Advances in Low-Level Color Image Processing. Lecture Notes in Computational Vision and Biomechanics, Volumen 11. Springer, 2014. 279-301.

DOI: http://dx.doi.org/10.1007/978-94-007-7584-8_9

Tensor Voting for Robust Color Edge Detection

Rodrigo Moreno and Miguel Angel Garcia and Domenec Puig

Abstract This chapter proposes two robust color edge detection methods based on tensor voting. The first method is a direct adaptation of the classical tensor voting to color images where tensors are initialized with either the gradient or the local color structure tensor. The second method is based on an extension of tensor voting in which the encoding and voting processes are specifically tailored to robust edge detection in color images. In this case, three tensors are used to encode local CIELAB color channels and edginess, while the voting process propagates both color and edginess by applying perception-based rules. Unlike the classical tensor voting, the second method considers the context in the voting process. Recall, discriminability, precision, false alarm rejection and robustness measurements with respect to three different ground-truths have been used to compare the proposed methods with the state-of-the-art. Experimental results show that the proposed methods are competitive, especially in robustness. Moreover, these experiments evidence the difficulty of proposing an edge detector with a perfect performance with respect to all features and fields of application.

Key words: Perceptual methods, tensor voting, perceptual grouping, nonlinear approximation, curveness and junctionness propagation.

R. Moreno
Center for Medical Image Science and Visualization and Dept. of Medical and Health Sciences, Linköping University, Campus US, 58185 Linköping, Sweden, e-mail: rodrigo.moreno@liu.se

M. A. Garcia
Dept. of Electronic and Communications Technology, Autonomous University of Madrid, Francisco Tomas y Valiente 11, 28049 Madrid, Spain, e-mail: miguelangel.garcia@uam.es

D. Puig
Intelligent Robotics and Computer Vision Group, Rovira i Virgili University, Av. Països Catalans 26, 43007 Tarragona, Spain, e-mail: domenec.puig@urv.cat

1 Introduction

Edge detection is an important problem in computer vision, since the performance of many computer vision applications directly depends on the effectiveness of a previous edge detection process. The final goal of edge detection is to identify the locations at which the image has "meaningful" discontinuities. The inherent difficulty in defining what a meaningful discontinuity is has fostered this research area during the last decades. However, in spite of all the efforts, the problem has not completely been solved yet and problems such as automatic tuning of parameters, edge detection in multiscale analysis or noise robustness are still under active research.

The raw output of a general purpose edge detector can be seen as an edginess map, that is, a map of the probability of every pixel being an edge. Since most applications require binary edge maps instead of edginess maps, post-processing steps, such as non-maximum suppression and thresholding with or without hysteresis, are applied to the edginess maps in order to generate such binary maps [5].

In gray-scale images, the Canny's edge detector [5] has consistently been reported as the best method in many comparisons, e.g., [4, 10, 29, 31, 23]. On the other hand, edge detectors specifically devised for color images usually outperform gray-scale edge detectors. For instance, [28, 3] and [1] have reported a better performance than Canny's edge detector applied to gray-scale images. Complete reviews of strategies on color edge detection are presented in [11, 12, 33, 25, 30].

Although a number of edge detectors have been proposed during the last years, only a few have been devised to deal with noise. This can be due to the fact that edges of noisy images can be extracted from denoised versions of the input images [6]. This strategy is followed, for example, in [32]. However, image denoising is not a trivial problem and is still one of the most active research areas in image processing. In addition, the application of image denoising before extracting edges makes it difficult to measure how good the edge detector is, since its performance will be directly related to the performance of the applied filtering stage.

Since the human visual system is able to detect edges in noisy scenarios, the use of perceptual techniques for robust edge detection appears promising. In this context, this paper explores a new approach to extract edges from noisy color images by applying tensor voting, a perceptual technique proposed by Medioni and collaborators [16] as a robust means of extracting perceptual structures from noisy clouds of points. Unfortunately, tensor voting cannot be directly applied to images, since it was devised for dealing with noise in clouds of points instead of in images. Thus, adaptations of this technique are mandatory in order to make it suitable to the problem of edge detection in images. In this chapter, we present two different adaptations of tensor voting to robust color edge detection The proposed methods take advantage of the

robustness of tensor voting to improve the performance in noisy scenarios. These methods are summarized in Sections 2 and 3.

Recently, we have also introduced a general methodology to evaluate edge detectors directly in gray-scale [22]. This methodology avoids possible biases generated by post-processing of edginess maps by directly comparing the algorithms in gray-scale. This methodology is summarized in Section 4.

The paper is organized as follows. Sections 2 and 3 detail the two proposed methods for color edge detection based on tensor voting. Section 4 summarizes the methodology of evaluation. Section 5 shows a comparative analysis of the proposed methods against some of the state-of-the-art color edge detection algorithms. Finally, Section 6 discusses the obtained results and makes some final remarks.

2 Color Edge Detection Through the Classical Tensor Voting

The first adaptation of tensor voting to robust color edge detection is based on using appropriate initialization and post-processing steps of the method proposed by Medioni and collaborators in [16], hereafter referred to as classical tensor voting, which is summarized in the following subsection.

2.1 Classical Tensor Voting

Tensor voting is a technique for extracting structure from a cloud of points, in particular in 3D. The method estimates saliency measurements of how likely a point lies on a surface, a curve, a junction, or it is noisy. It is based on the propagation and aggregation of the most likely normal(s) encoded by means of tensors. In a first stage, a tensor is initialized at every point in the cloud either with a first estimation of the normal, or with a ball-shaped tensor if a priori information is not available. Afterwards, every tensor is decomposed into its three components: a *stick*, a *plate* and a *ball*. Every component casts votes, which are tensors that encode the most likely direction(s) of the normal at a neighboring point by taking into account the information encoded by the voter in that component. Finally, the votes are summed up and analyzed in order to estimate surfaceness, curveness and junctionness measurements at every point. Points with low saliency are assumed to be noisy. More formally, the tensor voting at $\mathbf{p}$, $\mathrm{TV}(\mathbf{p})$ is given by:

$$\mathrm{TV}(\mathbf{p}) = \sum_{\mathbf{q} \in \mathcal{N}(\mathbf{p})} \mathrm{SV}(\mathbf{v}, \mathrm{S}_{\mathbf{q}}) + \mathrm{PV}(\mathbf{v}, \mathrm{P}_{\mathbf{q}}) + \mathrm{BV}(\mathbf{v}, \mathrm{B}_{\mathbf{q}}), \tag{1}$$

where $\mathbf{q}$ represents each of the points in the neighborhood of $\mathbf{p}$, $\mathcal{N}(\mathbf{p})$, SV, PV and BV are the *stick*, *plate* and *ball* tensor votes cast to $\mathbf{p}$ by every component of $\mathbf{q}$, $\mathbf{v} = \mathbf{p} - \mathbf{q}$, and $\mathrm{S}_\mathbf{q}$, $\mathrm{P}_\mathbf{q}$ and $\mathrm{B}_\mathbf{q}$ are the *stick*, *plate* and *ball* components of the tensor at $\mathbf{q}$ respectively. These components are given by:

$$\mathrm{S}_\mathbf{q} = (\lambda_1 - \lambda_2)\left(\mathbf{e_1}\mathbf{e_1}^T\right), \tag{2}$$

$$\mathrm{P}_\mathbf{q} = (\lambda_2 - \lambda_3)\left(\mathbf{e_1}\mathbf{e_1}^T + \mathbf{e_2}\mathbf{e_2}^T\right), \tag{3}$$

$$\mathrm{B}_\mathbf{q} = \lambda_3\left(\mathbf{e_1}\mathbf{e_1}^T + \mathbf{e_2}\mathbf{e_2}^T + \mathbf{e_3}\mathbf{e_3}^T\right), \tag{4}$$

where λ_i and $\mathbf{e_i}$ are the i-th largest eigenvalue and its corresponding eigenvector of the tensor at $\mathbf{q}$. Saliency measurements can be estimated from an analysis of the eigenvalues of the resulting tensors. Thus, $s_1 = (\lambda_1 - \lambda_2)$, $s_2 = (\lambda_2 - \lambda_3)$, and $s_3 = \lambda_3$ can be used as measurements of surfaceness, curveness and junctionness respectively.

A *stick* tensor is a tensor with only a single eigenvalue greater than zero. *Stick* tensors are processed through the so-called *stick* tensor voting. The process is illustrated in Fig. 1. Given a known *stick* tensor $\mathrm{S}_\mathbf{q}$ at $\mathbf{q}$, the orientation of the vote cast by $\mathbf{q}$ to $\mathbf{p}$ can be estimated by tensorizing the normal of a a circumference at $\mathbf{p}$ that joins $\mathbf{q}$ and $\mathbf{p}$. This vote is then weighted by a decaying scalar function, w_s. The *stick* tensor vote is given by [20]:

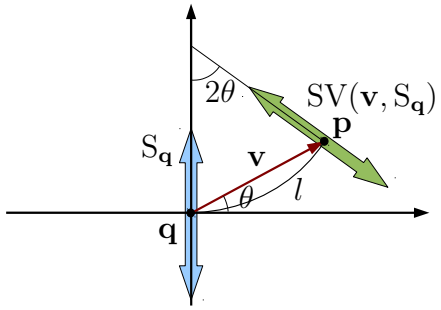

Fig. 1 *Stick* tensor voting. A *stick* $\mathrm{S}_\mathbf{q}$ casts a *stick* vote $\mathrm{SV}(\mathbf{v}, \mathrm{S}_\mathbf{q})$ to $\mathbf{p}$, which corresponds to the most likely tensorized normal at $\mathbf{p}$.

$$\mathrm{SV}(\mathbf{v}, \mathrm{S}_\mathbf{q}) = w_s \, R_{2\theta} \; \mathrm{S}_\mathbf{q} \; R_{2\theta}^T, \tag{5}$$

where θ is shown in Fig. 1 and $R_{2\theta}$ represents a rotation with respect to the axis $\mathbf{v} \times (\mathrm{S}_\mathbf{q} \; \mathbf{v})$, which is perpendicular to the plane that contains $\mathbf{v}$ and $\mathrm{S}_\mathbf{q}$; and w_s is an exponential decaying function that penalizes the arc-length l, and the curvature of the circumference, κ:

$$w_s = \begin{cases} e^{-\frac{l^2}{\sigma^2} + b\kappa^2} & \text{if } \theta \leq \pi/4 \\ 0 & \text{otherwise,} \end{cases} \tag{6}$$

where σ and b are parameters to weight the scale and the curvature respectively.

In turn, a *plate* tensor is a tensor with $\lambda_1 = \lambda_2 \geq 0$ and $\lambda_3 = 0$. *Plate* tensors are processed through the so-called *plate* tensor voting. The *plate* tensor voting uses the fact that any *plate* tensor P, can be decomposed into all possible *stick* tensors inside the *plate*. Let $\mathrm{S_P}(\beta) = R_\beta \mathbf{e_1} \mathbf{e_1}^T R_\beta^T$ be a *stick* inside the *plate* P, with $\mathbf{e_1}$ being its principal eigenvector, and R_β being a rotation with respect to an axis perpendicular to $\mathbf{e_1}$ and $\mathbf{e_2}$. Thus, the *plate* vote is defined as [20]:

$$\mathrm{PV}(\mathbf{v}, \mathrm{P_q}) = \frac{\lambda_{1_{\mathrm{P_q}}}}{\pi} \int_0^{2\pi} \mathrm{SV}(\mathbf{v}, \mathrm{S_{P_q}}(\beta))\ d\beta, \tag{7}$$

where $\lambda_{1_{\mathrm{P_q}}}$ is the largest eigenvalue of $\mathrm{P_q}$.

Finally, a *ball* tensor is a tensor with $\lambda_1 = \lambda_2 = \lambda_3 \geq 0$. The *ball* tensor voting is defined in a similar way as the *plate* tensor voting. Let $\mathrm{S_B}(\phi, \psi)$ be a unitary *stick* tensor oriented in the direction $(1, \phi, \psi)$ in spherical coordinates. Then, any *ball* tensor B can be written as [20]:

$$\mathrm{BV}(\mathbf{v}, \mathrm{B_q}) = \frac{3\lambda_{1_{\mathrm{B_q}}}}{4\pi} \int_\Gamma \mathrm{SV}(\mathbf{v}, \mathrm{S_{B_q}}(\phi, \psi))\ d\Gamma, \tag{8}$$

where Γ represents the surface of the unitary sphere, and $\lambda_{1_{\mathrm{B_q}}}$ is the largest eigenvalue of $\mathrm{B_q}$.

2.2 Color Edge Detection Through the Classical Tensor Voting

In [21], we showed that the classical tensor voting and the well-known structure tensor [8] are closely related. These similarities were used in [21] to extend classical tensor voting to different types of images, especially color images. This extension can be used to extract edges. This subsection summarizes that method for gray-scale and color images.

2.2.1 Gray-Scale Images

Tensor voting can be adapted in order to robustly detect edges in gray-scale images by following three steps. First, the tensorized gradient, $\nabla u \nabla u^T$, is used to initialize a tensor at every pixel. Second, the *stick* tensor voting is applied in order to propagate the information encoded in the tensors. In this case, it is not necessary to apply the *plate* and *ball* voting processes since the *plate* and *ball* components are zero at every pixel. Thus, tensor voting is reduced to:

$$\mathrm{TV}(\mathbf{p}) = \sum_{\mathbf{q}\in\mathcal{N}(\mathbf{p})} \mathrm{SV}(\mathbf{v}, \nabla u_{\mathbf{q}} \nabla u_{\mathbf{q}}^T). \tag{9}$$

Finally, the resulting tensors are rescaled by the factor:

$$\xi = \frac{\sum_{\mathbf{p}\in\Omega} trace(\nabla u_{\mathbf{p}} \nabla u_{\mathbf{p}}{}^T)}{\sum_{\mathbf{p}\in\Omega} trace(\mathrm{TV}(\mathbf{p}))}, \tag{10}$$

in order to renormalize the total energy of the tensorized gradient, where Ω refers to the given image.

After having applied tensor voting and the energy normalization step, the principal eigenvalue λ_1 of the resulting tensors can be used to detect edges, since it attains high values not only at boundaries but also at corners.

2.2.2 Color Images

Figure 2 shows the two possible options to extend tensor voting to color images using the adaptation proposed in the previous subsection. The first

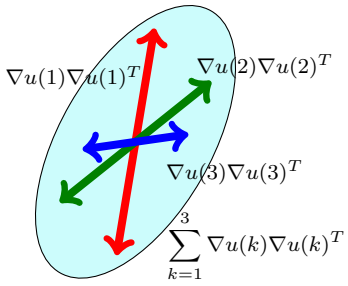

Fig. 2 Tensor voting can be applied to the color channels independently (the red, green and blue *sticks*) or to the sum of the tensorized gradients (the ellipse).

option is to apply the *stick* tensor voting independently to every channel and then adding the individual results, that is:

$$\mathrm{TV}(\mathbf{p}) = \sum_{k=1}^{3} \sum_{\mathbf{q}\in\mathcal{N}(\mathbf{p})} \alpha_k \ \mathrm{SV}(\mathbf{v}, \nabla u_{\mathbf{q}}(k) \nabla u_{\mathbf{q}}(k)^T), \tag{11}$$

where $\nabla u(k)$ is the gradient at color channel k, and α_k are weights used to give different relevance to every channel.

The second option is to apply (1) to the sum of tensorized gradients, with $\mathrm{S}_{\mathbf{q}}$, $\mathrm{P}_{\mathbf{q}}$ and $\mathrm{B}_{\mathbf{q}}$ being the *stick*, *plate* and *ball* components of $\mathrm{T}_{\mathbf{q}} = \sum_{k=1}^{3} \alpha_k \nabla u_{\mathbf{q}}(k) \nabla u_{\mathbf{q}}(k)^T$. For two-dimensional images, the computation of *plate* votes can be avoided since $\mathrm{P}_{\mathbf{q}} = 0$. Thus, the first option has the advantage that only the application of *stick* tensor voting is necessary, whereas the second option requires *stick* and *ball* tensor voting.

In practice, both strategies are very similar since $\mathrm{T}_{\mathbf{q}} \approx \mathrm{S}_{\mathbf{q}}$ in most pixels of images of natural scenes [21]. Thus, in the experiments of Section 5, the first option has been used for the majority of pixels, whereas the second one only in those pixels in which the aforementioned approximation is not valid. In practice, the first option can be applied when the angle between any pair of gradients is below a threshold.

Similarly to the case of gray-scale images, the classical tensor voting can be used to detect edges by means of the principal eigenvalue λ_1 of the resulting tensor, after an energy normalization step similar to the one of (10).

Since this method does not apply any pre-processing step, its robustness must completely rely on the robustness of the classical tensor voting. This could not be sufficient in highly noisy scenarios. Thus, in order to improve the results it is necessary to iterate the method. By iterating tensor voting, the most significant edges can be reinforced at the expense of discarding small ridges. According to our experiments, a few iterations (two or three) usually give good results for both noisy and noiseless images.

3 Color Edge Detection Through an Adapted Tensor Voting

It is important to remark that tensor voting is a methodology in which information encoded through tensors is propagated and aggregated in a local neighborhood. Thus, it is possible to devise more appropriate methods for specific applications by tailoring the way in which tensors are encoded, propagated and aggregated, while maintaining the tensor voting spirit. In this line, we introduced a method for image denoising [17, 19] that can also be applied to robust color edge detection, since both problems can be tackled at the same time [18]. The next subsections detail the edge detector.

3.1 Encoding of Color Information

Before applying the proposed method, color is converted to the CIELAB space. Every CIELAB channel is then normalized to the range $[0, \pi/2]$. In the first step of the method, the information of every pixel is encoded through

three second-order 2D tensors, one for each normalized CIELAB color channel.

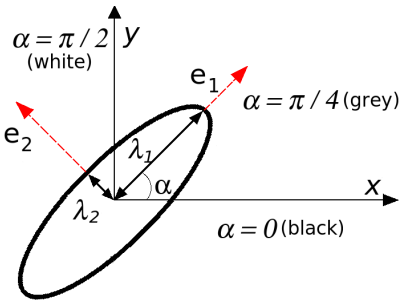

Fig. 3 Encoding process for channel L. Color, uniformity and edginess are encoded by means of α and the normalized saliencies $\hat{s_1} = (\lambda_1 - \lambda_2)/\lambda_1$ and $\hat{s_2} = \lambda_2/\lambda_1$ respectively.

Three perceptual measures are encoded in the tensors associated with every input pixel, namely: the normalized color at the pixel (in the specific channel), a measure of local uniformity (how edgeless its neighborhood is), and an estimation of edginess. Figure 3 shows the graphical interpretation of a tensor for channel L. The normalized color is encoded by the angle α between the x axis, which represents the lowest possible color value in the corresponding channel, and the eigenvector corresponding to the largest eigenvalue. For example, in channel L, a tensor with $\alpha = 0$ encodes black, whereas a tensor with $\alpha = \dfrac{\pi}{2}$ encodes white. In addition, local uniformity and edginess are encoded by means of the normalized $\hat{s_1} = (\lambda_1 - \lambda_2)/\lambda_1$ and $\hat{s_2} = \lambda_2/\lambda_1$ saliencies respectively. Thus, a pixel located at a completely uniform region is represented by means of three *stick tensors*, one for each color channel. In contrast, a pixel located at an ideal edge is represented by means of three *ball tensors*, one for every color channel.

Before applying the voting process, it is necessary to initialize the tensors associated with every pixel. The colors of the noisy image can be easily encoded by means of the angle α between the x axis and the principal eigenvector, as described above. However, since metrics of uniformity and edginess are usually unavailable at the beginning of the voting process, normalized saliency $\hat{s_1}$ is initialized to one and normalized saliency $\hat{s_2}$ to zero. These initializations allow the method to estimate more appropriate values of the normalized saliencies for the next stages, as described in the next subsection. Thus, the initial color information of a pixel is encoded through three *stick tensors* oriented along the directions that represent that color in the normalized CIELAB channels:

$$\mathrm{T}_k(\mathbf{p}) = \mathbf{t}_k(\mathbf{p})\ \mathbf{t}_k(\mathbf{p})^T, \tag{12}$$

where $\mathrm{T}_k(\mathbf{p})$ is the tensor of the k-th color channel (L, a and b) at pixel $\mathbf{p}$, $\mathbf{t}_k(\mathbf{p}) = [\cos(\mathrm{C}_k(\mathbf{p})) \quad \sin(\mathrm{C}_k(\mathbf{p}))]^T$, and $\mathrm{C}_k(\mathbf{p})$ is the normalized value of the k-th color channel at $\mathbf{p}$.

3.2 Voting Process

The voting process requires three measurements for every pair of pixels $\mathbf{p}$ and $\mathbf{q}$: the perceptual color difference, $\Delta E_{\mathbf{pq}}$; the joint uniformity measurement, $U_k(\mathbf{p}, \mathbf{q})$, used to determine if both pixels belong to the same region; and the likelihood of a pixel being impulse noise, $\eta_k(\mathbf{p})$. $\Delta E_{\mathbf{pq}}$ is calculated through CIEDE2000 [13], while

$$U_k(\mathbf{p}, \mathbf{q}) = \hat{s_{1}}_k(\mathbf{p})\ \hat{s_{1}}_k(\mathbf{q}), \tag{13}$$

and

$$\eta_k(\mathbf{p}) = \begin{cases} \hat{s_{2}}_c(\mathbf{p}) - \mu_{\hat{s_{2}}_c}(\mathbf{p}) & \text{if } \mathbf{p} \text{ is located at a local maximum} \\ 0 & \text{otherwise} \end{cases}, \tag{14}$$

where $\mu_{\hat{s_{2}}_c}(\mathbf{p})$ represents the mean of $\hat{s_{2}}_c$ over the neighborhood of $\mathbf{p}$.

In the second step of the method, the tensors associated with every pixel are propagated to their neighbors through a convolution-like process. This step is independently applied to the tensors of every channel (L, a and b). The voting process is carried out by means of specially designed tensorial functions referred to as *propagation functions*, which take into account not only the information encoded in the tensors but also the local relations between neighbors. Two propagation functions are proposed for edge detection: a *stick* and a *ball* propagation function. The *stick* propagation function is used to propagate the most likely noiseless color of a pixel, while the *ball* propagation function is used to increase edginess where required. The application of the first function leads to *stick* votes, while the application of the second function produces *ball* votes. *Stick* votes are used to eliminate noise and increase the edginess where the color of the voter and the voted pixels are different. *Ball* votes are used to increase the relevance of the most important edges.

A *stick* vote can be seen as a *stick*-shaped tensor, $\mathrm{ST}_k(\mathbf{p})$, with a strength modulated by three scalar factors. The proposed *stick* propagation function, $\mathrm{S}_k(\mathbf{p}, \mathbf{q})$, which allows a pixel $\mathbf{p}$ to cast a *stick* vote to a neighboring pixel $\mathbf{q}$ for channel k is given by:

$$\mathrm{S}_k(\mathbf{p}, \mathbf{q}) = GS(\mathbf{p}, \mathbf{q})\ \overline{\eta_k}(\mathbf{p})\ SV'_k(\mathbf{p}, \mathbf{q})\ \mathrm{ST}_k(\mathbf{p}), \tag{15}$$

with $\mathrm{ST}_k(\mathbf{p})$, $GS(\mathbf{p}, \mathbf{q})$, $\overline{\eta_k}(\mathbf{p})$ and $SV'_k(\mathbf{p}, \mathbf{q})$ being defined as follows. First, the tensor $\mathrm{ST}_k(\mathbf{p})$ encodes the most likely normalized noiseless color at $\mathbf{p}$. Thus, $\mathrm{ST}_k(\mathbf{p})$ is defined as the tensorized eigenvector corresponding to the

largest eigenvalue of the voter pixel, that is:

$$\mathrm{ST}_k(\mathbf{p}) = \mathbf{e}_{1k}(\mathbf{p})\ \mathbf{e}_{1k}(\mathbf{p})^T, \tag{16}$$

being $\mathbf{e}_{1k}(\mathbf{p})$ the eigenvector with the largest eigenvalue of the tensor associated with channel k at $\mathbf{p}$. Second, the three scalar factors in (15), each ranging between zero and one, are defined as follows. The first factor, $GS(\mathbf{p},\mathbf{q})$, models the influence of the distance between $\mathbf{p}$ and $\mathbf{q}$ in the vote strength. Thus, $GS(\mathbf{p},\mathbf{q}) = G_{\sigma_s}(||\mathbf{p}-\mathbf{q}||)$, where $G_{\sigma_s}(\cdot)$ is a decaying Gaussian function with zero mean and a user-defined standard deviation σ_s. The second factor, $\overline{\eta_k}(\mathbf{p})$ defined as $\overline{\eta_k}(\mathbf{p}) = 1-\eta_k(\mathbf{p})$, is introduced in order to prevent a pixel $\mathbf{p}$ previously classified as impulse noise from propagating its information. The third factor, SV'_k, takes into account the influence of the perceptual color difference, the uniformity and the noisiness of the voted pixel. This factor is given by:

$$SV'_k(\mathbf{p},\mathbf{q}) = \overline{\eta_k}(\mathbf{q})\ SV_k(\mathbf{p},\mathbf{q}) + \eta_k(\mathbf{q}), \tag{17}$$

where: $SV_k(\mathbf{p},\mathbf{q}) = [G_{\sigma_d}(\Delta E_{\mathbf{pq}}) + U_k(\mathbf{p},\mathbf{q})]/2$, and $\overline{\eta_k}(\mathbf{q}) = 1-\eta_k(\mathbf{q})$. $SV_k(\mathbf{p},\mathbf{q})$ allows a pixel $\mathbf{p}$ to cast a stronger *stick* vote to $\mathbf{q}$ either if both pixels belong to the same uniform region, or if the perceptual color difference between them is small. That behavior is achieved by means of the factors $U_k(\mathbf{p},\mathbf{q})$ and the decaying Gaussian function on $\Delta E_{\mathbf{pq}}$ with a user-defined standard deviation σ_d. A normalizing factor of two is used in order to make $SV_k(\mathbf{p},\mathbf{q})$ vary from zero to one. The term $\eta_k(\mathbf{q})$ in (17) makes noisy voted pixels, $\mathbf{q}$, to adopt the color of their voting neighbors, $\mathbf{p}$, disregarding local uniformity measurements and perceptual color differences between $\mathbf{p}$ and $\mathbf{q}$. The term $\overline{\eta_k}(\mathbf{q})$ in (17) makes SV'_k vary from zero to one. The effect of $\eta_k(\mathbf{q})$ and $\overline{\eta_k}(\mathbf{q})$ on the strength of the *stick* vote received at a noiseless pixel $\mathbf{q}$ is null.

In turn, a *ball* vote can be seen as a *ball*-shaped tensor, $\mathrm{BT}(\mathbf{p})$, with a strength controlled by the scalar factors $GS(\mathbf{p},\mathbf{q})$, $\overline{\eta_k}(\mathbf{p})$ and $BV_k(\mathbf{p},\mathbf{q})$, each varying between zero and one. The *ball* propagation function, $\mathrm{B}_k(\mathbf{p},\mathbf{q})$, which allows a pixel $\mathbf{p}$ to cast a *ball* vote to a neighboring pixel $\mathbf{q}$ for channel k is given by:

$$\mathrm{B}_k(\mathbf{p},\mathbf{q}) = GS(\mathbf{p},\mathbf{q})\ \overline{\eta_k}(\mathbf{p})\ BV_k(\mathbf{p},\mathbf{q})\ \mathrm{BT}(\mathbf{p}), \tag{18}$$

with $\mathrm{BT}(\mathbf{p})$, $GS(\mathbf{p},\mathbf{q})$, $\overline{\eta_k}(\mathbf{p})$ and $BV_k(\mathbf{p},\mathbf{q})$ being defined as follows. First, the *ball tensor*, represented by the identity matrix, $\mathbf{I}$, is the only possible tensor for $\mathrm{BT}(\mathbf{p})$, since it is the only tensor that complies with the two main design restrictions: a *ball* vote must be equivalent to casting *stick* votes for all possible colors using the hypothesis that all of them are equally likely, and the normalized $\hat{s_1}$ saliency must be zero when only *ball* votes are received at a pixel. Second, $GS(\mathbf{p},\mathbf{q})$ and $\overline{\eta_k}(\mathbf{p})$ are the same as the factors introduced in (15) for the *stick* propagation function. They are included for similar reasons to those given in the definition of the *stick* propagation function. Finally, the

scalar factor $BV_k(\mathbf{p},\mathbf{q})$ is given by:

$$BV_k(\mathbf{p},\mathbf{q}) = \frac{\overline{G_{\sigma_d}}(\Delta E_{\mathbf{pq}}) + \overline{U_k}(\mathbf{p},\mathbf{q}) + \overline{G_{\sigma_d}}(\Delta E^k_{\mathbf{pq}})}{3}, \tag{19}$$

where $\overline{G_{\sigma_d}}(\cdot) = 1 - G_{\sigma_d}(\cdot)$ and $\overline{U_k}(\mathbf{p},\mathbf{q}) = 1 - U_k(\mathbf{p},\mathbf{q})$. $BV_k(\mathbf{p},\mathbf{q})$ models the fact that a pixel $\mathbf{p}$ must reinforce the edginess at the voted pixel $\mathbf{q}$ either if there is a big perceptual color difference between $\mathbf{p}$ and $\mathbf{q}$, or if $\mathbf{p}$ and $\mathbf{q}$ are not in a uniform region. This behavior is modeled by means of $\overline{G_{\sigma_d}}(\Delta E_{\mathbf{pq}})$ and $\overline{U_k}(\mathbf{p},\mathbf{q})$. The additional term $\overline{G_{\sigma_d}}(\Delta E^k_{\mathbf{pq}})$ is introduced in order to increase the edginess of pixels in which the only noisy channel is k, where $\Delta E^k_{\mathbf{pq}}$ denotes the perceptual color difference only measured in the specific color channel k. The normalizing factor of three in (19) allows the *ball* propagation function to cast *ball* votes with a strength between zero and one.

The proposed voting process at every pixel is carried out by adding all the tensors propagated towards it from its neighbors by applying the above propagation functions. Thus, the total vote received at a pixel $\mathbf{q}$ for each color channel k, $\mathrm{TV}_k(\mathbf{q})$, is given by:

$$\mathrm{TV}_k(\mathbf{q}) = \sum_{p \in \mathcal{N}(\mathbf{q})} \mathrm{S}_k(\mathbf{p},\mathbf{q}) + \mathrm{B}_k(\mathbf{p},\mathbf{q}). \tag{20}$$

The voting process is applied twice. The first application is used to obtain an initial estimation of the normalized $\hat{s_1}$ and $\hat{s_2}$ saliencies, as they are necessary to calculate $U_k(\mathbf{p},\mathbf{q})$ and $\eta_k(\mathbf{p})$. For this first estimation, only perceptual color differences and spatial distances are taken into account. At the second application, the tensors at every pixel are initialized with the tensors obtained after the first application. After this initialization, (15) and (18) can be applied in their full definition, since all necessary data are available.

After applying the voting process described above, it is necessary to obtain eigenvectors and eigenvalues of $\mathrm{TV}_L(\mathbf{p})$, $\mathrm{TV}_a(\mathbf{p})$ and $\mathrm{TV}_b(\mathbf{p})$ at every pixel $\mathbf{p}$ in order to analyze its local perceptual information. The voting results can be interpreted as follows: uniformity increases with the normalized $\hat{s_1}$ saliency and edginess increases as the normalized $\hat{s_2}$ saliency becomes greater than the normalized $\hat{s_1}$ saliency. Hence, the map of normalized $\hat{s_2}$ saliencies can be directly used as an edginess map:

$$E(\mathbf{p}) = \sum_{k=1}^{3} \alpha_k \, \hat{s_{2k}}(\mathbf{p}), \tag{21}$$

where $E(\mathbf{p})$ is the edginess at $\mathbf{p}$ and α_k are weights that can be used to modulate the importance of every channel in the estimation of edginess.

The results can be improved by reducing the noise in the image. This denoising step can be achieved by replacing the pixel's color by the most likely normalized noiseless color encoded in its tensors. Similarly to the method

based on the classical tensor voting, this edge detector is expected to yield better results by iterating the process. Experimentally, it has been found that a few iterations (less than five in any case) can yield good results for both noisy and noiseless images.

3.3 Parameters of the CIEDE2000 formula

The CIEDE2000 formula [13], which estimates the perceptual color difference between two pixels $\mathbf{p}$ and $\mathbf{q}$, $\Delta E_{\mathbf{pq}}$, has three parameters, k_L, k_C and k_H, to weight the differences in CIELAB luminance, chroma and hue respectively. They can be adjusted to make the CIEDE2000 formula more suitable for every specific application by taking into account factors such as noise or background luminance, since those factors were not explicitly taken into account in the definition of the formula. These parameters must be greater than or equal to one. The following equations can be used to compute these parameters:

$$k_L = B_L\,\eta_L, \quad k_C = B_C\,\eta_C, \quad k_H = B_h\,\eta_h, \tag{22}$$

where Bm are factors that take into account the influence of the background color on the calculation of color differences for the color component m (L, C and h) and F_{η_m} are factors that take into account the influence of noise on the calculation of color differences in component m. On the one hand, big color differences in chromatic channels become less perceptually visible as background luminance decreases. Thus, the influence of the background on the CIEDE2000 formula can be modeled by $B_L = 1$ and $B_C = B_h = 1 + 3\ (1 - Y_B)$, where Y_B is the mean background luminance. On the other hand, big color differences become less perceptually visible as noise increases. The influence of noise on CIEDE2000 can be modeled by means of $\eta_m = MAD(I)_m - MAD(G)_m$, where I is the image, G is a Gaussian blurred version of I and $MAD(\cdot)_m$ is the median absolute difference (MAD) calculated on component m. In turn, η_m is set to 1 in noiseless regions.

4 Evaluation Methodology

In general, edge detectors apply three steps (cf. Figure 4). First, a filtering step is applied to the input image, since edge detectors are very sensitive to noise. Second, once the input image is noiseless, edge detectors estimate the likelihood of finding an edge for every pixel. The output of this step is an edginess map. Finally, post-processing is applied to the edginess map in order to obtain a binary edge map. The core of the edge detection algorithms embodies only the first two steps, leaving aside the post-processing step, since

the latter is usually application-dependent. In addition, it is not possible to separate the denoising and edginess estimation steps in general, since many algorithms carry out both processes in a unified framework.

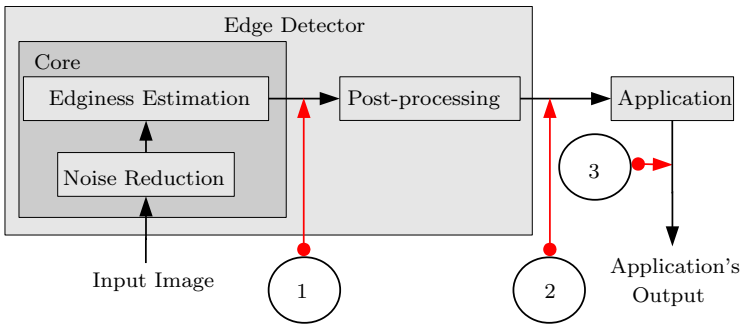

Fig. 4 The edge detection process. The performance of edge detectors can be assessed at the points marked in green.

The performance of edge detection algorithms can be assessed at three different points of the process, as shown in Figure 4. Measurements on edginess maps can be obtained at the first point, on binary edge maps at the second point, and application-dependent measurements can be made at the third point of the figure. Direct measurements at the output of the algorithms are made at the first and second points, while the performance at the third point is indirectly assessed by taking into account the application in which the edge detector is used. Indirect assessment is based on the assumption that the performance of an edge detector used in the context of a specific application is correlated with the general performance of that application. Assessing performance at the first point appears to be advantageous since the core of the edge detectors can be evaluated no matter the context or the applied post-processing. Many evaluation methodologies have been proposed to evaluate performance at the second e.g., [26, 10, 4, 15] and third points [29, 2]. On the other hand, to the best of our knowledge, the methodology introduced in [22] has been the first attempt to measure performance at the first point.

There are mainly four features that can be measured from edge detectors: completeness, discriminability, precision and robustness. Without loss of generality, completeness, discriminability and precision can be measured on non-maximum suppressed edginess maps, here referred to as NMSE maps, since the location of edges must be the same, disregarding the strength given to them by the detector. On the other hand, robustness can be directly assessed on the edginess maps. These features are described in the following paragraphs.

4.1 Completeness

Completeness is the ability of an edge detector to mark all possible edges of noiseless images. Completeness is a desirable feature of general purpose edge detectors since the decision of whether an edge is relevant or not only depends on the application. For instance, applications such as image edge enhancement based on edge detection, edge-based segmentation or texture feature extraction usually give a different relevance to every detected edge. Consequently, an edge detector will reduce its scope when it discards edges. Despite that, most edge detectors usually opt for decreasing their scope of use for the sake of improving their performance in other features, such as discriminability or robustness.

Complete ground-truths with all possible edges must be used to measure completeness. Unfortunately, that kind of ground-truth is not usually available. Thus, recall, the ground-truth dependent counterpart of completeness, can be used to give partial estimations of completeness. Let $D(\mathbf{p}_i)$ be the distance between the i-th pixel in the ground-truth $\mathbf{p}_i$, and its corresponding matching pixel $\mathbf{q}_i$ in the NMSE map or infinity if such a matching pixel does not exist, M and N be the number of marked pixels in the ground-truth and in the NMSE map respectively, and let $\phi(\cdot)$ be a radial decaying function in the range from zero to one. Function $\phi(x) = 1/(1 + (1/9)x^2)$ has been used in the experiments of Section 5. Recall can be estimated through the R-measurement defined as [22]:

$$R = \frac{1}{M} \sum_{i=1}^{M} \phi(D(\mathbf{p}_i)). \tag{23}$$

A problem associated with the measure of recall when $N > M$ is the fact that every edge detector generates a different number of edges. This can give advantage to detectors that generate a larger number of edges, since $D(\mathbf{p}_i)$ tends to be reduced when N increases. This bias can be suppressed by taking the same number of detected edges for all the edge detectors to be compared. This can be done by taking the N' strongest detected edges from the NMSE maps. R vs. N' plots can also be used to analyze the evolution of R.

4.2 Discriminability

Discriminability is the ability of an edge detector to discriminate between relevant and irrelevant edges. This feature is application-dependent since relevance can only be assessed in a specific scope. For example, the discriminability of an edge detector could be high when applied to image edge enhancing or low when applied to edge-based segmentation. Discriminability is one of the

most desirable features of edge detectors since low levels of discriminability make it necessary to use more sophisticated post-processing algorithms that can partially fix the drawbacks of the edge detector. Thus, global thresholding (which is the simplest post-processing) could be used for edge detectors with maximum discriminability.

Discriminability can be measured related to a specific ground-truth through the DS-measurement: the difference between the weighted mean edginess of the pixels that match the ground-truth and the weighted mean edginess of the pixels that do not match it. Let $E(\mathbf{q}_i)$ be the edginess at pixel $\mathbf{q}_i$ of the NMSE map, and $D(\mathbf{q}_i)$ be the distance between $\mathbf{q}_i$ and its matching pixel in the ground-truth or infinity if such a pixel does not exist. The DS-measurement is given by [22]:

$$DS = \frac{\sum_{i=1}^{N} E(\mathbf{q}_i)\,\phi(D(\mathbf{q}_i))}{\sum_{i=1}^{N} \phi(D(\mathbf{q}_i))} - \frac{\sum_{i=1}^{N} E(\mathbf{q}_i)\,(1-\phi(D(\mathbf{q}_i)))}{\sum_{i=1}^{N} 1-\phi(D(\mathbf{q}_i))}. \tag{24}$$

4.3 Precision

Precision measures the ability of an edge detector to mark edges as close as possible to real edges. Precision is mandatory for edge detection, since the performance of applications in which the detectors are used depends on this feature. Unlike discriminability, precision is, in essence, an application-independent feature. However, in practice, application-independent measures of precision are difficult to obtain since complete ground-truths are required. Thus, only precision measurements related to specific ground-truths can be obtained. Ideally, all edges of the ground-truth should be found at distance zero in the NMSE map. However, if hand-made ground-truths are used, it is necessary to take into account that those ground-truths are not precise, since some pixels can appear misplaced due to human errors. Despite that, those ground-truths can still be used to compare edge detectors, since all edge detectors are equally affected by those errors. Let $\overline{D}$ be the mean distance between pixels of the ground-truth and their corresponding matching pixels in the NMSE map. The P-measurement can be used to estimate precision:

$$P = \phi(\overline{D}). \tag{25}$$

Observe that pixels without a matching pixel in the NMSE are not considered for the P-measurement, since they are irrelevant for measuring precision. Notice that based on the definition of function ϕ described in Section 4.1, values of P below 0.69 and above 0.90 correspond to a mean distance between

matching points in the ground-truth and the NMSE above 2 pixels and below 1 pixel, respectively. Indeed, this behavior can be modified by varying function ϕ.

A feature related to precision is the false alarm rejection (FAR) feature, which represents the ability of edge detectors not to detect edges in flat regions. The FAR-measurement is given by:

$$FAR = \frac{1}{N} \sum_{i=1}^{N} \phi(D(\mathbf{q}_i)). \tag{26}$$

This measurement can be used as a numeric, ground-truth dependent estimation of false alarm rejection. Similarly to the R-measurement, the N' strongest detected edges from the NMSE map must be selected before computing the P and FAR-measurements in order to avoid biases related to N when $N > M$. Thus, plots of P vs. N' and FAR vs. N' can also be used to evaluate the evolution of the P and FAR-measurements.

4.4 Robustness

Robustness measures the ability of an edge detector to reject noise. Thus, an ideal robust edge detector should produce the same output for both noisy and noiseless images. Robustness is one of the most difficult features to comply with since edge detection is essentially a differential operation which is usually very sensitive to noise. In fact, most edge detectors include filtering steps in order to improve their robustness. However, most of those filters mistakenly eliminate important details by treating them as noise, thus reducing the completeness and recall features of the detector. Despite that, robustness is usually preferred to completeness.

Since edginess maps can be modeled by means of gray-scale images, measures of image fidelity can be used to measure robustness. The peak signal to noise ratio (PSNR) is the most widely used measure of image fidelity. Although PSNR is not suitable to measure precision [27], it is appropriate to measure robustness. The edge detector is applied to both the noiseless and the noisy version of the same image. The PSNR between both outputs is calculated in order to measure the difference between them. Unlike the aforementioned measurements, it is not necessary to use ground-truths or to apply non-maximum suppression to the edginess maps before computing PSNR.

4.5 Ground-Truths for Edge Detection Assessment

Ground-truths are very important for assessing edge detectors. However, they must be used carefully in order to obtain reliable and fair assessments [24]. Ground-truths can be classified into artificial, manual or generated by consensus. Artificial ground-truths are trivially obtained from synthetic images, manual ground-truths are obtained by manually annotating edges on the images, such as the Berkeley database presented in [14] for image segmentation and ground-truths generated by consensus are obtained from the output of a bank of edge detectors (e.g., [7]).

Artificial ground-truths are not generally used in comparisons since the results can barely be extrapolated to real scenes [4]. In turn, manual ground-truths are useful since edge detector outputs must correlate with the opinion of humans. However, manual ground-truths must be treated as partial ground-truths, since humans annotate edges depending on the instructions given by the experimenters. For example, humans do not usually mark the edges inside a textured region when the instruction is to annotate the edges necessary to separate regions. Thus, a manual ground-truth obtained for image segmentation should not be used for image edge enhancement, for example. Moreover, precision measurements using this kind of ground-truth can only be seen as estimates, since humans are prone to committing precision errors when marking edges. An additional problem is that gray-scale manual ground-truths are almost impossible to obtain for natural scenes [9].

Ground-truths generated by consensus rely on the hypothesis that the bank of edge detectors have a good performance in all contexts. This kind of ground-truth is easy to construct, including gray-scale ground-truths. However, the validity of these ground-truths directly depends on the choice of the bank of edge detectors.

5 Experimental Results

Fifteen images from the Berkeley segmentation data set [14] have been used in the experiments. In addition to the Laplacian of Gaussians (LoG), Sobel and Canny detectors, the methods proposed in [1], referred to as the LGC method, and [28], referred to as the Compass method, have been used in the comparisons, since they are considered to represent the state-of-the-art in color edge detection, and on top of that, implementations are available from their authors. The Compass, LoG and Canny algorithms have been applied with $\sigma = 2$, since the best overall performance of these algorithms has been attained with this standard deviation. Three iterations of the proposed methods have been run. The parameters of the method based on the classical tensor voting, referred to as CTV, have been set to $\sigma = 1.3$ and $b = 1$, while parameters $\sigma_s = 1.3$ and $\sigma_d = 2.5$ have been chosen for the edge

detector based on the denoiser described in Section 3, referred to as TVED. In addition, $\alpha_k = 1$ for all k. The efficient implementation proposed in [20] has been used for CTV.

Three ground-truths have been considered in the experiments: the NMSE map generated by the Prewitt's edge detector, a computer generated consensus ground-truth [7] and the hand-made ground-truth of the Berkeley segmentation data set [14]. We will refer to those ground-truths as GT1, GT2 and GT3, respectively. It is important to remark that the validity of GT3 has only been proven in segmentation related applications. We matched every detected edge to its closest pixel in the ground-truth, allowing for up to one match for every ground-truth pixel. Gaussian noise with different standard deviations has been added to the input images for the robustness analysis.

Table 1 Performance measurements for ground-truths GT1 and GT2. The best performance per column is marked in bold.

Method	R		DS		P		FAR	
	GT1	GT2	GT1	GT2	GT1	GT2	GT1	GT2
LoG	0.44	0.68	9.45	38.81	0.93	0.63	0.90	0.51
Sobel	**0.84**	**0.75**	9.89	34.07	0.94	0.88	0.84	0.74
Canny	0.23	0.29	7.98	39.65	0.96	0.79	0.93	0.74
LGC	0.15	0.40	4.46	**44.33**	0.96	0.85	0.93	0.78
Compass	0.57	0.61	8.62	41.10	0.93	0.71	0.89	0.57
CTV	0.23	0.34	7.17	21.66	**0.98**	**0.92**	**0.95**	**0.87**
TVED	0.20	0.53	**21.24**	34.39	**0.98**	0.75	**0.95**	0.69

Table 1 shows the performance of the different methods for GT1 and GT2. Evolution plots for the proposed performance measurements are not necessary for GT1 and GT2 since $M \geq N$ for them.

Regarding GT1, all the tested algorithms have a good precision and false alarm rejection but a poor discriminability. Although TVED has a better performance in discriminability than the others, some applications could require even better results. Only the Sobel detector has a good performance in recall. This result was expected since Prewitt detects significantly more edges than the other tested edge detectors, with the exception of the Sobel detector. TVED is the best method with respect to the other three measurements.

Regarding GT2, LGC is the best algorithm according to discriminability. However, LGC shows a poor performance in recall. On the other hand, CTV is the best in precision and false alarm rejection. However, CTV shows a poor performance in recall. Thus, LGC and CTV relinquish better recall figures for the sake of discriminability, and precision and false alarm rejection respectively. The Sobel detector is the best in recall and has a competitive performance in the other measures. Only LoG has a P value below 0.69, which means that it is the only method where the mean distance between the matched points in the ground-truth and the NMSE is above 2 pixels.

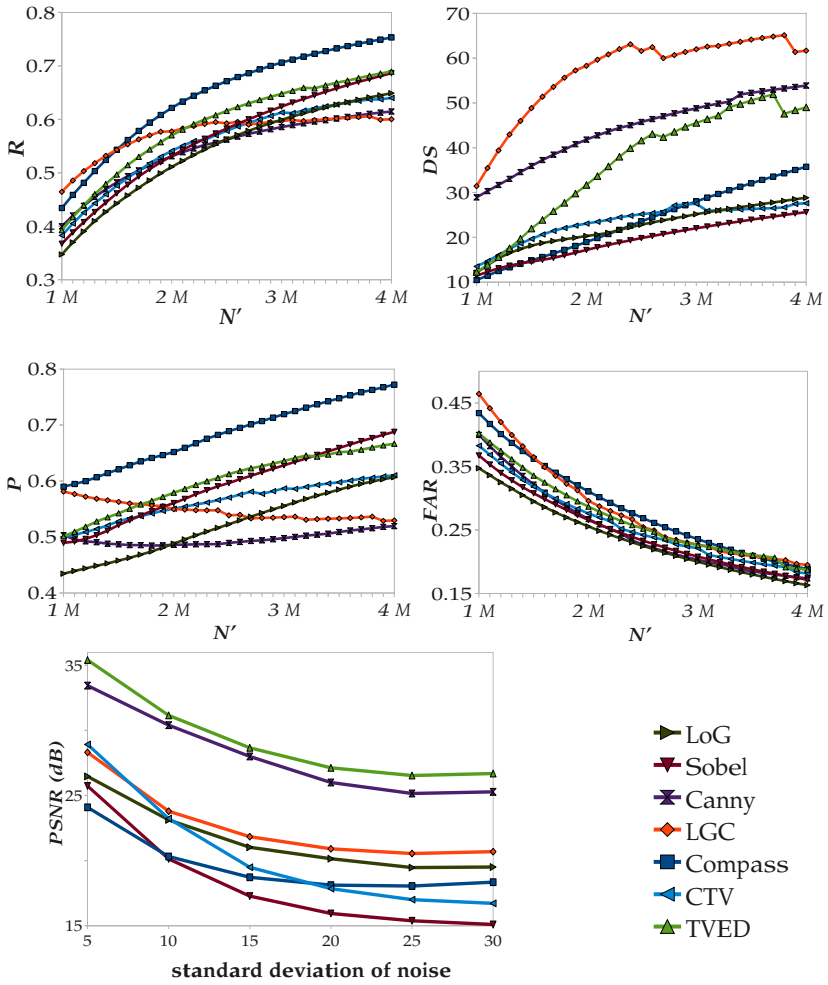

Fig. 5 Performance measurements for GT3. Top left: R vs. N' (recall); top right: DS vs. N' (discriminability); middle left: P vs. N' (precision); middle right: FAR vs. N' (false alarm rejection); bottom left: $PSNR$ vs. standard deviation of noise (robustness); bottom right: conventions.

Figure 5 shows the evolution of the proposed performance measurements for GT3 with N' and the robustness analysis. It can be observed that the Compass detector has the best evolution for the R and P-measurements, the LCG is the best for the DS-measurement and both have a similar performance with respect to FAR. The performance of the Sobel and LoG detectors is the worst, but it increases with N' even surpassing LGC in R and P for large values of N'. The proposed methods have competitive results, especially TVED. For example, unlike LGC, TVED has an increasing trend with N',

and outperforms Compass in *DS*. TVED is the most consistent method for GT3 as its performance is usually in the top three of the tested methods with respect to these four measurements. Although Canny has been outperformed in all measurements, it is still competitive in *DS*.

As for the robustness analysis, original images have been contaminated with additive white Gaussian noise (AWGN) with different standard deviations. TVED appears to be the most robust algorithm with around 1 dB above Canny, and 7dB above LGC. CTV has a good performance with low amounts of noise, but it rapidly decreases due to the appearance of artifacts (cf. Figures 6 and 7). This could mean that denoising and detecting edges at the same time seems a better alternative than iterating tensor voting in noisy scenarios. The LoG, Sobel and Compass detectors are more sensitive to noise.

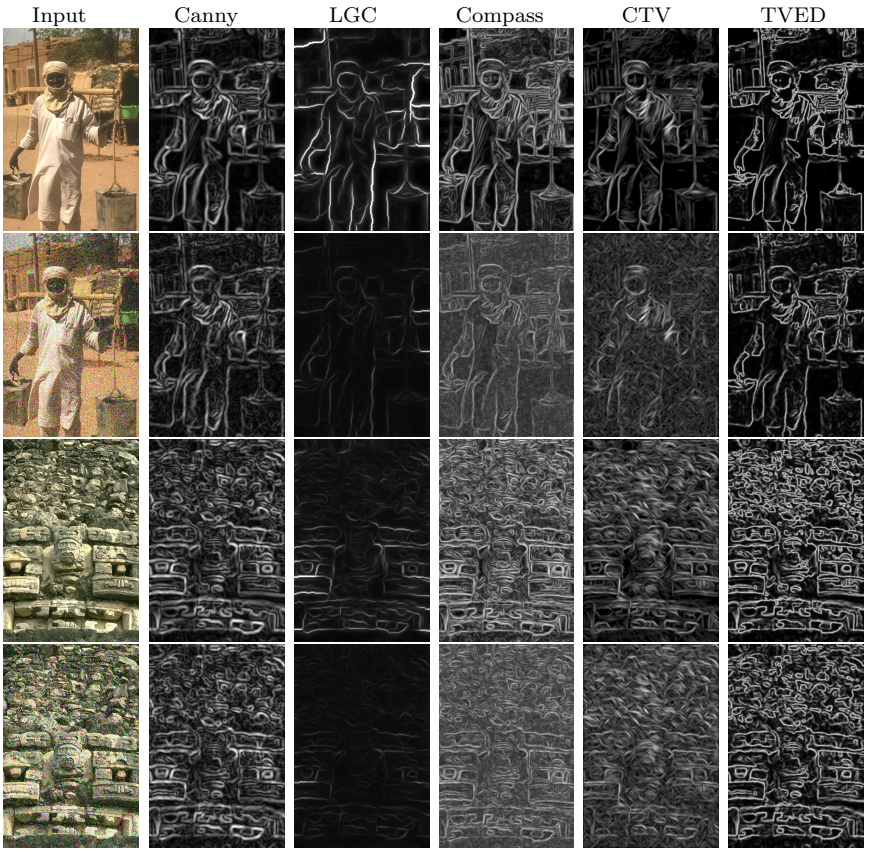

Fig. 6 Visual comparison of results. First column: original images and their noisy counterparts. Columns three to six: edginess maps generated by the Canny, LGC, Compass, CTV and TVED methods respectively for the corresponding images.

A visual comparison can also give noteworthy information of the properties of the tested edge detectors. Figures 6 and 7 show the edginess maps detected for some of the tested images and their noisy counterparts. The noisy images have been generated by adding AWGN with a standard deviation of thirty. This standard deviation of noise aims at simulating very noisy scenarios.

It can be appreciated that LGC generates fewer edges than the others, but also misses some important edges and their strength is reduced for the noisy images. The Compass operator generates too many edges and the number of edges increases with noise. CTV yields good results for noiseless images. However, its performance is largely degraded for noisy images, where undesirable cross-shaped artifacts are generated. This is mainly due to the fact that CTV is more prone to detecting straight lines by mistakenly joining noisy pixels. TVED and Canny have a better behavior, since they only detect the most important edges and are less influenced by noise. However, TVED generates sharper edges than Canny.

Regarding computational cost, the Sobel detector was the fastest of all tested algorithms when run on an Intel Core 2 Quad Q6600 with a 4GB RAM (0.06 seconds), followed by Canny (0.15 seconds), LoG (0.35 seconds), Compass (around 20 seconds), CTV (around 25 seconds), TVED (around 40 seconds). The slowest detector by far was LGC (2 minutes and 36 seconds).

Table 2 Examples of selection of edge detector.

Application	Feature	Best tested method
Image Segmentation	Discriminability	LGC
Image Segmentation	Precision	Compass
Image Segmentation	All	TVED
Image Edge Enhancement	Recall	Sobel
Image Edge Enhancement	Precision	CTV
Any	Robustness	TVED and Canny
Any	Speed	Sobel, Canny and LoG

6 Concluding Remarks

Two new methods for edge detection based on tensor voting have been presented: the first method based on the classical tensor voting, and the latter based on an adaptation of the tensor voting procedure. The evaluation has been performed by measuring the features of completeness, discriminability, precision and robustness of edge detectors.

Experimental results show that tensor voting is a powerful tool for edge detection. On the one hand, TVED has been found to be more robust than the state-of-the-art methods while having a competitive performance in recall, discriminability, precision, and false alarm rejection with respect to three

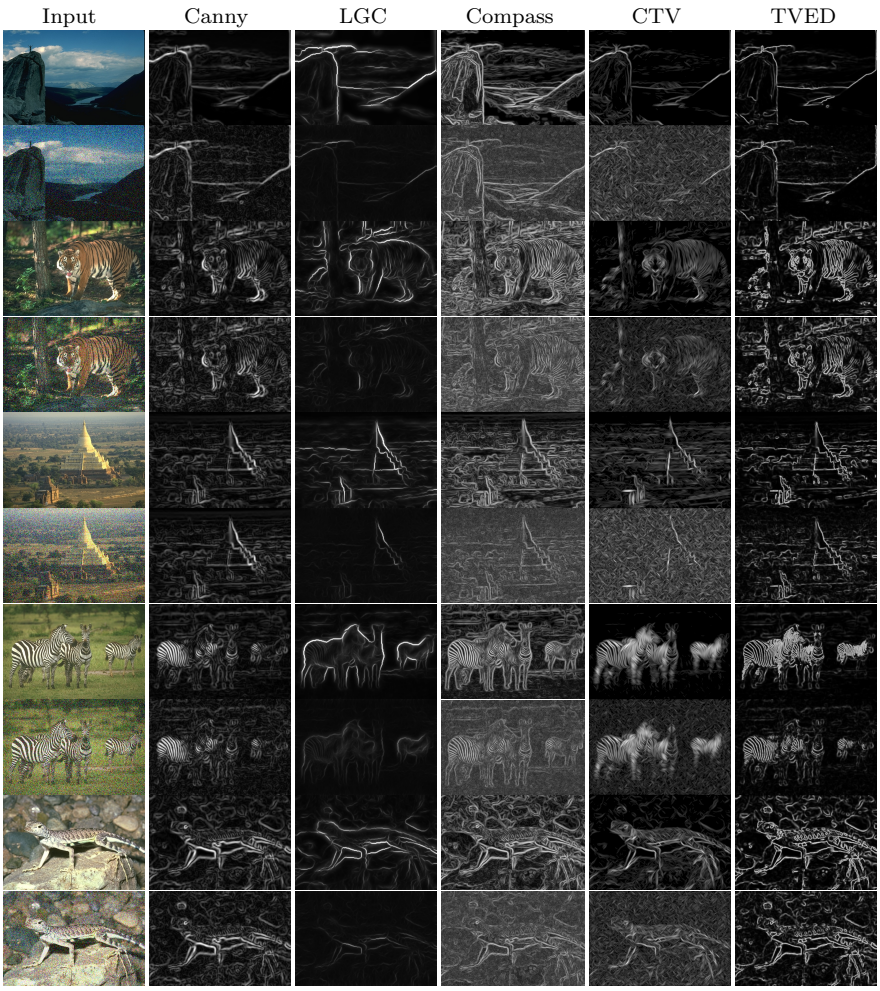

Fig. 7 Visual comparison of results. First column: original images and their noisy counterparts. Columns three to six: edginess maps generated by the Canny, LGC, Compass, CTV and TVED methods respectively for the corresponding images.

different ground-truths. TVED was the most consistent of the tested methods for image segmentation since, unlike other methods, it was usually in the top three of the tested methods under all measurements. CTV has demonstrated good properties of edge detection in both noiseless and images with a small amount of noise.

The results also show that it is difficult for an edge detector to have a good performance for all the features and applications. This means that every edge detector has strengths and weaknesses that makes it more suitable for some applications than for others under a specific measure. This fact should be

taken into account in order to choose the most appropriate edge detector for every context. For instance, Table 2 shows some examples of which method among the tested methods should be chosen for some particular scenarios.

Acknowledgements This research has been supported by the Swedish Research Council under the project VR 2012-3512.

References

1. Arbelaez, P., Maire, M., Fowlkes, C., Malik, J.: Contour detection and hierarchical image segmentation. IEEE Transactions on Pattern Analysis and Machine Intelligence **33**(5), 898–916 (2011)
2. Baker, S., Nayar, S.K.: Global measures of coherence for edge detector evaluation. In: Proc. IEEE Conf. on Computer Vision and Pattern Recognition, pp. II:373–379 (1999)
3. Batard, T., Saint-Jean, C., Berthier, M.: A metric approach to nD images edge detection with Clifford algebras. Journal of Mathematical Imaging and Vision **33**(3), 296–312 (2009)
4. Bowyer, K., Kranenburg, C., Dougherty, S.: Edge detector evaluation using empirical ROC curves. Computer Vision and Image Understanding **84**(1), 77–103 (2001)
5. Canny, J.F.: A computational approach to edge detection. IEEE Transactions on Pattern Analysis and Machine Intelligence **8**(6), 679–698 (1986)
6. De Micheli, E., Caprile, B., Ottonello, P., Torre, V.: Localization and noise in edge detection. IEEE Transactions on Pattern Analysis and Machine Intelligence **11**(10), 1106–1117 (1989)
7. Fernández-García, N., Carmona-Poyato, A., Medina-Carnicer, R., Madrid-Cuevas, F.: Automatic generation of consensus ground truth for the comparison of edge detection techniques. Image and Visual Computing **26**(4), 496–511 (2008)
8. Förstner, W.: A feature based correspondence algorithm for image matching. In: Int. Archives of Photogrammetry and Remote Sensing, vol. 26, pp. 150–166 (1986)
9. Heath, M., Sarkar, S., Sanocki, T., Bowyer, K.: Comparison of edge detectors: A methodology and initial study. Computer Vision and Image Understanding **69**(1), 38–54 (1998)
10. Heath, M., Sarkar, S., Sanocki, T., Bowyer, K.W.: A robust visual method for assessing the relative performance of edge-detection algorithms. IEEE Transactions on Pattern Analysis and Machine Intelligence **19**(12), 1338–1359 (1997)
11. Koschan, A.: A comparative study on color edge detection. In: Proc. Asian Conf. on Computer Vision, pp. 574–578 (1995)
12. Koschan, A., Abidi, M.: Detection and classification of edges in color images. IEEE Signal Processing Magazine **22**(1), 64–73 (2005)
13. Luo, M.R., Cui, G., Rigg, B.: The development of the CIE 2000 colour-difference formula: CIEDE2000. Color Research and Application **26**(5), 340–350 (2001)
14. Martin, D., Fowlkes, C., Tal, D., Malik, J.: A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In: Proc. IEEE Int. Conf. Computer Vision, pp. II:416–423 (2001)
15. Martin, D.R., Fowlkes, C.C., Malik, J.: Learning to detect natural image boundaries using local brightness, color and texture cues. IEEE Transactions on Pattern Analysis and Machine Intelligence **26**(1), 530–549 (2004)
16. Medioni, G., Lee, M.S., Tang, C.K.: A Computational Framework for Feature Extraction and Segmentation. Elsevier Science (2000)

17. Moreno, R., Garcia, M.A., Puig, D., Julià, C.: On adapting the tensor voting framework to robust color image denoising. In: Proc. Int. Conf. on Computer Analysis of Images and Patterns, Lecture Notes in Computer Science 5702, pp. 492–500 (2009)
18. Moreno, R., Garcia, M.A., Puig, D., Julià, C.: Robust color edge detection through tensor voting. In: Proc. IEEE Int. Conf. on Image Processing, pp. 2153–2156 (2009)
19. Moreno, R., Garcia, M.A., Puig, D., Julià, C.: Edge-preserving color image denoising through tensor voting. Compuer Vision and Image Understanding **115**(11), 1536–1551 (2011)
20. Moreno, R., Garcia, M.A., Puig, D., Pizarro, L., Burgeth, B., Weickert, J.: On improving the efficiency of tensor voting. IEEE Transactions on Pattern Analysis and Machine Intelligence **33**(11), 2215–2228 (2011)
21. Moreno, R., Pizarro, L., Burgeth, B., Weickert, J., Garcia, M.A., Puig, D.: Adaptation of tensor voting to image structure estimation. In: B. Burgeth, D. Laidlaw (eds.) New Developments in the Visualization and Processing of Tensor Fields, pp. 29–50. Springer (2012)
22. Moreno, R., Puig, D., Julià, C., Garcia, M.A.: A new methodology for evaluation of edge detectors. In: Proc. IEEE Int. Conf. on Image Processing, pp. 2157–2160 (2009)
23. Nguyen, T.B., Ziou, D.: Contextual and non-contextual performance evaluation of edge detectors. Pattern Recognition Letters **21**(9), 805–816 (2000)
24. Papari, G., Petkov, N.: Edge and line oriented contour detection: State of the art. Image and Vision Computing **29**(2–3), 79–103 (2011)
25. Plataniotis, K., Venetsanopoulos, A.: Color Image Processing and Applications. Springer (2000)
26. Pratt, W.K.: Digital Image Processing: PIKS Scientific Inside, fourth edn. Wiley-Interscience (2007)
27. Prieto, M., Allen, A.: A similarity metric for edge images. IEEE Transactions on Pattern Analysis and Machine Intelligence **25**(10), 1265–1273 (2003)
28. Ruzon, M., Tomasi, C.: Edge, junction, and corner detection using color distributions. IEEE Transactions on Pattern Analysis and Machine Intelligence **23**(11), 1281–1295 (2001)
29. Shin, M.C., Goldgof, D.B., Bowyer, K.W., Nikiforou, S.: Comparison of edge detection algorithms using a structure from motion task. IEEE Transactions on Systems, Man, and Cybernetics—Part B: Cybernetics **31**(4), 589–601 (2001)
30. Smolka, B., Venetsanopoulos, A.: Noise reduction and edge detection in color images. In: R. Lukac, K.N. Plataniotis (eds.) Color Image Processing: Methods and Applications, pp. 88–120. CRC Press (2006)
31. Spreeuwers, L.J., van der Heijden, F.: Evaluation of edge detectors using average risk. In: Proc. Int. Conf. on Pattern Recognition, vol. 3, pp. 771–774 (1992)
32. Xue-Wei, L., Xin-Rong, Z.: A perceptual color edge detection algorithm. In: Proc. Int. Conf. on Computer Science and Software Engineering, vol. 1, pp. 297–300 (2008)
33. Zhu, S.Y., Plataniotis, K.N., Venetsanopoulos, A.N.: Comprehensive analysis of edge detection in color image processing. Optical Engineering **38**(4), 612–625 (1999)