LSD: A Fast Line Segment Detector with a False Detection Control

Rafael Grompone von Gioi, Jérémie Jakubowicz, Jean-Michel Morel, and Gregory Randall

Abstract—We propose a linear-time line segment detector that gives accurate results, a controlled number of false detections, and requires no parameter tuning. This algorithm is tested and compared to state-of-the-art algorithms on a wide set of natural images.

Index Terms—Line segment detection, NFA, Helmholtz principle, a contrario detection.

1 Introduction

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 economic 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 [14], crack detection in materials [17], image compression [11], and satellite image indexation [19].

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 [4] followed by a Hough transform [1] 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. 1). Ignoring the orientation of the edge points, such algorithms obtain line segments with aberrant 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. 1).

Another classic method starts from edge points, chains them into curves and then cuts the chains into line segments by a straightness criterion [10]. A standard chaining method is due to Etemadi [9]. This method is parameterless and usually gives accurate results. Also, it is one of the few algorithms that simultaneously detect line segments and arcs.[1] Nevertheless, the result is not completely satisfactory, as illustrated in Fig. 1. Many detected straight and small edge curves are false positives: Here comes the fundamental threshold problem again.

Burns et al. [3] 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. [15], [16]. The line segments given by this algorithm are well localized, but the threshold problem is still there. The foliage of the tree in Fig. 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. [18], [12]. 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. 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. 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

1. The detected arcs are not shown in Fig. 1.

- *R. Grompone von Gioi is with the CMLA, ENS Cachan, CNRS, UniverSud, 61 Avenue du President Wilson, F-94230 Cachan, France, and with the IIE, Facultad de Ingenieria, Universidad de la Republica, Julio Herrera y Reissig 565, CP 11300, Uruguay. E-mail: grompone@cmla.ens-cachan.fr.*
- *J. Jakubowicz and J.-M. Morel are with the CMLA, ENS Cachan, CNRS, UniverSud, 61 Avenue du President Wilson, F-94230 Cachan, France. E-mail: {jakubowi, morel}@cmla.ens-cachan.fr.*
- *G. Randall is with the IIE, Facultad de Ingenieria, Universidad de la Republica, Julio Herrera y Reissig 565, CP 11300, Uruguay. E-mail: randall@fing.edu.uy.*

Manuscript received 19 May 2008; revised 14 Nov. 2008; accepted 1 Dec. 2008; published online 14 Dec. 2008.
Recommended for acceptance by M. Lindenbaum.
For information on obtaining reprints of this article, please send e-mail to: tpami@computer.org, and reference IEEECS Log Number TPAMI-2008-05-0294.
Digital Object Identifier no. 10.1109/TPAMI.2008.300.

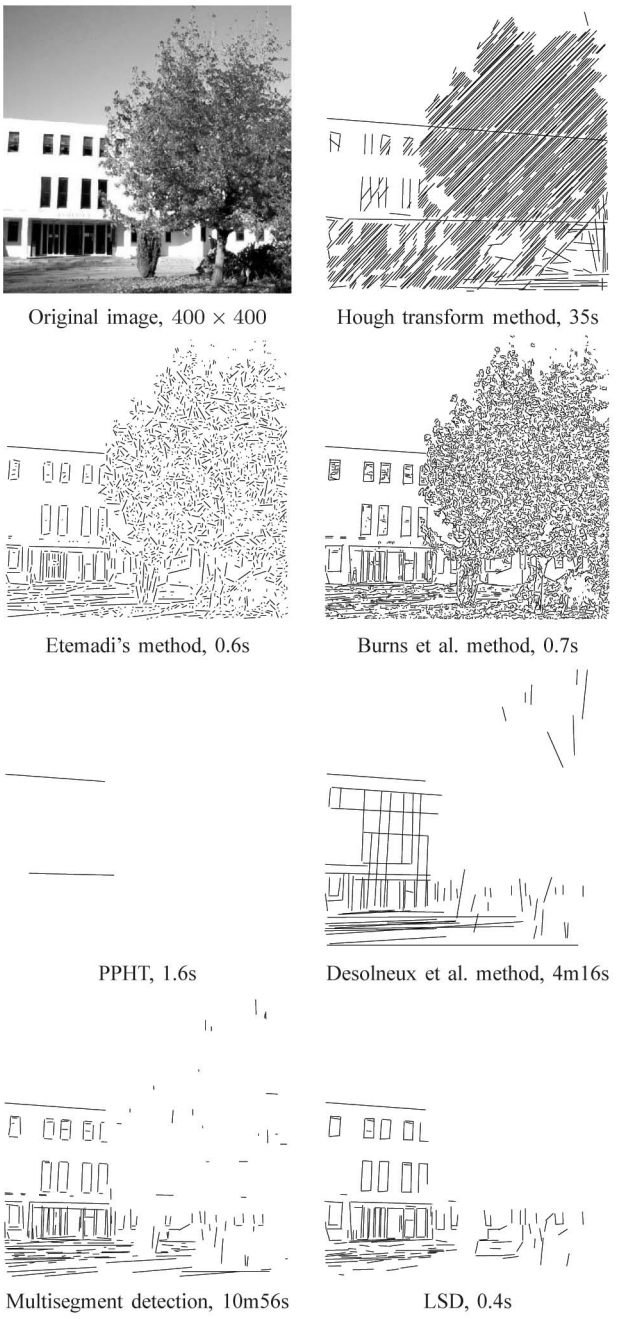

Fig. 1. A comparison of various line segment detection methods. The processing times on an Apple PowerBook G4 1.5 GHz are indicated. The Hough transform, Etemadi, and Burns et al. methods detect many irrelevant small line segments on the tree. PPHT produces no false detection but fails to detect small 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.

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, as Figs. 10 and 11 show. 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 [6], [8]. 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 [8], 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 [7], [8]. 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 [7], [8] 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 [21], [22]. 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. 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 an $O(N^4)$ complexity, where N is the image perimeter. The multisegment has an $O(N^5)$ 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. 1).

Sections 2 and 3 present an improved version of the Burns et al. algorithm that provides the line segment candidates. Section 4 describes the validation criterion. Section 5 gives a global picture of the algorithm and discusses its most

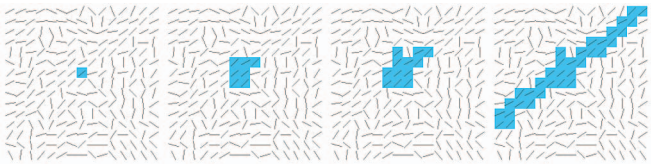

Fig. 2. Growing process of a region of aligned points. The level-line orientation field (orthogonal to the gradient orientation field) is represented by dashes. Marked pixels are the ones forming the region. From left to right: first, second, and third iterations, and final result.

important properties. Section 6 comments on the experimental results, and Section 7 concludes the paper.

2 Line-Support Regions

In contrast to classic edge detectors, the Burns et al. method defines a line segment as an image region, the *line-support region*, namely a straight region whose points share roughly the same image gradient angle. Such line segments are roughly oriented along the average level-line direction. The Burns et al. algorithm extracts line segments in three steps:

1. Partition the image into *line-support regions* by grouping connected pixels that share the same gradient angle up to a certain tolerance.
2. Find the line segment that best approximates each line-support region.
3. Validate or not each line segment based on the information in the line-support region.

Considering that this method was a real breakthrough, the proposed algorithm shares the core ideas of steps 1 and 2, with some improvements. Step 3 is, however, completely different and based on the Desolneux et al. *a contrario* method. A new version of step 1 is described in this section. Steps 2 and 3 will be described in the following two sections.

Our version of step 1 is a region growing algorithm. Fig. 2 illustrates the procedure. Each region starts with just one pixel and the *region angle* set to the level-line angle at that pixel (orthogonal to the gradient angle). Then, the pixels adjacent[2] to the region are tested; the ones with level-line orientation equal to the region angle up to a certain precision[3] are added to the region. At each iteration, the region angle is updated to a pseudomean level-line orientation of the region's pixels, defined by

$$\arctan\left(\frac{\sum_i \sin(\text{ang}_i)}{\sum_i \cos(\text{ang}_i)}\right).$$

The process is repeated until no new point can be added. The pseudocode Algorithm 1 gives more details. Seed pixels with larger gradient magnitude are tested first[4] as they are more likely to belong to straight edges. When a pixel is added to a region, it is marked and never visited again. This key property makes the algorithm greedy and therefore linear.

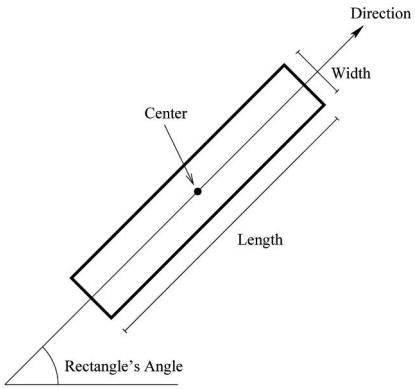

Fig. 3. Line segments are characterized by a rectangle determined by its center point, angle, length, and width.

Algorithm 1. REGIONGROW

input: An image I; a starting pixel (x, y); an angle tolerance τ; an image $Status$ where pixels used by other regions are marked.

output: A list $Region$ of pixels.

$Region \leftarrow (x, y)$;
$\theta_{region} \leftarrow \text{LevelLineAngle}(x, y)$;
$S_x \leftarrow \cos(\theta_{region})$;
$S_y \leftarrow \sin(\theta_{region})$;
foreach *pixel P in Region* **do**
  foreach $\bar{P}$ *neighbor of* P *and* $Status(\bar{P}) \neq Used$ **do**
    if $Diff(LevelLineAngle(\bar{P}),\ \theta_{region}) < \tau$ **then**
      Add $\bar{P}$ to $Region$;
      $\text{Status}(\bar{P}) \leftarrow$ Used;
      $S_x \leftarrow S_x + \cos(\text{LevelLineAngle}(\bar{P}))$;
      $S_y \leftarrow S_y + \sin(\text{LevelLineAngle}(\bar{P}))$;
      $\theta_{region} \leftarrow \arctan(S_y/S_x)$;
    end
  end
end

Whenever a large and well contrasted straight edge is present in the image, the algorithm usually finds the same line-support region, whatever the starting point. In contrast, the result may depend on the starting point when a nonstraight curve is being approximated by line segments (for example, when a circle is present in the image). In this case, the obtained decomposition of a circle into line segments would be as good as any other obtained from a different seed point. The fact that connected regions with common orientation would almost always coincide with straight edges is a surprising empirical discovery due to Burns et al. We mentioned the case of curves as a harmless first counterexample. Since the antiquity, smooth curves have been handled as concatenations of (small) straight segments.

3 Rectangular Approximation of Regions

Prior to the validation step, the line-support region (a set of pixels) must be associated with a line segment (actually, a rectangle). A line segment is determined by its endpoints

2. We use 8-connected pixel neighborhood.
3. We use a 22.5 degree tolerance. The relevance of this parameter and how to set it will be commented in Section 5.
4. Sorting algorithms have a typical $O(N \log N)$ complexity. Ordering the pixels by strict gradient magnitude value would be computationally too expensive. Instead, a pseudo-ordering is done: Pixels are classified into a finite number of bins according to their gradient magnitude value, then the pixels from higher bins are visited first and pixels in the lower bins later.

Fig. 4. Example of a rectangular approximation of a line-support region. Left: Image. Middle: One of the line-support regions. Right: Rectangular approximation superposed to the line-support region.

and its width or, equivalently, its center, angle, length, and width. Its rectangular approximation as shown in Fig. 3 includes all of these parameters.

The first idea that comes to mind is to use the mean level-line angle as the main direction for the line segment. This procedure can lead to an erroneous line angle estimation when the background shows a slow intensity variation, see [20]. In LSD (the idea was first proposed by Kahn et al. in [15], [16]), the center of mass is used to select the center of the rectangle, and the first inertia axis to select the rectangle orientation. The gradient magnitude is used as the pixel's mass. Points with large gradient norm correspond better to the observed edges. Then, the length and the width are chosen in such a way as to cover the line-support region. Fig. 4 shows an example of the result.

4 Line Segment Validation

The two key points of the Desolneux et al. [6] approach are the use of gradient orientation and a new framework to deal with parameter setting. Their method is illustrated in Fig. 5. The gradient of the input image is computed and only the level-line orientation is kept. In Fig. 5, this information is codified in the angle of the dashes. Given a line segment, the algorithm counts the number of *aligned points*, i.e., points having the level-line orientation equal to the line segment angle up to a certain tolerance τ. All potential line segments on the image must be tested; those that satisfy a threshold criterion based on their length l and their number of aligned points k are kept as valid detections.

On natural images, the gray-level transition corresponding to edges can be many pixels thick. This happens, for

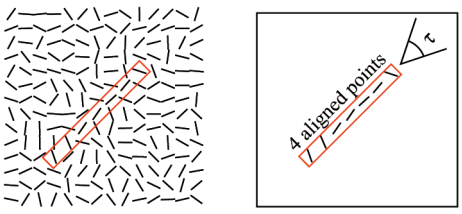

Fig. 5. Left: One line segment shown over the level-line orientation field (orthogonal to the gradient orientation field). Right: The number of aligned points up to an angular tolerance τ is counted for each line segment. The line segment shown has four aligned points among seven.

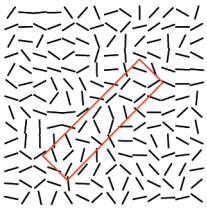

Fig. 6. One rectangle shown over the level-line orientation field (orthogonal to the gradient orientation field). The rectangle has nine aligned points out of 20.

example, with the straight boundary of an out of focus object. In that case, the Desolneux et al. algorithm gives many parallel detections and it requires a lot of effort to go back to a correct interpretation [8]. To deal with this problem, *rectangles* (line segments with a certain width) will be used instead of line segments. Fig. 6 illustrates the concept.

In the Desolneux et al. *a contrario* approach, detection is treated as a simplified hypothesis testing problem. Indeed, in the classic decision framework, two probabilistic models are required: one for the background and one for the objects to be detected. In the *a contrario* approach, the objects are directly detected as outliers of the background model. In addition, the background model is reduced to the simplest of all, namely white noise. As Desolneux et al. showed, a suitable background model is just one in which all gradient angles are independent and uniformly distributed. They showed that this is the case for a Gaussian white noise image. More formally, an image X under the background model H_0 is a random image (defined on the grid $\Gamma = [1, N] \times [1, M] \subset \mathbb{Z}^2$) such that

1. $\forall m \in \Gamma, \text{Angle}(\nabla X(m))$ is uniformly distributed over $[0, 2\pi]$;
2. the family $\{\text{Angle}(\nabla X(m))\}_{m \in \Gamma}$ is composed of independent random variables.

This is indeed a good model for flat zones of images which usually present, due to the acquisition process, a white noise distribution. But, more importantly, this model represents well isotropic zones, while straight edges are exactly the opposite: highly anisotropic zones. Thus, in practice, a set of pixels will not be accepted as a line segment if it could have been formed by an isotropic process. A good example is shown in Fig. 1. The foliage of the tree is far from being a white noise process, but it is an isotropic structure; as a consequence, no line segment is detected there.

There are as many statistical tests T_r to be performed as potential rectangles r in the image. Each test relies on the statistics $k(r, x)$ which is the number of aligned points in the rectangle r and image x. The detection step is as follows: Reject H_0 if $k(r, x) \geq k_r$, accept H_0 otherwise. For this test, non-H_0 will also be denoted by H_r, i.e., rectangle detection. Thus, we are led to the question of fixing a threshold k_r for each rectangle r. Following Desolneux et al., k_r must be fixed in a way that guarantees a control of the expected number of false alarms under H_0. We define the Number of False Alarms of a rectangle $r \in \mathcal{R}$ and an image x, as

$$\text{NFA}(r, x) = \#\mathcal{R} \cdot \mathbb{P}_{H_0}[k(r, X) \geq k(r, x)],$$

where x is the observed image, X is a random image under H_0, and $\#\mathcal{R}$ is the number of potential rectangles in the image. The smaller the $\mathrm{NFA}(r,x)$, the more meaningful r is, i.e., the less likely it is to appear in an image drawn under the H_0 model. Rejecting H_0 if and only if $\mathrm{NFA}(r,x) \leq \varepsilon$ gives what we call *ε-meaningful rectangles*.

The method is justified by the following proposition:

Proposition.

$$\mathbb{E}_{H_0} \sum_{r \in \mathcal{R}} \mathbb{1}_{\mathrm{NFA}(r,X) \leq \varepsilon} \leq \varepsilon.$$

In other terms, the expected number of detections under the background model is less than ε. Thus, in agreement with the Helmholtz principle, only a very few detections (ε) could have arisen just by chance.

In practice, only the line-support regions found by step 1 of the algorithm are tested as candidates for a line segment detection. However, this does not imply that one can set $\#\mathcal{R}$ equal to the number of line-support regions effectively tested. These line-support regions have complex statistics, different from white noise statistics. Indeed, each one of them is selected as the bounding box of a connected component of pixels sharing the same direction. Let us call $\mathcal{B}(r)$ the event that r is such a bounding box. Computing an NFA adapted to the effectively tested bounding boxes would be possible only if we knew how to compute the conditional probability $\mathbb{P}_{H_0}[k(r,X) \geq k(r,x) \mid \mathcal{B}(r)]$. This probability is intuitively much higher than $\mathbb{P}_{H_0}[k(r,X) \geq k(r,x)]$. In the absence of a closed formula for this complex probability, we must be content with the estimate given by the proposition stated above, counting the number of potential rectangles in the image.

Considering a one pixel precision, there are N^4 potential oriented line segments in an $N \times N$ image (starting and ending on a point of the grid Γ). If we set to N the number of possible width values for the rectangle (an overestimation), the number of potential tests goes up to $\#\mathcal{R} = N^5$. This estimation is rough, the exact number depending on the exact precision considered. What is important here is the order of magnitude, that allows the thresholds to adapt to different image sizes. See [8] for further discussion on the number of tests.

If the angle tolerance τ is set to $\tau = \pi p$, the probability (under H_0) that a given point has its level-line aligned with a rectangle is p. Since the gradient is independent at different image points under H_0, $k(r)$ follows a binomial law with parameters $n(r)$ and p, where $n(r)$ is the total number of points in the rectangle. Using the number of tests estimated before, we can write

$$\mathrm{NFA}(r) = N^5 \cdot b\big(n(r), k(r), p\big),$$

where $b(n,k,p) = \sum_{i=k}^{n} \binom{n}{i} p^i (1-p)^{n-i}$ stands for the binomial tail.

The dependence of the method on ε is very weak (actually logarithmic), see [6]. Thus, as advised by Desolneux et al., one can fix $\varepsilon = 1$ once for all. This corresponds to accepting, on average, one false positive detection per image on the nonstructured model. It also guarantees that all discarded line segments are indeed likely to appear in noise (no false negatives in this sense).

5 The Complete LSD Algorithm

5.1 Details of the Algorithm

Algorithm 2 shows a pseudocode for the complete algorithm. The subroutine **Grad** computes the image gradient and gives three outputs: the level-line angles, the gradient magnitude, and an ordered list of pixels. The parameter ρ is a threshold: Points with gradient magnitude smaller that ρ are discarded.[5] This parameter will be discussed later in Section 5.2. To construct the list, the pixels are classified into bins according to their gradient magnitude; the list starts with the pixels belonging to the bin with high gradient and ends with the pixels belonging to the bin with low gradient. The list is roughly ordered in decreasing gradient magnitude order.

Algorithm 2. LSD: Line Segment Detector

input: An image I, parameters ρ, τ and ε.
output: A list *out* of rectangles.

```
(LLAngles, GradMod, OrderedListPixels) ← Grad(I, ρ);
Status(allpixels) ← NotUsed;
foreach pixel P in OrderedListPixels do
    if Status(P) = NotUsed then
        region ← RegionGrow(P, τ, Status);
        rect ← RectApprox(region);
        nfa ← NFA(rect);
        nfa ← ImproveRect(rect);
        if nfa < ε then
            Add rect to out;
            Status(region) ← Used;
        else
            Status(region) ← NotIni;
        end
    end
end
```

The list of pixels is used to give priority to pixels as seeds in the search of line-support regions. The first pixels in the list are the ones with higher gradient magnitude because they are more likely to belong to edges. Starting from this pixel, the algorithm described in Section 2, **RegionGrow**, is used to obtain a line-support region. Then, the algorithm described in Section 3, **RectApprox**, gives a rectangle approximation of the region, and the method described in Section 4 computes the line segment's NFA.

In our context, the best rectangular approximation of a line-support region is the one that gives the smaller NFA value. The routine **ImproveRect** tries several perturbations to the initial approximation in order to get a better approximation. This step is not significant for large and well contrasted line segments, but it extends the detection limit for small and noisy ones. The tested perturbations are variations in width and lateral position.[6] The justification is that the width of the line segments is the worst estimated parameter on the first rectangular approximation, but also a very influential one; an error that makes the rectangle one pixel thicker adds a large number of nonaligned points, as

5. For the validation, these points are considered as *not* being part of an edge, thus as *nonaligned*.

6. In our current implementation, five half-pixel variations in width are tested and five quarter-pixel lateral variations to each side are tested.

many as the length of the line segment. This can increase the NFA value, rising the nondetection risk.

The first computation of the NFA value is done using the probability p (that one point is aligned by chance) equal to $p = \frac{\tau}{\pi}$, where τ is the tolerance used to get the line-support region. But, the observed level-line angles could be much more precise, resulting in a better NFA value. Starting from $p = \frac{\tau}{\pi}$, **ImproveRect** also tries dyadic precision levels, covering the practical range of p values.[7] With a finer precision some points may stop from being aligned, so $k(r)$ may diminish; the balance between a smaller p and a smaller $k(r)$ can increase or decrease the NFA value depending on the case. The best one is kept. Notice that this loop eliminates p as a method parameter.

Status is used to keep track of pixels used or tested by line-support regions, as explained in Section 2. However, a third possible state is added: **NotIni**, which is assigned to pixels in a line-support region that did not qualified as a detection. This will prevent them from being used as seed points for new regions but will allow to use them if a neighbor region grows into them. In some cases, this helps improving the search, see [20].

5.2 Internal Parameters

There are three parameters involved in LSD: ρ, τ, and ε.

ρ is a threshold on the gradient magnitude: Pixels with small gradient are not considered. The reason is to cope with the quantization of the image intensity values. Desolneux et al. [5] showed that gray-level quantification produces errors in the gradient orientation angle. This error is negligible when gradient magnitude is large, but can be dominant for a small gradient magnitude. It can create patterns and therefore spurious detections. A side effect of the threshold ρ is a speed improvement by reducing the number of considered pixels.

The criterion we use to set ρ is to leave out points where the angle error is larger than the angle tolerance. Let u denote the image and $\tilde{u}$ the quantized one. Then, $\tilde{u} = u + n$ and

$$\nabla\tilde{u} = \nabla u + \nabla n,$$

where n is the quantization noise. In [5], it is shown that the error of the image gradient angle can be bound by

$$|\text{angle error}| \leq \arcsin\left(\frac{q}{|\nabla u|}\right),$$

where q is a bound to $|\nabla n|$, see Fig. 7. Imposing that $|\text{angle error}| \leq \tau$ one obtains

$$\rho = \frac{q}{\sin\tau}.$$

For example, on images quantized to gray values in $0, 1, 2, \ldots, 255$, $|\nabla n|$ can be bounded by $q = 2$ (the worst case is when adjacent points gets errors of $+1$ and -1). If τ is set to 22.5 degrees (which corresponds to eight different angle bins) the threshold to the gradient magnitude is $\rho = 5.2$.

τ is the angle tolerance used in the search for line-support regions. A small value is more restrictive, leading to an over-partition of line segments. A large value results in large regions and in the merging of unrelated ones. When a well contrasted and large enough line segment is present in the image, the line-support region obtained depends little on this parameter (and the rectangle approximation even less, as a result of the use of the gradient magnitude weighting). In critical size regions or when much noise is present, this parameter is more important.

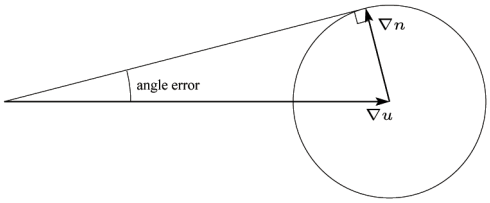

Fig. 7. Relation between image noise and gradient angle error.

Burns et al. [3] proposed using a 22.5 degree angle that corresponds to eight different angle bins. Independently, we arrived at the same value as the one that gives the best results. There is no theory behind this parameter value, but it is supported by the results on thousands of images.

Note that τ is also used to set the first angle tolerance used in the NFA computation. But, all of the practical range of values of p are also tested. As a result, τ has no influence in the validation step nor in the theoretic definition of line segments.

Finally, as already stated in Section 4, the detection parameter ε is not a critical one; one can safely set its value to 1 once for all. This means that, on average, one casual detection per image is acceptable.

All in all, LSD can be used without parameter tuning, and all of the above parameters can be considered as internal parameters. This fact is supported by tests on thousands of images of very different kinds and origins.

5.3 Complexity of the Algorithm

The computation of the gradient magnitude and angle is proportional to the number of pixels. Pixels are pseudo-ordered by a classification into bins, operation that can be done in linear time. The computational time of the line-support region finding algorithm is proportional to the number of visited pixels. If there were no overlap between regions, this number would be equal to the total number of pixels in the regions plus the border pixels of each one. The **NotIni** condition allows for some overlap between regions. This overlap is equivalent to having thicker frontiers between regions. Thus, the number of visited pixels remains proportional to the total number of pixels of the image. The rest of the processing can be divided into two kinds of tasks. The first kind, e.g., summing the region mass or counting aligned points, are proportional to the total number of pixels involved in all regions. The second kind, e.g., computing inertia moments or computing the NFA value from the number of aligned points, are proportional to the number of regions. Both the total number of pixels involved and the number of regions are at most equal to the number of pixels. All in all, LSD has an execution time proportional to the number of pixels in the image.

7. Five dyadic precision steps are considered before adjusting the rectangle width, and again, five dyadic precision steps afterward. Thus, precisions from $p = \frac{1}{8}$ to $p = \frac{1}{8192}$ are tested. In practice, a precision with $p \leq \frac{1}{256}$ is rarely obtained.

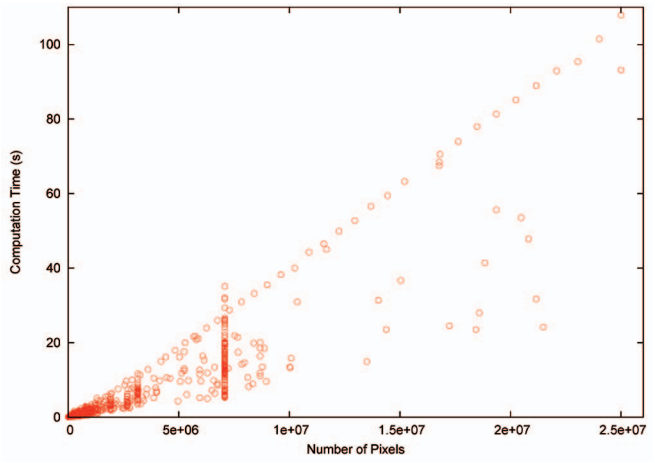

Fig. 8. Computation time (in seconds) required by LSD for the processing of 1,297 images as a function of the image size (total number of pixels). The computations were done with LSD implemented on Megawave2 version 2.31a, running on an Apple PowerBook G4 1.5 GHz. The slanted linear series corresponds to Gaussian white noise images. The vertical linear series corresponds to images of size $3{,}072 \times 2{,}304$, a popular camera resolution today.

6 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 of the experiments were done without tuning any parameter at all. See [20] for a deeper analysis of the experiments. More results, including some videos, a demo version, and a source code for the algorithm, can be found at http://iie.fing.edu.uy/~jirafa/lsd. 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. 9 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

Fig. 9. Result of LSD on natural images.

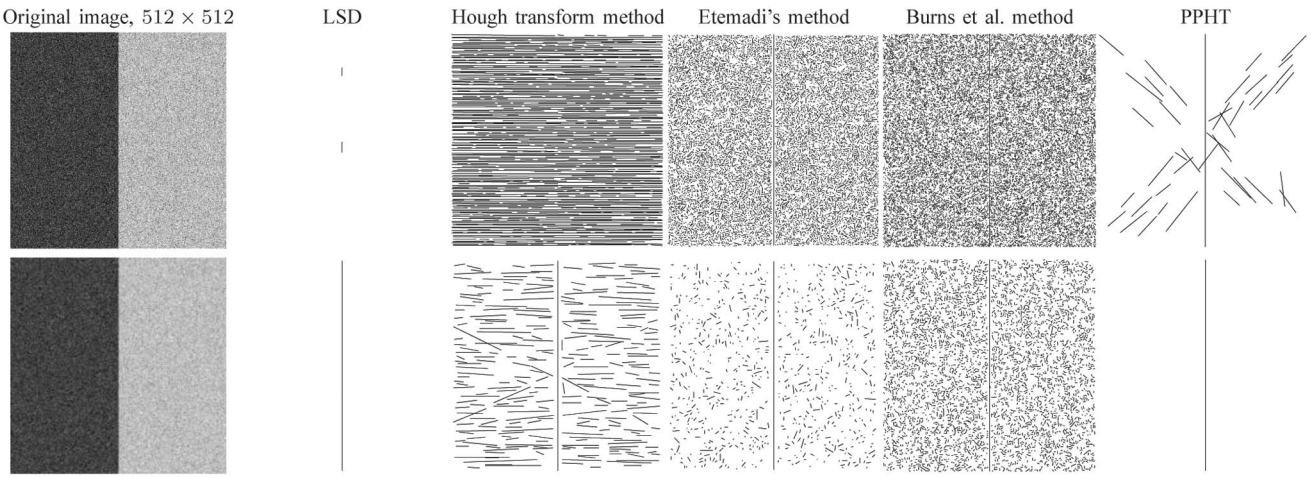

Fig. 10. Analysis of a noisy edge image at two different scales by five line segment detection algorithms. When a lot of noise is present (here Gaussian noise with $\sigma = 50$), LSD will not detect some of the line segments that we can see; it will not produce false detections either. The methods without a false detection control produce many false detections, making the result useless. PPHT false detection control failed in this case. PPHT and Etemadi's method did, indeed, detect the edge at full resolution; this is because the detection relies on Canny edge points, that involve a Gaussian filtering. A standard way to cope with noise is by Gaussian subsampling. When LSD processes the half resolution image (second row) it succeeds in detecting the edge; the Hough transform method, Etemadi's method, and the Burns et al. method still produce false detections.

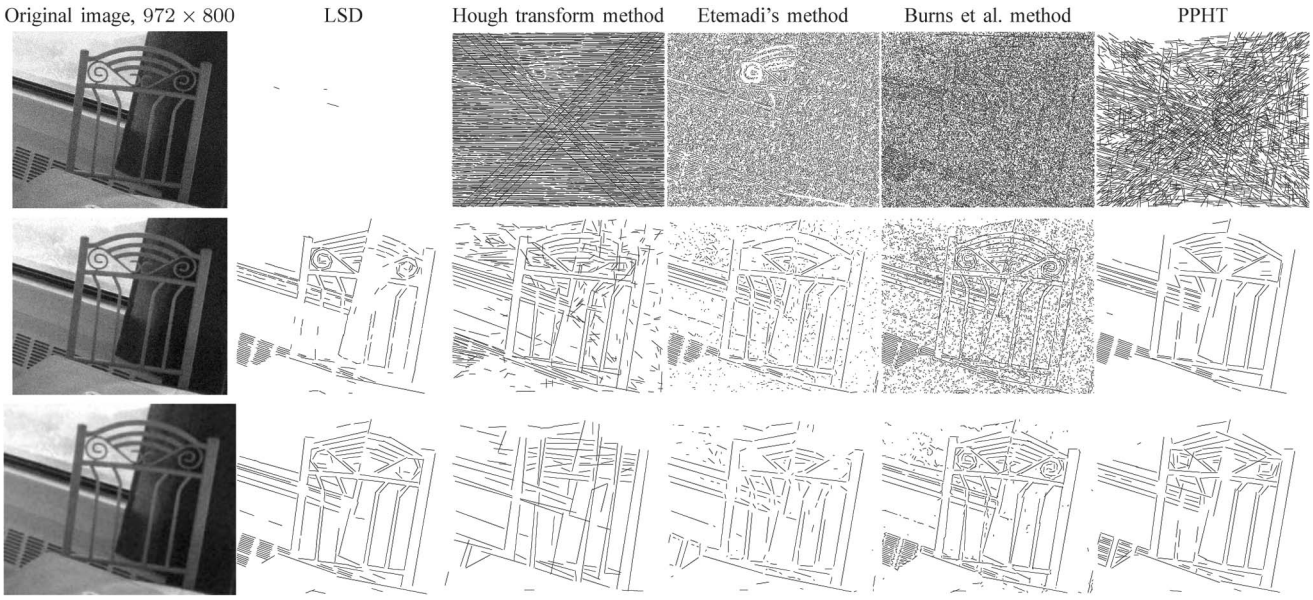

Fig. 11. The same effect shown in Fig. 10 can be seen on a noisy natural image. The three rows correspond to the analysis at full scale, 1/2 resolution and 1/4 resolution by Gaussian filtering. At full resolution, LSD fails to detect the structure of the image, but produces no false detection. Algorithms without a false detection control, like the Hough transform method, Etemadi's method, and Burns et al.'s method produce useless results. PPHT false detection control fails in this image and produces many false detections. The structure of the image can't be obtained at full resolution when noise dominates. But the image can be analyzed at a different scale (as probably do human vision). Most of the structure is detected by LSD at half resolution. Note that the PPHT method and Etemadi's method detect part of the structure of the image at full resolution; the reason is that both approaches use Canny points, that involve a Gaussian filtering.

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.

The presence of noise can deteriorate the performance of LSD. Helmholtz principle states that no detection should be made on white noise images. As a result, 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

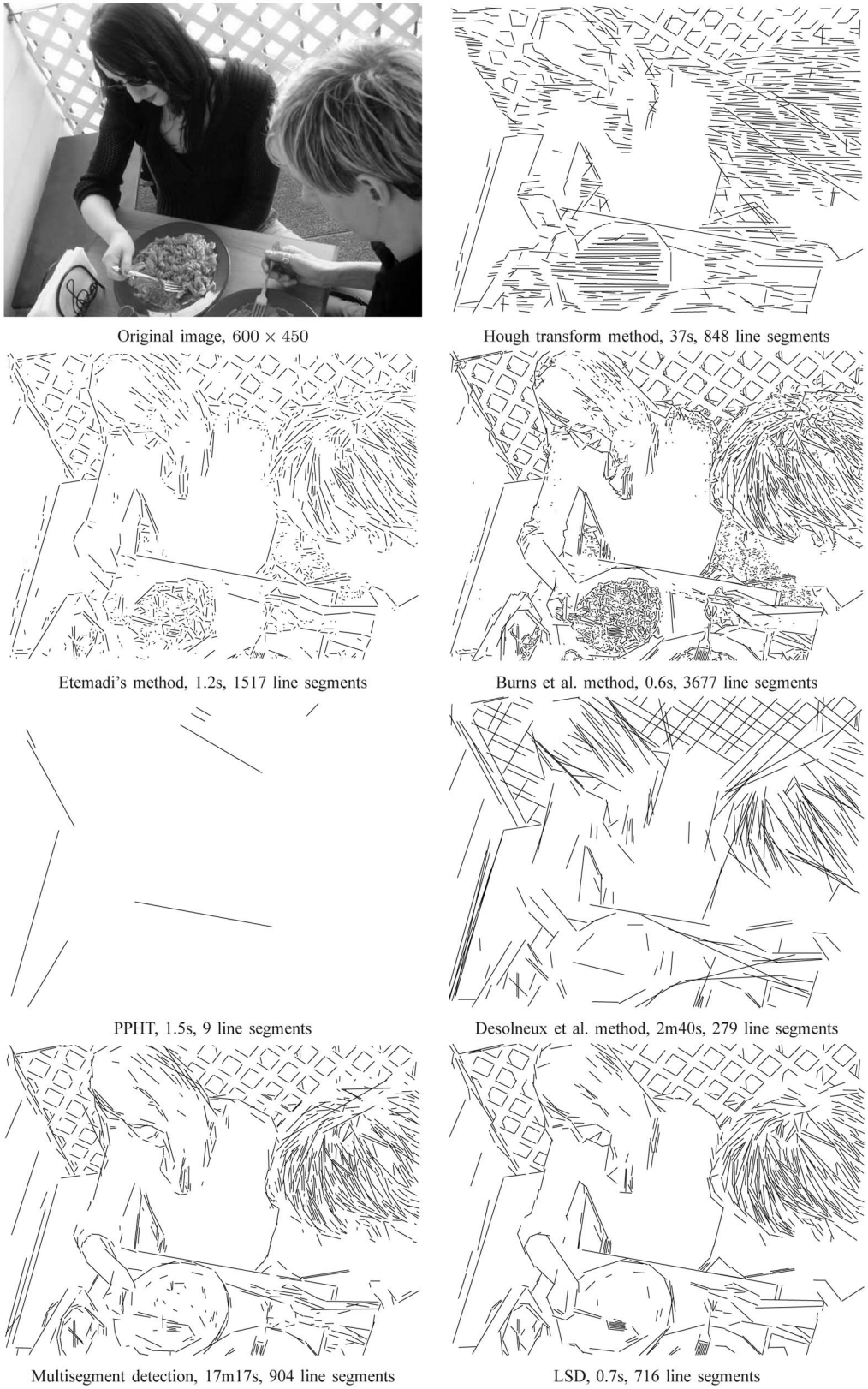

Fig. 12. A comparison of various line segment detection methods. The processing times on an Apple PowerBook G4 1.5 GHz are indicated. Etemadi's method detects line segments and arcs at the same time; here only line segments are shown. Hough and Etemadi have many wrong detections. Burns et al.'s method is accurate but has an overwhelming number of false detections. PPHT produces no false detection but fails to detect many line segments. Desolneux et al.'s method has no false detection but is too global. Multisegments and LSD give similar results in very dissimilar times.

moderate noise power, the variations to the level-line angles become larger and the angle difference can reach the angle tolerance value τ. 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. 10 (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. 10. The same effect happens on real images, see Fig. 11.

A criticism can be raised to the theory: Almost no line segment is detected in the images of Figs. 10 and 11 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. 10 (bottom) and 11 (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. See [20]. 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. 12 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 Xhoughtool [13]; Etemadi's algorithm in its original implementation ORT-2.3; Burns et al.'s algorithm, using an implementation by Ross Beveridge [2] (personal communication); PPHT as implemented in the freely available RAVL libraries; Desolneux et al.'s algorithm, as implemented in the module `align_mdl` included in the freely available image processing framework MegaWave2 http://megawave.cmla.ens-cachan.fr; the Multisegment detector; finally the implementation of LSD, available at http://iie.fing.edu.uy/~jirafa/lsd.

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. 12. Also, when slow gradients are present, like in the shadow of the table on the left of the image, this produces multiple parallel

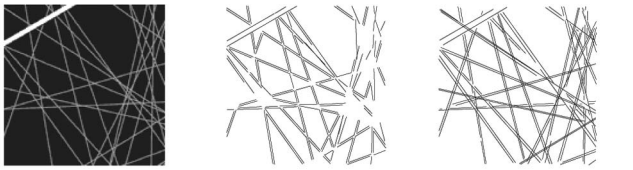

Fig. 13. A case where the multisegment detector gives a better result than LSD. Some line segments are too small to be meaningful for LSD. This means that their line support region could have arisen in noise. The global analysis performed on each line allows the multisegment detector to obtain the right line interpretation.

detections instead of one wider line segment. Multisegment detection produces good results (even if the parallel detections problem is still present and in some cases hallucinates global aligned structures not present, see [20]). 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 multisegment detector and LSD are similar in most cases. Let us comment briefly on the main differences. The multisegment 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. 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 multisegment. Being much more local, LSD does not present this problem. But, the global analysis of the multisegment detector is needed, however, to grasp the right interpretation on other cases, as Fig. 13 shows. In this image, LSD cannot detect small line segments that are too short, while the multisegment algorithm succeeds in reconstructing the global structure.

7 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.

ACKNOWLEDGMENTS

The authors are indebted to their collaborators for many remarks and corrections, and more particularly to Andrés Almansa, Gabriele Facciolo, Enric Meinhardt-Llopis, and Pablo Musé. They wish to thank J. Ross Beveridge for providing the software for the Burns et al. algorithm. The authors also wish to extend thanks to the editor and the anonymous reviewers for their valuable suggestions and comments. The research was partially financed by the ALFA project CVFA II-0366-FA, the Centre National d'Etudes Spatiales, and the US Office of Naval Research under grant N00014-97-1-0839.

REFERENCES

[1] D.H. Ballard, "Generalizing the Hough Transform to Detect Arbitrary Shapes," *Pattern Recognition,* vol. 13, no. 2, pp. 111-122, 1981.
[2] J. Ross Beveridge, C. Graves, and C. Lesher, "Some Lessons Learned from Coding the Burns Line Extraction Algorithm in the Darpa Image Understanding Environment," Technical Report CS-96-125, Computer Science Dept., Colorado State Univ., 1996.
[3] 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.
[4] J. Canny, "A Computational Approach to Edge Detection," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 8, no. 6, pp. 679-698, Nov. 1986.
[5] A. Desolneux, S. Ladjal, L. Moisan, and J.M. Morel, "Dequantizing Image Orientation," *IEEE Trans. Image Processing,* vol. 11, no. 10, pp. 1129-1140, Oct. 2002.
[6] A. Desolneux, L. Moisan, and J.M. Morel, "Meaningful Alignments," *Int'l J. Computer Vision,* vol. 40, no. 1, pp. 7-23, 2000.
[7] A. Desolneux, L. Moisan, and J.M. Morel, "Computational Gestalts and Perception Thresholds," *J. Physiology-Paris,* vol. 97, pp. 311-324, 2003.
[8] A. Desolneux, L. Moisan, and J.M. Morel, *From Gestalt Theory to Image Analysis, A Probabilistic Approach.* Springer, 2008.
[9] A. Etemadi, "Robust Segmentation of Edge Data," *Proc. Int'l Conf. Image Processing and Its Applications,* pp. 311-314, 1992.
[10] O. Faugeras, R. Deriche, H. Mathieu, N.J. Ayache, and G. Randall, "The Depth and Motion Analysis Machine," *Int'l J. Pattern Recognition and Artificial Intelligence,* vol. 6, pp. 353-385, 1992.
[11] P. Fränti, E.I. Ageenko, H. Kälviäinen, and S. Kukkonen, "Compression of Line Drawing Images Using Hough Transform for Exploiting Global Dependencies," *Proc. Joint Conf. Information Sciences,* 1998.
[12] C. Galambos, J. Kittler, and J. Matas, "Gradient Based Progressive Probabilistic Hough Transform," *IEE Proc. Vision, Image and Signal Processing,* vol. 148, no. 3, pp. 158-165, June 2001.
[13] H. Kälviäinen, P. Hirvonen, and E. Oja, "Houghtool—A Software Package for the Use of the Hough Transform," *Pattern Recognition Letters,* vol. 17, no. 8, pp. 889-897, 1996.
[14] C.X. Ji and Z.P. Zhang, "Stereo Match Based on Linear Feature," *Proc. Int'l Conf. Pattern Recognition,* pp. 875-878, 1988.
[15] P. Kahn, L. Kitchen, and E.M. Riseman, "Real-Time Feature Extraction: A Fast Line Finder for Vision-Guided Robot Navigation," Technical Report 87-57, COINS, 1987.
[16] 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.
[17] S. Mahadevan and D.P. Casasent, "Detection of Triple Junction Parameters in Microscope Images," *Proc. SPIE,* pp. 204-214, 2001.
[18] 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.
[19] R. Grompone von Gioi and J. Jakubowicz, "Geometry-Based Unsupervised Urban-Area Detection," manuscript in preparation, 2007.
[20] R. Grompone von Gioi, J. Jakubowicz, J.M. Morel, and G. Randall, "LSD: A Line Segment Detector," technical report, Centre de Mathématiques et de leurs Applications (CMLA), Ecole Normale Supérieure de Cachan (ENS-CACHAN), 2008.
[21] R. Grompone von Gioi, J. Jakubowicz, J.M. Morel, and G. Randall, "On Straight Line Segment Detection," *J. Math. Imaging and Vision,* vol. 32, no. 3, pp. 313-347, Nov. 2008.
[22] R. Grompone von Gioi, J. Jakubowicz, and G. Randall, "Multi-segment Detection," *Proc. IEEE Int'l Conf. Image Processing,* 2007.

Rafael Grompone von Gioi received the BSc degree from the Universidad de la República, Uruguay, in 2004 and the MSc degree from the Ecole Normale Supérieure de Cachan, France, in 2006. He is currently a PhD student at the Ecole Normale Supérieur de Cachan.

Jérémie Jakubowicz received the MS degree in image processing (2004) and the PhD degree in image processing (2007) from the Ecole Normale Supérieure de Cachan, France. He is currently a postdoctoral researcher at the Centre National des Etudes Spatiales, France.

Jean-Michel Morel received the PhD degree in 1980. He has been a professor of applied mathematics at the Ecole Normale Supérieure de Cachan since 1997. Since 1990 his research has been focused on the mathematical analysis of image analysis and processing. He is co-author of four books on this subject: *Variational Methods in Image Segmentation* (Birkhäuser 1994); *Contrast Invariant Image Analysis and PDE's* (SIAM, to appear); *From Gestalt Theory to Image Analysis: A Probabilistic Approach* (Springer, 2008); and *A Theory of Shape Identification* (Springer, 2008).

Gregory Randall received the electrical engineering degree in 1983 from ISPJAE, Cuba, the DEA degree in robotics from the University of Paris VI, and the PhD degree in applied informatics from the University of Paris XI in 1991. He is a full professor in the Electrical Engineering Department at the Universidad de la República (UR), Uruguay. From 1988 to 1991, he was a member of the Robotvis project at INRIA. From 1992 to 1994, he worked at COSE, Paris, and in 1994 he moved to Uruguay, where he founded the Image Processing Group. From 1998 to 2004, he was the head of the Electrical Engineering Department at UR. During 2005-2006, he was a visiting professor in the Electrical and Computer Engineering Department and an IMA general member at the University of Minnesota. His research interests are in image processing and computer vision.