ALMA MATER STUDIORUM - UNIVERSITÀ DI BOLOGNA

School of Engineering and Architecture

Department of Electrical, Electronics and Information Engineering - DEI

Master Thesis
In
Automation Engineering

Finding Objects in Images by Line Segments

Candidate:
Majid Mazaheri

Supervisor:
Prof. L. Di Stefano

Academic Year [2016/17]
Session [III]

Table of Contents

Table of Figures

Acknowledgment

I have been extremely fortunate to have Prof. Luigi Di Stefano as my academic advisor and mentor and I would express my sincere gratitude to him, for his support, guidance, help and critical appraisal which have always been with me.

I am glad that I decided to do my dissertation with him, and again thanks to him for suggesting me such a good topic, and even though I was not sure of my abilities to do it, he encouraged me to do, and finally I did a great job. I appreciate that, although he is very busy with university stuff, but he never delayed on answering my emails more than a day

Chapter 1

1 Introduction

As we know, nowadays, Computer vision plays a big role in modern world , it is used in vast variety technologies such as:
1-inteligent transportation systems
2-moving object tracking
3- defense surveillance
4-robots
…

One of the very useful and challenging topics of Computer vision is about detecting objects in images. Detecting objects should be accurate as much as possible and in same time to be fast to be applicable in real time applications.

The aim of this thesis is to find objects in images. The work has been based on two important tools in Computer vision, Generalized Hough Transform and LSD line segment detection.

Firstly, we find lines in images by means of LSD, then by using the Generalized Hough Transform we use these lines to find favorite object in image. All work is done by Open cv 3.1, based on C++ in visual Studio 2015 environment.

Chapter 2

2 Hough Transform

The Hough transform is a Object detection technique used in image analysis, computer vision, and digital image processing. The purpose of the technique is to eliminate imperfect instances of objects within a certain class of shapes by a voting procedure. This voting procedure is carried out in a parameter space, from which object candidates are obtained as local maxima in a so-called accumulator space that is explicitly constructed by the algorithm for computing the Hough transform.

The classical Hough transform was concerned with the identification of lines in the image, but later the Hough transform has been extended to identifying positions of arbitrary shapes, most commonly circles or ellipses. The Hough transform as it is universally used today was invented by Richard Duda and Peter Hart in 1972, who called it a "generalized Hough transform" after the related 1962 patent of Paul Hough.The transform was popularized in the computer vision community by Dana H. Ballard through a 1981 journal article titled "Generalizing the Hough transform to detect arbitrary shapes". [1]

2.1 Generalized Hough Transform

The Hough transform was initially developed to detect analytically defined shapes (e.g., line, circle, ellipse etc.). In these cases, we have knowledge of the shape and aim to find out its location and orientation in the image. The Generalized Hough Transform or GHT, introduced by Dana H. Ballard in 1981, is the modification of the Hough Transform using the principle of template matching. This modification enables the Hough Transform to be used for not only the detection of an object described with an analytic function. Instead, it can also be used to detect an arbitrary object described with its model.

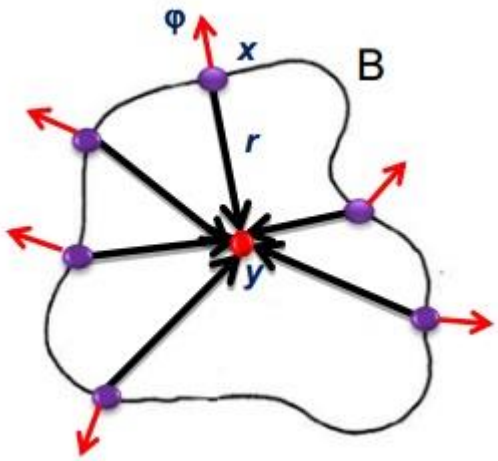
The problem of finding the object (described with a model) in an image can be solved by finding the model's position in the image. With the generalized Hough Transform,

the problem of finding the model's position is transformed to a problem of finding the transformation's parameter that maps the model into the image. As long as we know the value of the transformation's parameter, the position of the model in the image can be determined.The original implementation of the GHT uses edge information to define a mapping from orientation of an edge point to a reference point of the shape. [2]

Generalized hough Transform has 2 phases ,offline and online phase. [3]

2.2 Offline phase

First phase is offline phase,and it is about creating R-Table .We have our sample object edge profile, and each edge pixel has its own gradient direction(it has an angle).We choose a refrence point ,preferably inside our object.Make a “r” vector from this refrence point to all edge pixels of our model.Now Gradient direction is quantized according to a chosen step Δφ ,so it depends on our choice ,Store “r” as a function of Δφ (R-Table),for example, I choose step Δφ = 5, so the table has 360/5= 72 rows, and I have totally 10 edge points with gradient angle between 0 and 5, so in the table in first row I put 10 different r vectors.



i	ϕ_i	R_{ϕ_i}
0	0	$\{r \mid y - r = x, x \in B, \phi(x) = 0\}$
1	$\Delta\phi$	$\{r \mid y - r = x, x \in B, \phi(x) = \Delta\phi\}$
2	$2\Delta\phi$	$\{r \mid y - r = x, x \in B, \phi(x) = 2\Delta\phi\}$
...	...	...

An entry in the R-Table can contain several r vectors.

2.3 Online phase

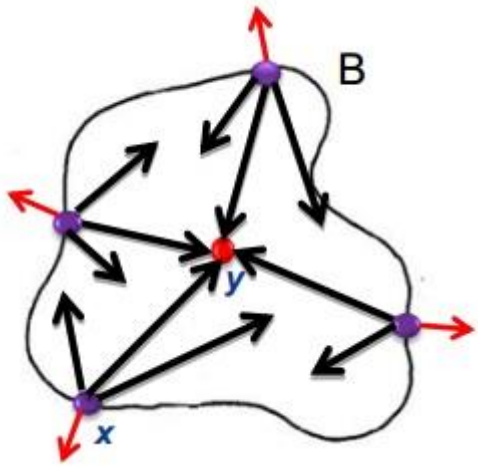
Second phase is online phase;in this phase we do as below :
1. An image *A[y]* is iniatialized as accumulator array.
For each edge pixel *x* of the input image :
2. Compute gradient direction *φ*

3. Quantize φ to index the R-Table. For each *r*
vector stored into the accessed row:

a) Compute the position the reference point *y*:
$y=x+r$
b) Cast a vote into the accumulator array:
A[y]++

4. As usual with the HT, instances of the sought object are
detected by finding peaks of the accumulator array.

In generalised Hough method ,is possible to handle rotation and scaling,or both together,but it is expensive in terms of time.



i	ϕ_i	R_{ϕ_i}
0	0	$\{r \mid y - r = x, x \in B, \phi(x) = 0\}$
1	$\Delta\phi$	r1,r2,r3
2	$2\Delta\phi$	$\{r \mid y - r = x, x \in B, \phi(x) = 2\Delta\phi\}$
...	...	...

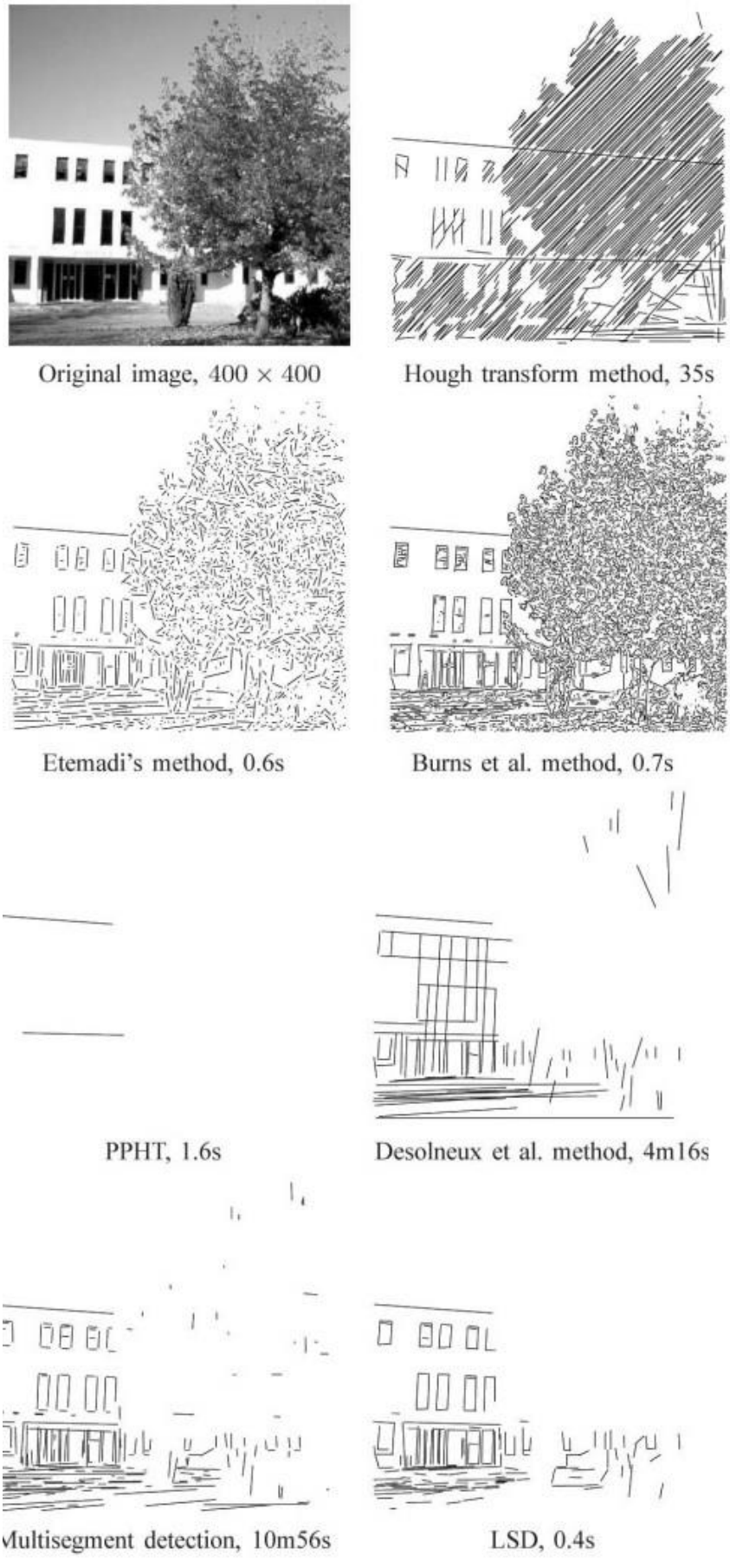
Chapter 3

3 LSD: A Fast Line Segment Detector with a False Detection Control [4]

LINE segments give important information about the geometric content of images. First, because most human-made objects are made of flat surfaces; second,because many shapes accept an compact description in terms of straight lines. Line segments can be used as low level features to extract information from images or can serve as a basic tool to analyze and detect more elaborated shapes.

As features, they can help in several problems such as stereo analysis , crack detection in materials and image compression.Ideally, one would like to have an algorithm that accurately detects the line segments present in an image,without false detection, and without the need to manually tune parameters for each image or group of images.

To evaluate how far the Line Segment Detector (LSD) presented in this paper goes in that direction, all experiments will be made with the same parameters regardless of the different image origin, scene, and resolution.Line segment detection is an old and recurrent problem in computer vision. Standard methods first apply Canny edge detector followed by a Hough transform extracting all lines that contain a number of edge points exceeding a threshold. These lines are thereafter cut into line segments by using gap and length thresholds. The Hough transform method has serious drawbacks. Textured regions that have a high edge density can cause many false detections (see the slanted lines on the tree of Fig. 3.1).



The processing times on an Apple PowerBook G4 1.5 GHz are indicated. TheHough transform, Etemadi, and Burns et al. methods detect many irrelevant small line segments on the tree. PPHT produces no false detection but fails to detect small line segments. Desolneux et al. controls the false detections but gives an inaccurate interpretation when aligned line segments are present. The multisegment detector gives a good result, but in prohibitive time. LSD gives a similar result in linear time.

figure 3. 1 comparison of various line segment detection methods

Ignoring the orientation of the edge points, such algorithms obtain line segments with wrong directions. Also, setting thresholds is a fundamental problem for all detection methods. Using fixed thresholds can lead to a significant number of false positives or false negatives (see Fig. 3.1). Another classic method starts from edge points, chains them into curves and then cuts the chains into line segments by a straightness criterion . A standard chaining method is due to Etemadi[5] . This method is parameterless and usually gives accurate results. Also, it is one of the few algorithms that simultaneously detect line segments and arcs. Nevertheless, the result is not completely satisfactory, as illustrated in Fig. 3.1. Many detected straight and small edge curves are false positives: Here comes the fundamental threshold problem again. Burns et al[6]. introduced a linear-time line segment detection method with a key new idea. Their algorithm does not start with edge points, and actually ignores gradient magnitudes, using only gradient orientations. This algorithm was improved by Kahn et al[7]. . The line segments given by this algorithm are well localized, but the threshold problem is still there. The foliage of the tree in Fig. 3.1 could be described as a texture, as an object, but certainly not as a set of line segments. The examination of these methods suggests that a selection criterion should be added as a final step. There were some propositions of such criteria for the classic methods. A good example is the Progressive Probabilistic Hough Transform (PPHT) proposed by Matas et al[8]. . Like many similar methods, it accelerates the computing time by a random selection of the edge points. But the improvements came from the use of the image gradient information and a false detection control. Fig. 3.1 shows a clear improvement over the standard Hough transform method. Nevertheless, the false detection control used is not completely satisfactory. First, the mechanism is well adapted for whole lines and not for line segment detection. Long line segments (similar in length to lines in the image) produce detections, but small ones do not. As a result, many short line segments are missing, as Fig. 3.1 shows. Second, the detection parameter of the method is the probability of getting a false detection each time an edge point is analyzed. But, the number of edge points analyzed depends on the image size and the expected number of false detections too. The default value of this parameter is set to control false detections on image sizes of about 256 256. For larger images, the false detections are not controlled anymore. Any fixed value produces false detections on large images and misses on small ones. This threshold question was thoroughly analyzed by Desolneux, Moisan and Morel[9] . Their line segment detection method succeeds in controlling the number of false positives. The method counts the number of aligned points (points with gradient direction

approximately orthogonal to the line segment) and finds the line segments as outliers in a nonstructured, a contrario model. This method is based on a general perception principle, the Helmholtz principle [9], according to which an observed geometric structure is perceptually meaningful when its expectation in noise is less than 1. Applying the Helmholtz principle guarantees the lack of false positives in the weak sense that, on average, only one false detection would be made in a white noise image of the same size as the analyzed image. It also guarantees no false negative, in the sense that a line segment that could arise in noise must be considered as a true negative. The detection of line segments by Helmholtz principle was subject to a controlled psychovisual comparison with human perception . This comparison uses synthetic images. The ground truth is made of deterministic alignments. It is superposed to a background clutter made of small random line segments. By varying the parameters of the background and of the ground truth, this setting gives experimental detection-rejection curves that can be compared to the theoretical ones predicted by Helmholtz principle. The results in show a convincing agreement, both qualitative and quantitative, between the predicted and observed curves on 20 subjects. The Desolneux et al. method has been extensively tested. Experimental evidence, including all images presented here, confirms that it indeed finds the line segments in the image where alignments are intuitively present (no false negative). It has few false positives, as guaranteed by the method. Unfortunately, it often misinterprets arrays of aligned line segments (see, for example, the windows in Fig. 1). A detailed analysis of this defect was performed, and a satisfactory solution found in . The solution involves a more sophisticated use of the Helmholtz principle computing and comparing the meaningfulness of all possible arrays of line segments (multisegments) on each line. The misinterpretations of the Desolneux et al. method were corrected, giving a much more accurate line segment detector (see Fig. 3.1). Unfortunately, the Desolneux et al. and the multisegment detectors are exhaustive algorithms. The Desolneux et al. method tests every possible line segment in the image and has complexity. Thus, these detectors are doomed to be used only for offline applications. The aim of this paper is to present a linear-time algorithm that cumulates most of the advantages of the previous algorithms without their drawbacks. The Burns et al. line segment finder, that made a breakthrough in the extraction of the line segments, will be improved and combined with a validation criterion inspired from Desolneux et al. The result is LSD, a linear-time line segment detector that requires no parameter tuning and gives accurate results (see Fig. 3.1).

3.1 EXPERIMENTS

The processing time for a 512 512 image is a fraction of a second. This permitted to test the algorithm on thousands of images, images of different kinds, origins, sizes, and noise levels; some tests were also performed on videos. It should be emphasized that all the experiments were done without tuning any parameter at all. The results shown in this paper were obtained on some of the most challenging images. Fig. 8 shows a plot of the computational time in seconds versus the image size on an Apple PowerBook G4 1.5 GHz. For most images the computation time is shorter than for a white noise image of the same size (the slanted linear series of points on the plot). The main reason is that, in natural images, many pixels are discarded by the gradient threshold. Some images, however, require longer processing time than white noise images, as can be seen on the plot. It is usually the case in images with noise-like structures (like a grass background) and long range structures (like objects on the foreground). Fig. 3.2 shows a series of experiments on natural images. Note that the detected line segments represent well the structure of these diverse images. The images in the left-hand column contain highly geometrical contents. Almost all the expected line segments were found, the perceptual exceptions being small ones that lay beyond the meaningfulness limit. In accordance with the theory, there are very few false detections. The images in the right-hand column show results on some nongeometric images. Most detections in this second group of images do not correspond to real straight or flat objects; they correspond to locally straight edges. In cases like the profile of an arm, a line segment interpretation is not strictly correct in the sense that an arm is not straight, but it is a reasonable interpretation in terms of the 2D structure present on the image at a given resolution. For curved edges, like the one in the wheel on the middle-right image, the line segment interpretation is only an economic approximation to a curve. Such results are acceptable in the sense that every detection corresponds to a locally straight structure in the image and every locally straight structure has a line segment associated.

figure 3. 2 Result of LSD on natural images.

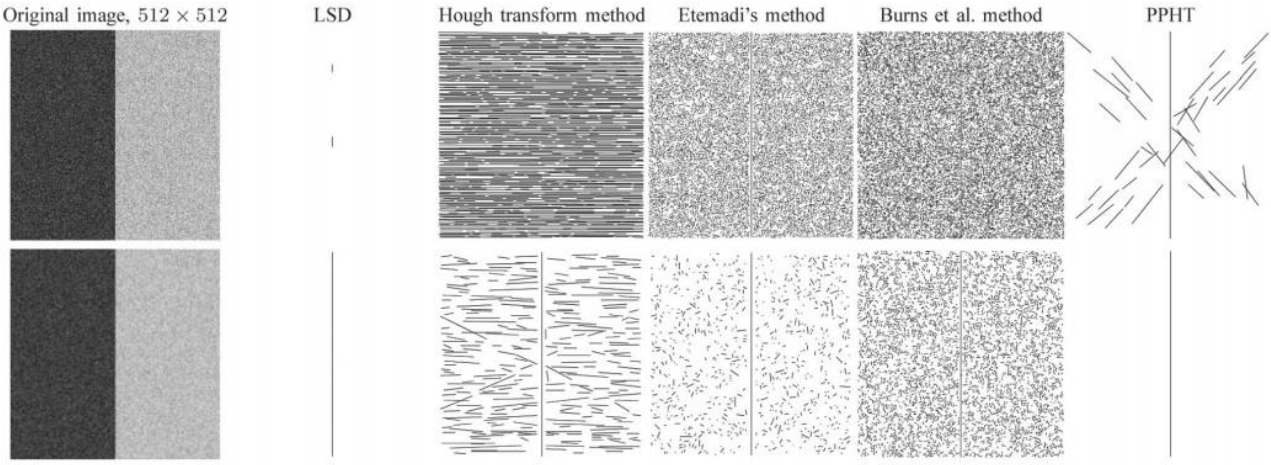


figure 3. 3 Analysis of a noisy edge image at two different scales by five-line segment detection algorithms.

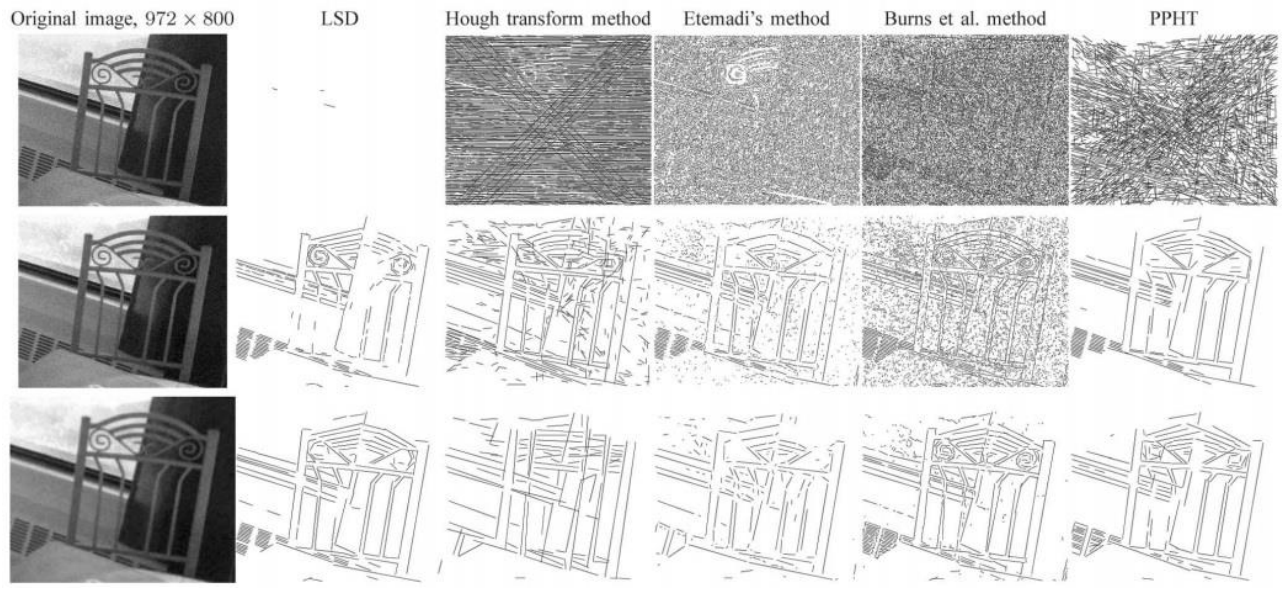


figure 3. 4 The same effect shown in Fig. 3.3 can be seen on a noisy natural image.

The presence of noise can deteriorate the performance of LSD. Helmholtz principle states that no detection should be made on white noise images. Thus, when noise with increasing variance is added, the NFA value of meaningful line segments increases. Eventually, noise dominates the image, the NFA becomes larger than 1, and the line segment is no longer detected. See [20] for more details. Noise affects the region growing algorithm too and this effect is critical for LSD. Noise produces variations to the level-line angles. Low power noise produces negligible variations and the results are not affected. With moderate noise power, the variations to the level-line angles become larger and the angle difference can reach the angle tolerance value. Then, the region growing algorithm is no more able to follow the edge and the line support regions become fragmented. In the presence of strong noise, only small line-support regions are formed and no detection is made. Fig. 3.3 (top) shows an example of a synthetic noisy image; only two small fractions of the edge were detected by LSD. Nevertheless, no false detections were made, in striking contrast to state-of the-art methods, see Fig. 3.3. The same effect happens on real images, see Fig. 3.4. A criticism can be raised to the theory: Almost no line segment is detected in the images of Figs. 3.3 and 3.4 at full resolution, even if they are perfectly visible for us. The answer is that the human visual system uses a multiscale analysis. For a fair comparison one must authorize the line segment detection theory to analyze the image at a different scale too. (The Hough transform methods, Etemadi's method, and PPHT include such a step since they rely on Canny points, whose processing implies Gaussian filtering. Figs. 3.3 (bottom) and 3.4 (middle and bottom) show that the line segments masked by noise can be detected by the very same algorithm at coarser scales. Analyzing at a coarser scale may also help detecting global structures masked at full scale by details of the image. However, this experiment made with strong artificial noise does not imply that a multiscale theory for line segment detection is necessary. It only points out that noise should be detected, and a denoising step or a zoom-in performed when the noise happens

to be unusually strong. Fig. 3.5 shows a comparison of seven-line segment detection algorithms on a natural image. The computation time on an Apple PowerBook G4 1.5 GHz is shown for all of them, as well as the number of line segments found. The algorithms used were: the Hough transform method as implemented in the freely available package Hough tool [13]; Etemadi's algorithm in its original implementation ORT-2.3; Burns et al.'s algorithm, using an implementation by Ross Beveridge ; PPHT as implemented in the freely available RAVL libraries; detector; As shown above, the Hough transform method often produces false detection, simply because it does not take\ into account edge orientation: see the horizontal ones on the hair of the man. Etemadi's and Burns et al.'s methods have the threshold problem: A decision rule is needed to select the good line segments; otherwise, the detection is useless (1,517 and 3,677 line segments detected, respectively). PPHT fails to detect many small line segments due to too strict detection thresholds, well adapted for line detection but not for line segment detections. Desolneux et al.'s method produces no false detection but fails to get the right interpretation when aligned line segments are present, as in the balustrade at the background of Fig. 3.5. Also, when slow gradients are present, like in the shadow of the table on the left of the image, this produces multiple parallel detections instead of one wider line segment. Multi segment detection produces good results (even if the parallel detections problem is still present and in some cases, hallucinates global aligned structures not present, see). Its computation time is prohibitive, though. LSD produces a good description of the image structure with 716 line segments in 0.7 seconds. The results of the multi segment detector and LSD are similar in most cases. Let us comment briefly on the main differences. The multi segment detector processes every line that crosses an image, producing a global interpretation for each one of them. This process occasionally leads to the hallucination of a global structure that is not present in the image. Fig. 3.1 shows an example: The small line segments in the foliage that are horizontal or vertical, and aligned with some true alignment present on the image are kept. This obviously casual alignment reinforces the meaningfulness of the line. As a consequence, the small line segments, nonmeaningful by themselves, are interpreted as parts of a larger multi segment. Being much more local, LSD does not present this problem. But, the global analysis of the multi segment detector is needed, however, to grasp the right interpretation on other cases, as Fig. 3.5 shows. In this image, LSD cannot detect small line segments that are too short, while the multi segment algorithm succeeds in reconstructing the global structure.

figure 3. 5 A comparison of various line segment detection methods

3.3 CONCLUSIONS

An examination of the experimental results (and of many others that the reader may wish to perform online) indicates that a line segment detection can be accurate and have a small amount of false positive and false negative detections. This promising result can be primarily attributed to Burns et al., who procured the right edge representation as a region of aligned orientation pixels, and to Desolneux et al., who proposed a general method to eliminate false positives. The experimental results substantiate the idea that a reliable raw primal sketch is doable in digital images. However, experiments also show that the story is not complete, but just starting. A circular plate is detected as a concatenation of straight line segments. This representation must lead to the notion of the curve. A more accurate representation should distinguish real polygons from curves. It should also permit building up more elaborate gestalts such as bars, regular polygons, periodic grids, or stripes, to name a few that are visible in the test images presented in this paper.

Chapter 4

4 Finding objects By LSD

4.1 Finding line segments in smple object Image

As a first step ,LSD function takes sample object image and findes line segments and saves them into a Coordinates matrix.Each line is saved in one row (startpoint X,startpoint Y ,endpoint X,endpoint Y)

X and Y are pixel coordinates of image.

4.2 Making r-table

For each line segment of model object ,after finding the angle and middle point ,we save them in a Matrix ,then we find masscenter of these mid_points and subtract mid point of each line segment from masscenter ,so it is like, we have an arrow per each line from line's middle point toward masscenter point.

It is not necessary to compute masscenter ,which is :

[(∑x)/(number of lines), (∑y)/(number of lines)] ,

It can be any arbitrary point inside or outside of our object(preferably inside),just to compose **R** arrows.So now we have a matrix **R** containing all our **R** arrows and angle Matrix,contributed to each line segment of model object .Now we create a table, which has row numbers equal to **180/teta_step**(*) .We categorise all lines in this table ;It is easier to describe with an example.Imagine that teta_step = 5◦ ,so all lines with slope [0 ~ 5) degrees, goes to first row .And all lines with slope [5 ~ 10) degrees goes to second row,and so on till the end.As we see it is possible that, in this matrix, a row can have zero **R** vector or many **R** vectors.

*Teta_step is defined by user at the beginning of code,

*180 degrees, because for example if a line has 182◦ slope ,it has same slope as 2◦ slope line ,so 0-180 degrees interval is enough .

4.3 Finding line segments in main Image

As second step ,lsd function takes main image(the image that we wish to find our object inside it),and gives us back a matrix ,containing all line segments inside our main image,in same format as sample object image lines and store them in a matrix.

we make another angle matrix containing all angles of lines of main image,each angle goes to a row that corrosponds to its line row number.

4.4 Applying R-table to main image lines

We create a Bin,a matrix bigger than our main image size ,which,here is twice size;the reason, is that, maybe my desired object is at edge of image and barycenter falls outside of image,so by making bin size bigger than image,the barycenter is still inside of bin.Now that, we have all our lines info (slope and midpoint) of main image, we look at angle of each line of main image and we pick from R_table,corrosponding row and we add all available R vectors in that specific row of R_table to middpoint of our chosen line,so we get coordinate of a point and we vote 1 to correspandance point in bins. After finishing all ,we find the pixel with highest vote in this bin ,and most probably, that is our desired object place.now we transfer this point to our maine image coordinates .now this is actually our barycenter so we add all R vectors of sample object to this point(max bin) ,and we get our searched object.

4.5 Handling rotation

For handling rotation ,we assume that our searched object probably is rotated in our main image.So what we do is , we rotate all line segments of sample object by **rotation_step** angle ,which is defined by user ,so the only difference is that we have new line segments coordinate matrix,and the rest procedure is same as before.and we continue this rotation until end of **rotation_interval** which is defined by user.

Lets see it, with an example;lets assume ,our object is rotated 30 degrees.

We set rotation interval [10 ~ 50] and rotation step= 5 degrees;

So at first step we imagine,our object has been rotated 10 degrees ,consecuently all lines and also r vectors connecting barycenter to these lines has been rotated 10 degrees,we apply this rotation and we get new R vectors and new angles for lines.

From this point ,the procedure is same as without rotation procedure, we search for object in our main image according to this new rotated lines and r vector,and we find the point at bin,with more votes .and we save coordinates of this point,the angle of

rotation and number of max votes in **All_possible_objects** matrix .Then we add one rotation-step to our rotation ,new teta = 10 + rotation step = 10+5=15 .So we repeat the rotation procedure this time for 15 degrees,and at the end we save rotation angle,max bin,and coordinate of max bin.

We continue this procedure till we reach the end of our rotation interval,here 45 degrees.then we compare all our max bins and choose the max of them ,we have coordinate of that point and also we know the rotation angle so we can find our rotated r vectors,and add them to this barycenter and show the found rotated object

notes:

1: one possibility is that for each rotation angle we make separate bin,and save all bins for all rotation angle steps,another is the way I have used ,to save only the max point coordinate and number of votes of it and angle of rotation;I choosed second for memory efficiency

2:depending on how many objects we are searching we can change slighty our approach,if there is one object ,the way that I described is ok,but if there is several objects ,then we should define “good enough max” and then at each rotation level if the max been is bigger than “good enough” then we choose it as our rotated object and then we continue our search for other rotated samples.

3: this code does not work if the object is scaled,for that, is very important to have pictures of model and main be taken from same distance and later if there is some edits on pictures it should be same for model and main picture so that pixels of those 2 picture are in same depth

Chapter 5

5 Experiments and Conclusions

For all experiments these parameters are same:

table step = 3

Rotation step = 3

good_enough_max_bin = 25

kernel size = 21

supress_line_check = no

Sample 1

Experiment 1

Figure 5. 1 Sample 1_ Red lines are found lines by LSD

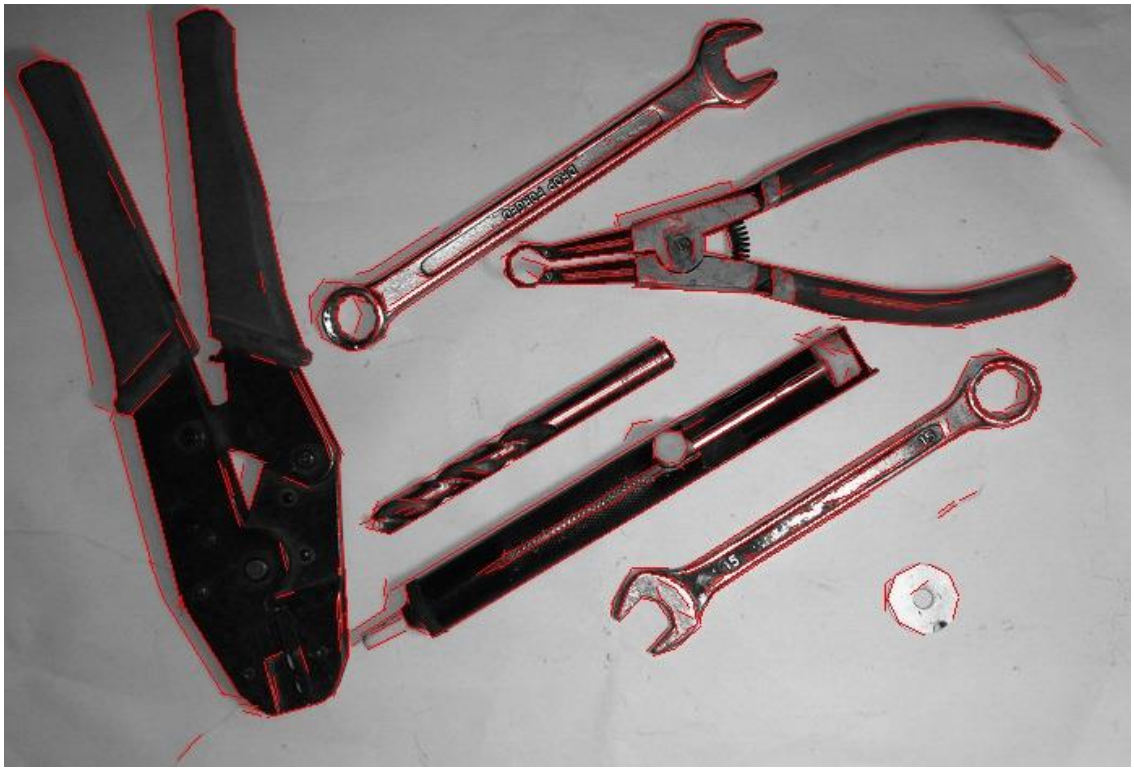


Figure 5. 2 _The result of LSD on image 1;Red lines are found lines

Figure 5. 3 _Original image ;Green lines and red dots are showing the found object

number of lines of sample object = 116

number of lines of main picture = 452

candidates for found object:

A: (459,142) →angle = -83° , max bin = 36

B: (456,134) →angle = -80° ,max bin = 33

preferred : A

Rotation angle : [-179 , 180]

Experiment 2

Figure 5. 4 _The result of LSD on image 2;Red lines are found lines

Figure 5. 5_Original image 2 ;Green lines and red dots are showing the found object

number of lines of sample object = 116

number of lines of main picture = 547

candidates for found object:

A: (406,169) →angle = -83^{o} , max bin = 35

B: (410,155) →angle = -80^{o} ,max bin = 26

B: (286,161) →angle = -65^{o} ,max bin = 26

chosen: A

Rotation angle : [-179 , 180]

Time elapsed : 4589 ms

Experiment 3

Figure 5. 6_Original image 3 ;Green lines and red dots are showing the found object

number of lines of sample object = 112

number of lines of main picture = 673

candidates for found object:

A: (338,285) →angle = -83^{o} , max bin = 31

B: (371,291) →angle = 64^{o} ,max bin = 51

C: (363,296) →angle = 67^{o} ,max bin = 28

D: (363,296) →angle = 97^{o} ,max bin = 27

preferred : B

Rotation angle : [-179 , 180]

Time elapsed : 3323 ms

Experiment 4

Figure 5. 7 _Original image 4 ;Green lines and red dots are showing the found object

number of lines of sample object = 112

number of lines of main picture = 533

candidates for found object:

A: (621,285) →angle = 1◦ , max bin = 31

B: (614,281) →angle = 4◦ ,max bin = 34

preferred : B

Rotation angle : [-179 , 180]

Time elapsed : 4217 ms

Sample 2

Experiment 1

Figure 5. 8 Sample 2_ Red lines are found lines by LSD

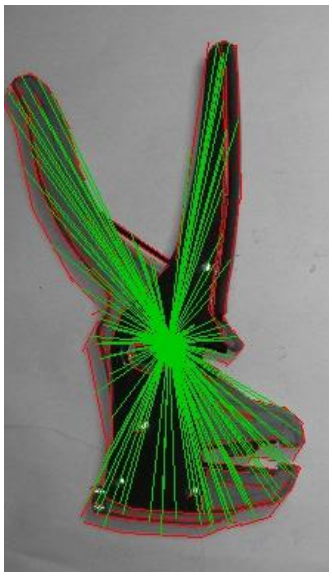


Figure 5. 9 Sample 2_Green arrows are arrows from each found line by LSD, toward refrence point for making r_table

Figure 5. 10_Original image 5 ;Green lines and red dots are showing the found object

number of lines of sample object = 149

number of lines of main picture = 575

candidates for found object:

A: (609,290) →angle = -5◦ , max bin = 47

B: (612,287) →angle = -2◦ ,max bin = 39

preferred : A

Rotation angle : [-179 , 180]

Time elapsed : 3593 ms

Experiment 2

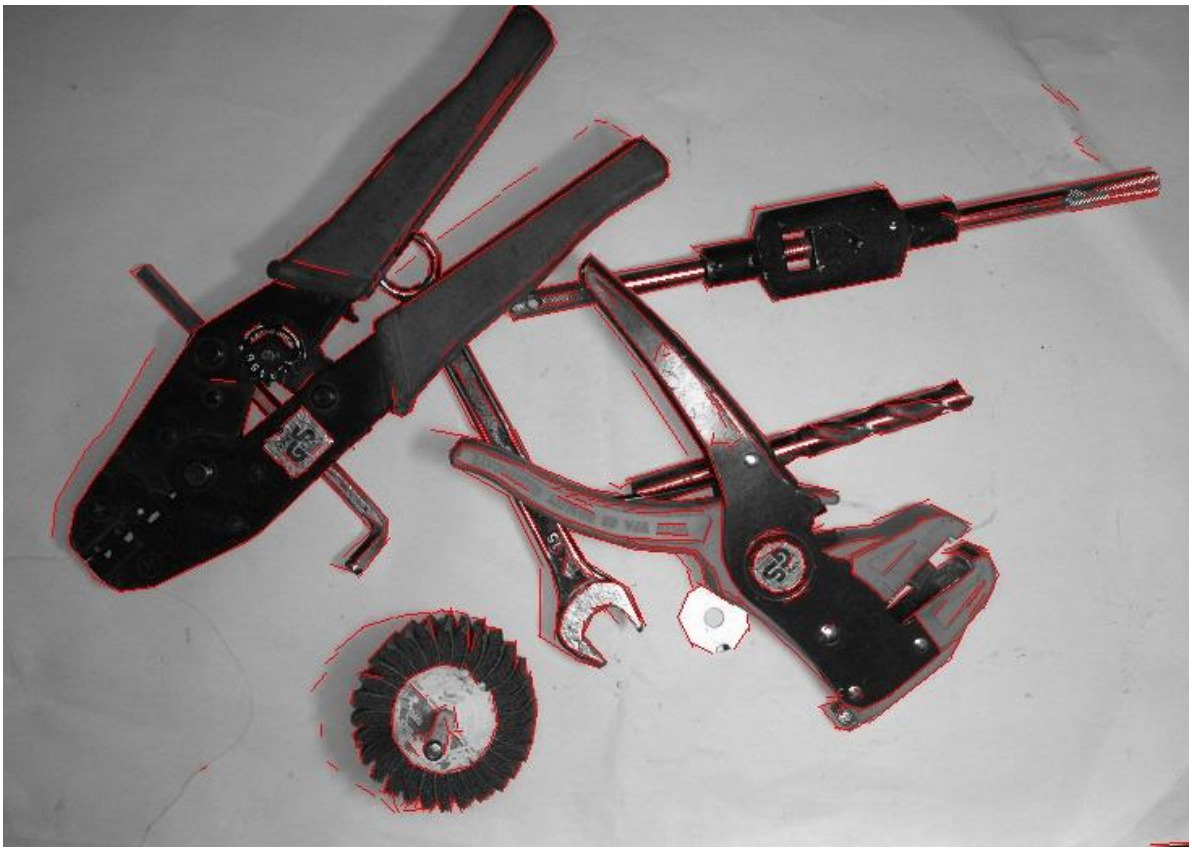


Figure 5. 11_ Result of LSD on image 6;Red lines are found lines

Figure 5. 12_Original image 6,The object has been found correctly

number of lines of sample object = 149

number of lines of main picture = 506

candidates for found object:

A: (474,331) →angle = 40◦ , max bin = 28

B: (473,329) →angle = 43◦ ,max bin = 46

C: (475,325) →angle = 46◦ ,max bin = 38

chosen: B

Rotation angle : [-179 , 180]

Time elapsed : 6030 ms

Experiment 3

Figure 5. 13_Original image 7,object has been found correctly

number of lines of sample object = 149

number of lines of main picture = 502

candidates for found object:

A: (527,334) →angle = 32◦ , max bin = 38

B: (532,329) →angle = 35◦ ,max bin = 46

C: (529,336) →angle = 38◦ ,max bin = 29

chosen: B

Rotation angle : [20 , 60]

Time elapsed : 3942 ms

As you see time decreased, because we decreased rotation angle

Experiment 4

Figure 5. 14_Original image 8,Object has been found correctly

number of lines of sample object = 149

number of lines of main picture = 409

candidates for found object:

A: (122,292) →angle = 0◦ , max bin = 56

B: (127,294) →angle = 3◦ ,max bin = 40

chosen: B

Rotation angle : [0 , 20]

Time elapsed : 1528 ms

Experiment 5

Figure 5. 15_ Original image 9,Object has been found correctly

number of lines of sample object = 149

number of lines of main picture = 547

candidates for found object:

A: (228,314) →angle = 72◦ , max bin = 26

B: (218,310) →angle = 75◦ ,max bin = 50

C: (225,306) →angle = 78◦ , max bin = 43

D: (220,299) →angle = 81◦ , max bin = 29

chosen: B

Rotation angle : [0 , 90]

Time elapsed : 3327 ms

Experiment 6

Figure 5. 16_ Original image 10,Object has been found correctly

number of lines of sample object = 149

number of lines of main picture = 673

candidates for found object:

A: (178,349) →angle = 45◦ , max bin = 38

B: (183,346) →angle = 48◦ ,max bin = 51

C: (189,348) →angle = 51◦ , max bin = 31

chosen: B

Rotation angle : [0 , 90]

Time elapsed : 1865 ms

Experiment 7

Figure 5. 17_ Original image 11,Object has been found correctly,even though it is partialy occluded

number of lines of sample object = 149

number of lines of main picture = 663

candidates for found object:

A: (410,254) →angle = 60° , max bin = 29

B: (414,254) →angle = 63° ,max bin = 44

C: (410,249) →angle = 66° , max bin = 59

D: (416,247) →angle = 69° ,max bin = 26

E: (405,245) →angle = 75° , max bin = 27

chosen: C Rotation angle : [0 , 90] Time elapsed : 2823 ms

Experiment 8

Figure 5. 18_ Original image 12,Object has been found correctly,even though it is partialy invisable

number of lines of sample object = 149

number of lines of main picture = 663

candidates for found object:

A: (487,49) →angle = -42◦ , max bin = 28

chosen: A Rotation angle : [-90 , 0] Time elapsed : 3781 ms

Experiment 9

Figure 5. 19_ Original image 13,Object has been found correctly,even though it is partialy invisable

number of lines of sample object = 149

number of lines of main picture = 673

candidates for found object:

A: x,y= (509,-22) , angle = -42◦ , max bin = -45 B: x,y= (515,-11) , angle = -42◦ , max bin = 19

C: x,y= (289,179) , angle = -36◦ , max bin = 18 D: x,y= (262,260) , angle = -18◦ , max bin = 18

A: x,y= (234,291) , angle = -3◦ , max bin = 19

chosen: B Rotation angle : [-90 , 0] Time elapsed : 1853 ms

in this case it couldn't find the object with before settings, so I changed good-enough filter to 17, and as you see it found.

Sample 3

Experiment 1

Figure 5. 20_ Sample 3_ Red lines are found lines by LSD,there are some extra lines because of shadow

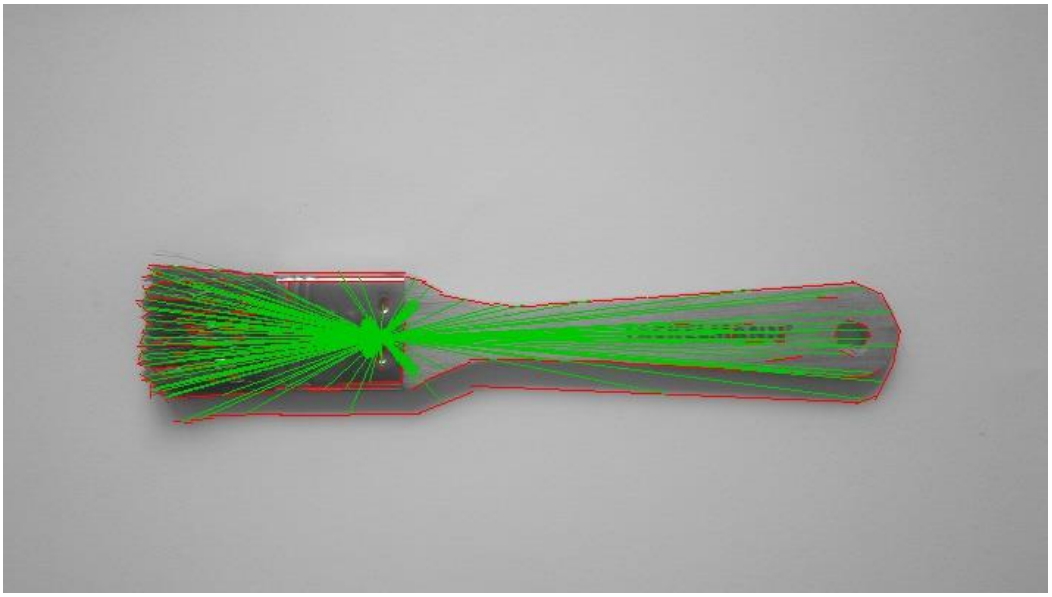


Figure 5. 21_ Sample 3_Green arrows are arrows from each found line by LSD,

toward refrence point for making r_table

Figure 5. 22_ Original image 14,Object has been found correctly,even though it is partialy occluded

number of lines of sample object = 115

number of lines of main picture = 311

candidates for found object:

A: (346,234) →angle = -5◦ , max bin =26

B: (348,230) →angle = -2◦ ,max bin = 40

preferred : B

Rotation angle : [-179 , 180]

Time elapsed : 6820 ms

Experiment 2

Figure 5. 23_ Original image 15,Object has been found correctly

number of lines of sample object = 115

number of lines of main picture = 339,112 506

candidates for found object:

A: (483,241) →angle = -179◦ , max bin =41

B: (482,245) →angle = -176◦ ,max bin = 34

preferred : A

Rotation angle : [-179 , 180]

Time elapsed : 3735 ms

Time is more because of printing last_bin

Sample 4

Experiment 1

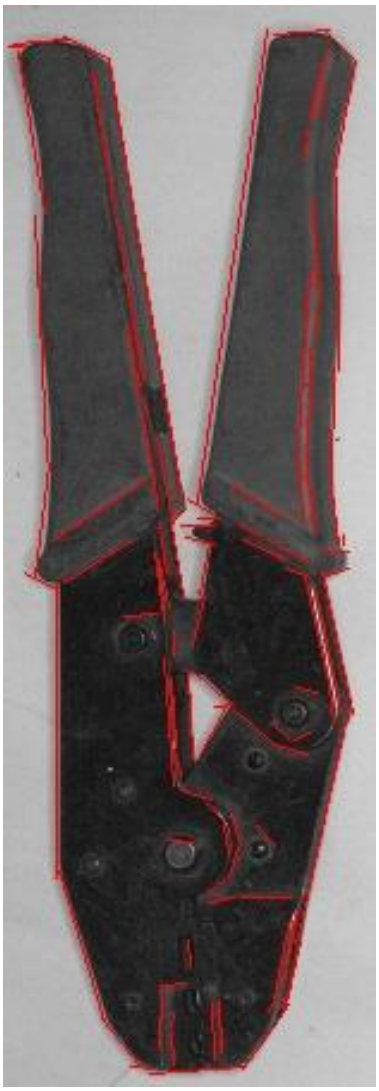


Figure 5. 24_ Sample 4_ Red lines are found lines by LSD

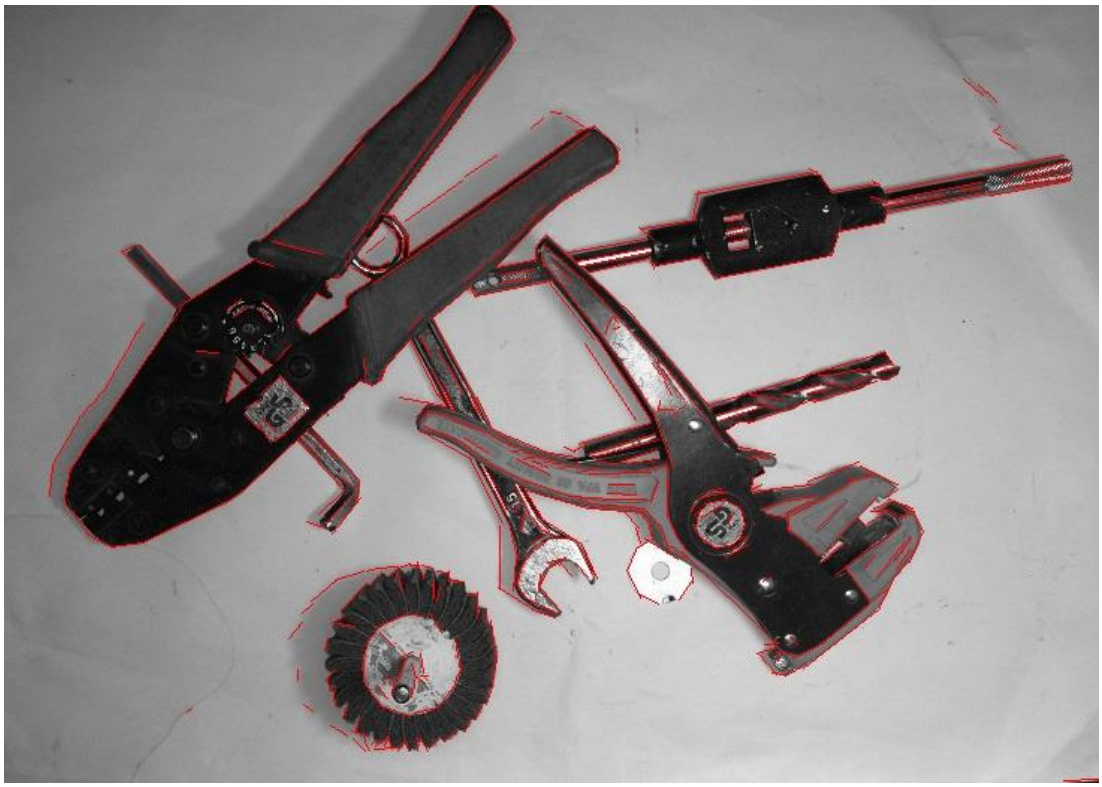


Figure 5. 25_ Result of LSD on image 16;Red lines are found lines

Figure 5. 26_ Original image 16;Object not found correctly

As we see in this case it could not find , probably because of light on main picture , so lines of object is not found perfectly

Experiment 2

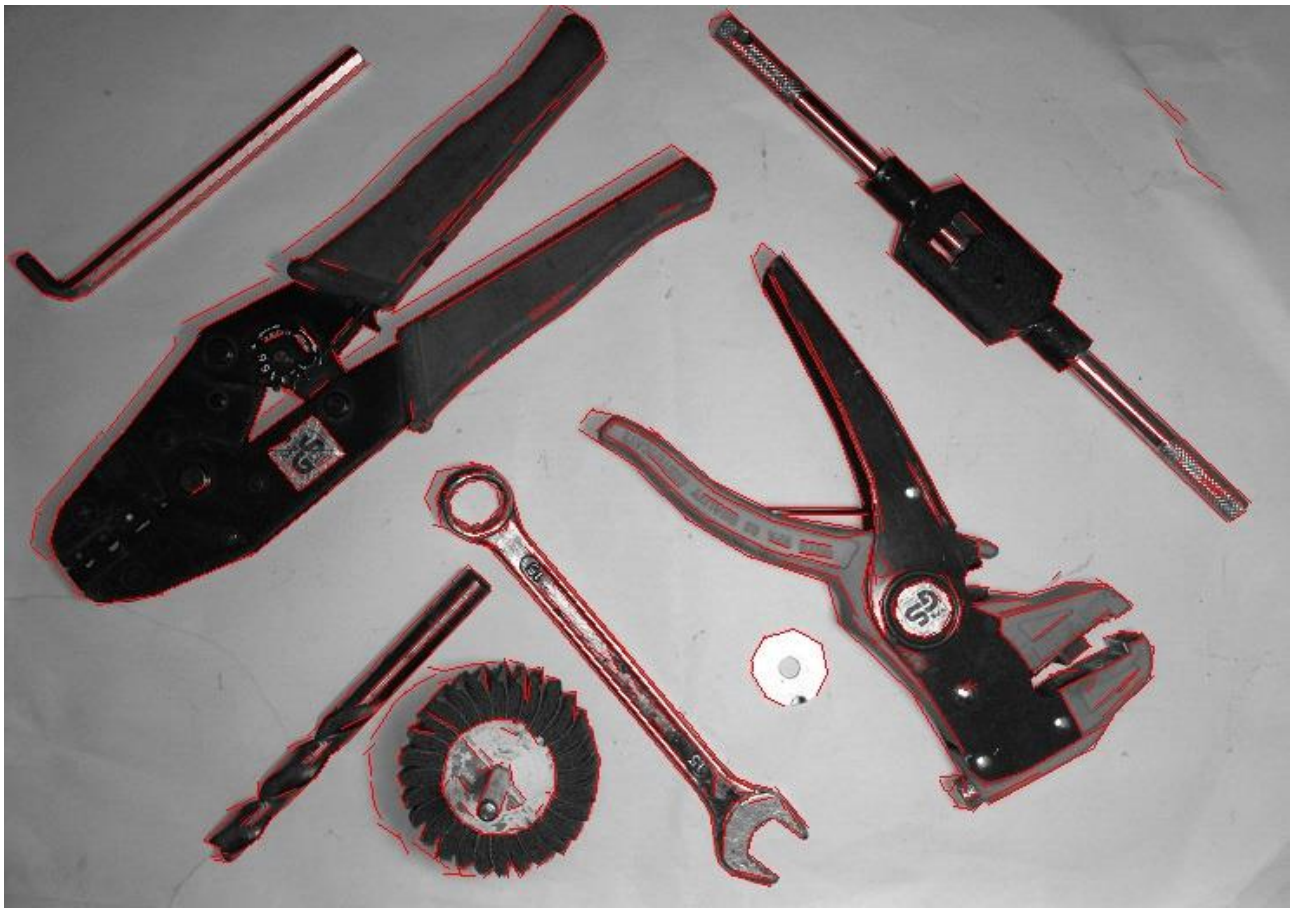


Figure 5. 27_ Result of LSD on image 17;Red lines are found lines

Figure 5. 28_ Original image 17;Object found correctly

number of lines of sample object = 112

number of lines of main picture = 502

candidates for found object:

A: x,y= (308,351) , angle = -149◦ , max bin = 25 B: x,y= (324,383) , angle = -143◦ , max bin = 21

C: x,y= (201,212) , angle = -50◦ , max bin = 29 D: x,y= (319,27) , angle = 130◦ , max bin = 22

preferred : C

Rotation angle : [-179 , 180]

Time elapsed : 5451 ms

Time is more because of printing last_bin

Experiment 3

Figure 5. 29_ Result of LSD on image 17;Red lines are found lines

Figure 5. 30_ Original image 18;Object found correctly

number of lines of sample object = 112

number of lines of main picture = 472

candidates for found object:

A: x,y= (550,352) , angle = -8◦ , max bin = 27 B: x,y= (124,276) , angle = 4◦ , max bin = 47

C: x,y= (516,148) , angle = 172◦ , max bin = 25 D: x,y= (285,298) , angle = 178◦ , max bin = 27

preferred : B

Rotation angle : [-179 , 180]

Time elapsed : 4361 ms

Time is more because of printing last_bin

Sample 5

Experiment 1

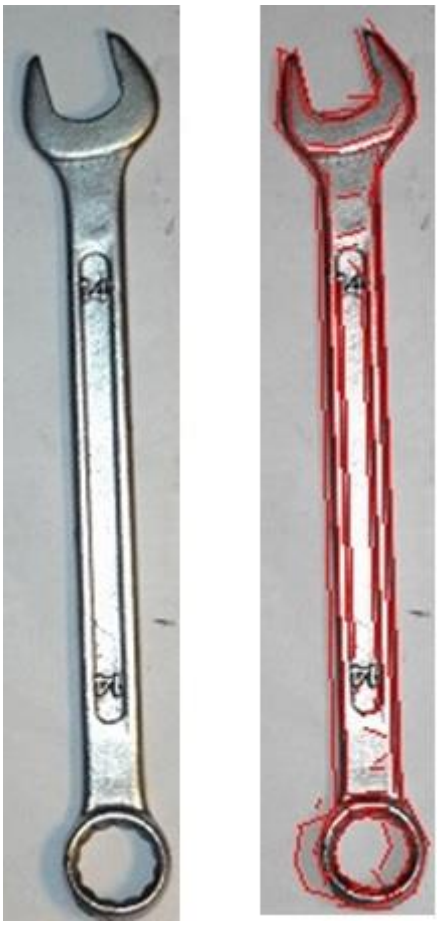


Figure 5. 31_Sample 5 , left : original photo of sample,right: sample after LSD

Figure 5. 32_Original image 19,object found correctly

number of lines of sample object = 86 number of lines of main picture = 575

Good_enough-max_bin = 20

candidates for found object:

A: x,y= (246,248) , angle = -8◦ , max bin = 23 B: x,y= (335,259) , angle = -2◦ , max bin = 24

C: x,y= (246,210) , angle = 169◦ , max bin = 22

preferred : B

Rotation angle : [-179 , 180]

Time elapsed : 3530 ms

Experiment 2

Figure 5. 33_Original image 20,object found correctly even though it is partialy occluded

number of lines of sample object = 86 number of lines of main picture = 575

Good_enough-max_bin = 22

candidates for found object:

A: x,y= (287,248) , angle = -152◦ , max bin = 25 B: x,y= (494,139) , angle = 100◦ , max bin = 25

C: x,y= (407,301) , angle = -74◦ , max bin = 23 C: x,y= (416,334) , angle = 85◦ , max bin = 23

preferred : A

Rotation angle : [-179 , 180]

Time elapsed : 4584 ms

As we see in this case ,false cases are very near to real one,and only because of structure of code ,by chance it choses correct one.

Experiment 3

Figure 5. 34_ Original image 21,object found correctly

number of lines of sample object = 86 number of lines of main picture = 502

Good_enough-max_bin = 20

candidates for found object:

A: x,y= (287,248) , angle = -140◦ , max bin = 22

preferred : A

Rotation angle : [-179 , 180]

Time elapsed : 4460 ms

Experiment 4

Figure 5. 35_ Original image 22,object found correctly

number of lines of sample object = 86 number of lines of main picture = 547

Good_enough-max_bin = 20

candidates for found object:

A: x,y= (280,139) , angle = -62◦ , max bin = 28 B: x,y= (274,141) , angle = -59◦ , max bin = 29

C: x,y= (484,326) , angle = 118◦ , max bin = 26 D: x,y= (302,129) , angle = 121◦ , max bin = 24

preferred : B

Rotation angle : [-179 , 180]

Time elapsed : 3498 ms

In this case ,C,D are the other similar object ,but since the code is not adjusted to find several objects ,it choses best candidate

Experiment 5

Figure 5. 36_ Original image 23,object found correctly

number of lines of sample object = 86 number of lines of main picture = 673

Good_enough-max_bin = 20

candidates for found object:

A: x,y= (320,270) , angle = -77◦ , max bin = 25 B: x,y= (310,277) , angle = -74◦ , max bin = 25

C: x,y= (425,248) , angle = 103◦ , max bin = 23 D: x,y= (368,419) , angle = 112◦ , max bin = 29

preferred : D

Rotation angle : [-179 , 180]

Time elapsed : 4204 ms

In this case ,A,B are for other similar object ,in both direction and C is false positive,but since the code is not adjusted to find several objects ,it choses best candidate

Experiment 6

Figure 5. 37_ Original image 24,object found correctly even though with some occlusion

number of lines of sample object = 86 number of lines of main picture = 673

Good_enough-max_bin = 20

candidates for found object:

A: x,y= (256,172) , angle = -161◦ , max bin = 27 B: x,y= (340,281) , angle = -74◦ , max bin = 21

C: x,y= (262,199) , angle = 16◦ , max bin = 23 D: x,y= (277,185) , angle = 19◦ , max bin = 23

E: x,y= (355,272) , angle = 106◦ , max bin = 26 F: x,y= (358,422) , angle = 109◦ , max bin = 26

preferred : A

Rotation angle : [-179 , 180]

Time elapsed : 3694 ms

In this case ,it finds the object,but as we see there are very close false positives cases available,also because similarity of both heads of object,it found the object in reverse angle,and actually,correct angle is 29.

Experiment 7

Figure 5. 38_ Original image 25,object found correctly

number of lines of sample object = 86 number of lines of main picture = 277

Good_enough-max_bin = 20

candidates for found object:

A: x,y= (268,219) , angle = -149◦ , max bin = 22 B: x,y= (277,238) , angle = 28◦ , max bin = 22

preferred : A

Rotation angle : [-179 , 180]

Time elapsed : 4101 ms

In this case ,since our object's both heads are similar ,so in fact case A and B are referring to same object only in reverse angle,and as we see since there is not many objects,it easily finds it.

Experiment 7

Figure 5. 39_ Original image 26,object found correctly

number of lines of sample object = 86 number of lines of main picture = 673

Good_enough-max_bin = 20

candidates for found object:

A: x,y= (339,345) , angle = -143◦ , max bin = 25 B: x,y= (330,354) , angle = -140◦ , max bin = 30

C: x,y= (505,181) , angle = -110◦ , max bin = 24 D: x,y= (503,187) , angle = -107◦ , max bin = 23

E: x,y= (398,416) , angle = 40◦ , max bin = 23 F: x,y= (526,200) , angle = 70◦ , max bin = 22

G: x,y= (542,191) , angle = 73◦ , max bin = 26

preferred : B

Rotation angle : [-179 , 180]

Time elapsed : 4101 ms

A and **B** are referring to same object (very similar position and angle) . **C**,**D** are referring to other similar object .**E** is the same found object in reverse angle,and **F**,**G** is the similar object in reverse angle

Sample 6

Experiment 1

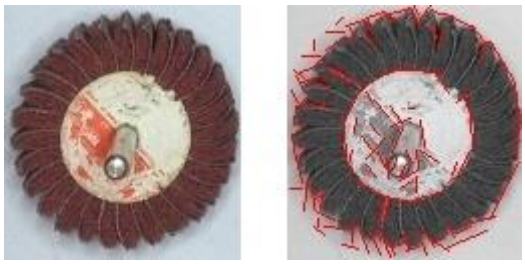


Figure 5. 40_Sample 6 ;left : original photo of sample ,right : sample after LSD

Figure 5. 41_ Original image 27,object found correctly

number of lines of sample object = 117 number of lines of main picture = 575

Good_enough-max_bin = 10

candidates for found object: A: x,y= (234,427) , angle = 0◦ , max bin = 21

preferred : A

Rotation angle : since the object is circular ,so I turned off the Rotation

Time elapsed : 1232 ms

Experiment 2

Figure 5. 42_ Original image 28,object not found correctly

number of lines of sample object = 117 number of lines of main picture = 663

Good_enough-max_bin = 10

candidates for found object: A: x,y= (485,265) , angle = 0◦ , max bin = 17

preferred : It fails to find correct object

Rotation angle : since the object is circular ,so I turned off the Rotation

Time elapsed : 2691 ms

Experiment 3

Figure 5. 43_ Original image 29,object not found correctly

number of lines of sample object = 117 number of lines of main picture = 553

Good_enough-max_bin = 10

candidates for found object: A: x,y= (420,145) , angle = 0◦ , max bin = 14

preferred : It fails to find correct object

Rotation angle : since the object is circular ,so I turned off the Rotation

Time elapsed : 855 ms

Sample 7

Experiment 1

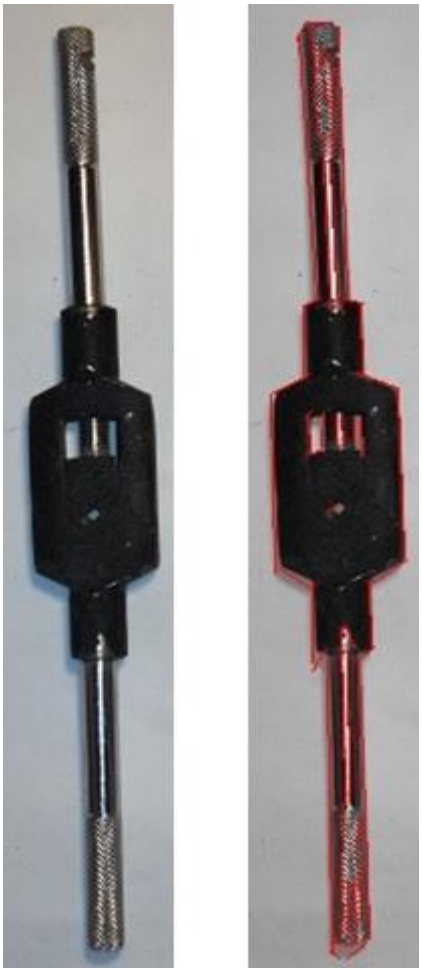


Figure 5. 44_ Sample 7 ;left : original photo of sample ,right : sample after LSD

Figure 5. 45_ Original image 30,object found correctly

number of lines of sample object = 54 number of lines of main picture = 575

Good_enough-max_bin = 20

candidates for found object:

A: x,y= (138,290) , angle = -2◦ , max bin = 25 B: x,y= (330,354) , angle = 175◦ , max bin = 25

C: x,y= (148,253) , angle = 178◦ , max bin = 32

preferred : C

A,B,C all are showing same object ,A is reverse angle

Rotation angle : [-179,180]

Time elapsed : 3574 ms

Experiment 2

Figure 5. 46_ Original image 31,object found correctly

number of lines of sample object = 54 number of lines of main picture = 506

Good_enough-max_bin = 20

candidates for found object:

A: x,y= (593,128) , angle = -80◦ , max bin = 21 B: x,y= (515,146) , angle = 97◦ , max bin = 24

C: x,y= (509,151) , angle = 100◦ , max bin = 30

preferred : C

A,B,C all are showing same object ,A is reverse angle

Rotation angle : [-179,180]

Time elapsed : 5559 ms

Experiment 3

Figure 5. 47_ Original image 32,object found correctly

number of lines of sample object = 54 number of lines of main picture = 673

Good_enough-max_bin = 21

candidates for found object:

A: x,y= (435,159) , angle = -107◦ , max bin = 22 B: x,y= (145,167) , angle = -35◦ , max bin = 22

C: x,y= (537,277) , angle = 61◦ , max bin = 28 D: x,y= (529,279) , angle = 64◦ , max bin = 22

E: x,y= (478,180) , angle = 73◦ , max bin = 23

preferred : C

Rotation angle : [-179,180]

Time elapsed : 5172 ms

Chapter 6

6 Conclusion

The work that has been done proves clearly that LSD method can be successfully used in order finding objects in images. It can be implemented in industries where the distance of camera and objects are fixed.

Experimental tests showed that, this type of object finding is fast, comparing to Generalized Hough transform, since it deals with line segments rather than edge points. It is accurate most of times but when the object has very simple shape and therefore few lines, it fails; also, it is robust to partially occlusion of object. In this work, I have considered only rotation, and not scaling, but anyway to some extent it is robust also to scaling, according to my tests. It should be noted that the sample picture and main picture should have same pixel density, to avoid having scaling.

This work leaves some interesting open research activities that could be performed in the future like, finding several objects in one image, and handling scaling in images.

References

[1] Wikipedia ,Hough Transform, https://en.wikipedia.org/wiki/Hough_transform.

[2] Wikipedia,Generalized Hough Transform, https://en.wikipedia.org/wiki/Generalised_Hough_transform.

[3] L. D. Stefano,"Computer vision " ,course slides,Object detection vol. 11,P. 36 ,2015.

[4] R. Grompone v. Gioi1 , J. Jakubowicz2 , J.M. Morel3 , G. Randall. ," LSD: a Line Segment Detector" IEEE transactions on pattern analysis and machine intelligence, VOL. 32, NO. 4, APRIL 2010

[5] A. Etemadi, "Robust Segmentation of Edge Data," Proc. Int'l Conf.

Image Processing and Its Applications, pp. 311-314, 1992.

[6] J.B. Burns, A.R. Hanson, and E.M. Riseman, "Extracting Straight

Lines," IEEE Trans. Pattern Analysis and Machine Intelligence, vol. 8, no. 4, pp. 425-455, July 1986.

[7] P. Kahn, L. Kitchen, and E.M. Riseman, "A Fast Line Finder for Vision-Guided Robot Navigation," IEEE Trans. Pattern Analysis and Machine Intelligence, vol. 12, no. 11, pp. 1098-1102, Nov. 1990.

[8] J. Matas, C. Galambos, and J. Kittler, "Robust Detection of Lines Using the Progressive Probabilistic Hough Transform," Computer Vision and Image Understanding, vol. 78, no. 1, pp. 119-137, 2000.

[9] A. Desolneux, L. Moisan, and J.M. Morel, From Gestalt Theory to Image Analysis, A Probabilistic Approach. Springer, 2008.