EXTRACTION OF LINE SEGMENTS IN CLUTTERED IMAGES VIA MULTISCALE EDGES

Rui F. C. Guerreiro, Pedro M. Q. Aguiar

Institute for Systems and Robotics / Instituto Superior Técnico, Lisboa, Portugal
{rfcg,aguiar}@isr.ist.utl.pt

ABSTRACT

Current methods for line segment extraction often fail in challenging scenarios that abound in real-life images, *e.g.*, those containing corrupted lines, of various widths, with multiple crossings, and immersed in clutter. We propose a method that tackles these issues by combining multiscale edges while taking line segment connectivity into account. In particular, we use two scales originating what we call *contextual* and *local edges*, obtained with filters of, respectively, large and small footprints. Contextual edges are robust to noise and our method uses them validate local edges, *i.e.*, to only select the local edges that correspond to the same intensity transition (dark-to-bright or vice-versa). Line segment connectivity is enforced by joining the valid local edges whose distance does not exceed a threshold. To enable dealing with situations where the edges divide regions of non-uniform intensity distributions, *e.g.*, textures, the contextual edges are decided by using a two-sample statistical test. We present experiments that illustrate how our method is efficient in extracting complete segments in several situations where current methods fail.

***Index Terms*—** Line segment extraction, Multiscale edges, Two-sample statistical tests, Connectivity

1. INTRODUCTION

Methods to extract line segments from real-world images are key to applications that range from understanding to vizualization (*e.g.*, the automatic generation of cartoon/sketch effects, see `http://papercamera.net/`). Although several approaches to this problem have been proposed, current solutions make (most often implicit) assumptions that limit their applicability. As a consequence, the detection of line segments in realistic scenarios remains an open frontier, see [1, 2, 3] for examples of recent advances.

The the most popular method to detect lines in images is through the so-called Hough transform (HT), which consists of an efficient way to count the votes for each of all possible lines, see, *e.g.*, [4]. Its success comes precisely from the global nature of the voting scheme. However, several limitations of the HT have been identified and experimentally observed [2, 3] for the reasons we briefly synthesize. The majority of methods that use the HT perform an initial step of edge detection, an ill-posed problem that requires a delicate balance between edge localization and noise reduction. Spurious edge points, arising from noise and/or textures, vote for in inexistent lines, compromising in a particularly critical way the detection of short line segments. Although it was pointed out as early as in the 70s that the effect of spurious votes could be reduced by enforcing the *connectivity* of edge points contributing to each segment [4], this topic received little attention (exceptions are [5, 6, 3]). The method *Segment exTRAction by connectivity-enforcInG HT* (STRAIGHT) [3] obviates this but at a prohibitive computational cost for many applications, since it computes one HT per edge point. Finally, there are cases that are not appropriately coped with by the HT-based methods, such as dealing with wide line segments that arise from smooth edges (the segment receiving highest number of votes is the diagonal of the rectangle of edge points, not its center line).

Other methods extract line segments by performing local decisions, see [2] for examples. Most of them use three steps: first, edge detection and chaining; then, from the edge chain, an initial (rough) estimate of the segment direction is computed; finally, the segment is extended and the estimate of its parameters is refined. Although this sequence of steps may fail in several situations (see [3] for a description of some), the computational simplicity of these methods is undoubtedly attractive, which justifies the popularity of the state-of-the-art local method – the *Line Segment Detector* (LSD) [2].

In this paper we propose a new line segment extraction method. Its main distinctive characteristic is the usage of multiscale edges. In particular, our method combines *contextual edges*, which are obtained with filters of large footprint, with *local edges*, which are thresholded image derivatives obtained with filters of very small footprint, *e.g.*, Sobel, Prewitt, or Roberts operators. Contextual edges are naturally robust to the noise but exhibit imprecise localization, since the intensity transitions are smoothed by the large footprint. Local edges are precisely localized but sensitive to noise. Our method only accepts as valid local edges those that agree with the type of image intensity transition (dark-to-bright or the opposite) predicted by the contextual edge computed at the same location. The method further accounts for line connectivity by using a maximum distance threshold to select valid local edges as belonging to the same segment. The resulting pixel-thin connected lines are finally grown and a simple rectangle fitting methodology produces the representation of segments of all lengths and widths.

A typical approach to compute contextual edges would use a low-pass filter, followed by differentiation and binarization with a fixed threshold. However, the inevitable presence of textures, clutter from interfering line segments, and non-rectilinear image data, common in real-life images, motivate the usage of a statistical approach. In our method, the intensity distributions each region is modelled as Gaussian; the sample mean and variance are obtained for each set of pixels and the detection of a contextual edge obeys a two-sample statistical test that basically discards the null hypothesis that both distributions are the same.

The proposed method results computationally simple, being even competitive in speed with the fast local methods. Furthermore, the results of our method when dealing with challenging images compare favorably with the ones of HT [4], LSD [2], and STRAIGHT [3], as the experiments we report in the paper illustrate.

This work was partially supported by *Fundação para a Ciência e a Tecnologia* (FCT) [PEst-OE/EEI/LA0009/2011]. R. Guerreiro was also supported by FCT grant SFRH/BD/48602/2008, being currently affiliated to Philips Research Eindhoven, The Netherlands.

2. CONTEXTUAL AND LOCAL EDGES

2.1. Statistical test for contextual edge detection

We say that a contextual edge exists if the intensity distributions of the regions bordering the candidate location are significatively different. This is done by using a two-sample statistical test, as illustrated in Fig. 1. Two sets of pixels, forming a long and thin footprint, are analyzed, being considered as samples of two underlying probability distributions, $\boldsymbol{X}_T$ and $\boldsymbol{X}_B$. A contextual edge exists if $\boldsymbol{X}_T$ and $\boldsymbol{X}_B$ are different, *i.e.*, if the two sample sets can not be considered as having been generated by the same intensity distribution.

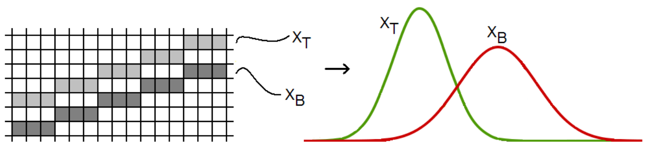

Fig. 1. Two-sample statistical test for contextual edge detection.

We model $\boldsymbol{X}_T$ and $\boldsymbol{X}_B$ as Normal distributions and measure its (dis)similarity using the Total Variation (TV) distance. Although other authors have used more flexible models, including nonparametric ones, and other distances, our experience has shown that this simple method is robust and results computationally very cheap. Our method thus computes the sample means and variances of the two sets of pixels and decides for the presence of a contextual edge whenever the TV distance between $\boldsymbol{X}_T$ and $\boldsymbol{X}_B$, defined as

$$\delta(\boldsymbol{X}_T, \boldsymbol{X}_B) = \frac{1}{2}\int_{-\infty}^{\infty} \left|\mathcal{N}_{\widehat{\mu}_T,\widehat{\sigma}_T}(x) - \mathcal{N}_{\widehat{\mu}_B,\widehat{\sigma}_B}(x)\right| dx\,,$$

is above a threshold C (obviously, $\delta(\boldsymbol{X}_T, \boldsymbol{X}_B) \in [0,1]$; we verified that $C = 0.7$ leads to meaningful results). We further classify each contextual edge as *positive* or *negative*, *i.e.*, we also store the sign of the contextual edge, defined as $\operatorname{sgn}(\widehat{\mu}_T - \widehat{\mu}_B)$.

We developed an efficient procedure to evaluate in practice the TV distance $\delta(\boldsymbol{X}_T, \boldsymbol{X}_B)$. In first place, the integral in the expression above can be easily expressed in terms of differences of the cumulative density function of the Normal distribution. Furthermore, it can be shown that $\delta(\boldsymbol{X}_T, \boldsymbol{X}_B)$ only apparently depends on the four parameters $\widehat{\mu}_T, \widehat{\sigma}_T, \widehat{\mu}_B$, and $\widehat{\sigma}_B$; in fact, only the (normalized) absolute difference of the means and the ratio of variances matter. As a consequence, a two-dimensional look-up table suffices to enable obtaining all the needed TV distances.

Our method detects contextual edges in linear arrangements of $M = 15$ pixels along $N = 32$ equally spaced directions, *i.e.*, along angles $\theta_n = 180°(n-1)/N$, with $n \in \{1, \dots, N\}$. For efficiency, along each direction, we use a running average to compute the sample means and variances of the two sets of pixels.

2.2. Local edge detection

Local edges are detected by thresholding image derivatives obtained by convolving the image with kernels $\boldsymbol{K}_\theta$ of very small footprint, the so-called central difference kernels. We use four 3×3 kernels, corresponding to the vertical, horizontal, and diagonal central differences, *i.e.*, we compute

$$\boldsymbol{\nabla}_\theta \boldsymbol{I} = \boldsymbol{I} * \boldsymbol{K}_\theta\,, \quad \theta \in \{0°, 45°, 90°, 135°\}\,,$$

and classify a pixel $\boldsymbol{p}$ as a local edge along direction θ when $|\boldsymbol{\nabla}_\theta I(\boldsymbol{p})| > L$. We found that better results are obtained if the threshold L is allowed to depend on the "strength" of the corresponding contextual edge, rather than kept fixed; in particular, if the absolute difference of the means of the regions forming the corresponding contextual edge is large, the threshold for the detection of local edges should also be large. In our experiments, we ended up setting $L = \max(3, |\hat{\mu}_T - \hat{\mu}_B|/2)$. As for the contextual edges, we also classify a local edge point $\boldsymbol{p}$ along direction θ as *positive* or *negative*, depending on $\operatorname{sgn}\boldsymbol{\nabla}_\theta \boldsymbol{I}(\boldsymbol{p})$.

3. EXTRACTION OF CONNECTED LINE SEGMENTS

Contextual edges are used to select *valid* local edges, *i.e.*, local edges whose intensity transition agrees with the one of the contextual edge. Connectivity is ensured by marking as edge points also the pixels that fall between valid local edges that are close, as detailed below.

3.1. Finding a valid edge point

The first step consists of scanning the image in order to find contextual and local edge points whose spatial location, direction, and intensity transition coincide. Fig. 2 illustrates the scenario with a set of $M = 15$ pixels that define a contextual edge along a direction θ_n. For simplicity, we name the m-th pixel of the set as $\boldsymbol{p}_m$.

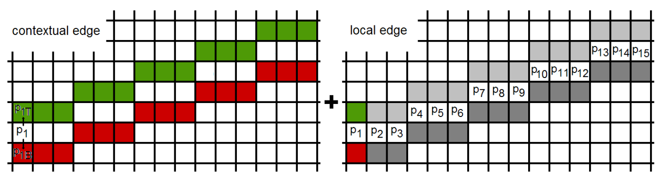

Fig. 2. Contextual edge (left) and valid local edge (right).

Basically, our method checks if there is a local edge with matching direction and sign at each of the pixels of the contextual edge. For $\boldsymbol{p}_1$, this means checking if $|\boldsymbol{\nabla}_{Q(\theta_n)}\boldsymbol{I}(\boldsymbol{p}_1)| \geq L$, where $Q(\theta_n)$ is just the angle $\theta \in \{0°, 45°, 90°, 135°\}$ that best approximates θ_n, and if $\operatorname{sgn}(\widehat{\mu}_T - \widehat{\mu}_B) = \operatorname{sgn}\boldsymbol{\nabla}_{Q(\theta_n)}\boldsymbol{I}(\boldsymbol{p}_1)$. When this happens (situation illustrated in Fig. 2), $\boldsymbol{p}_1$ is called a *valid* local edge point.

3.2. Finding a connected line segment

Once a valid edge point is found, a second step checks if it is in fact part of a line segment or just the result of an image artifact. Our method considers that there is a line segment when the M pixels between $\boldsymbol{p}_1$ and $\boldsymbol{p}_M$ contain valid local edges that are "sufficiently connected to each other". This notion is implemented through a maximum distance threshold that prevents accepting larger gaps, *i.e.*, we consider that valid local edges are connected if they are not separated by more than a threshold $d = 5$.

Fig. 3 illustrates the scenario, where pixels $\boldsymbol{p}_m \in \{\boldsymbol{p}_1, \dots, \boldsymbol{p}_M\}$ are checked sequentially. If a gap between valid local edge points exceeds the maximum distance threshold d, we consider there is not a line segment; if none gap exceeds d, all the pixels are considered as edge points with the sign of the corresponding contextual edge, $\operatorname{sgn}(\widehat{\mu}_T - \widehat{\mu}_B)$. In the illustration of Fig. 3, we also display the actual "gap counter": if the maximum distance threshold is set to $d = 2$, our method considers there is not a line segment, since there is a length-4 gap between local edge points with "correct" sign (between $\boldsymbol{p}_8$ and $\boldsymbol{p}_{13}$); in opposition, with $d = 5$, we would have detected a line segment in this situation.

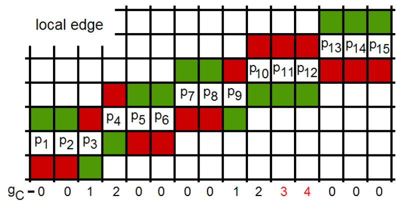

Fig. 3. Checking line segment connectivity by comparing the local edge signs with the one of the contextual edge of Fig. 2. If the "gap counter" (g_C) becomes larger than the maximum distance threshold, there is not a connected segment.

It is important to emphasize that other approaches to extract connected line segments, such as the one in [5], judge edge point connectivity uniquely from their spatial location. In opposition, our method requires that the signs of the edge points match the corresponding contextual edge, *i.e.*, local edges with opposite signs — even if they are "strong" edges — are discarded just like non-edges.

3.3. Extending the line segment

For each connected segment found, a third step progressively checks the pixels along its direction in order to extend the segment to its correct length. This basically amounts to checking along the line if there are contextual and local edges whose direction and sign matches and are sufficiently close, *i.e.*, with gaps smaller than the maximum distance threshold d. When a gap larger than d is found, the segment is considered to end. The output of this process is thus a set of connected edge points for each direction, as illustrated in Fig. 4.

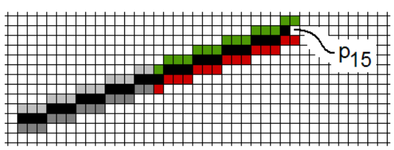

Fig. 4. Extending the connected line segments. Connected edge points are shown in black and pixel $\boldsymbol{p}_{15}$ is singled out as where the contextual and local edges are being checked.

4. DEALING WITH WIDE SEGMENTS

As pointed out in the Introduction, one of our motivations was the need to cope with multiple widths of line segments that are often present in real photographic images. Since the connected segments obtained as described in the previous section are pixel-thin, a wide line segment will be made up of various thin connected segments. We now describe the final step that our method uses to merge these thin lines into a single region corresponding to the actual line segment, as illustrated in Fig. 5.

Basically, we fit a rectangle to each region corresponding to a collection of line segments of direction θ_n. Since these segments are connected, the fitting scheme can be made very simple: we start by fitting lines to the upper and lower limits of the region, as illustrated in Fig. 5; then, using the mean slope the fitted lines we obtain the start and end of each line segment. A validation step eliminates regions corresponding to non-rectilinear structures in the image: we require the slopes of these lines to be similar and to lie inside the angle range of direction θ_n, *i.e.*, $[\theta_n - 180°/2N, \theta_n + 180°/2N]$.

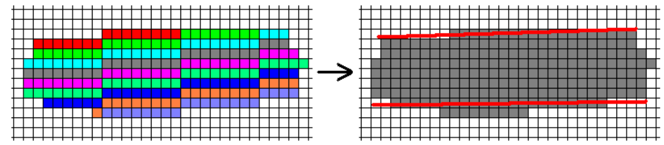

Fig. 5. Several connected pixel-thin segments (left) are merged into a single region corresponding to the actual (wide) line segment (right).

5. EXPERIMENTS

We start by comparing the behavior of LSD [2], STRAIGHT [3], and the proposed method when dealing with textures. We use the synthetic images in the left column of Fig. 6. In the top image, one of the regions is noiseless, *i.e.*, of constant intensity. The middle image simulates a scenario where both regions are textured. In the bottom image, a more challenging situation, where the mean of both regions is equal, being their variance the discriminating factor. Due to the noise, the local edge detection performed by LSD fails and only for the top image some segments are correctly extracted. Both STRAIGHT and the proposed method succeed in extracting the line segments from the top and middle images (the few short segments correspond to accidental connected alignments in the random texture). For the bottom image, only the proposed method was able to extract the majority of segments separating the textured regions, due to the two-sample statistical test it uses to detect contextual edges.

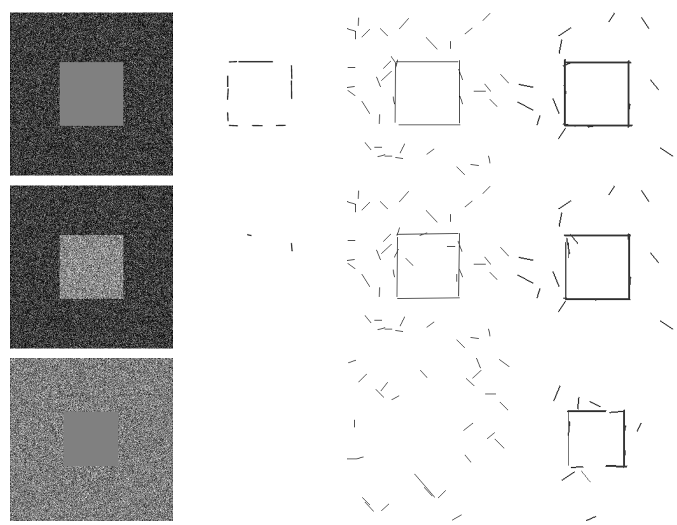

Fig. 6. In each line, from left to right: input image and results of, respectively LSD [2], STRAIGHT [3], and the proposed method.

The real image in Fig. 7 is illustrative of a complex arrangement of line segments: a scene, containing many segments, several of them with low contrast, is occluded by a net that is out-of-focus, thus formed by wide segments. LSD fails to extract several low contrast segments and fragments several others, particularly when there are crossings. STRAIGHT correctly extracts some entire segments but still fails in processing low contrast and/or wide ones, as easily seen from the several erroneous line segments that correspond to the thick lines that form the net. In opposition, the proposed method extracts most line segments with little fragmentation.

Fig. 8 displays sample results of the proposed method when dealing with challenging real images. Note in particular that al-

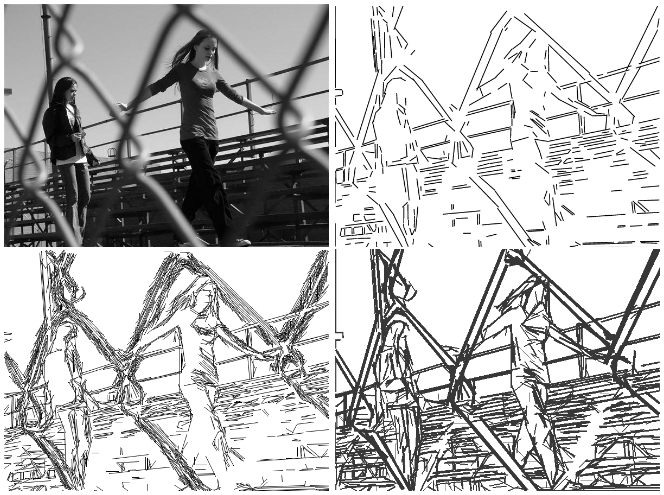

Fig. 7. Top left: input image. Top right, bottom left and bottom right: results of LSD [2], STRAIGHT [3], and the proposed method.

though some of the edges in these images are not straight lines, our method succeeds in approximating them in a piecewise way, *i.e.*, by a sequence of rectilinear line segments. We end by pointing out that these good results are obtained at a low computational cost. In fact, the computational cost of the proposed method grows only linearly with the number of pixels, as it also happens with local methods such as LSD. As a result, on an Intel© 2.67 GHz machine, we were able to process about 97K pixels per second, making the extraction of line segments from a 512×512 image take about 2.7 seconds. This contrasts with the huge computational cost of STRAIGHT, that takes several minutes to process the same image.

6. CONCLUSION

We propose a new method for line segment extraction that uses contextual edges, computed via a two-sample statistical test, and line connectivity, to validate local edges. The method results computationally simple and our experiments illustrate its effectiveness.

7. REFERENCES

[1] S. Du et al., "An improved Hough Transform neighborhood map for straight line segments," *IEEE T-IP*, vol. 19, no. 3, 2010.

[2] R. von Gioi et al., "LSD: A fast line segment detector with a false detection control," *IEEE T-PAMI*, vol. 32, no. 4, 2010.

[3] R. Guerreiro and P. Aguiar, "Connectivity-enforcing Hough Transform for the robust extraction of line segments," *IEEE T-IP*, vol. 21, no. 12, 2012.

[4] R. Duda and P. Hart, "Use of the Hough Transformation to detect lines and curves in pictures," *Communications of the ACM*, vol. 15, no. 1, 1972.

[5] S. Yuen, T. Lam, and N. Leung, "Connective Hough Transform," *Image and Vision Computing*, vol. 11, no. 5, 1993.

[6] M. Yang et al., "Hough Transform modified by line connectivity and line thickness," *IEEE T-PAMI*, vol. 19, no. 8, 1997.

Fig. 8. Illustrative sample results of the proposed method.