On Straight Line Segment Detection

Rafael Grompone von Gioi
CMLA, ENS Cachan, France
grompone@cmla.ens-cachan.fr

Jérémie Jakubowicz
CMLA, ENS Cachan, France
jakubowi@cmla.ens-cachan.fr

Jean-Michel Morel
CMLA, ENS Cachan, France
morel@cmla.ens-cachan.fr

Gregory Randall
IIE, Universidad de la República, Uruguay
randall@fing.edu.uy

Abstract

In this paper we propose a comprehensive method for detecting straight line segments in any digital image, accurately and controlling both false positive and false negative detections. Based on Helmholtz principle, the proposed method is parameterless. At the core of the work lies a new way to interpret binary sequences in terms of unions of segments, for which a dynamic programming implementation is given. The proposed algorithm is extensively tested on synthetic and real images and compared with the state of the art.

1 Introduction

It might seem futile to address the problem of detecting segments, that is, local straight edges in an image. First of all, edge detection has been the object of intense research for more than thirty years and many more and more sophisticated algorithms have been proposed. Second, alignments often coincide with edges (most edges are locally straight or smooth as illustrated e.g. in [7]).

On the other hand, the definition of an edge has never been fixed because it depends on a highly subjective and varying measurement, namely the contrast. Segments instead, or alignments, can receive a first purely geometric definition, as intervals of a straight line in an image which are orthogonal to the image gradient at many of their points. This requirement is independent of contrast and relies only on image geometry. Thus, it is expected that segments in images will give important information about their geometric content, and so it is. Segments as features can help in several problems. To cite just a few: stereo analysis [25], classification tasks like on-board selection of relevant images [31], crack detection in materials [32], stream and roadbeds detection [45], and image compression [18]. They serve as a basic tool when trying to find sailing ships and their V-shaped wakes [27, 8], to detect filamentous structures in cryo-electron microscopy images [47], or to get rid of lines in forms in order to process them [46].

In addition, segments are one of the basic shapes in graphics. The gestalt school [26] has gone so far as to analyze human perception with drawings essentially made of straight segments. Most human made environments and in particular architectures are based on alignments. All photographs made in those environments show alignments as an essential perspective feature.

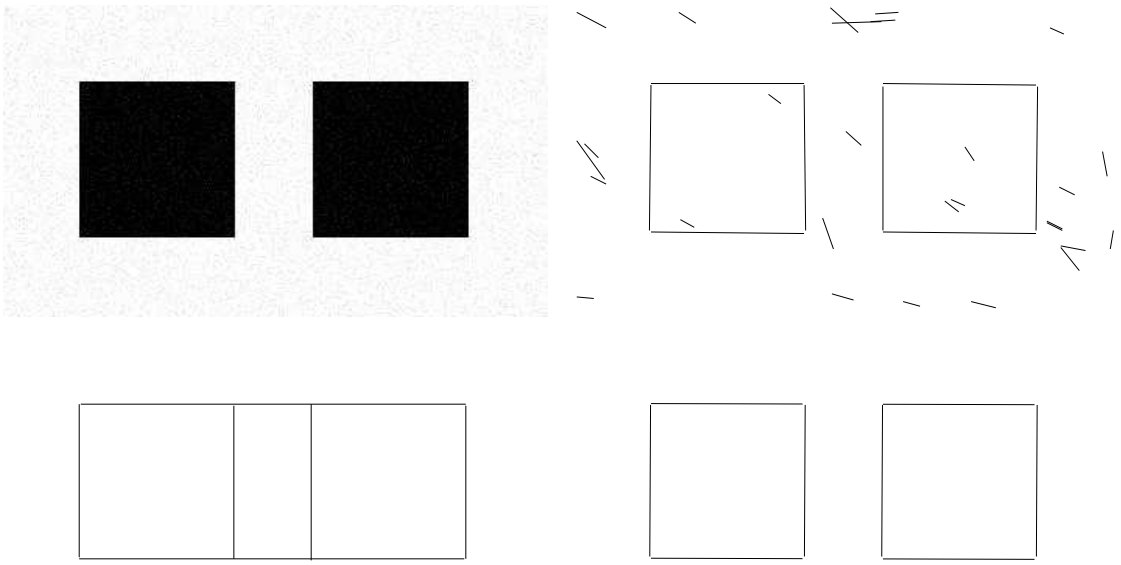

Figure 1: Why another segment detector? Up left: a synthetic image formed by black squares on a white background with Gaussian white noise added. Up right: Segment detection by Hough Transform Method (HTM). There are many false positive detections. Bottom left: result of DMM algorithm: No false positive detections but the top and the bottom of each square become one segment. Bottom right: result of our algorithm. All the experiments in this paper were run without tuning the parameters.

It would seem that such a basic problem as segment detection is a simple one and has been tackled once for ever. We hope to convince the reader that it has not. We intend to show that it is a more complex event than anticipated in the former theories. This explains why, to the best of our knowledge, all existing algorithms have serious drawbacks. Worse than that, they do not deliver a result reliable enough to found a hierarchical image analysis. Three issues have to be solved: over-detection (false positive), under-detection (false negative) and the accuracy of each detection.

There are, however, some milestones in the past attempts. The first milestone is definitely the involvement of the Hough transform followed by thresholds [24]. Given a set of dots, this transform computes quickly the frequency of dots on each digital line. Then all lines where an "excess" of dots is observed supposedly contain alignments. This method is not directly applicable to images, but it can be applied to their edge map. The detection method is coarse to fine, as opposed to the digital segment detector proposed by Faugeras and his collaborators who have investigated digital segments thoroughly and derived interesting applications [16, 11, 19]. To detect segments, they group edge maps into chains and do polygonal approximations. Such grouping is based on empirical thresholds. This approach shows the same problems as the standard Hough Transform Method which we will discuss thoroughly. As seen in the noisy image of Fig. 1, there can be many false positive detections using methods with fixed detection thresholds. A more sophisticated version of the chaining technique is due to P. Musé and F. Sur [36, 44]. It detects flat pieces of level lines. The chaining part does not rely on an *a priori* scheme, but rather adapts to the image itself since it follows its level lines. The flatness is measured by how much a given piece of level line turns with respect to its endpoints chord. This algorithm uses two thresholds and adjust them using an *a contrario* model. We will discuss its properties later as it can be considered a state of the art chaining algorithm.

Another milestone is the Desolneux *et al.* theory [14], since they definitely tackled two of the

issues raised above, namely the control of both the false positive and the false negative detections. Details will be given later but, in a few words: They guarantee with a parameterless method that *not more than one segment on the average will be a false positive*. They also guarantee that *all discarded segments could occur in white noise* and therefore cannot be accepted. Their segment detector (which from here on we will call DMM algorithm) obeys what they called the *Helmholtz principle*. Quoting from [13]: *"According to this principle, an observed geometric structure is perceptually 'meaningful' if the expectation of its occurrences (in other terms, its number of false alarms (NFA)) is small in a random image. [...] this definition leads to a parameter free method, compatible with phenomenology."* As we shall demonstrate extensively, DMM algorithm indeed never misses a line where at least a *meaningful* segment is present, but can instead misinterpret the segment organization in a line, up to the point where a segment can be missed in presence of a more meaningful one, or two distinct segments can be merged into a single one. We refer to Fig. 1 and Fig. 17 to illustrate these phenomena.

Our intention here is to generalize the DMM algorithm in such a way as to keep its two main detection properties; but also to tackle the last issue, namely the detection accuracy. We shall give mathematical arguments and experiments explaining why and how false positive, false negative and inaccurate results occur with the two milestone algorithms. We hope to demonstrate that the slightly more sophisticated method we shall propose is actually necessary, and hopefully sufficient, to solve all of the three mentioned issues.

Since we will often refer to that algorithm, let us now clarify what is meant by the Hough Transform Method (HTM) for segment detection. There are three necessary steps: the gray-level image is first processed by an edge detector to give an edge mask (a binary image where only edge points are marked). Then the standard Hough transform is applied to the mask in order to detect the lines. Finally, the segments are extracted from the detected lines using thresholds on two geometric parameters: the minimal length of a segment and the gap allowed between two consecutive segments. In a Hough Transform approach there are several parameters, three of which are critical: the threshold on the number of votes in the Hough transform going from edges to lines, and two more to go from lines to segments. HTM is described in more detail in section 3.

The Hough transform is a valid method for every parametric family of shapes. Segments form a four-parameter family. One could accumulate the votes directly into the *segment* parameter space and select the peaks. Why detecting lines and cut them into segments instead? Because the largest segments would be favored. Indeed, if the image were purely random, a large segment could receive more votes than a smaller one just because it would contain more potential voters. This is a general drawback of the Hough transform. When the parameter space is not homogeneous enough, a uniform threshold does not work well. Ideally the threshold process should be performed locally in the parameter space. In the case of segments, larger segments should have a more conservative threshold. The difficulty with the local threshold procedure is to find a good threshold function.

This was done in [14] where Desolneux *et al.* found a function to perform this thresholding which has a natural description in terms of meaningfulness. In fact, they introduced a general definition of meaningfulness based on the Helmholtz principle, "no structure is perceived in white noise images." According to their definition, the meaningfulness of a segment depends on the number of votes it gathered and its length in a nonlinear way. They showed that selecting peaks in meaningfulness and thresholding according to meaningfulness was no more favoring large segments. They also showed the link between false alarms and meaningfulness and made everything dependent on the expected number of false alarms in a white noise model. When using their algorithm, the only parameter the user has to tune is the maximum number of false alarms she/he is ready to get. It turns out that this method is parameterless because the dependency on this parameter vary very slowly (it is a $\sqrt{\log}$ dependency). In practice, setting the allowed maximum number of false alarms to 1, acts as an universal threshold. As already seen in the noisy synthetic image, the detected segments are not always accurate. Fig. 2(c) shows that this accuracy problem is not specific to synthetic images but also appears in natural

Figure 2: (a): the original image. (b): result obtained by HTM. (c): result obtained by the DMM algorithm. (d): result of the algorithm shown in this paper.

images (see around the windows).

We shall present a method that controls the number of false alarms in white noise and performs better on synthetic and natural images (see Fig. 1 and Fig. 2). The method is also based on the Helmholtz principle but the 1D binary sequence segmentation step is more sophisticated than Desolneux *et al.*'s. It is adaptive as it uses the sequence itself to automatically set its thresholds. For example, we will see that, when two consecutive segments are large, the minimal size of the gap that keeps them separate is smaller than for two smaller segments.

This paper is organized as follows: Section 2 introduces some notation and definitions. The HTM is presented in section 3. Section 4 presents the DMM algorithm and section 5 the Flat Piece Detector of Musé and Sur. The theoretical considerations of multisegment detection are presented in section 6. Its properties to segment 1D binary sequences are studied in section 7. The dynamic programming step and other implementation details are given in section 8. Section 9 presents the results of the new algorithm tested on real images. Section 10 concludes the paper.

2 Objects and Notations

A digital image is a function x from a grid $\Gamma = [1, N] \times [1, M] \subset \mathbb{Z}^2$ to $\mathbb{R}$. The symbol $\mathcal{S}$ denotes the set of all possible (oriented) segments: $\mathcal{S} = \{(a, b) \in \Gamma^2\}$. So $\#\mathcal{S} = \#\Gamma^2$, where $\#\cdot$ stands for the number of elements of a set. A segment joining two borders of the grid Γ is called a line. There are $O(M + N)^2$ lines in a $[1, N] \times [1, M]$ grid. In fact it is easy to check that there are exactly $2(M + N)^2 + 4MN - 16M - 16N + 28$ of them when $M, N > 1$.

Definition 1. A pixel $m \in \Gamma$ is aligned with an oriented segment s up to precision p, or simply p-aligned, if and only if,

$$\text{Angle}(\nabla x(m), s) \in \left]\frac{\pi}{2} - p\pi, \frac{\pi}{2} + p\pi\right[$$

where $\nabla x(m)$ stands for the gradient of image x in the pixel m. See Fig. 3.

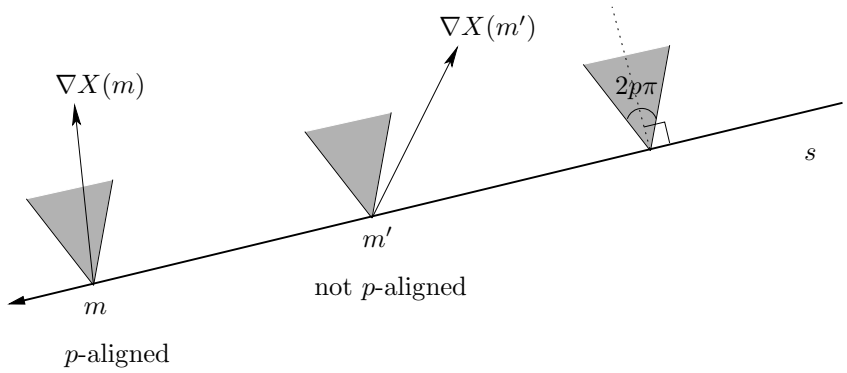

Figure 3: Pixel m is p-aligned with segment s, whereas m' is not because the image gradient in m' is out of the tolerance cone. The tolerance cone alone is also represented.

Definition 2. Given two collinear segments s_1 and s_2, their convex hull is a segment denoted by $\overline{s_1 s_2}$.

A segment $s = [a, b]$ will be identified with a digital straight segment in the sense of Rosenfeld [41], considering the digital straight segment that joins the center of pixel a to the center of pixel b (*cf.* Fig. 4). The length of the digital segment is denoted by $l(s)$.

Given a precision parameter p, an image x and a segment s, $k(s, x)$ denotes the number of p-aligned pixels in s (p has no special role in the analysis that follows so it is not included in

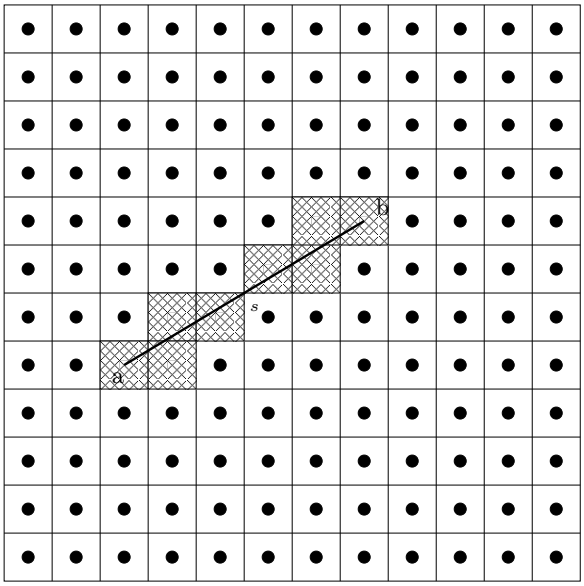

Figure 4: Discretization of a segment. Pointed squares are the pixels and shaded ones are the ones crossed by the segment s.

the notation of k). Usually $k(s, x)$ is simply denoted by $k(s)$ when there is no ambiguity about the image x or even k when s is also implied. We shall denote it by $K(s)$ when considered a random variable.

Definition 3. Given a line L, an n-multisegment with support in L is an n-tuple $(s_1, \ldots, s_n)$ of n *disjoint* segments s_i *contained* in L (*cf.* Fig. 5). The set of all n-multisegments with support in L will be denoted by $\mathcal{M}(n, L)$.

Note that $\#\mathcal{M}(n, L) = \binom{l(L)}{2n}$.

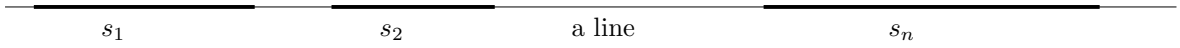

Figure 5: An n-multisegment is an n-tuple $(s_1, \ldots, s_n)$ of n *disjoint* and *collinear* segments

Definition 4. Given an image x and an n-multisegment $(s_1, \ldots, s_n)$, we write $k(s_1, \ldots, s_n, x) \in \mathbb{R}^n$ for the vector whose components are the numbers $k(s_i, x)$ of p-aligned pixels in s_i. Formally,

$$k(s_1, \ldots, s_n, x) = (k(s_1, x), \ldots, k(s_n, x)) .$$

When there is no ambiguity about the image x, we shall write $k(s_1, \ldots, s_n)$ for $k(s_1, \ldots, s_n, x)$.

3 Segment Detection Using Hough Transform

The Hough Transform (HT) is a common tool for line detection. It was introduced in the 60's [21, 40]. Since then much effort has been made to improve and understand it better. Here we recall the basics very briefly, and refer to [24, 28] for more details.

3.1 Hough Transform

Assume one wants to find, among a parametric family of shapes, which ones are present in a given *binary* image I. Assume also that these shapes are described through a parametric

Cartesian equation:

$$f(\lambda, m) = 0,$$

meaning that $f(\lambda, m) = 0$ whenever the pixel $m \in \Omega \subset \mathbb{R}^2$ belongs to the shape defined by the parameter $\lambda \in \Lambda \subset \mathbb{R}^d$. A standard example is the family of lines given by the normal parameterization: $m_1 \cos\theta + m_2 \sin\theta - \rho = 0$, where the parameters are $\lambda = (\rho, \theta) \in \mathbb{R}_+ \times [0, 2\pi[$ and $m = (m_1, m_2) \in \mathbb{R}^2$, see Fig. 6.

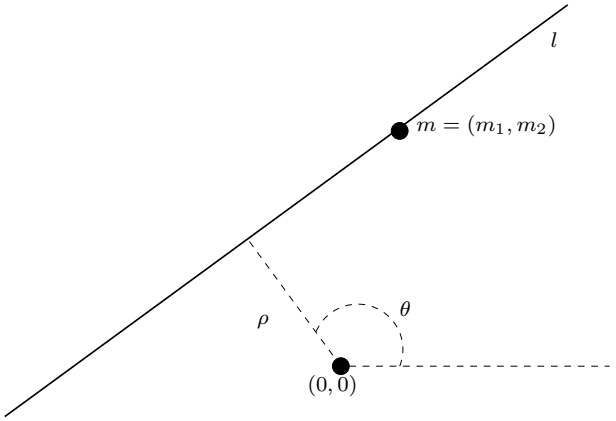

Figure 6: Normal parameterization (ρ, θ) of a line l: $m = (m_1, m_2) \in l \Leftrightarrow m_1 \cos\theta + m_2 \sin\theta - \rho = 0$.

The Hough transform is defined by the integral

$$R(\lambda) = \int_\Omega x(m)\delta\big(f(\lambda, m)\big)dm$$

where δ is the Dirac delta function. The integrand will respond only when the pixel m is "on" and belongs to the shape defined by λ. A peak of $R(\lambda)$ at λ_0 correspond to a good match with the shape of parameter λ_0. In this way the shape detection problem is turned into a peak detection problem, which is an easier one since it is local. This is the idea underpinning the Hough transform.

To compute this integral numerically, it is necessary to discretize the parameter space Λ and the image space Ω. Let us call Λ_d the discrete parameter space, Γ the image grid, and use an approximation of δ, say ϕ. Then the integral is approximated by the sum

$$\forall \lambda \in \Lambda_d \quad R(\lambda) = \sum_{m \in \Gamma} x(m)\phi(f(\lambda, m)). \tag{1}$$

For example, when dealing with lines, one can use a discrete approximation of the normal coordinates (ρ, θ):

$$\Lambda_d = \{u\Delta\rho, u \in [0, Z-1]\} \times \{v\Delta\theta, v \in [0, Z-1]\}$$

where $\Delta\rho$ is the ρ-resolution, and $\Delta\theta = \frac{2\pi}{Z}$ with Z defined by the user. Let us normalize the image support to $\Omega = [0, 1]^2$ and use

$$\Gamma = \{u\Delta m_1, u \in [0, Z-1]\} \times \{v\Delta m_2, v \in [0, Z-1]\}$$

with $\Delta m_1 = \Delta m_2 = \frac{1}{Z}$. In this case, the maximum distance between a pixel and the center of the image (set at coordinates $0, 0$) is $\frac{\sqrt{2}}{2}$. So one should take $\Delta\rho = \frac{1}{\sqrt{2}Z}$.

It is possible to compute the sum of equation (1) in two ways. As a consequence there are two algorithms that compute the same transform. They are traditionally referred to as many to one HT and one to many HT. Many to one HT indicates that many (aligned) points are going to vote for a single line at each step whereas one to many HT indicates that one point is going to vote for all lines going through it.

Algorithm 1: many to one Hough Transform

input : A binary image I, the resolution parameter Z, the size of the window for local maxima detection w.
output: A list of (ρ, θ) corresponding to the detected lines

```
Allocate an array H of size (Z+1)^2 indexed by ρ and θ;
foreach θ ∈ {uΔθ, u ∈ [0, Z]} and ρ ∈ {vΔρ, v ∈ [0, Z]} do
    foreach pixel m belonging to the line ρ, θ do
        increment H(ρ, θ) by x(m)
    end
end
Return the ρ, θ that are local maxima according to a given neighborhood size w × w;
```

Algorithm 2: one to many Hough Transform

input : A binary image I, the resolution parameter Z, the size of the window for local maxima detection w.
output: A list of (ρ, θ) corresponding to the detected lines

```
Allocate an array H of size (Z+1)^2 indexed by ρ and θ;
foreach m ∈ Γ do
    foreach ρ, θ such that the associated line goes through m do
        increment H(ρ, θ) by x(m)
    end
end
Return the ρ, θ that are local maxima according to a given neighborhood size w × w;
```

The difference between both algorithms is the order of summation: The first one scans the parameter space while the second scans the image space. Both algorithms give the same output with different complexity. The first one is more suitable if one wants to know $H(\rho, \theta)$ for a few values of (ρ, θ) and the last one is more suited for sparse images. The second algorithm is the traditional presentation of the Hough transform but the first one shares the strongest links with the DMM algorithm which will be explained in section 4. It is also linked with the Radon Transform [9].

3.2 Segment Detection

A common way to extract segments from images is what we call here the Hough Transform Method (HTM):

- Use an edge detector to produce a binary image.
- Apply the standard Hough transform to detect lines on the binary image.
- Break the lines into segments.

The last step is usually crude: It takes two thresholds, f and g, a binary sequence representing the underlying line (1 if the point is aligned and 0 otherwise) and returns n couples of indexes (b_i, e_i) with $1 \leqslant i \leqslant n$, such that $e_i - b_i \geqslant f$ for all $1 \leqslant i \leqslant n$ and $b_{i+1} - e_i \geqslant g$ for all $1 \leqslant i \leqslant n-1$, that represents the n disjoint segments found on the line. Note that f corresponds to the minimum length of a segment and g to the minimum gap between segments.

The pseudo-code Algorithm 3 corresponds to this method. It depends on the algorithm Extract-Runs that extracts maximal consecutive sequences of "1" in a binary input sequence. Each such sequence is usually called a *run*.

There are several parameters involved in the HTM. The edge detector step involves at least a SNR-related parameter and a scale-related parameter. The standard Hough transform usually

Algorithm 3: 2-Thresholds Segmentation

input : A couple of integers (f, g) and a binary array Line
output: A list of integers Segs: i^{th} segment going from Segs$(2i)$ to Segs$(2i+1)$ (excluded)

```
1  R ← Extract-Runs(Line);
2  l ← Size (R);
3  Segs ← [];
4  Add R(0) to Segs;
5  for i ← 0 to l − 2 do
6      if R(2(i + 1)) − R(2i + 1) ⩾ g then
7          Add R(2i + 1) to Segs;
8          Add R(2(i + 1)) to Segs;
9      end
10 end
11 Add R(2l − 1) to Segs;
12 i ← 0;
13 while i < Size(Segs) do
14     if R(2i + 1) − R(2i) < f then
15         Remove Segs(2i) from Segs;
16         Remove Segs(2i + 1) from Segs;
17     else
18         i++
19     end
20 end
```

involves four parameters, one scale-related (the size of the neighborhood used for the local maxima search), a second one related to the false positive/false negative trade-off, and the other two related to the discretization of the space parameters ρ and θ. The last step involves at least two parameters: one is the minimal gap between two segments and the second is the minimal length of a segment. All of these parameters are scale and SNR dependent.

The resolution in ρ and θ can be computed from the resolution of the image but the setting of all the other parameters can be problematic. When correctly set, they can lead to good results as seen in Fig. 2(b). But using fixed thresholds can lead to false positive or false negative detections (see Fig. 7). So the problem is to correctly adjust the parameters.

Unfortunately there are no general rules to fix these parameters automatically. This parameter tuning problem has already been addressed in the framework of decision theory. Princen *et al.* (*cf.* [39]) say "$[\cdots]$ HT is a hypothesis testing method. Each sample in the HT array implements a test to determine whether a curve with the given parameters fits the edge point data. This view allows the performance of HT algorithms to be quantified. The power function, which gives the probability of rejection as a function of the underlying parametric distribution of data points, is shown to be the fundamentally important characteristic of HT behavior." Lindenbaum [29] computed a probability of false alarm linked to the DMM method which we will describe in the sequel. Arias-Castro *et al.* [3] formalized the segment detection problem as a generalized likelihood ratio test. This point of view had also been adopted by those who randomized the HT in order to speed up the computations [33, 43]. The next section deals with the Desolneux *et al.* theory to tackle the problem of setting the thresholds.

4 Meaningful Segments

In [14], Desolneux *et al.* came up with a general framework to deal with parameter thresholds in image analysis. The main idea is to set the thresholds of a detection algorithm in order to control its expected number of false detections under a background model. This *a contrario*

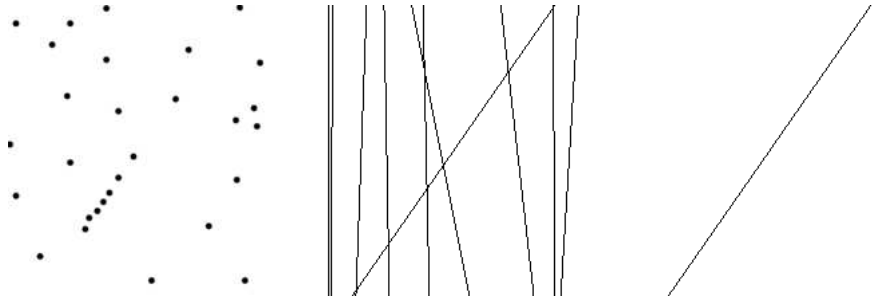

Figure 7: The Hough transform can lead to correct results when its thresholds are correctly set, but there are no general rules to fix them. Left: Random points drawn according to a uniform law and 8 aligned points. Middle: Lines detected by the Hough transform using a threshold set to 1. All detections but one are false positive. Right: Lines detected by the Hough transform using a threshold set to 25. This is the correct result. If the threshold is set to 100, there is no detection at all, which means one false negative.

approach can be opposed to the classical method which consists of finding a statistical model for the objects to be detected. Here instead, the background only is roughly modeled and the features detected as outliers to the background model. The authors claimed that they simply developed a program proposed by D. Lowe in [30]: "We need to determine the probability that each relation in the image could have arisen by accident. Naturally, the smaller that this value is, the more likely the relation is to have a causal interpretation. [...] Given an image relation that holds within some degree of accuracy, we wish to calculate the probability that it could have arisen by accident to within that level of accuracy. This can only be done in the context of some assumption regarding the surrounding distribution of objects, which serves as the null hypothesis against which we judge significance."

This framework has been successfully applied to the detection of various structures in images: segments [14], modes of histograms [14, 10], edges [15], clusters [5, 6], vanishing points [2], stereoscopic matching [34], and shape matching [37].

Concerning segments, the DMM method in [14] deals directly with the parametric family of all segments without any previous line detection step. Straight line segments form a parametric family described by four parameters. The first stage of the method performs a many to one-like HT associated with the family of *segments* (and not lines). A pixel votes for a segment when it is p-aligned to it (for notation please refer to section 2 and Fig. 3). Formally, given an image x, every segment s in the image is mapped to $k(s, x)$.

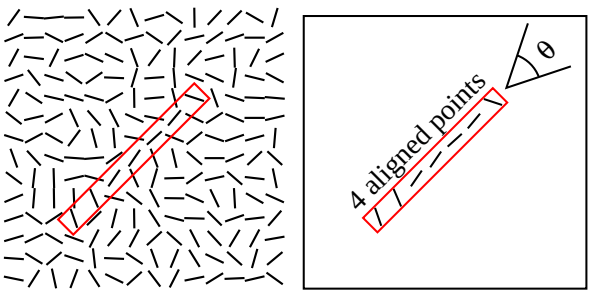

Figure 8: Left: One segment shown over the level-line orientation field (orthogonal to the gradient orientation field). Right: The number of aligned points is counted up to an angular tolerance of $\theta = 2p\pi$. Here the shown segment receives 4 votes among 7.

In the standard Hough transform, after filling the accumulator, the local maxima are selected. Selecting peaks in the accumulator favors the largest segments: large segments are more likely

to receive more votes because there are more pixels that can vote for them. Thus the thresholds should be *non-uniform* over the accumulator. Small segments should be detected with a smaller threshold than large segments. But how to set the thresholds?

The authors of [14] found a solution to this problem. They map the Hough accumulator to a new one using a non-linear transformation. A uniform thresholding over the mapped accumulator is equivalent to a non-uniform one in the Hough one. Moreover, the transform used can be interpreted in terms of number of false alarms. When the threshold is set to ε, the overall expected number of false alarms in white noise is less than ε. So the user can set the threshold according to the number of false alarms he can accept. This is also another advantage over the classical HT approach which does not give any *a priori* idea to the user of what to expect from the chosen threshold.

In the next subsection, we will briefly comment the underlying statistical framework, since we are going to need it for our own method. For more details see [12].

4.1 Statistical Framework

An image X from the background H_0 model is a random image such that:

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

When $X(i, j)$ are independent Gaussian random variables with any mean μ and variance σ^2 the two previous assertions hold true[1] (*cf.* [12]). Such a white noise image and its associated gradient orientation are represented in Fig. 9.

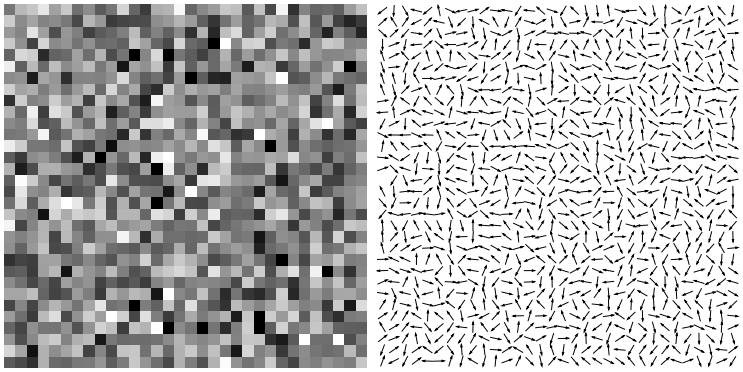

Figure 9: Background model: no structures. Left: A Gaussian white noise image. Right: The level line orientation field (orthogonal to the gradient). Gaussian white noise yields uniformly distributed gradient angles.

There are as many tests T_s as possible segments in the image. Each test relies on the statistics $k(s)$:

$$\text{Reject } H_0 \text{ if } k(s) \geqslant k_s$$
$$\text{Accept } H_0 \text{ otherwise,}$$

for a threshold k_s that controls false detections. For this test T_s, non-H_0 is also denoted H_s.

If there were only one test, a significance level ε would be chosen and the test would be rejected whenever its p-value would be less than ε. This would imply $k_s = \min\{k, \mathbb{P}_{H_0}[k(s) \geqslant k] < \varepsilon\}$. When dealing with multiple tests the p-value is replaced by the notion of Number of False Alarms (*abbrev.* NFA).

[1] This is true under some other condition on the sampling, see section 9.2.

Definition 5 (NFA(s),[14])**.** Given a segment $s \in \mathcal{S}$ and an image x, one defines

$$\mathrm{NFA}(s, x) = \#\mathcal{S} \cdot \mathbb{P}_{H_0}[k(s, X) \geqslant k(s, x)],$$

where X is a random image on H_0. When there is no ambiguity about the image x it is simply denoted by NFA(s). The NFA(s) is $\#\mathcal{S}$ times the p-value of the test $k(s)$.

Rejecting H_0 if and only if $\mathrm{NFA}(s) \leqslant \varepsilon$ is equivalent to applying the so-called Bonferroni correction [42]. Just like for the p-value, the smaller NFA(s) the more meaningful the observed segment s is, *i.e.*, the less likely it is to appear in an image drawn by the H_0 model. In this way, ε controls the average number of detected segments under the H_0 hypothesis. The next proposition gives a precise meaning to this fact.

Proposition 1 ([14])**.** *If* NFA(ε) *stands for the quantity*

$$\mathbb{E}_{H_0} \sum_{s \in \mathcal{S}} \mathbb{1}_{\mathrm{NFA}(s,X) \leqslant \varepsilon}$$

Then

$$\mathrm{NFA}(\varepsilon) \leqslant \varepsilon.$$

In other terms, the expectation of the number of ε-meaningful segments in the background model is less than ε.

Proof.

$$\mathbb{E}_{H_0} \sum_{s \in \mathcal{S}} \mathbb{1}_{\mathrm{NFA}(s,X) \leqslant \varepsilon} = \sum_{s \in \mathcal{S}} \mathbb{P}_{H_0}[\mathrm{NFA}(s, X) \leqslant \varepsilon].$$

By definition of NFA(s, X),

$$\mathbb{P}_{H_0}[\mathrm{NFA}(s, X) \leqslant \varepsilon] = \mathbb{P}_{H_0}\left[U(s, X) \leqslant \frac{\varepsilon}{\#\mathcal{S}}\right]$$

where $U(s, x)$ stands for the quantity $\mathbb{P}_{H_0}[k(s, X) \geqslant k(s, x)]$. The result then comes from the fact that, for every $\alpha > 0$,

$$\mathbb{P}_{H_0}[U(s, X) \leqslant \alpha] \leqslant \alpha.$$

□

Note that NFA(ε) is not the same as NFA(s). The expression NFA(ε) is not attached to any segment. Rather, it is attached to the method itself. The above proposition says that if the detected segments are those such that $\mathrm{NFA}(s) \leqslant \varepsilon$, then the overall expected number of false alarms under the H_0 assumption is less than ε.

Computations can be done explicitly because under the H_0 hypothesis, $k(s)$ follows a binomial law of parameters $l(s)$ and p, so that $\mathrm{NFA}(s) \leqslant \varepsilon$ becomes

$$b\Big(l(s), k(s), p\Big) \leqslant \frac{\varepsilon}{\#\Gamma^2}$$

where $b(l, k, p) = \sum_{i=0}^{i=k} \binom{l}{i} p^i (1-p)^{l-i}$ stands for the binomial tail, that is, the probability for a binomial of parameters l and p to be larger than k.

Desolneux *et al.* claim to have a parameterless method: *"This [ε] seems to contradict our notion of a parameter-less theory. Now, it does not, since the ε-dependency of meaningfulness will be low (it will be in fact a $\log \varepsilon$-dependency)."* [14]. They suggest to take $\varepsilon = 1$ as a general rule. All the experiments in this paper were done using $\varepsilon = 1$.

Algorithm 4 sketches the method. Note that the role of the number of votes in the Hough transform is now played by NFA(s).

Fig. 10, middle, shows what happens when algorithm 4 is applied to the house image. There are many more detected segments than what we would normally perceive. Still, all the relevant segments are among the detected ones and all detections seem to stem for an actual alignment. The next subsection explains why this happens and what Desolneux *et al.* did in order to select only the most relevant segments, which removes the main drawback of their method.

Algorithm 4: Meaningful Segments, [14]

input : An image x.
output: The list Segs of all ε-meaningful segments.

```
1 Segs ← [];
2 foreach segment s ∈ S do
3     if NFA(s, x) ⩽ ε then
4         Add s to Segs
5     end
6 end
```

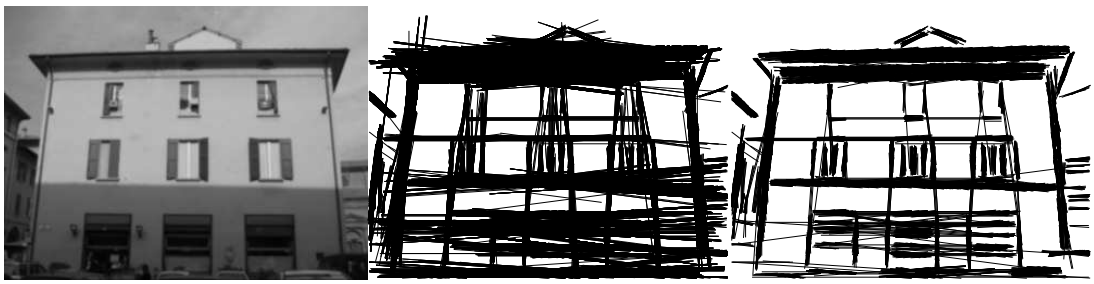

Figure 10: Left: A digital image. Middle: All 1-meaningful segments. Right: All maximal meaningful segments. The meaningful segments give redundant information. There are many more meaningful segments than perceptual ones. The ones we do perceive are among these detected segments. Indeed the presence of alignments induces multiple detections for two reasons: a) Almost every meaningful segment contains shorter meaningful segments and is contained in longer meaningful segments. This multiple detection can be fixed by retaining only maximal 1-meaningful segments, as shown on the right image. b) since the image is blurry the alignments are usually thick and multiple alignments are detected, slanted up to the precision p. The exclusion principle introduces a competition between meaningful segments stemming from the same "true segment" and keeps only the relevant ones (see Fig. 2(c) for the result on this image).

4.2 Maximality and Exclusion Principle

The example illustrated by figure 11 is a typical situation. Whenever a segment has a central part with many p-aligned points, it is systematically detected as meaningful because its aligned portion still makes it very unlikely to appear by chance, *i.e.* under H_0 (cf. fig. 11).

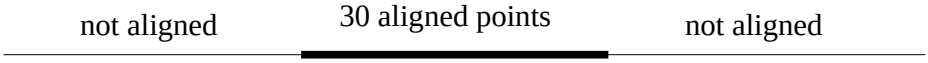

Figure 11: A segment of length 90 with only 30 points aligned in its central part is still unlikely to appear in a random 512×512 image (NFA $= 8 \cdot 10^{-4}$). Thus it is detected as a 1-meaningful segment. Its central part (30 aligned points) is much more meaningful (NFA $= 5 \cdot 10^{-26}$); in fact, it is the only maximal meaningful segment in this case.

In our exposition the number of false alarms has been used to establish a threshold. Now, $\text{NFA}(s)$ also yields a measure of the meaningfulness of s. When it is low $k(s)$ is large, *i.e.* it contains many p-aligned points. So $\text{NFA}(s)$ can be used as a criterion for selecting the right segment when two compete. The best segments have the lowest NFA.

Definition 6 (maximality, [14])**.** A segment s is maximal when

- $\forall s' \subseteq s, \quad \text{NFA}(s) \leqslant \text{NFA}(s')$
- $\forall s' \supset s, \quad \text{NFA}(s) < \text{NFA}(s')$

Note that the second inclusion is strict.

We can define the neighborhood of s as $\mathcal{N}(s) = \{s' | s \subset s' \text{ or } s \supset s'\}$. Selecting the local minima of $\text{NFA}(s)$ according to this neighborhood system gives algorithm 5.

Algorithm 5: Maximal Meaningful Segments, [14]

input : An image x
output: The list maxSegs of all ε-maximum meaningful segments.

```
1  maxSegs ← [];
2  foreach Line l do
3      Segs ← [];
4      foreach segment s with support in l do
5          if NFA(s, x) ⩽ ε then
6              Add s to Segs
7          end
8      end
9      foreach s in Segs do
10         if for all s' ∈ Segs ∩ N(s) is true that NFA(s) ⩽ NFA(s') then
11             Add s to maxSegs
12         end
13     end
14 end
```

When keeping only maximal segments as defined above, the result improves in a very significative way. But still, many parallel or slanted adjacent segments are simultaneously detected due to the same perceptual alignment (*cf.* Fig. 10, right). This redundancy is due to the image blur which thickens alignments. Since maximality only makes sense for segments supported by the same line, picking only maximal meaningful segments is not enough to remove this last redundancy. In order to get rid of such redundant detections, Desolneux *et al.* proposed in [14] an

exclusion principle, which was extended by Almansa [1, 12]. Stated as a sequential algorithm, the exclusion principle leads to pick first the segment with the lowest NFA. All of its points and all points of a thick neighborhood of this segment can no more be used in the calculation of the meaningfulness of other segments. This process is repeated with the next most meaningful segment and so on, until there are no more meaningful segments. Algorithm 6 is the corresponding pseudo-code. It uses the procedure RemovePoints(s, x) that prevents all pixels belonging to a thick neighborhood of s from being used afterwards when computing the NFA of other segments.

Algorithm 6: Exclusion Principle, [12]

input : An image x, a list of segments L
output: A list of segments `Lout`.

```
1  Lout ← [];
2  while Size(L) > 0 do
3      foreach s ∈ L do
4          Compute NFA(s, x);
5          if NFA(s, x) ⩾ 1 then
6              Remove s from L
7          end
8      end
9      Find s_min ∈ L with the lowest NFA;
10     if NFA(s_min, x) < 1 then
11         RemovePoints(s_min, x);
12         Move s_min to Lout
13     end
14 end
```

Remark 1. The exclusion principle does not bring new segment detections. It just creates competition between segments and eliminates losers.

Remark 2. Throughout this paper, when we talk about the DMM algorithm, the exclusion principle is implicitly included.

The result of this algorithm was shown in Fig. 2(c).

5 The Flat Pieces Detector

For the sake of completeness we shall explain the ideas underpinning the Flat Pieces Detector (FPD) of Musé and Sur [36, 44].

The core of this algorithm detects straight pieces of a given curve. These pieces are called Flat Pieces. When performed on all meaningful curves of an image it is an alternative to a straight segment detector. It can be seen as a chaining algorithm combined with an *a contrario* model which allows to control the average number of false alarms.

5.1 The curve selection step

The first step consists in finding the paths in the image where to look for flat pieces, as done in classical chaining methods. The level lines are a natural candidate for this task because, on the one hand they locally coincide with edges and encode the image geometry, and on the other hand they can be very efficiently extracted using the Fast Level Set Transform of Monasse *et al.* [35]. Here, there are essentially two main options: either keep all the level lines tree using the FLST, or select only the meaningful ones in the sense of Cao *et al.* [7]. But the detected flat pieces will be very much the same in both cases because the flat part usually coincide with the meaningful level lines. The difference between these two options is illustrated in Fig. 12.

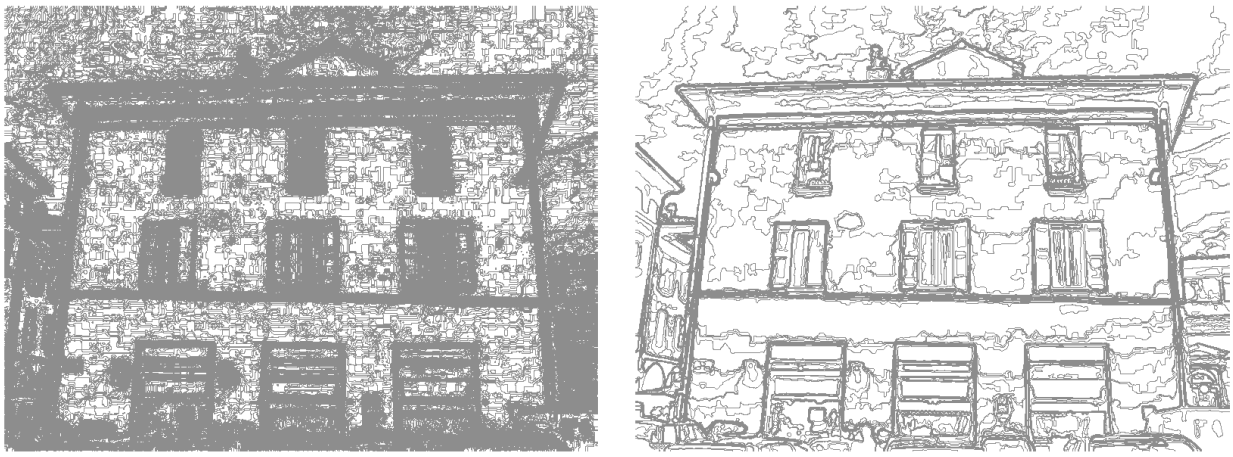

Figure 12: Left: all level lines. Right: only the meaningful level lines.

5.2 The *a contrario* model

The straightness of a piece of curve is measured by the maximum angle between the chord at its endpoints and the tangents of the curve (*cf.* Fig. 13).

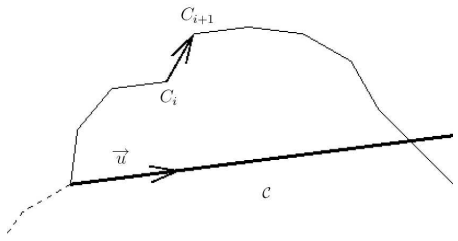

Figure 13: The flatness of a piece of curve is measured by the maximum angle between the chord direction $\vec{u}$ and the tangents $\overrightarrow{C_iC_{i+1}}$. Formally $\max_i |\text{Angle}(C_iC_{i+1}, \vec{u})|$.

How much the piece of curve is allowed to deviate from its endpoints chord is specified by two thresholds. The first threshold α^* is a coarse threshold: it specifies how much each tangent of the piece is allowed to deviate from the chord direction $\vec{u}$. If (C_i) denotes the points belonging to the given piece of curve, one must have $\max_i |\text{Angle}(C_iC_{i+1}, \vec{u})| < \alpha^*$ or reject the piece. The second threshold takes into account the maximal deviation angle and the length of the piece (see algorithm 7 for the details). The authors of the FPD systematically set α^* to 1 and the second threshold is automatically adjusted using an *a contrario* model. Namely, in this *a contrario* model, every point of the given curve has independent and uniform deviation angles.

The last part of the process consists in reducing redundancy (*cf.* line 16 of the algorithm 7). The flattest piece of a curve prevents all other overlapping pieces of the same curve from being tagged as flat. It plays the same role than the maximality principle of Desolneux *et al.* (see again algorithm 7 for the details).

Fig. 14 shows the result of this algorithm. The algorithm performs quite well, but it has two drawbacks. The first drawback is that it has been primarily conceived as a flat piece detector for *one* curve. How to select the curves one wants to extract the flat pieces from is left to the user. Actually this selection is quite an intricate problem. For this purpose we used the meaningful boundaries of Cao *et al.* [7]. This algorithm sometimes has false positive detections as can be

Algorithm 7: Flat Piece Detector, [36, 44]

input : An image x
output: A list of flat piece of curves Lout.

1 Lout $\leftarrow$ [];
2 $Lin \leftarrow$A list of curves (usually the level lines of the image x);
3 **foreach** $c \in Lin$ **do**
4 **foreach** *Chord of* c **do**
5 Compute $\alpha = \max_i |\text{Angle}(C_i C_{i+1}, \vec{u})|$;
6 **if** $\alpha > \alpha^*$ **then**
7 reject the considered piece
8 **else**
9 compute $p(\alpha, l) = \left(\frac{\alpha}{\alpha^*}\right)^l$ where l is the length of the considered piece
10 **end**
11 **if** $p(\alpha, l) > p^*$ **then**
12 Reject the piece
13 **end**
14 **end**
15 Select the piece π with the lowest $p(\alpha, l)$ and add it to Lout;
16 Drop all the pieces of c overlapping π;
17 Go to line 15 and iterate until no more pieces are available;
18 **end**

seen in the sky in the house image. When this step has failed, its errors propagate to the flat piece detections.

Another drawback is related to redundancy reduction. The flat pieces which are located on the same curves are dealt with properly. But whenever two flat pieces are located on close curves there is nothing done in order to get rid of redundancy. This is why one can observe many parallel flat pieces detected as shown in Fig. 15. It lacks an exclusion principle.

Figure 14: The detected flat pieces of meaningful level lines for the image 2. One can see some false positives (for example in the sky).

The advantage of this algorithm is that it can be used with any curve selection step. So it is possible to use it with other chaining rules than the level line one. We did not investigate in this direction. This is why we are going to focus our comparisons with the HTM and DMM methods only.

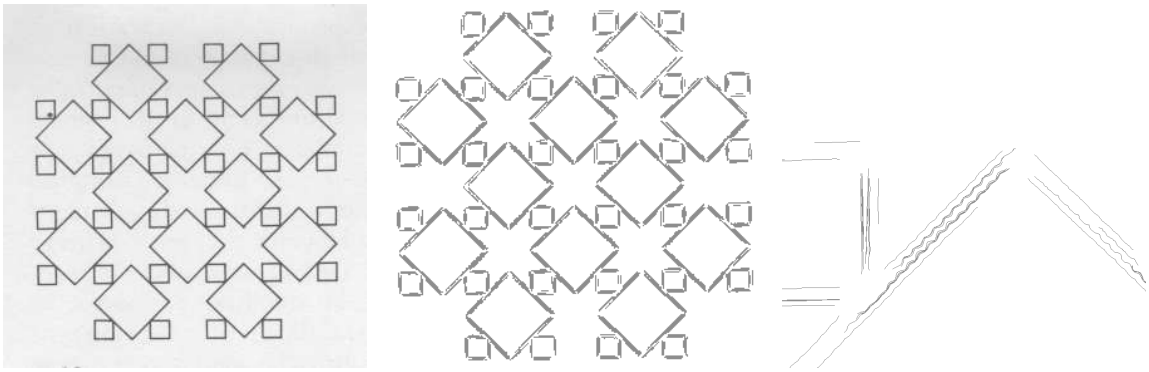

Figure 15: Left: a synthetic noisy image. Center: its detected flat pieces. Right: If one look carefully at the detected pieces, one can see that there are some redundancies.

6 Breaking Lines Into Pieces: Multisegments

Fig 2(c) showed how algorithms 5+6 perform on the house image. The visual impression is not correct because some segments are unduly non local and should be cut into smaller segments. For example, the top and bottom edges of some windows are joined. This "window problem" is not a bug and can be reproduced on a synthetic image as shown in Fig. 16 where both top and bottom edges are merged and detected as a unique segment. The reason is that when deciding whether two segments A and B should be merged into segment $C = \overline{AB}$ or not, algorithm 5 compares NFA(A) against NFA(C), and NFA(B) against NFA(C). If NFA(C) $<$ NFA(A) and NFA(C) $<$ NFA(B), A and B are merged. If NFA(A) $<$ NFA(C) and NFA(B) $<$ NFA(C) A and B are kept separated. Otherwise, one of them disappears. All this cases might happen as illustrated by the numerical computation shown in Fig. 17. The "window problem" corresponds to the first case when A and B are merged and should not. The last case, where A or B disappears also is wrong. Even if one segment on the line, say A, is correctly detected, the second one can be masked. This is not really a false negative, but rather a case we could term as *masked detection*. Indeed, if A is removed, then B is detected. These drawbacks of the *a contrario* detection theory have been pointed out for histogram segmentation [10] and for shape grouping algorithms [5, 6]. In [10] Delon *et al.* explain why large histogram modes may mask small ones when using a similar maximality principle. In [5, 6] Cao *et al.* showed that the proper way to decide whether or not a cluster C should be refined into two disjoint sub-clusters C_1 and C_2 is not only to compare the meaningfulness of C_1 (*resp.* C_2) against the NFA of C but also to compare the joint NFA of the group of clusters formed by C_1 and C_2 against the meaningfulness of their union C.

In summary, the problem pointed out in Fig 2(c) is that the DMM theory does not address correctly the problem of whether A and B should be merged or kept separated. In order to decide if A and B (or more segments on the same line) should be kept separated or should be merged, one should also compare NFA(A and B) against NFA($\overline{AB}$). The DMM method addresses the question "what is the most meaningful event: A, B or $\overline{AB}$?". A more accurate comparison should answer the question "what is the most meaningful event: A alone, B alone, A and B, or $\overline{AB}$?". Answering this question gives the best interpretation. The difficulty is to give a meaning to NFA(A and B). The method proposed in this paper will be called Multisegment detection. It aims at giving a global interpretation of the line in terms of segments and is in spirit close to the solution found in [5, 6] for cluster detection. Fig. 18 shows what a meaningful multisegment looks like.

(a) (b)

Figure 16: Maximality and Exclusion Principle do not always select the best interpretation in terms of meaningful segments: The top and bottom edges of the two squares (a) are detected as a single bigger segment (b).

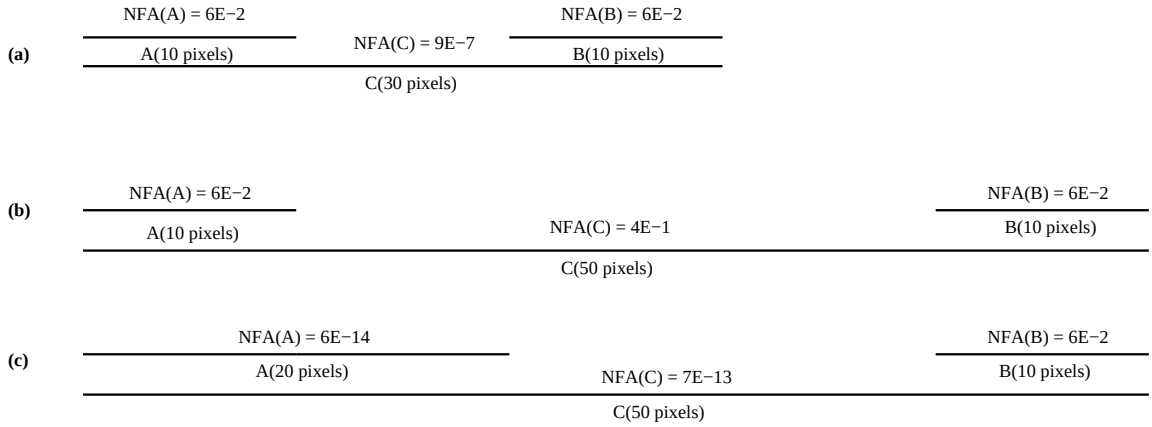

Figure 17: A problem with Maximality as defined by Desolneux *et al.*: (a) $C = \overline{AB}$ is the most meaningful segment. In consequence the perceptually correct segments A and B are merged. (b) A and B are both more meaningful than $C = \overline{AB}$. They stay separated but this state is unstable as illustrated in (c). (c) A is more meaningful than C which is more meaningful than B. So A stays alone and B disappears, a clear miss in the detection. All of these situations appear in the experiment of Fig 2(c). This will be corrected by defining a meaningfulness for multisegments. In the computation of the NFA values we assumed an image size of 512×512 pixels and a precision $p = 1/16$.

6.1 Giving a Meaning to $\mathrm{NFA}(s_1, \ldots, s_n)$

The same steps as for a single segment case are involved in the derivation of an expression for the number of false alarms $\mathrm{NFA}(s_1, \ldots, s_n)$ for the multisegment $(s_1, \ldots, s_n)$. The *a contrario* framework still holds: H_0 is a Gaussian white noise.

In the case of a single segment s, the DMM test had the form $b\big(l(s), k(s), p\big) \leqslant \alpha$ with α equal to $\frac{\varepsilon}{\#\mathcal{S}}$. Since the segments of a multisegment are not overlapping, the number of aligned points for each one are independent random variables. Thus the tests associated to a multisegment $(s_1, \ldots, s_n)$ have the form

$$\prod_{i=1}^{n} b\Big(l(s_i), k(s_i), p\Big) \leqslant \alpha.$$

Definition 7. The number of false alarms of an n-multisegment $(s_1, \ldots, s_n)$ in $\mathcal{M}(n, L)$ sup-

Figure 18: A multisegment is a global interpretation in terms of segments for the whole line. Left: Original image. Right: Segments detected by multisegments. One multisegment formed by 4 segments has been drawn in bold. The multisegment structure also tells us that these segments are aligned.

ported by the line L in an image x is defined by

$$\mathrm{NFA}(s_1,\ldots,s_n,x) = \#\mathcal{L}\cdot\binom{l(L)}{2n}b\Big(l(s_1),k(s_1),p\Big)\prod_{i=2}^{n}\big(l(s_i)+1\big)b\Big(l(s_i),k(s_i),p\Big)$$

where $\#\mathcal{L}$ stands for the total number of lines in x, *i.e.* $2(M+N)^2+4MN-16M-16N+28$ for an $M\times N$ image. A multisegment $(s_1,\ldots,s_n)$ is said ε-meaningful whenever $\mathrm{NFA}(s_1,\ldots,s_n,x)\leqslant\varepsilon$. The $\mathrm{NFA}(s_1,\ldots,s_n,x)$ will often be abbreviated as $\mathrm{NFA}(s_1,\ldots,s_n)$ when there is no ambiguity about x.

This expression is formed by two parts: the number of tests and a sharp upper bound of the probability of the event. The number of tests is the product of the number of lines in an image $\#\mathcal{L}$ by the number of possible n-multisegments in the given line $\binom{l(L)}{2n}$. The upper bound for the probability of the observed event is

$$b\Big(l(s_1),k(s_1),p\Big)\prod_{i=2}^{n}\big(l(s_i)+1\big)b\Big(l(s_i),k(s_i),p\Big).$$

The necessity of this definition will be clarified by Proposition 2. For one segment ($n=1$), the above definition gives $\mathrm{NFA}(s)=\#\mathcal{L}\cdot\binom{l(L)}{2}b\big(l(s),k(s),p\big)$. There is a small difference with the previous definition of $\mathrm{NFA}(s)$. The number $\mathrm{NFA}(s)$ depends now on the line that supports s. Considering that in an $N\times N$ image most of the lines have a length approximately equal to N, the new definition gives $\mathrm{NFA}(s)$ proportional to $N^4b\big(l(s),k(s),p\big)$, which is the definition given by Desolneux *et al.* The reason for this difference is only technical: there is a closed form for the number of simple segments in a line L – namely $\binom{L}{2}$ – as well as in the whole image – $(N\times M)^2$ – for an image of size $N\times M$; but for multisegments there is no such formula for the whole image.

The following proposition is a sanity check for the method: it shows that the number of detections in the background model is controlled.

Proposition 2. *Set for all $n>0$,*

$$\mathrm{NFA}(\varepsilon)=\mathbb{E}_{H_0}[\textit{number of }\varepsilon\textit{-meaningful }n\textit{-multisegments}]$$

Then

$$\mathrm{NFA}(\varepsilon)\leqslant\varepsilon$$

Proof. One has:

$$\text{number of } \varepsilon\text{-meaningful } n\text{-multisegments} = \sum_{L\in\mathcal{L}} \sum_{(s_1,\ldots,s_n)\in\mathcal{M}(n,L)} \mathbb{1}_{\mathrm{NFA}(s_1,\ldots,s_n)\leqslant\varepsilon}.$$

$\mathrm{NFA}(s_1,\ldots,s_n) \leqslant \varepsilon$ is equivalent to

$$\prod_{i=1}^{n} b\Big(l(s_i),k(s_i),p\Big) \leqslant \frac{\varepsilon}{\#\mathcal{L}\cdot\binom{l(L)}{2n}\prod_{i=2}^{n}\big(l(s_i)+1\big)}.$$

Setting $U_i = b\big(l(s_i),k(s_i),p\big)$, and α being a positive number,

$$\mathbb{P}_{H_0}\left[\prod_{i=1}^{n} U_i \leqslant \alpha\right] = \sum_{(u_2,\ldots,u_n)} \mathbb{P}_{H_0}\left[\prod_{i=1}^{n} U_i \leqslant \alpha | U_2=u_2,\ldots,U_n=u_n\right]\mathbb{P}_{H_0}\left[U_2=u_2,\ldots,U_n=u_n\right].$$

Since the s_i are disjoint the U_i are independent,

$$\mathbb{P}_{H_0}\left[\prod_{i=1}^{n} U_i \leqslant \alpha\right] = \sum_{(u_2,\ldots,u_n)} \mathbb{P}_{H_0}\left[U_1 \leqslant \frac{\alpha}{u_2\cdots u_n}\right]\mathbb{P}_{H_0}\left[U_2=u_2,\ldots,U_n=u_n\right].$$

Now using that $\mathbb{P}[U_i\leqslant\alpha]\leqslant\alpha$ as already seen in proposition 1 and $\mathbb{P}_{H_0}\left[U_2=u_2,\ldots,U_n=u_n\right] \leqslant \mathbb{P}_{H_0}\left[U_2\leqslant u_2,\ldots,U_n\leqslant u_n\right]$ one gets, since there are $l(s_i)+1$ possible values for U_i,

$$\mathbb{P}_{H_0}\left[\prod_{i=1}^{n} U_i \leqslant \alpha\right] \leqslant \prod_{i=2}^{n}\big(l(s_i)+1\big)\alpha.$$

Setting $\alpha = \frac{\varepsilon}{\#\mathcal{L}\cdot\binom{l(L)}{2n}\prod_{i=2}^{n}\left(l(s_i)+1\right)}$ gives the wanted result. □

Remark 3. The inequality $\mathbb{P}_{H_0}\left[U_2=u_2,\ldots,U_n=u_n\right] \leqslant \mathbb{P}_{H_0}\left[U_2\leqslant u_2,\ldots,U_n\leqslant u_n\right]$ is crude but suffices.

6.2 Using $\mathrm{NFA}(s_1,\ldots,s_n)$

Let us use this new NFA to analyze the numerical examples in Fig. 17. As before, we assume an image size of 512×512 pixels, a line length of 512 pixels, and a precision $p=1/16$.

(a) $\mathrm{NFA}(A)=\mathrm{NFA}(B)=1.3\,\mathrm{E}{-3}$, $\mathrm{NFA}(\overline{AB})=2\,\mathrm{E}{-8}$, $\mathrm{NFA}(A,B)=6.5\,\mathrm{E}{-12}$

(b) $\mathrm{NFA}(A)=\mathrm{NFA}(B)=1.3\,\mathrm{E}{-3}$, $\mathrm{NFA}(\overline{AB})=9.5\,\mathrm{E}{-3}$, $\mathrm{NFA}(A,B)=6.5\,\mathrm{E}{-12}$

(c) $\mathrm{NFA}(A)=1.2\,\mathrm{E}{-15}$, $\mathrm{NFA}(B)=1.3\,\mathrm{E}{-3}$, $\mathrm{NFA}(\overline{AB})=1.6\,\mathrm{E}{-14}$, $\mathrm{NFA}(A,B)=5.9\,\mathrm{E}{-24}$

As already pointed out, these figures are different from the figures in Fig. 17 because the definition of $\mathrm{NFA}(s)$ has changed slightly. But what matters is the order of magnitude of $\mathrm{NFA}(A)$, $\mathrm{NFA}(B)$ and $\mathrm{NFA}(\overline{AB})$. It is easy to check that it has the same values with both $\mathrm{NFA}(s)$ definitions. This is indeed what can be observed in the previous computations.

When the new $\mathrm{NFA}(A,B)$ is taken into account, the wrong merging of example (a) vanishes and the segment B of example (c) does not disappear anymore. On the three examples, the most significant event is the 2-multisegment (A,B). This experiment shows that introducing the multisegment meaningfulness $\mathrm{NFA}(A,B)$ overcomes the difficulties encountered with the Desolneux *et al.* notion of maximality. Concerning case (c) we have:

Proposition 3. *If $s_1,\ldots,s_n$ are n 1-meaningful segments ($\mathrm{NFA}(s_i)\leqslant1$), then*

$$\forall i \quad \mathrm{NFA}(s_1,\ldots,s_n) < \mathrm{NFA}(s_i)$$

(a) (b)

Figure 19: (a) Original image. (b) multisegment detection. The top and bottom edges of the two squares (a) are well separated. Compare with Fig 16.

Proof. According to the definitions,

$$\mathrm{NFA}(s_1,\ldots,s_n) = \frac{\binom{l(L)}{2n}}{\binom{l(L)}{2}^n} \frac{\prod_{i=2}^{n}\big(l(s_i)+1\big)}{\#\mathcal{L}^{n-1}} \mathrm{NFA}(s_1)\cdots\mathrm{NFA}(s_n).$$

But $\binom{l(L)}{2n} \leqslant \binom{l(L)}{2}^n$ because $\binom{n}{a+b} \leqslant \binom{n}{a}\binom{n}{b}$. Also, $l(s_i)$ is less than the diameter of the image ($\sqrt{M^2+N^2}$), and it is easy to check that $\sqrt{M^2+N^2}+1$ is less than $\#\mathcal{L}$ as soon as $M>1$ and $N>1$. □

This proposition tells us that whenever two segments A and B are 1-meaningful, the 2-multisegment they form is more meaningful than each segment alone. The only comparison that must be done is between $\mathrm{NFA}(A,B)$ and $\mathrm{NFA}(AB)$. If $\mathrm{NFA}(AB) < \mathrm{NFA}(A,B)$ then A and B must be merged, otherwise they must be kept separated. In any case they must not be discarded. Fig. 19 shows the result of multisegment detection over the test image of the squares (compare with Fig. 16).

Implementation details will be treated in a separate section. The main issue to be dealt with will be the potential combinatorial blow-up in the search for all possible multisegments in an image.

7 Binary Sequence Segmentation

All previously mentioned algorithms rely on a 1D step. For the HTM, this 1D step is the 2-thresholds algorithm. For the DMM algorithm this 1D step is the selection of the maximal 1-meaningful segments. For multisegments, the 1D step is the selection of the maximal 1-meaningful multisegment (*i.e.*, the multisegment that has the lowest NFA on the given line provided that it is less than 1). This 1D step is really the core of each method in order to go from lines to segments. The multisegment approach has the more sophisticated one. Let us discuss this issue.

Some terminology is needed. A 1D sequence is a finite binary sequence $(x_n)_{1\leqslant n\leqslant L}$ ($\forall n\ x_n = 0$ or $x_n = 1$). See Fig. 20 for an example. In this context, a segment is a couple of increasing indexes $[i,j]$ with $i \leqslant j$ and an n-multisegment is characterized by an increasing family of indexes $(i_k)_{1\leqslant k\leqslant 2n}$ such that $i_{2k} < i_{2k+1}$. The underlying segments s_k then correspond to each $[i_{2k-1}, i_{2k}]$. The number of aligned points $k(s_k)$ is $\sum_{i\in s_k} x_i$ and $l(s_k) = i_{2k} - i_{2k-1} + 1$. To graphically represent such a multisegment one just draws the segments $[i_{2k-1}, i_{2k}]$ over the base sequence. See Fig. 21 as an example. By a "run" we mean an interval $[i_1, i_2]$ such that for all $i \in [i_1, i_2]$, $x_i = 1$ and which is not contained in such a larger interval. In this work, such a binary sequence comes from a line in a 2D image through the notion of p-aligned pixel. A pixel is mapped to 1 if it is p-aligned with the line and 0 otherwise. But what follows in this section

could be applied to any binary sequence – not just those stemming from an image. Indeed, the factor $\#\mathcal{L}$ in the definition of $\mathrm{NFA}(s_1, \ldots, s_n)$ is common to all n. So one can compare the NFA of two different multisegments without knowing it. If one wants to segment a single binary sequence, it is altogether possible to take $\#\mathcal{L} = 1$.

Figure 20: An example of Type I sequence: the sequence has length 128 and 64 p-aligned points represented by dashes. This sequence has been generated by randomly drawing a subset of size 64 of a set of size 128 with a uniform law over all possible such subsets. It is referred to as sequence1.

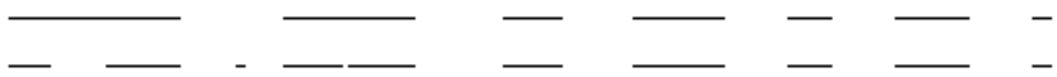

Figure 21: A Type II sequence (down) that has length 128 and 64 p-aligned points grouped into 10 runs and the most meaningful multisegment associated with this sequence (up), which is a 7-multisegment represented above. This base sequence is referred to as sequence2.

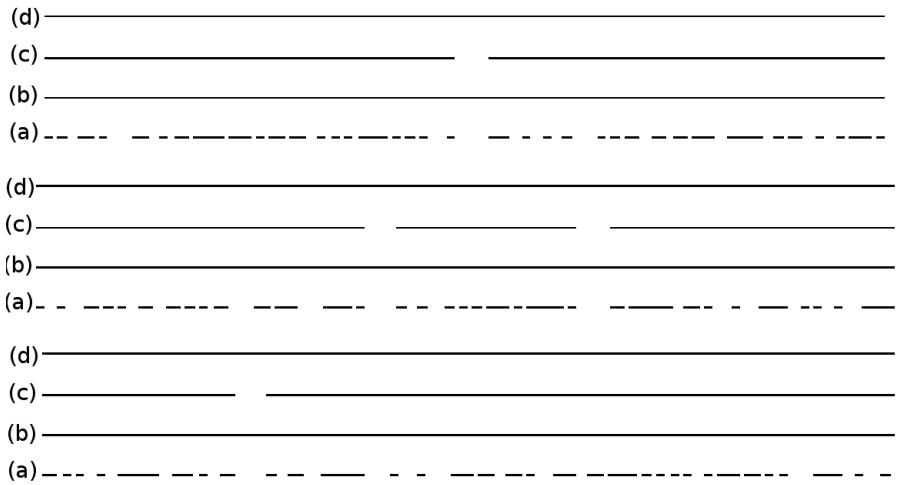

Figure 22: Segmentation of three different Type I sequences. In each of the three experiments we have: (a) The sequence. (b) The best associated 1-meaningful multisegment. (c) The 2-thresholds result with thresholds set to 10 for the minimal length and 5 for the gap. (d) The maximal 1-meaningful segments in Desolneux *et al.* sense. One can see that the Multisegment segmentation results in just one segment. This is the correct interpretation for sequences from the family sequence1 which are generated in a random and uniform way. See Fig. 23 for the same experiments with Type II sequences.

The following experiments show why Multisegment detection can be seen as an adaptive 2-threshold algorithm. The first experiment compares the segmentation obtained by the 2-thresholds algorithm (with thresholds 10 for the length and 5 for the gap, default values of the XHOUGHTOOL program [4]), the maximality segmentation of Desolneux *et al.* and Multisegment segmentation for two different classes of randomly generated sequences. The first class, Type I, is composed of sequences such as sequence1 (Fig. 20) that are generated by a Bernoulli process conditioned by its number of heads. The second class, Type II, is composed of sequences randomly drawn but organized by runs. Precisely, the number of runs is given as well as the

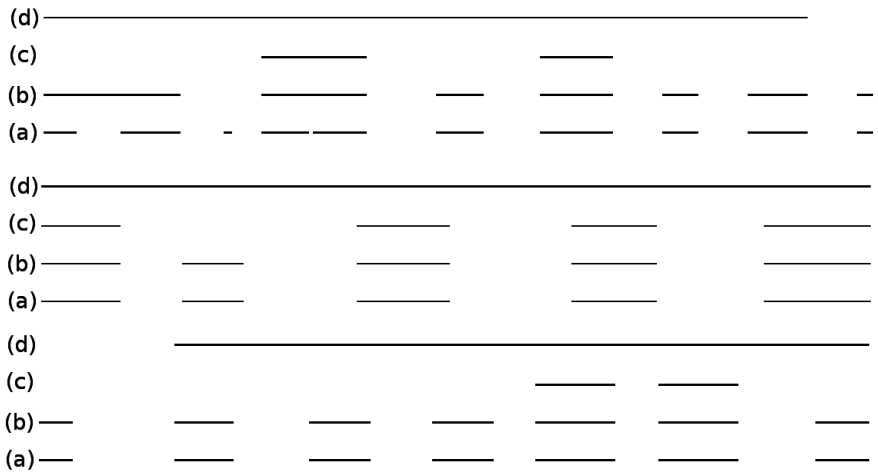

Figure 23: Segmentation of three different Type II sequences. In each of the three experiments we have: (*a*) Input sequence. (*b*) Result of Multisegment method. (*c*) Result of the 2-thresholds method. (*d*) Result of the DMM algorithm. Multisegments method segments the sequences in a way compatible with a bloc-wise interpretation, which is precisely how the sequences were generated.

total number of aligned points and the total number of points. Given this restrictions, the length of each run is randomly drawn. The same is done for the holes separating two consecutive runs. For example, sequence2 (Fig. 21) was obtained in this way with 128 total points, 64 of them aligned and organized in 10 runs.

The results are given in Fig. 22. For nearly all Type I sequences, the most meaningful multisegment is the whole line. This reflects the fact that these sequences were generated according to a Bernoulli process. The same observation holds for the Desolneux *et al.* maximality criterion. From time to time a gap larger than 5 appears in the sequence. Multisegments adapts to it in the sense that it does not break the line into pieces owing to this gap. The 2-thresholds algorithm instead doesn't adapt, being based on fixed thresholds, and breaks the line. For nearly all Type II sequences, the most meaningful multisegment is formed by the runs of the sequence themselves. For the 2-thresholds algorithm most of them disappear because they are not long enough. The maximality criterion of Desolneux *et al.* has a clear tendency to select the whole line as if the sequences were Type I.

This experiment shows that, as far as the Multisegment algorithm is concerned, a size 5 gap is sometimes enough to cut a sequence, as in sequence2, and sometimes not, as in sequence1. The most meaningful multisegment adapts. The same remark holds for the minimum length threshold. This clearly shows that no fixed thresholds can give a correct account of segments. The correct length of segments and gaps to be detected depends upon the interaction of all segments in the line, an effect which can only be predicted by a global criterion. One can also observe that the maximality criterion of Desolneux *et al.* gives more credit to the longest segments. The next experiment captures this effect in a simpler framework.

Lets say that A and B denote segments of same length t separated by a gap of length g. The graph on Fig 24 represents the decision taken by 2-threshold and Multisegment algorithms depending on the value of (t, g). There are three different regions for both algorithms: the region with a small gap length g where the segments are merged, the region with a large g where the segments are separated and a region where the segments have too small a length to be recognized as meaningful (thus are simply discarded). This overall organization is shared by both algorithms. Looking carefully one sees some differences. First, the shape of the region where segments are discarded is convex for the Multisegment algorithm and not for the 2-thresholds algorithm. Consider, for example, the segment joining the points $(t = 4, g = 1)$ to $(t = 4, g = 6)$ for the 2-threshold algorithm. The point $(t = 4, g = 1)$ is in the discard region because the total length of the two segments plus the gap is less than 10. The points $(t = 4, g = 2)$, $(t = 4, g = 3)$,

$(t = 4, g = 4)$ are in the merge region because the total length exceeds 10, but the gap is below the separation threshold. They are then discarded again at point $(t = 4, g = 6)$ because the composing segments are considered separately and have a length below the threshold. In that example, even if the length of each segment is fixed to a very small value while the gap grew, the 2-threshold algorithm moves from the discard region to the merge one and then goes back again to the discard region. Moreover, the boundaries between regions are not straight for the Multisegment algorithm: the thresholds adapts. For example, when the length of the surrounding segments is large, a smaller gap suffices to separate them.

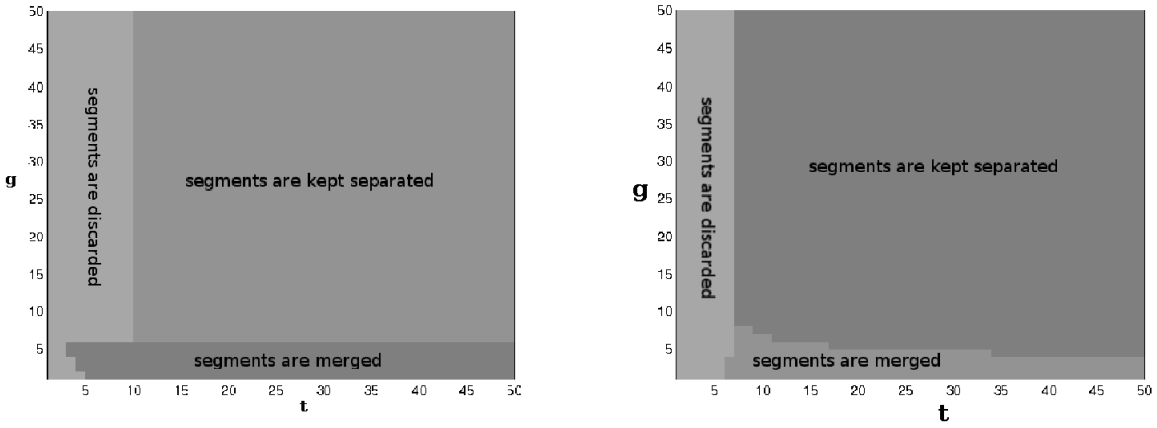

Figure 24: The three possible cases for two segments of length t separated by a gap of size g when processed by either the 2-threshold algorithm (left) or by the Multisegment algorithm (right): segments are discarded and disappear, segments merge, or segments are kept separated.

The following proposition proves that when the lengths of the two segments grow (as well as the image size that contains them) a gap of size 4 is enough to separate the two segments from the Multisegment point of view.

Proposition 4. *For all $p > 0$ and for an image of size $N \times N$ containing two collinear segments of length t (t p-aligned points among t) separated by a gap of size g (no p-aligned points at all) in a line L of length $l(L)$ such that $l(L) = O(t)$, if $g > 3$ then for large t, the 2-multisegment is the most meaningful event.*

Proof. Let us denote by A and B the two segments. Then

$$\mathrm{NFA}(A, B) = \#\mathcal{L} \cdot \binom{l(L)}{4}(t+1)p^{2t}$$

and

$$\mathrm{NFA}(\overline{AB}) = \#\mathcal{L} \cdot \binom{l(L)}{2} b(2t+g, 2t, p).$$

Clearly, $\binom{l(L)}{4} \leqslant l^2(L)\binom{l(L)}{2}$. Also, $b(2t+g, 2t, p) \geqslant \binom{2t+g}{2t}p^{2t}(1-p)^g$. By assumption, $l^2(L) = O(t^2)$, so that proving

$$t^3 = o\left(\binom{2t+g}{2t}(1-p)^g\right)$$

for $g > 3$ would be enough. The fact that $t^g = O(\binom{2t+g}{2t})$ finishes the proof. □

7.1 A Digression Over the Choice of NFA$(s_1, \ldots, s_n)$

When deciding whether a segment s should be considered meaningful or not, we computed its number of p-aligned points $k(s)$. To see how exceptional it is, we computed, under the *a contrario* randomness assumptions that there are uniformly and independently distributed gradient angles, the probability that the random variable $K(s)$ is larger than the one observed: $\mathbb{P}_{H_0}[K(s) \geqslant k(s)]$. It was shown that a good decision procedure to answer the question "is s meaningful?" is to compare $\#S \cdot \mathbb{P}_{H_0}[K(s) \geqslant k(s)]$ to 1.

One may ask why we use the event $\{K(s) \geqslant k(s)\}$ and not the event $\{K(s) \leqslant k(s)\}$ or $\{|K(s) - \frac{l(s)}{2}| \geqslant |k(s) - \frac{l(s)}{2}|\}$ or more generally $\{f(K(s)) \geqslant f(k(s))\}$ for a certain function f (f corresponding to $k \mapsto k$ in the first case, $k \mapsto -k$ in the second case and $k \mapsto |k - l(s)/2|$ in the last case)? All of the previous examples make sense: Looking for segments s such that $\mathbb{P}_{H_0}[K(s) \geqslant k(s)]$ is low leads to look for segments with such a high number of aligned points that they are unlikely to have arisen by chance. Looking for segments s such that $\mathbb{P}_{H_0}[K(s) \leqslant k(s)]$ is low leads to look for segments with such a low number of aligned points that they are unlikely to have arisen by chance. Looking for segments such that $\mathbb{P}_{H_0}\left[|K(s) - \frac{l(s)}{2}| \geqslant |k(s) - \frac{l(s)}{2}|\right]$ is low leads to look for segments with a number of aligned points either low or high enough to be likely to have happened by chance. Yet our interest is in segments having a high number of aligned points, and not a low number of aligned points or a "low or high" number of aligned points. Therefore, $\{K(s) \geqslant k(s)\}$ is the only relevant event in segment detection and corresponds to any increasing function f.

Unfortunately, for multisegments the situation is not as simple as for segments. The events to be considered have the form $\{f\big(K(s_1), \ldots, K(s_n)\big) \geqslant f\big(k(s_1), \ldots, k(s_n)\big)\}$ where f is a function from $\mathbb{R}^n$ to $\mathbb{R}$. Without loss of generality we shall assume $n = 2$ in the discussion. Let us consider four possible functions $f_1(k_1, k_2) = k_1 + k_2$, $f_2(k_1, k_2) = \min(\frac{k_1}{l_1}, \frac{k_2}{l_2})$, $f_3(k_1, k_2) = b(l_1, k_1, p)b(l_2, k_2, p)$ and $f_4(k_1, k_2) = k_1 - k_2$. To a function $f : \mathbb{R}^n \to \mathbb{R}$ we associate the function

$$U_f : (k_1, \ldots, k_n) \mapsto \mathbb{P}_{H_0}\left[f\big(K(s_1), \ldots, K(s_n)\big) \geqslant f\big(k(s_1), \ldots, k(s_n)\big)\right].$$

This maps each point $(k_1, \ldots, k_n)$ to the probability measure of the corresponding level set of f. Fig. 25 represents U_f for f_1, f_2, f_3 and f_4 when $n = 2$. The lower the value of U_f the darker the point. Some level lines are also represented. The good events to consider for segment detection are of the form $\{f(K(s)) \geqslant f(k(s))\}$ with f increasing. For multisegments, the good events are the ones having the form $\{f(K(s_1), \ldots, K(s_n)) \geqslant f(k(s_1), \ldots, k(s_n))\}$ where f is non-decreasing with respect to the order defined by $(k_1, \ldots, k_n) \leqslant (k'_1, \ldots, k'_n) \Leftrightarrow \forall i \; k_i \leqslant k'_i$. One can check that f_1, f_2 and f_3 share this property. In all cases we shall get meaningful 2-multisegments when both k_1 and k_2 are large enough. The function f_4 is not an appropriate criterion for 2-multisegments, in the same way that $\{K(s) \leqslant k(s)\}$ was not relevant for segments: it induces a low probability for multisegments with large k_1 but small k_2. It turns out that f_1, f_2 and f_3 have very different properties and this is related to the concavity or lack of concavity of their level lines.

The level sets associated with f_1 and f_2 are convex (see Fig. 25) whereas the level sets associated with f_3 are concave. As a byproduct, when comparing a level set of f_1 or f_2 and a level set of f_3 for a given probability, the former contains smaller k_1, k_2 in the center whereas the latter contains smaller k_1, k_2 in the periphery (see Fig. 26).

The case of inhomogeneous multisegments – the ones with k_1, k_2 far from the center – gives us a clue about what criterion among the three is best suited for meaningful multisegments. Indeed, inhomogeneous multisegments should be favored. The inhomogeneity of its parts encourages viewing their parts as separated. This simple requirement rules out f_1 and f_2 as shown in Fig. 26. We did the following experiment to support this claim. An inhomogeneous n-multisegments formed by two parts was designed. The first part was dotted and the other plain. Using the criterion f_2 we interpreted it as a unique segment whereas when using criteria f_1 or f_3 it was viewed as a 2-multisegment. When the inhomogeneity is bigger, only the criterion f_3 can separate the compounding segments as shown in Fig. 27 and 28.

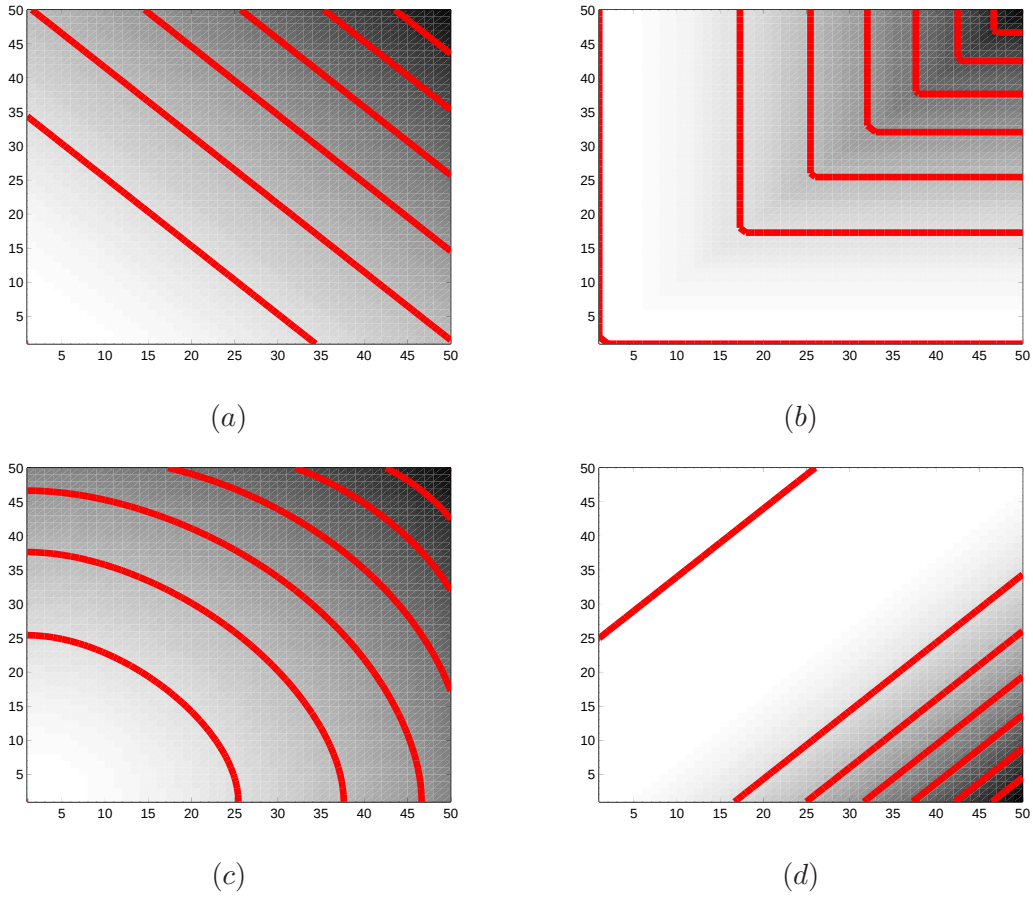

Figure 25: Illustration of four different ways to define NFA(s_1, s_2). The graph represents the decision space for a 2-multisegment where both segments are 50 pixels long. Each one can have a number of aligned points between 0 and 50. The abscissa represents the number of aligned points of the first segment; the ordinate represents the number of aligned points of the second segment. The lines represent different families of decision frontiers. They are defined as the level sets and level lines of functions: (a) U_{f_1}, takes the mean density of aligned points on both segments as criterion. (b) U_{f_2} uses the minimal density of aligned points between the two segments. (c) U_{f_3}, the criterion we propose. It permits to accept a difference on the aligned points density between the segments, to some degree. (d) U_{f_4}, a useless example.

This is why f_3 was chosen in our NFA$(s_1, \ldots, s_n)$ design. For other applications it may be interesting to choose a different criterion. For example, for region merging in [22, 38, 23], Igual, Preciozzi and Almansa chose the criterion f_1.

We now prove a result weaker than the concavity of the level sets associated with f_3.

Lemma 1. Let $b(l, k, p)$ stand for the probability that a variable following the binomial law of parameters l and p be greater or equal to k. Then for any $k \in [0, l]$ and h such that $k + h \in [0, l]$ and $k - h \in [0, l]$,

$$b(l, k + h, p) \cdot b(l, k - h, p) \leqslant b^2(l, k, p)$$

Proof. This amounts to proving that $\log b(l, k, p)$ is concave in k. We abbreviate b_k for $b(l, k, p)$, p_k for $\binom{l}{k} p^k (1-p)^{l-k}$ and $h_k = \frac{p_k}{b_{k+1}}$. Then one can check that

$$h_{k-1} = \frac{1-p}{p} \frac{k}{l-k+1} \frac{h_k}{h_k + 1}.$$

One can define $h_l = +\infty$, so $h_{l-1} < h_l$. Also, the recursion formula can be expressed as $h_{k-1} = \frac{1-p}{p} \frac{k}{l-k+1} f(h_k)$ with $f(t) = \frac{t}{t+1}$. Since f is increasing and $\frac{k}{l-k+1} < \frac{k+1}{l-k}$, it can be shown

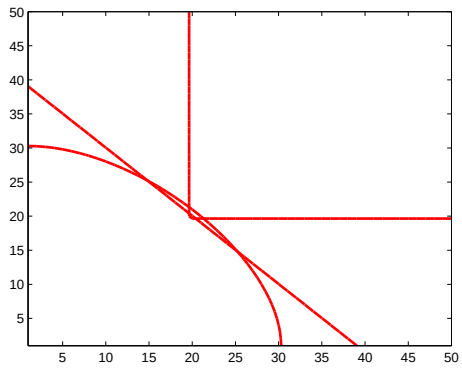

Figure 26: Level lines of U_{f_1}, U_{f_2} and U_{f_3} corresponding to level sets with the same probability.

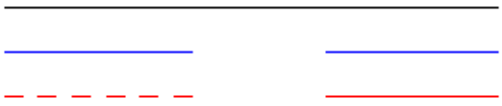

Figure 27: An inhomogeneous 7-multisegment (bottom) is seen as a 2-multisegment with criteria f_1 and f_3 (middle), but as a single segment by criterion f_2 (top).

Figure 28: Same experiment as in Fig. 27 but for an even larger multisegment: only the criterion f_3 decides to interpret it as a 2-multisegment (middle); criteria f_1 and f_2 view it as a segment (top).

that $h_{k-1} \leqslant h_k = \frac{1-p}{p}\frac{k+1}{l-k} f(h_{k+1})$. An induction proves that h_k is an increasing sequence. Hence $\frac{b_k}{b_{k+1}} = 1 + h_k$ also forms an increasing sequence. So $\frac{b_{k-1}}{b_k} \leqslant \frac{b_k}{b_{k+1}}$ and $b_{k+1}b_{k-1} \leqslant b_k^2$. Using the same increasing sequence one step further in both directions one gets $\frac{b_{k-2}}{b_{k-1}} \leqslant \frac{b_{k-1}}{b_k} \leqslant \frac{b_k}{b_{k+1}} \leqslant \frac{b_{k+1}}{b_{k+2}}$ which shows that $b_{k+2}b_{k-2} \leqslant b_{k+1}b_{k-1}$. Repeating h times gives the result. □

8 Implementation

This section deals with computational issues. Multisegment detection is split into three loops. The first loop scans all lines in the image. It has complexity $O(M+N)^2$ for an image of size $M \times N$. The second loop searches for the most meaningful multisegment associated with the pending line, if there is one. A brute force approach would require $\sum_n \binom{L}{2n}$ steps for a line of length L and would be unfeasible. We shall show that a dynamic programming (*cf.* [4]) setup can achieve the same goal as a brute force search but with complexity $O(r^3)$, where r denotes the number of runs in the pending line. The last loop is devoted to the exclusion principle which aims to avoid redundant detections. Indeed, since the best multisegment is chosen line per line, some comparisons between lines must be performed to avoid detecting the same segment in several close by lines, due to the angular tolerance.

8.1 Dynamic Programming Algorithm

One can restrict the search for n-multisegments to the ones where each of the n segments starts at a run start and ends at a run end. Otherwise, one segment would start or end with a non-aligned point. In such cases a shorter segment without that non-aligned point would have a smaller NFA and would be preferred [14, 12].

Let us denote by $M(n,i)$ the most meaningful n-multisegment using only the first i runs on the line. We shall show that $M(n,i)$ can be found knowing $M(n,i-1)$ and $M(n-1,k)$ for k in $[n-1, i-1]$.

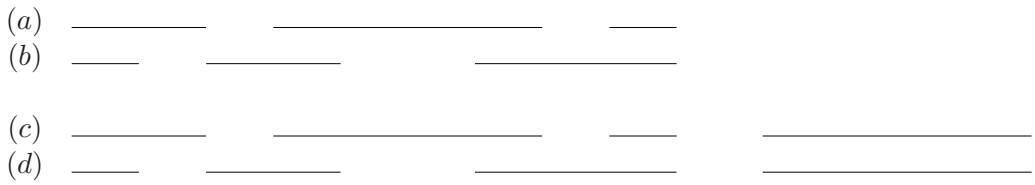

Figure 29: (a) and (b) are two different 3-multisegments for the same sequence. (c) and (d) are 4-multisegments where the same fourth segment was added to the multisegment on (a) and (b). If $\mathrm{NFA}_a < \mathrm{NFA}_b$ then $\mathrm{NFA}_c < \mathrm{NFA}_d$. This fact permits to solve the n-multisegment problem using the solution for the $(n-1)$-multisegment.

This fact is illustrated using two 3-multisegments associated with the same sequence, as shown in Fig. 29 (a) and (b). Their NFA are:

$$\mathrm{NFA}_a = \#\mathcal{L} \cdot \binom{l(L)}{6} b\big(l(s_1^a), k(s_1^a), p\big) \prod_{i=2}^{3} \big(l(s_i^a)+1\big)\, b\big(l(s_i^a), k(s_i^a), p\big)$$

and

$$\mathrm{NFA}_b = \#\mathcal{L} \cdot \binom{l(L)}{6} b\big(l(s_1^b), k(s_1^b), p\big) \prod_{i=2}^{3} \big(l(s_i^b)+1\big)\, b\big(l(s_i^b), k(s_i^b), p\big).$$

If one adds the same extra segment to both multisegments, one gets the multisegments shown in Fig. 29 (c) and (d), with NFAs:

$$\mathrm{NFA}_c = \#\mathcal{L} \cdot \binom{l(L)}{8} b\big(l(s_1^a), k(s_1^a), p\big) \prod_{i=2}^{3} \big(l(s_i^a)+1\big)\, b\big(l(s_i^a), k(s_i^a), p\big) \cdot \big(l(s_4)+1\big)\, b\big(l(s_4), k(s_4), p\big)$$

and

$$\mathrm{NFA}_d = \#\mathcal{L}\cdot\binom{l(L)}{8} b\big(l(s_1^b),k(s_1^b),p\big)\prod_{i=2}^{3}\big(l(s_i^b)+1\big)\,b\big(l(s_i^b),k(s_i^b),p\big)\cdot\big(l(s_4)+1\big)\,b\big(l(s_4),k(s_4),p\big).$$

The multiplicative form of the NFA implies that the NFAs of multisegments (a) and (c) share common factors: only the factor $\binom{l(L)}{2n}$ and the term $\big(l(s_4)+1\big)\,b\big(l(s_4),k(s_4),p\big)$ differ. If $\mathrm{NFA}_a < \mathrm{NFA}_b$ then $\mathrm{NFA}_c < \mathrm{NFA}_d$, since

$$b\big(l(s_1^a),k(s_1^a),p\big)\prod_{i=2}^{3}\big(l(s_i^a)+1\big)\,b\big(l(s_i^a),k(s_i^a),p\big) < b\big(l(s_1^b),k(s_1^b),p\big)\prod_{i=2}^{3}\big(l(s_i^b)+1\big)\,b\big(l(s_i^b),k(s_i^b),p\big).$$

This principle generalizes well. The segment that covers from run j to run i is denoted by $\overline{j,i}$. If one wants $M(n+1,i) = (s_1,\ldots,s_{n+1})$, the best $(n+1)$-multisegment using the first i runs, one considers all the possible values for s_{n+1}. The segment s_{n+1} can be: $\overline{i,i}$, $\overline{i-1,i}$,..., $\overline{n+1,i}$. It cannot be $\overline{n,i}$ or more generally $\overline{k,i}$ with $k \leqslant n$ because $(s_1,\ldots,s_n)$ has to be a non-degenerate n-multisegment. It is also possible that $s_{n+1} = \overline{k,j}$ with $j < i$, *i.e.*, the run i is not part of the last segment. Since the NFA factorizes, in the previous cases one can tell that $(s_1,\ldots,s_n)$ must be $M(n,i-1)$ if $s_{n+1} = \overline{i,i}$, $M(n,i-2)$ if $s_{n+1} = \overline{i-1,i}$, ..., $M(n,n)$ if $s_{n+1} = \overline{n+1,i}$, and $M(n+1,i)$ if $s_{n+1} = \overline{k,j}$ with $j < i$. This means that to find $M(n+1,i)$, one needs to know first the $i-n+1$ multisegments $M(n,i-1)$, ..., $M(n,n)$ and $M(n+1,i)$: With $M(n,n)$ to $M(n,i-1)$ and $M(n+1,i-1)$ already computed $M(n+1,i)$ can be computed with complexity $O(i-n+1)$. Using this method the following table is computed, and the configuration with smaller NFA is chosen.

$M(1,1)$	$M(1,2)$	...	$M(1,r-1)$	$M(1,r)$
	$M(2,2)$	$\ddots$	$\ddots$	$M(2,r)$
		$\ddots$	$\ddots$	$\vdots$
			$M(r-1,r-1)$	$M(r-1,r)$
				$M(r,r)$

All in all, the best multisegment for a line with r runs can be computed with complexity $O(r^3)$. A pseudo-code for the algorithm is shown in Algorithm 8. We note $M(n,i)$.part the part of the NFA that can be re-used:

$$M(n,i).\mathrm{part} = b\big(l(s_1),k(s_1),p\big)\prod_{i=2}^{n}\big(l(s_i)+1\big)b\big(l(s_i),k(s_i),p\big)$$

Some simplification was done for clarity. The first row of the table, $M(1,1)$ to $M(1,r)$, should be computed with some slight changes. The term $(l_{\overline{j+1,i}}+1)$ on lines 9 and 11 must be omitted for $n=1$ and the undefined components should be considered null. For example, $M(-1,i).\mathrm{part} = 1$ and $M(n,n-1).\mathrm{Segs} = []$.

8.2 Exclusion Principle, complexity

Applying the previous algorithm to search for 1-meaningful multisegment in each line of the image provides a list of multisegments that is to a large extent redundant. Indeed, if there is a segment somewhere in the image it is likely that several neighboring lines share it. The problem was the same for the maximal segments of Desolneux *et al.* These authors and later Almansa proposed the exclusion principle (*cf.* algorithm 6) as a work-around in [1, 12]. Algorithm 9 is the pseudo-code that corresponds to the multisegment exclusion principle. In analogy with the segment case, the procedure RemoveMultisegmentPoints(l,x) prevents the pixels belonging to a thick neighborhood of the best multisegment interpretation of line l from being used afterwards when computing the NFA of other lines.

Algorithm 8: MULTISEGMENT DETECTION

input : A binary array Line
output: A list of integers Segs

$R \leftarrow$ EXTRACT-RUNS(Line);
$r \leftarrow$ NUMBER-RUNS(Line);
Segs $\leftarrow$ [];
nfa $\leftarrow \infty$;
for $n \leftarrow 1$ **to** r **do**
  for $i \leftarrow n$ **to** r **do**
    $M(n,i) \leftarrow M(n,i-1)$;
    for $j \leftarrow n+1$ **to** i **do**
      if $M(n-1,j-1).part \cdot (l_{\overline{j,i}}+1) \cdot b(l_{\overline{j,i}}, k_{\overline{j,i}}) < M(n,i).part$ **then**
        $M(n,i)$.Segs $\leftarrow M(n-1,j-1)$.Segs $\cup\ \overline{j,i}$;
        $M(n,i)$.part $\leftarrow M(n-1,j-1)$.part $\cdot (l_{\overline{j,i}}+1) \cdot b(l_{\overline{j,i}}, k_{\overline{j,i}})$;
        $M(n,i)$.nfa $\leftarrow \#\mathcal{L} \cdot \binom{l(Line)}{2n} \cdot M(n,i)$.part;
      end
    end
    if $M(n,i).nfa <$ *nfa* **then**
      nfa $\leftarrow M(n,i)$.nfa;
      Segs $\leftarrow M(n,i)$.Segs;
    end
  end
end

Algorithm 9: Multisegment Exclusion Principle

input : An image x, a list of lines L
output: A list of multisegments Mout.

Mout $\leftarrow$ [];
while `Size`$(L) > 0$ **do**
  foreach $l \in L$ **do**
    Compute multisegment NFA(l,x);
    if NFA$(l,x) \geqslant 1$ **then**
      `Remove` l from L
    end
  end
  `Find` $l_{min} \in L$ with the lowest NFA;
  if NFA$(l_{min},x) < 1$ **then**
    REMOVEMULTISEGMENTPOINTS(l_{min},x);
    `Add` l_{min} best multisegment interpretation to Mout;
    `Remove` l_{min} from L;
  end
end

The complexity of the whole Multisegment detection algorithm is $O(N^5)$ for a square image of size N. Here we are considering $O(N^2)$ number of lines on the image and assuming the number of runs per line proportional to N. With the dynamic programming algorithm running at complexity $O(r^3)$, one gets the global complexity. Thus, the overall algorithm speed performance is poor. It cannot be used for online applications so far. For example, it takes about 3 minutes to process a 256 by 256 image.

Improvements for the three loops of the algorithm will be the subject of further investigation. In particular, the line loop should concentrate only on worthwhile lines. The current algorithm runs the 1D segmentation in all the lines in the image. Fernandez [17] improved the exclusion principle algorithm: by keeping the segments in an highly ordered structure, his algorithm minimize the number of NFA to recompute at each iteration.

9 Results

The Multisegment algorithm was tested on hundreds of images. In this section we will show some results and experiments that illustrate some of its properties. We shall compare the results obtained by Multisegment detection with the ones obtained via the HTM method and the DMM algorithm.[2] More experiments can be found at `http://www.cmla.ens-cachan.fr/Utilisateurs/grompone/multi.htm`.

9.1 Detection in Presence of Noise

The Multisegment algorithm, just like the DMM algorithm, controls the number of false detections. The first experiment runs the Multisegment detector on images that are pure white noise. Thus each detection made will be a casual, false detection. Segments were also searched for using HTM. Table 1 clearly shows that the number of false detections is controlled when using the Multisegment detector and not when using HTM.

Image Size	32×32	64×64	128×128	256×256	512×512
HTM	8.0	20.0	39.8	150.2	671.4
Multisegment	0.0	0.0	0.0	0.0	0.0

Table 1: Performance of the Multisegment detection method and the HTM on a noise image. Mean number of detections in 10 images of Gaussian white noise.

This table is confirmed by performing segment detection in an artificially noisy image. On Fig. 30 one can see how the detections are degraded for both methods as white noise is added. The number of detections of HTM grows quickly as more noise is added. Inversely the number of detections by the Multisegment method decreases as noise is added. When the image is dominated by noise no detection should be possible anymore. Indeed in such a case the gradient orientations become random uniform independent variables.

Now, HTM is taking here advantage of its use of the Canny edge detector, which includes a denoising step. To make the comparison fairer, Fig. 31 shows the same experiments where an unitary Gaussian filter was applied before both detection algorithms were tested. These figures show that HTM is quite sensitive to noise. Thus, even when the noise level is reduced by the Gaussian filter, detection results remain poor. This is not the case for Multisegments where one sees that a decrease in the noise level improves the results.

[2]For the experiments in this paper related to HTM we have used the XHoughtool package, freely available on the Internet, without tuning any parameters [20].

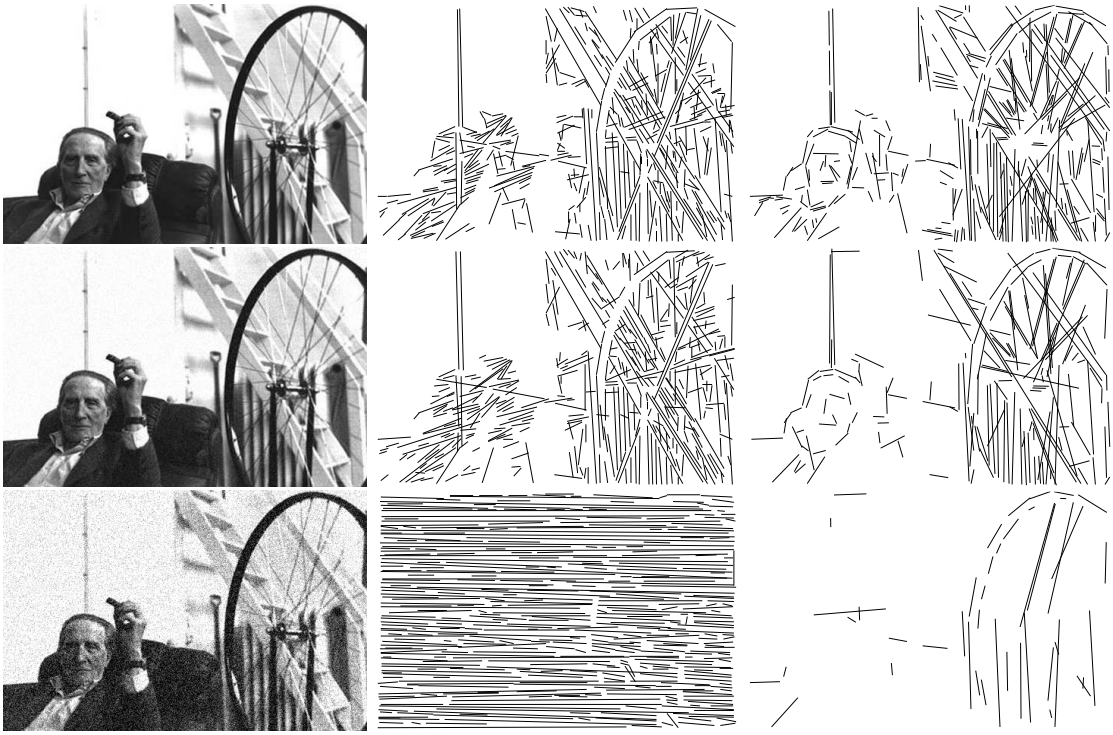

Figure 30: Performance of Hough Transform Method and Multisegments in a noisy image. Left: Image. Middle: Segments found by HTM. Right: Segments found by Multisegment detection. From top to bottom: Original image, then the same image with added noise, $\sigma = 8$ and $\sigma = 32$. The results of both algorithms are degraded as noise level increases. But the behavior is different: HTM gives a big number of false detections while Multisegments gives less but true detections. Its performance is restored by adding a previous denoising step (see Figure 31.)

9.2 Distance Between Used Points

The theory of *a contrario* segment detection by the Helmholtz principle entails a control of the number of false detections in white noise. As shown, the number of false detections can be bounded assuming independent gradient orientation for each pixel. This independence relies on a trade-off between the support of the discrete operator chosen to approximate the gradient and the discretization resolution. If the gradient is computed using a 2×2 support

$$\nabla u(i,j) = \begin{pmatrix} \frac{u(i+1,j)-u(i,j)+u(i+1,j+1)-u(i,j+1)}{2} \\ \frac{u(i,j+1)-u(i,j)+u(i+1,j+1)-u(i+1,j)}{2} \end{pmatrix},$$

the algorithm should only consider segment points at a distance larger than two in order to guarantee the independence of their gradient orientation. One way to do this is to use the digital straight segment discretization between the considered couple of end points and then to drop one point out of two.

Thus in the Desolneux *et al.* theory, gradient independence is achieved by dropping half the image information, as we use only half the points, which may lead to an under-detection. This could be avoided at the price of a more complex background model. Even if there is dependence in gradient information of adjacent points, the number of false detections can still be controlled modeling this dependency in white noise. Unfortunately a theoretical bound for the number of false detections under the dependency model is hard to get. However, numerical experiments on white noise show that the number of false detections does not blow up when all points in the discretization are kept and we assume independence. This means that the Helmholtz principle (no detection in noise) is still valid without any down-sampling. In order to illustrate the effect

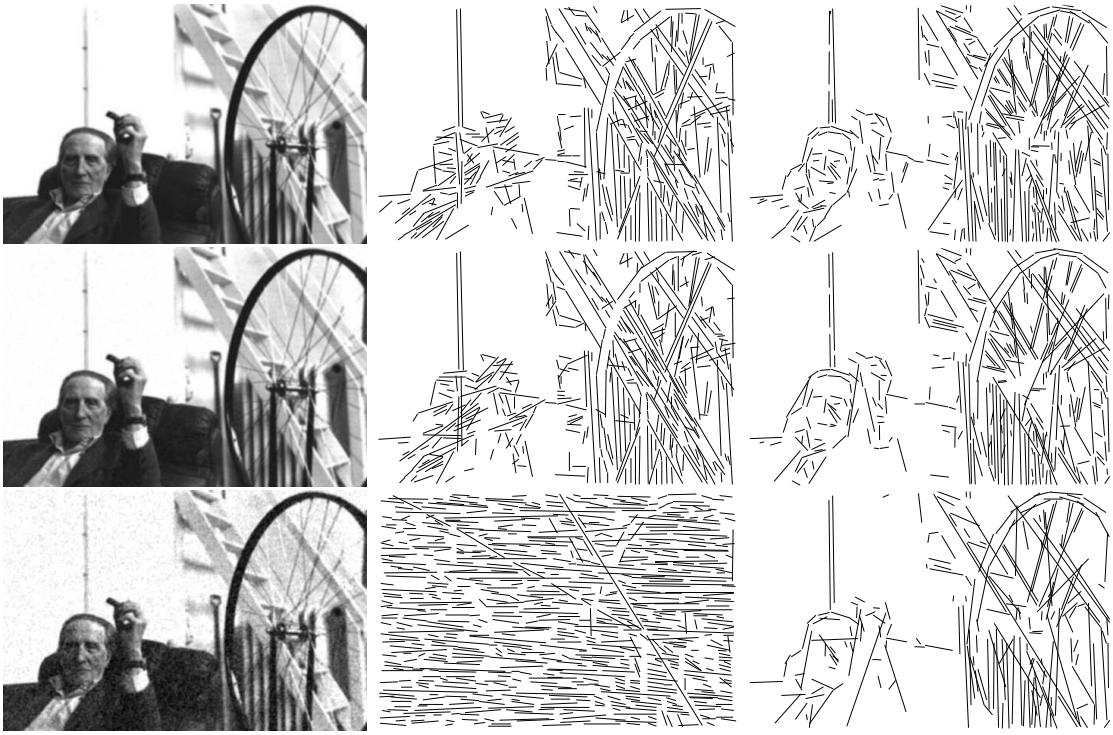

Figure 31: Performance of both algorithms in a prefiltered image. Left: Image filtered with a Gaussian filter of $\sigma = 1$. Middle: Segments found by HTM. Right: Segments found by Multisegment detection. From Top to Bottom: Original image, then the same image with added noise, $\sigma = 8$ and $\sigma = 32$.

of the use of all points instead of one out of two, Fig. 32 shows the results on one image with both approaches. Some details are only obtained using all points.

We will show now, experimentally, that the number of false detections doesn't blow up when all points are used in the computation. To this aim we estimated the number of false detections in white noise. White noise images were generated and the gradient orientation computed at each point using a 2×2 window. Each segment with integer coordinates tips was tested, thus scanning N^4 segments on a $N \times N$ image.

Three variants of the experiment were done. The first detection (labeled "1/2 points") was done in the classical way: with the points far enough to assure independence. The second experiment (labeled "all points, independent gradient") started with a $2N \times 2N$ white noise image. Then the gradient was computed in the usual way. Finally, a $N \times N$ image of independent gradient orientations was obtained by sub-sampling. Segment detection was performed using all points (without down-sampling). This experiment was the nearest to the theoretical computation of the bounds, where an $N \times N$ image of independent gradient orientation was assumed. The last experiment (labeled "all points, dependent gradient") was done computing the gradient from a $N \times N$ white noise image and using all points in the segment detection.

Table 2 shows the mean number of detections for the three cases and for different values of ε. The number of detections using all points has the same order of magnitude as in the other two cases. This experiment suggests that the same thresholds can be used without a degradation of the number of false detections. In other terms Helmholtz principle nearly holds without down-sampling. This is not that surprising. In fact it can be checked that gradient orientations at neighboring pixels computed with a 2×2 mask are fairly de-correlated in white noise (the correlation between neighbor angles is about 0.2).

The last experiment was done with the DMM algorithm. We expect the Multisegments detection to show the same controlled number of false detections using adjacent points. However,

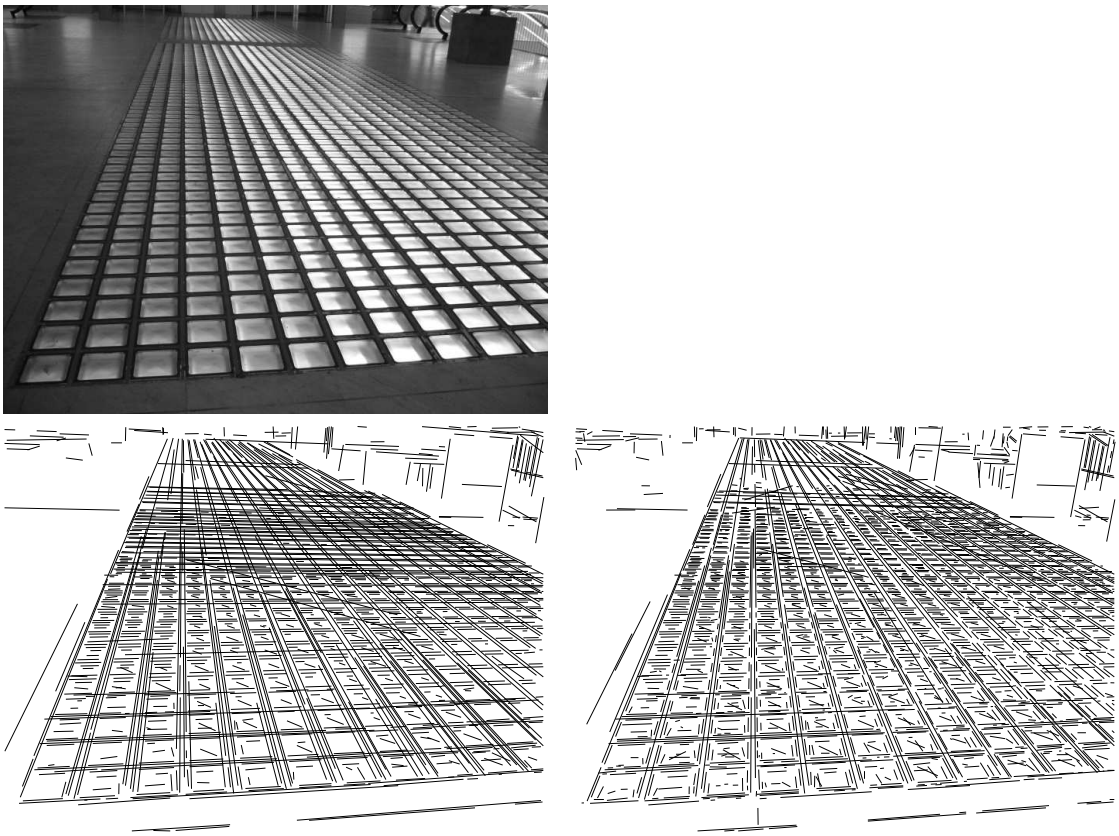

Figure 32: Illustration of the effect of Multisegment detection using one point out of two or all points. Up: CDG Image. Down-left: multisegments using one point out of two. Down-right: multisegments using all points. In the second case the tilling is better resolved. No false detection occurs.

ε	100000	10000	1000	100	10	1
1/2 points down-sampling	45988	3797	295.3	32.26	2.72	0.20
all points, independent gradient	50590	4291	395.9	40.06	3.96	0.41
all points, dependent gradient	50976	4300	388.2	37.51	3.61	0.25

Table 2: Measuring the sensitivity of DMM algorithm to the pixel independence on white noise. Mean number of detections in 100 white noise images of size 50×50 for different values of ε. It can be seen that the "all points" approach gives results similar to the theoretical model. In fact in white noise gradient orientations computed with a 2×2 mask are fairly independent.

an exhaustive count of all the meaningful multisegments would have been a computational burden. Instead, the mean number of meaningful multisegments after the exclusion principle was computed and is shown in table 3. The result indicates clearly that the false detection rate is fully under control without the down-sampling applied by DMM algorithm.

9.3 Experiments on Images

We now illustrate the performance of HTM, DMM and the Multisegment algorithms on three representative images of increasing complexity in Figs. 33, 34 and 35.

The image of Fig. 33 is a line drawing extracted from a Kanizsa book [26], representing a cube partially hidden by three white bars. We see that HTM produces many expected segments

ε	100000	10000	1000	100	10	1
Multisegments	74.19	19.74	3.78	0.81	0.17	0.03

Table 3: Sensitivity of Multisegments detection algorithm to pixel independence. Mean number of detections in 100 white noise images of size 50×50 for different values of ε. This number is computed after the exclusion principle has been applied.

but not all of them. Some important segments are absent (see the upper right corner) due to the fixed thresholds on segment and gap minimum lengths. Moreover, the position and angle of the detected segments are not always correct. The DMM algorithm gives good angle precision but fails to detect the right segments. Some are missing (see the upper right corner) and some are incorrectly joined into bigger ones (*eg.* the large diagonal). These effects have been predicted and explained in Section 6 and Fig. 17. Such imprecisions would be particularly disturbing when trying to interpret the scene and, for instance, to detect the hidden cube. In that sense, the Multisegment approach gives a result which raises hopes to obtain an automatic interpretation of such a simple scene. We insist that DMM algorithm and HTM would be a dead end and not at good starting point to such an interpretation. Notice that the detection of the small segments on the upper right corner is due to adaptive length thresholds implicitly computed in the algorithm. The absence of the long joined segments is due to the adaptive gap threshold, also implicit in the algorithm.

On Fig. 34 we see a portion of an Escher drawing. This image is more complex because we have texture in addition to lines. HTM produces many false detections due to the fine structure of the engraving, which are wrongly interpreted as segments. In this example DMM and Multisegment algorithms give comparable results.

Fig. 35 shows the results of all three algorithms for a classical INRIA image. This is no more a drawing but a gray level image of a typical interior scene. Both, HTM and DMM results lack some segments, incorrectly join some segments and give imprecise position and angle in some cases. The Multisegment detector gives a much better result in terms of localization of segment accuracy.

Note that the Multisegment detection gives both: the explanation of the scene in terms of segments, and a more global description in terms of aligned segments (multisegments). The image of Fig. 36 is a line drawing extracted from a Kanizsa book [26]. DMM algorithm produces a very good explanation in terms of global alignment but fails to detect individual segments. Fig. 36 bottom-right shows the global structure of multisegments.

Fig. 37 shows some results on natural images containing many segments. As all the examples shown in this paper, no parameter tuning was used. Note that a segment explanation of this type of images is very descriptive. Fig. 38 illustrates the performance of the Multisegment algorithm on natural images that has less segments. Even for this type of images, the description in terms of segments is informative. For a richer semantic description of the scene we need to detect other gestalts and solve the problem of gestalts interaction.

10 Conclusion

In this paper we proposed a new line segment detection algorithm based on the Helmholtz principle. The method is definitely an extension of the Desolneux *et al.* approach, since it delivers a two scales result, one of them being the global alignments and the other one their segmentation into segments. As in the DMM original method, no parameter tuning is needed and the number of false detections is controlled. In our opinion, this correct interpretation in terms of line segments of an image opens the way to many applications. All comparative experiments indeed indicate that inaccuracy in this detection stops short of giving any way to interpret images. For instance in Figure 36 it is clear that only the accurate multisegment

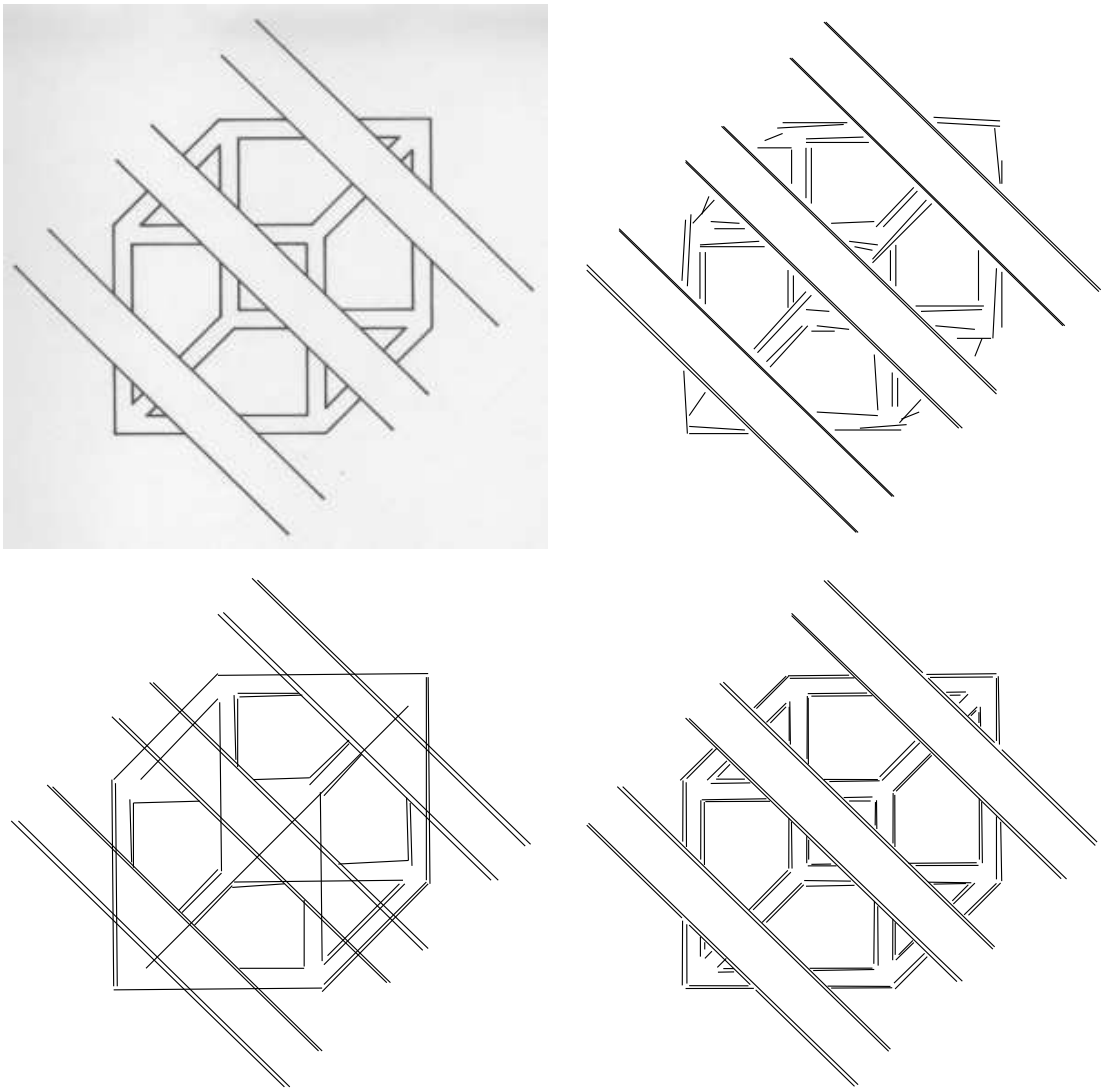

Figure 33: A comparison of the segments found by HTM, DMM and Multisegment algorithms on a Kanizsa drawing (up-left). Up-right: segments found by HTM. Down-left: segments found by DMM algorithm. Down-right: detections found by Multisegments, using all points. HTM gives many segments but not all those we expect. Some segments are missing (see the upper right corner) and the position and angle of the detected segments are inaccurate. DMM algorithm gives good angle precision but fails to detect all the right segments. Some are missing, and some are incorrectly joined into bigger ones (*eg.* the big diagonal). These effects have been predicted and explained in Section 6 and Figure 17. The ensuing misinterpretation would be particularly disturbing when trying to construct the hidden cube. Finally, the Multisegment approach gives the expected result. The Multisegment result also gives back the global alignments (see Figure 36 for an illustration.)

interpretation raises hopes of interpreting correctly the drawing, simple though it is.

We compared our approach with the well known Hough Transform Method and showed that the main differences are: a principled way to set the threshold of meaningful segments; and the fact that the length and gap between detected segments adapts to the image contents. In fact, Hough transforms have been designed as a shortcut to the problem. They are $O(N^3)$ algorithms, while DMM algorithm is basically $O(N^4)$ and Multisegments method would be up to $O(N^5)$ where N is the image perimeter. Indeed, we hope to have demonstrated that only a thorough exploration of all segments can lead a fully reliable and complete geometric interpretation. This fact is not in contradiction with neurophysiological data, which indicate that billions of neurons

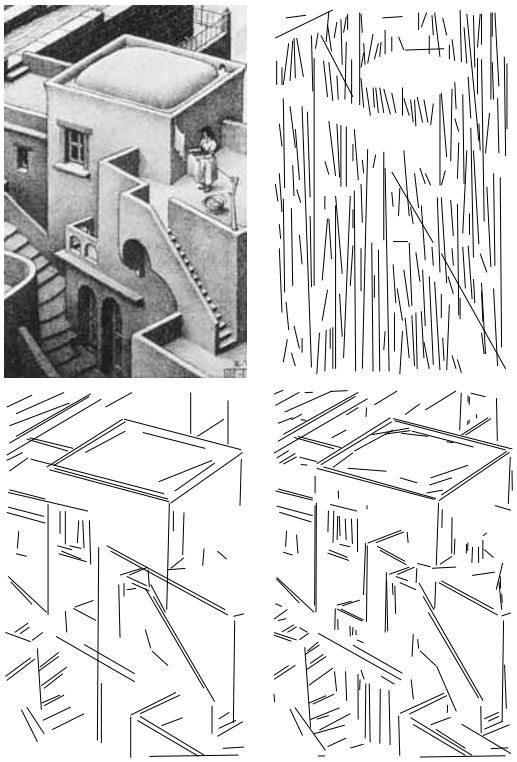

Figure 34: Comparison of the segments found by HTM, DMM and Multisegment algorithms on a detail of Escher's lithograph "Waterfall." Such drawings or etchings are a good test because they don't have the same structure as photographs. Up-left: Image. Up-right: segments found by HTM. Down-left: segments found by the DMM algorithm. Down-right: multisegments using all points. The HTM produces many false detections due to the fine structure of the engraving, which are wrongly interpreted as segments. In this case DMM and Multisegment algorithms give similar results, with a little advantage for the Multisegment detector.

are involved at the earliest stage of primate vision. As we pointed out, there are N^4 segments and their full exploration is $O(N^5)$. The hypercolumn organization indicates the early neural stage to involve not less than four parameters or dimensions, namely position, orientation and scale.

We have shown how dynamic programming techniques actually reduce the computational burden to $O(N^5)$. Actually we think that adequate multiscale techniques might lead to a hopefully real time algorithm, necessarily involving some heuristics, however. This was not the object of the present article, which was to show that a fairly simple and parameterless stochastic model gives back a complete geometric account of the scene, regardless of any computational consideration.

We would also like to point out that the core of the algorithm is nothing but a general binary sequence segmentation algorithm, looking for any meaningful runs or sets of runs.

Viewed as part of a general program aimed at fully explaining an image as a collaboration of different partial gestalts, multisegments can be seen as an attempt to cope with interactions between segments.

Current work is focused on a real time version of the algorithm and on the use of multisegments in automatic scene interpretation.

Acknowledgments

The authors are indebted to their collaborators for many remarks and corrections, and more particularly to Andrès Almansa, Enric Meinhardt, Guillermo Sapiro, and Sylvain Arlot. The

Figure 35: A comparison of the segments found by HTM, DMM and Multisegment algorithms on the INRIA office image. Up-left: Image. Up-right: segments found by HTM. Down-left: segments found by the DMM algorithm. Down-right: multisegments using all points. In this case HTM gives more or less all the segments present on the image. Nevertheless the angle and position of many of them is not precise, see near the chair. DMM algorithm gives a good line localization but a wrong decoupling into segments. Multisegments gives a fairly complete result with no false detection in flat (and noisy) image regions. This is in accordance with Helmholtz principle: Flat regions of an image are essentially a white noise.

research was partially financed by the ALFA project CVFA II-0366-FA, the Centre National d'Etudes Spatiales, the Office of Naval research under grant N00014-97-1-0839 and Direction Générale de l'Armement. Many thanks to Bernard Rougé, Olivier Goretta and Wen Masters for their interest and constant support.

References

[1] A. Almansa. *Echantillonnage, interpolation et détection. Applications en imagerie satellitaire.* PhD thesis, ENS Cachan, 2002.

[2] A. Almansa, A. Desolneux, and S. Vamech. Vanishing point detection without any a priori

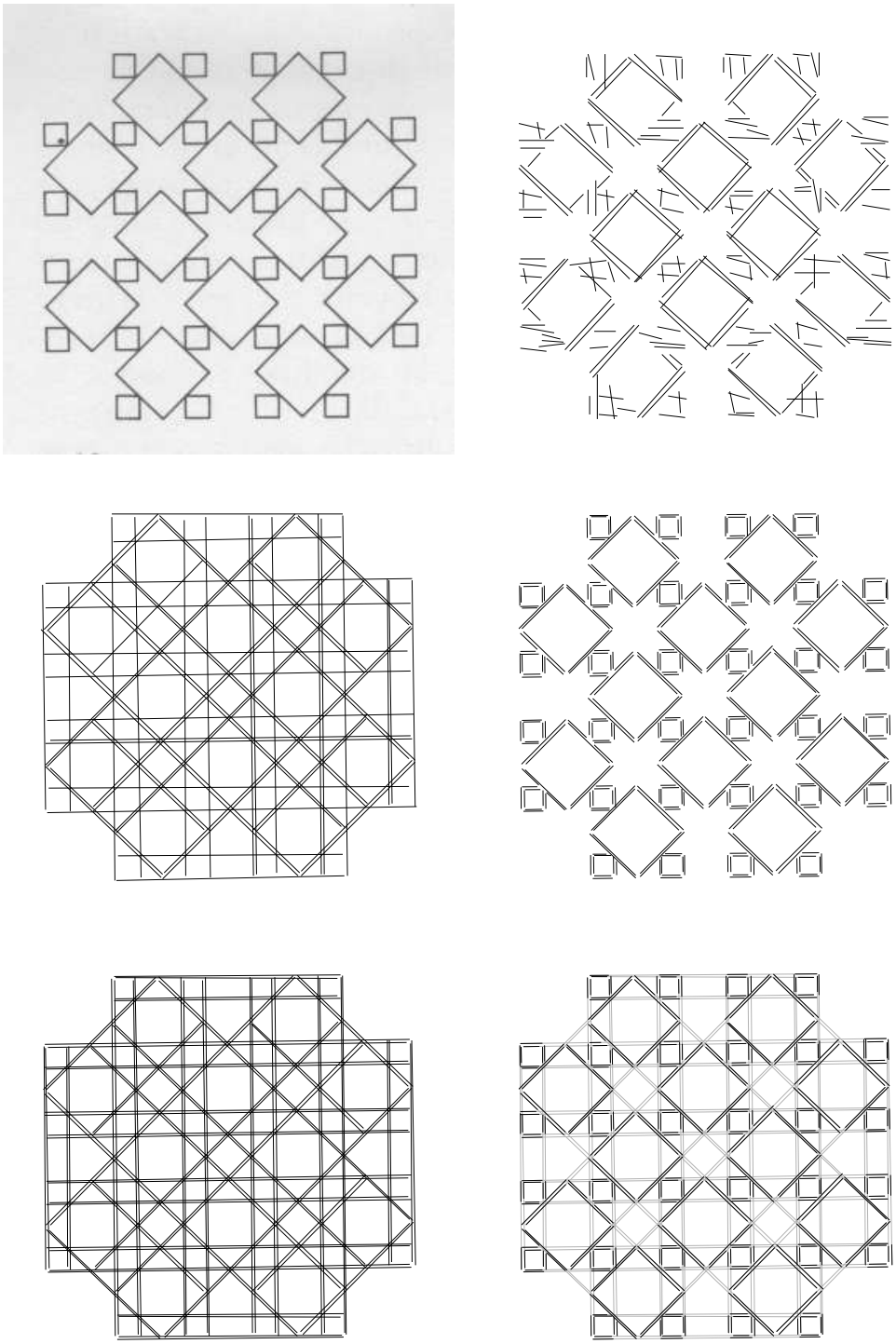

Figure 36: Multisegments produce local and global alignment information. Up-left: Image from Kanizsa. Up-right: segments found by HTM. Middle-left: segments found by DMM algorithm (the long alignments are more meaningful than their isolated segments). Middle-right: segments found by the Multisegments algorithm. Bottom-left: baselines of each multisegment. Bottom-right: segments found by the Multisegments algorithm in black; the baselines of each multisegment are shown in grey. Thus, the multisegment structure produces information at two scales; in the first one long alignments are detected like by DMM and in the second scale the right segments causing the alignment are singled out. The bottom right image representation summarizes these two kinds of information.

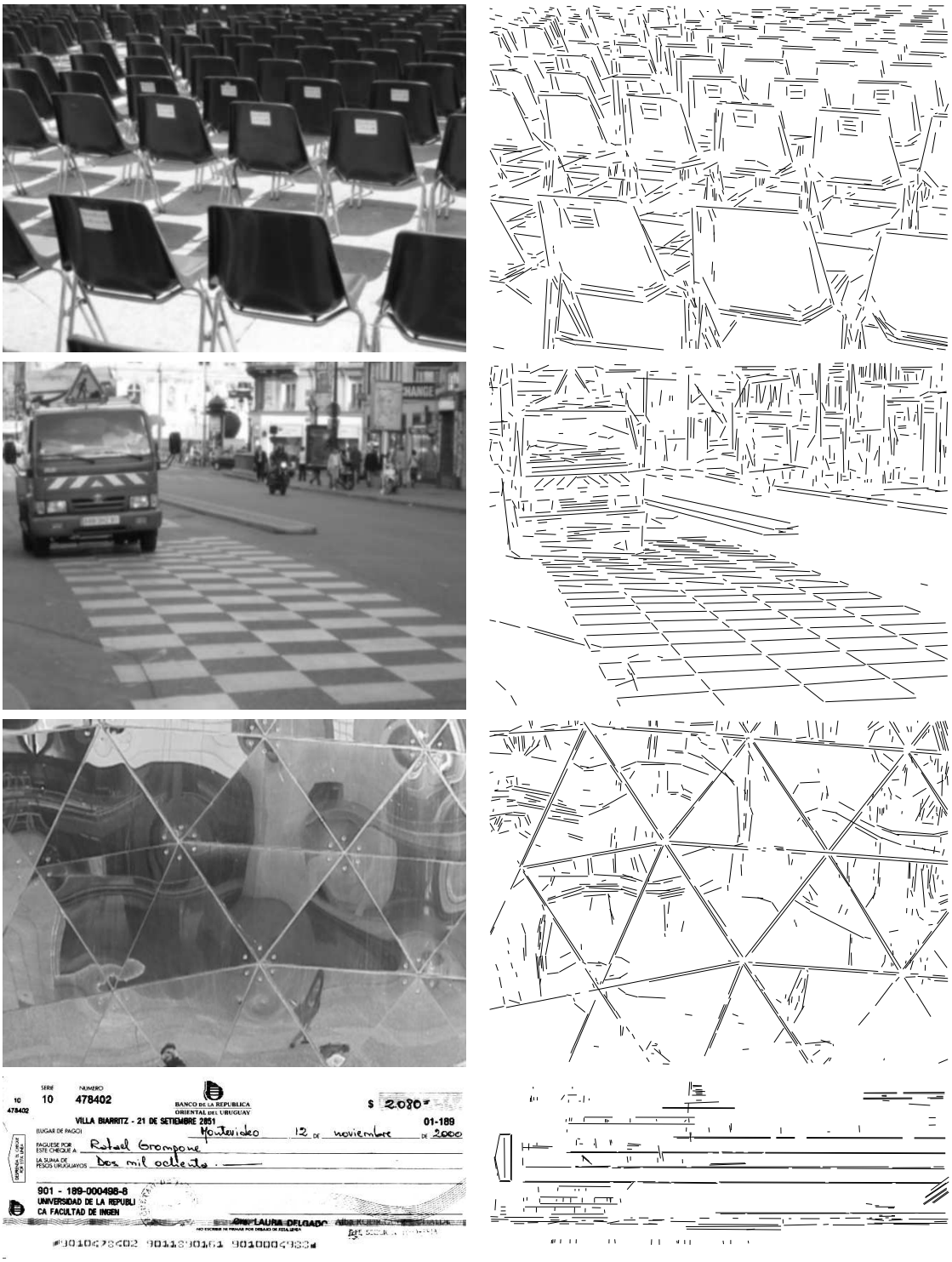

Figure 37: Experimental results of Multisegment detection on some natural images. Left column: Original image. Right column: multisegments found. All multisegment experiments were produced without parameter tuning. In presence of polygons, alignment detection does not lead to a connected set of alignments. Indeed all images are slightly blurry (following Shannon constraints) and therefore angles or corners are rounded and do not belong to alignments.

information. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(4):502–507, 2003.

[3] E. Arias-Castro, D.L. Donoho, and X. Huo. Near-optimal detection of geometric objects by fast multiscale methods. *IEEE Trans. Inform. Theory*, 51(7):2402–2425, 2005.

[4] Richard E. Bellman. *Dynamic programming*. Princeton University Press, 1957.

[5] F. Cao, J. Delon, A. Desolneux, P. Musé, and F. Sur. An a contrario approach to hierarchical clustering validity assessment. *preprint CMLA No.2004-13*, 2003.

[6] F. Cao, J. Delon, A. Desolneux, P. Musé, and F. Sur. A unified framework for detecting groups and application to shape recognition. *To appear in JMCV*, 2006.

[7] F. Cao, P. Musé, and F. Sur. Extracting Meaningful Curves from Images. *Journal of Mathematical Imaging and Vision*, 22(2):159–181, 2005.

[8] A.C. Copeland, G. Ravichandran, and M.M. Trivedi. Radon transform based ship-wake detection. *GeoRS*, 33(1):35–45, January 1995.

[9] Stanley R. Deans. Hough transform from the radon transform. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 3:185–188, 1981.

[10] J. Delon, A. Desolneux, J.-L. Lisani, and A.-B. Petro. A nonparametric approach for histogram segmentation. *IEEE Transactions on Image Processing*, 16(1):253–261, 2007.

[11] R. Deriche and O. Faugeras. Tracking line segments. *IVC*, 8(4):261–270, November 1990.

[12] A. Desolneux, L. Moisan, and J.-M. Morel. Gestalt theory and image analysis: A probabilistic approach, February 2006. `http://www.cmla.ens-cachan.fr/Utilisateurs/morel/lecturenote.pdf`.

[13] A. Desolneux, L. Moisan, and J.M. Morel. Maximal meaningful events and applications to image analysis. Technical Report 2000-22, CMLA, ENS-CACHAN, 2000. available at `http://www.cmla.ens-cachan.fr/Cmla/Publications/2000/CMLA2000-22.ps.gz`.

[14] A. Desolneux, L. Moisan, and J.M. Morel. Meaningful alignments. *International Journal of Computer Vision*, 40(1):7–23, 2000.

[15] A. Desolneux, L. Moisan, and J.M. Morel. Edge detection by helmholtz principle. *Journal of Mathematical Imaging and Vision*, 14(3):271–284, 2001.

[16] O. Faugeras, R. Deriche, H. Mathieu, N.J. Ayache, and G. Randall. The depth and motion analysis machine. *PRAI*, 6:353–385, 1992.

[17] Fernando Fernández. Mejoras al detector de alineamientos. Technical report, InCO, Universidad de la República, Uruguay, 2006.

[18] P. Fränti, E.I. Ageenko, H. Kälviäinen, and S. Kukkonen. Compression of line drawing images using hough transform for exploiting global dependencies. In *JCIS 1998*, 1998.

[19] B. Giai Checa, P. Bouthemy, and T. Vieville. Segment-based detection of moving objects in a sequence of images. In *ICPR94*, pages A:379–383, 1994.

[20] E. Oja H. Kälviäinen, P. Hirvonen. Houghtool — a software package for the use of the hough transform. *Pattern Recognition Letters*, 17:889–897, 1996.

[21] P.V.C. Hough. *Method and Means for Recognizing Complex Patterns*. U.S. Patent 3,069,654, Dec. 18, 1962.

[22] L. Igual. *Image Segmentation and Compression Using the Tree of Shapes of an Image. Motion Estimation*. PhD thesis, Universitat Pompeu Fabra, 2006.

[23] L. Igual, J. Preciozzi, L. Garrido, A. Almansa, V. Caselles, and B. Rougé. Automatic low baseline stereo in urban areas. *Inverse Problems and Imaging*, 1(2):319–348, 2007.

[24] J. Illingworth and J. Kittler. A survey of the Hough transform. *Computer Vision, Graphics, and Image Processing*, 44(1):87–116, 1988.

[25] C.X. Ji and Z.P. Zhang. Stereo match based on linear feature. In *ICPR88*, pages 875–878, 1988.

[26] Gaetano Kanizsa. *Grammatica del vedere*. Il Mulino, 1980.

[27] L. Kelvin. On ship waves. *Proc. Inst. Mech. Eng.*, 3, 1887.

[28] V.F. Leavers. Survey: Which hough transform? *CVGIP: Image Understanding*, 58(2):250–264, September 1993.

[29] M. Lindenbaum. An integrated model for evaluating the amount of data required for reliable recognition. *PAMI*, 19(11):1251–1264, 1997.

[30] D. Lowe. *Perceptual Organization and Visual Recognition*. Kluwer Academic Publishers, 1985.

[31] E. Magli, G. Olmo, and L. Lo Presti. On-board selection of relevant images: an application to linear feature recognition. *IEEE Transactions on Image Processing*, 10(4):543–553, 2001.

[32] S. Mahadevan and D.P. Casasent. Detection of triple junction parameters in microscope images. In *SPIE*, pages 204–214, 2001.

[33] J. Matas, C. Galambos, and J.V. Kittler. Progressive probabilistic hough transform. In *BMVC98*, 1998.

[34] L. Moisan and B. Stival. A probabilistic criterion to detect rigid point matches between two images and estimate the fundamental matrix. *International Journal of Computer Vision*, 57(3):201–218, 2004.

[35] P. Monasse and F. Guichard. Fast computation of a contrast-invariant image representation. *IEEE Transactions on Image Processing*, 9(5):860–872, 2000.

[36] P. Musé. *On the definition and recognition of planar shapes in digital images*. PhD thesis, ENS Cachan, 2004.

[37] P. Musé, F. Sur, F. Cao, and Y. Gousseau. Unsupervised thresholds for shape matching. *IEEE Int. Conf. on Image Processing, ICIP*, 2003.

[38] J. Preciozzi. "Dense Urban Elevation Models from Stereo Images by an Affine Region Merging Approach". Master's thesis, Universidad de la República, Montevideo, Uruguay, 2006.

[39] J. Princen, J. Illingworth, and J.V. Kittler. Hypothesis testing: A framework for analysing and optimizing hough transform performance. *PAMI*, 16(4):329–341, April 1994.

[40] A. Rosenfeld. *Picture Processing by Computer*. Academic Press, 1969.

[41] A. Rosenfeld. Digital straight line segments. *TC*, 23(12):1264–1269, December 1974.

[42] J. P. Shaffer. Multiple hypothesis testing. *Annu. Rev. Psychol.*, 46:561–584, 1995.

[43] D. Shaked, O. Yaron, and N. Kiryati. Deriving stopping rules for the probabilistic hough transform by sequential-analysis. *CVIU*, 63(3):512–526, May 1996.

[44] F. Sur. *A contrario decision for shape recognition*. PhD thesis, ENS Cachan, 2004.

[45] F. Tupin, H. Maître, J.-F. Mangin, J.-M. Nicolas, and E. Pechersky. Detection of linear features in SAR images: Application to the road network extraction. *IEEE Trans. Geosci. Remote Sensing*, 36(2):434–453, Mar. 1998.

[46] Y. Zheng, H. Li, and D. Doermann. A parallel-line detection algorithm based on HMM decoding. *IEEE Trans. PAMI*, 27(5), May 2005.

[47] Y. Zhu, B. Carragher, D. Kriegman, R. Milligan, and C. Potter. Automated identification of filaments in cryo-electron microscopy images. *Journal of Structural Biology*, 135:302–312, September 2001.

Figure 38: Experimental results of Multisegment detection on some natural images. Left column: Original image. Right column: multisegments found. All multisegment experiments in this paper were produced without parameter tuning. Smooth curves in an image lead to the detection of alignments because they have long enough tangent lines. This is why alignments are detected along curvy hair, eyeglasses, zebra stripes, etc. Many natural objects (branches, straws, etc.) are actually straight or piecewise straight.