A Similarity Metric for Edge Images

Miguel Segui Prieto and Alastair R. Allen, *Member*, *IEEE*

Abstract—The performance of several discrepancy measures for the comparison of edge images is analyzed and a novel similarity metric aimed at overcoming their problems is proposed. The algorithm finds an optimal matching of the pixels between the images and estimates the error produced by this matching. The resulting Pixel Correspondence Metric (PCM) can take into account edge strength as well as the displacement of edge pixel positions in the estimation of similarity. A series of experimental tests shows the new metric to be a robust and effective tool in the comparison of edge images when a small localization error of the detected edges is allowed.

Index Terms—Edge detection, image similarity, performance evaluation, pixel correspondence metric, weighted matching in bipartite graphs, matching problem, assignment problem.

1 Introduction

During the last few years, the number of applications of automated image analysis has increased considerably. Many of these applications require very fast processing of the images and it is often necessary to modify the algorithms of an application to reduce their execution time. In most cases, such modifications introduce a certain amount of error in the results obtained. In order to decide whether an implementation is suitable or not, the quality of the results has to be carefully studied by comparing the obtained image (distorted) with the desired image (undistorted) and a compromise between the speed of an algorithm and the quality of its results must be reached.

Many distortion metrics used to evaluate the similarity between two images belong to the group of *pixel-based metrics*. This kind of metric measures the difference between the pixels corresponding to every position of the undistorted and the distorted image. The most popular distortion metrics in the field of image processing are the *Signal to Noise Ratio* (*SNR*) and the *Peak Signal to Noise Ratio* (*PSNR*). In spite of their popularity, the simplicity of these metrics may in some cases result in misleading quantitative distortion measurements. This problem may occur, for example, during the construction of an application for image segmentation in which an early stage of the process is the detection of the edges from the original images. It could be the case that, in such an application, specific requirements in the execution of this stage force the designer to choose between different edge detection algorithms. In order to choose which algorithmic implementation should be included, it would be necessary to observe the edges that are identified by the different algorithms. The position of the edges in this application would not be crucial since maybe a small error in their location would possibly have little influence on the extraction of features from the processed images. In this example, if the edge locations were slightly offset, pixel-based metrics would return a very high error, making it difficult for the designer of the application to decide which algorithmic modification to apply.

Many algorithms have been proposed to evaluate the performance of image segmentation techniques [1], [2], [3], [4], [5], [6], [7]. Most of these methods study the location, size, and shape of the objects of the image by comparing the edges obtained by the segmentation algorithm with a ground-truth (GT) image. The GT image is obtained by manually defining the position of the ideal edges and areas of interest in the original image. From the comparison with the GT, the three main factors in the performance evaluation are: the detection rate, false alarm rate, and localization error for the detected edges. Discrepancy measures which do not take into account the localization error for the detected edges may give equal discrepancy values for images segmented differently. Consequently, a certain localization error is tolerated, allowing a detected edge pixel not being at exactly the same pixel location as the GT edge pixel to be declared as a correct detection.

A commonly used discrepancy measure which allows a certain localization error is the *Figure Of Merit* (*FOM*) proposed by Pratt [8] (see Section 3.2). One of the problems with *FOM* is that the measure allows more than one detected edge pixel to correspond to the same GT edge pixel, a circumstance which might produce misleading discrepancy values (the problems of this metric can be found in [3]). Similarly, the distance transform-based technique [5] allows a multiple-to-one correspondence between the detected edge pixels and the GT edge pixels, thus giving unrealistic measured values [7]. Considering the sort of problems that this correspondence produces, it is preferable to restrict to one the number of detected edge pixels to be associated with each GT edge pixel.

Following this idea, Bowyer et al. [9] proposed a one-to-one matching between the edge pixels obtained by an edge operator and the edge pixels of the GT. In their work, when an edge pixel has multiple potential matches, they choose the closest-distance match (see Section 3.1), although they mention the existence of several other methods that could be used instead. Bowyer et al. mention that, in their experience, the method used in the matching changes the

- *M.S. Prieto is with the School of Engineering & Physical Sciences, University of Aberdeen, Fraser Noble Building, Aberdeen, AB24 3UE, Scotland, UK. E-mail: m.segui@abdn.ac.uk.*
- *A.R. Allen is with the School of Engineering & Physical Sciences and the Department of Bio-Medical Physics & Bio-Engineering, University of Aberdeen, Aberdeen, AB24 3UE, Scotland, UK. E-mail: a.allen@abdn.ac.uk.*

Manuscript received 31 July 2002; revised 5 Mar. 2003; accepted 22 Apr. 2003.
Recommended for acceptance by R. Sharma.
For information on obtaining reprints of this article, please send e-mail to: tpami@computer.org, and reference IEEECS Log Number 117046.

count of the false and true edge pixels slightly, but does not affect the overall evaluation of edge detectors.

Liu and Haralick [7] present a method based on combinatorial assignment problems to deal with the association of GT edge pixels with the detected edge pixels. The idea of adapting the solution from a combinatorial problem to find the optimal matching of the edges seems the most appropriate method to find the best solution. Nevertheless, in their experiments, they apply the optimal matching algorithm only after a first step designed to resolve the simple cases. The problem is that this first step misclassifies some matches making the overall solution suboptimal. In the first step of their algorithm, when a detected edge pixel has exactly the same pixel location as a GT edge pixel, then it assumes that the edge pixel has been correctly identified and, therefore, it automatically matches them. A case in which this assumption causes misclassification of pixels can be observed in the following example:

$$GT_f = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} EM_f = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix},$$

where GT_f is the ideal edge map (GT) of an image f, EM_f is the detected edge map of f to be evaluated and the 1s in the matrices indicate the position of the edges. In this example, the first step of the algorithm would match the common edge pixel from the two images and attempt to match the rest of the edges by applying the optimal matching algorithm. If the matching algorithm was set to allow a maximum localization error of distance one (threshold $\tau = 1$ in [7]), then the result of the matching would identify only one correctly detected edge pixel, one misdetection, and one false alarm, when it is clear that both edge pixels have just been displaced diagonally one pixel and should be classified as correctly detected.

The contents of this paper present the work carried out toward the development of a reliable metric for the evaluation of the similarity between edge images, allowing a small localization error in the detected edges. The idea is that the metric must first find an optimal matching (using tools obtained from graph theory) of the pixels between the edge images and then estimate the error produced by this matching. In contrast with [7], our method to find the optimal matching has been designed to take into account not only the distance between the edges to be matched, but also the difference of edge strength value between them. The reason for this constraint is that it is sometimes useful to work with gray-level edge maps instead of binary edge maps. Some edge operators first detect potential edges and then apply some sort of thresholding to eliminate those which are not relevant. This stage may leave some *weak edges* which, even though they are only just over the lower threshold, are not relevant to the analysis of the image. The problem with the weak edges is that even though their strength is very low, they become as important as the relevant edges once the edge maps have been binarized. It is for this reason that using gray-level edge maps allows the metric to identify the weak edges and return a more reasonable estimate of the similarity between the images. Most discrepancy measures ignore the edge strength, discarding useful information. We will show that finding an optimal one-to-one correspondence between the edge images, using the strength of the edge pixels, is a useful tool that could also be used as a supplement to other existing measures.

Two experiments are presented in this report in which the metric we propose is compared with the traditional metric *PSNR*, the Figure of Merit, and a metric using the closest-distance matching. The experiments have been performed using gray-level edge maps as well as binary in order to analyze the implications of using both kinds of images and to study the significance of the error introduced by non-optimal matchings. The purpose of these measurements is not to evaluate different edge detectors, but to test the new metric as a means of comparing edge images.

The remainder of this paper is organized as follows: The next section presents the theoretical study of a metric for the estimation of similarity between edge images and a possible implementation. Section 3 contains the definition of the discrepancy measures used in the experiments. Section 4 describes the two experiments designed to evaluate the performance of the metric introduced in Section 2. An analysis of the results obtained from the experiments is included in Section 5. Finally, Section 6 draws some conclusions from the results obtained in the previous sections.

2 Pixel Correspondence Metric

This section presents the formal description and a possible implementation of a new metric, named *Pixel Correspondence Metric* (*PCM*), for the estimation of similarity between edge images.

The general idea of the algorithm is to take every pixel $f(i,j)$ from one of the images and find its match within a neighborhood of radius η of the corresponding pixel $g(k,l)$ in the other image. The pixels from both images that have not found any match are measured as errors. Possibly, there are some pixels from f that have more than one possible correspondence in g. In this case, the match is chosen from the different possibilities so that the overall similarity estimate between the two images $PCM(f,g)$ is maximized.

Considering that the metric takes into account both, the distance and the strength of the pixels to be matched, a single value representative of the error in relation to these two factors must be assigned, instead of the method (for binary images) of counting true and false edge pixels. More specifically, the error introduced by false edge pixels (unmatched) is their actual strength (value of the edges) and the error introduced by true pixels (matched) is in direct relation to the spatial distance between the matched pixels and their difference in strength.

2.1 Theoretical Algorithm

Let f and g be two images of dimensions $X \times Y$ with pixel values between 0 and Z. We define the separation S between two pixels in positions (i,j) and (k,l) as:

$$S((i,j),(k,l)) = E(\max(|k-i|,|l-j|)), \tag{1}$$

where $i,j,k,l \in \mathbb{N}$, $i,k \in [0,M]$, $j,l \in [0,N]$, and $E(d)$ is a normalized function ($E(d) \in [0,1]$, $E(d) \in \mathbb{R}$) that represents a weighting dependent on the chess board distance d (8-connectivity) between the pixels. The function $E(d)$ used in the experiments in Section 4 is: $E(d) = (1, 0.9, 0.69, 0.5) \mid d = 0 \ldots 3$. By using these numbers, the cost of matching two pixels of exactly the same value (at any distance) is the same as the error that *FOM* would estimate for these two pixels when using $a = 1/9$ (see Section 3.2).

The similarity between two images f and g can be estimated from the cost of the optimal matching $\mathcal{M}_{opt}(f,g)$ between their pixels. An optimal matching is a matching ($\mathcal{M}$) with minimal cost among all possible matchings. We define the cost of a possible match between a pixel $f(i,j)$ and a pixel $g(k,l)$ as:

$$C(f(i,j),g(k,l)) = 1 - S((i,j),(k,l))\left(1 - \frac{|f(i,j) - g(k,l)|}{Z}\right). \quad (2)$$

The cost of a match between two pixels will always be a real number between 0 and 1 and is normalized so that it equals zero for a perfectly detected edge pixel and is directly proportional to the difference of their magnitude and to the distance between them. For example, the difference between two pixels $f(3,35) = 140$ and $g(5,36) = 130$ from images with 256 levels of gray would be:

$$\begin{aligned} C(f(3,35),g(5,36)) &= 1 - S((3,35),(5,36)) \\ &\quad \cdot \left(1 - \frac{|140-130|}{255}\right) \\ &= 1 - E(2)\left(1 - \frac{10}{255}\right) \\ &= 1 - 0.69(0.961) \\ &= 0.337. \end{aligned}$$

Considering that a matching $\mathcal{M}$ between two images cannot always find a pair for every pixel, the cost of $\mathcal{M}$ must not only count the cost of every match (μ) included in it ($\sum C(\mu)$, $\forall \mu \in \mathcal{M}$), but also the cost of all the pixels left unmatched from both images, where the cost of an unmatched pixel is the value of the pixel itself. Finally, we can define the PCM similarity value (percentage) between two images f and g as:

$$PCM_\eta(f,g) = 100\left(1 - \frac{\mathcal{C}(\mathcal{M}_{opt}(f,g))}{(|f \cup g|)}\right), \quad (3)$$

where $\mathcal{C}(\mathcal{M}_{opt}(f,g))$ is the cost of an optimal matching between the images, η is the maximum localization error allowed between the pixels and $|f \cup g|$ is the total number of pixels that are not zero in f or in g. The search for the optimal matching (the matching with minimal cost) is a rather complex process. Fortunately, there is a similar kind of problem in graph theory which may be used in our method. This kind of problem is known as weighted matching in bipartite graphs.

2.2 Weighted Matching in Bipartite Graphs

Let $\mathcal{G}(V,E)$ be an undirected graph with vertex set V and edge set E, where $|V| = n$ and $|E| = m$. The graph $\mathcal{G}(V,E)$ is bipartite if the vertex set V can be partitioned into two sets V^+ and V^-. A *matching* $\mathcal{M}$ is a subset of E such that no two edges share a vertex. A vertex is *matched* if it is incident to an edge in $\mathcal{M}$ and *unmatched* otherwise. Similarly, an edge is *matched* if it is contained in $\mathcal{M}$ and *unmatched* otherwise. An *alternating path* (with respect to $\mathcal{M}$) in $\mathcal{G}(V,E)$ is a path whose edges are alternately in $E \backslash \mathcal{M}$ and $\mathcal{M}$. An *augmenting path* (with respect to $\mathcal{M}$) is an alternating path between two unmatched vertices.

If we are able to find a valid representation of our system as a graph, then the problem of finding $\mathcal{M}_{opt}$ is equivalent to adapting to our system any algorithm able to find the solution over the graph. In the representation of our system as a graph, the vertex set V contains all the pixels from the images f and g so that the pixels from f are contained in the set V^+ and the pixels from g in V^-, thus forming a bipartite graph. An edge between two vertices exists if $\max(|k-i|,|l-j|) \leq \eta$, $f(i,j)$ and $g(k,l)$ being the pixels equivalent to the vertices under study. The cost of a match between $f(i,j)$ and $g(k,l)$ is represented by assigning a weight $W = \lceil C(f(i,j),g(k,l))Z \rceil$ to the edge between the corresponding vertices.

Similarly to Section 2.1, we define the cost of a matching $\mathcal{M}$ as the accumulated value of all the weights of the matched edges in $\mathcal{M}$ plus the accumulated value of all the unmatched vertices (the value of an unmatched vertex is the value of the pixel that the vertex represents). In this way, we define an *optimal matching* $\mathcal{M}_{opt}$ as the matching with the minimum cost among all the possible matchings in $\mathcal{G}(V,E)$.

The first algorithm that solved in polynomial time the problem of finding the optimal matching of bipartite graphs was created by Kuhn [10]. After that, other algorithms with better performance have been developed using newer techniques (a review of several methods available can be found in [11]). In particular, we shall consider the algorithm conceived by Gabow and Tarjan [12], which has a complexity of $O(m\sqrt{n}\log(nU))$, where U is the greatest absolute value among edge weight values. In order to decide whether this algorithm can be applied to our system, we need to describe its complexity in terms of the parameters of our original problem (X, Y, η, and Z). In this way, the number of vertices n would become the number of pixels from both images ($2XY$), the number of edges m would become the total number of pixels times the number of neighbors of each pixel ($4\eta(\eta+1)$), and U would become Z (as W returns an integer value between 0 and Z). Consequently, the complexity of finding $\mathcal{M}_{opt}$ would be

$$\begin{aligned} &O\big(m\sqrt{n}\log(nU)\big) \\ &= O\Big(8XY\eta(\eta+1)\sqrt{2XY}\log(2XYZ)\Big) \\ &= O\left(\eta^2\sqrt{(XY)^3}\log(XYZ)\right). \end{aligned}$$

Expressed in numbers, the time required to find $\mathcal{M}_{opt}$ for a system with parameters $X = 256$, $Y = 256$, $\eta = 2$, and $Z = 256$ would be approximately 2^{28} phases, where each phase takes the time equivalent to finding an augmenting path in $\mathcal{G}$.

Considering that the algorithm is very time-consuming and it requires much memory to execute, it might not be appropriate for some applications. For this reason, we propose a method that reduces the complexity of the problem. The method included in the next section produces an approximation to the optimal matching and it is the method used in the experiments.

2.3 Approximation to the Weighted Matching Algorithm

Bearing in mind the problems stated above, an algorithm that searches for an approximation to the optimal matching has been designed. The main idea is to divide the graph into sections (subgraphs) and find their optimal matching separately. The size of the subgraphs is determined by the depth s of the recursive algorithm that constructs the subgraphs.

First, the main algorithm inspects the image f for unmatched pixels and, if it finds one, it calls the recursive algorithm, passing this pixel and the depth s. When the recursive algorithm receives this pixel, it marks the pixel as visited (avoiding possible loops in the recursive calls), inspects the neighbors in the other image, and performs a recursive call for every possible match with the depth decreased by one $(s-1)$. When the recursive algorithm reaches depth 0, the algorithm goes back in the recursivity, keeping track of all the visited pixels and constructing the subgraph formed by these pixels. Once the subgraph has been created, the main algorithm finds the optimal matching of this subgraph and continues inspecting f for more unmatched pixels. Given the reduced size of the subgraphs, any of the algorithms to find the optimal matching [11] can now be used.

In the experiments presented in this paper, the depth s has been fixed at five. As shown in the results, this depth is already sufficient to provide a good estimate of the similarity between the images, although a bigger depth could be used if memory and speed requirements allowed it.

3 Discrepancy Measures

3.1 CDM

The first metric to be compared with PCM is the *Closest Distance Metric* (*CDM*). This metric follows the method by Bowyer et al. [9] to count the number of edge pixels correctly identified and the number of false detections. Basically, the idea is that each edge pixel in the detected edge map inspects the reference image for a possible match, even if displaced by a small distance. If there is no possible match, the edge pixel is left unmatched and, if it has multiple potential matches, the edge pixel from the reference image that is closest to the detected edge pixel is chosen.

In order to compare the performance of this matching method with the method used in PCM, the present work has extended the CDM method to work with edge maps containing edge strength information. In this way, if an edge pixel has two possible matches at equal distance, the match that is more similar in strength is chosen. Then, by calculating the cost of this matching, a similarity between the images can be estimated in a similar way to PCM:

$$CDM_\eta(f,g) = 100\left(1 - \frac{\mathcal{C}(\mathcal{M}_{cd}(f,g))}{(|f \cup g|)}\right), \tag{4}$$

where η is the neighborhood radius used in the matching, $\mathcal{C}(\mathcal{M}_{cd}(f,g))$ is the cost of the matching found using the closest-distance criterion, and $|f \cup g|$ is the total number of pixels that are not zero in f or in g.

3.2 FOM

The second metric is the *Figure of Merit*, a widely used discrepancy measure for edge images:

$$FOM = \frac{1}{I_N}\sum_{i=1}^{I_A}\frac{1}{1 + a \times d^2(i)}, \tag{5}$$

where $I_N = \max(I_I, I_A)$ and I_I and I_A represent the number of ideal and actual detected edge pixels, a is a scaling constant (usually $a = 1/9$), and $d(i)$ is the separation distance between an actual edge pixel i and its correct position in the GT. Actually, to make it easier to compare the results with PCM, the value included in the results corresponds to 100 FOM, which is the percentage of similarity between the images.

3.3 PSNR

The last metric used in the experiments is *PSNR*. It is usually measured in decibels (dB): $PSNR(\mathrm{dB}) = 10\log_{10} PSNR$ and its definition for the error between two images f and g (of dimensions $X \times Y$ pixels) is:

$$PSNR(f,g) = \frac{XY \max_{x,y}\left(f_{x,y}^2\right)}{\sum_{x,y}\left(f_{x,y} - g_{x,y}\right)^2}. \tag{6}$$

In order to compare *PSNR* with the results obtained by the other metrics, it is worth mentioning that *PSNR* works exactly like a metric in which the neighborhood assigned to the matching is zero. For that reason, it is valid to say that the three metrics follow the idea of searching for a matching between the images (with different methods and neighborhoods) and then estimate the similarity between the images from the addition of the errors caused by the matched and the unmatched pixels. Furthermore, since *PSNR* is equivalent to a zero-neighborhood matching, it is therefore useful as a reference value when comparing the results obtained by the other metrics.

4 Description of the Experiments

This section includes a description of the two experiments designed to evaluate the performance of PCM. It is not the purpose of this paper to evaluate the performance of different edge detectors, but rather the viability of PCM as a tool for comparing edge images. All the experiments are therefore based on a single algorithm for edge detection in which the results obtained from different implementations of this algorithm are compared using PCM and the discrepancy measures defined in the previous section. All the images used in these experiments had 256 levels of grayscale.

4.1 Experiment 1

The first experiment included in this paper attempts to measure the robustness of various discrepancy measures. In this experiment, three classical test images (Lena, Peppers, and Goldhill [13]) of size 256×256 and a synthetic test image of size 64×64 were used. The synthetic image consists of a bright disk of a constant value against a darker background (a more detailed description of how to create such an image can be found in [5]). The four images were exposed to three different kinds of noise of varying intensity:

- *Gaussian*: with zero mean and variances 0.005, 0.01, and 0.015.
- *Salt & Pepper*: with densities 0.025, 0.05, and 0.075.
- *Speckle*: with variances 0.02, 0.04, and 0.06.

For each case, the *Canny edge detector* [14] with a Gaussian mask of standard deviation 1.25 (see Section 4.3 for more details) was applied. Then, the edge maps obtained from the noisy images were compared with the edge map corresponding to the original images (without noise) and the similarity values obtained by four discrepancy measures were recorded. The four measures were: PCM_2, CDM_2, FOM (with $a = 1/9$), and *PSNR*.

The idea behind this experiment is that a certain increment of noise in the original image produces a certain amount of error in the detection of the edges. This error is directly related to the degree of noise added to the image and the texture of the image. A metric can be considered robust if it reflects this relationship by giving similarity values which decrease according to the quantity of noise added to the original image. At the same time, a robust metric should obtain similar measurements for images of similar texture (Lena and Peppers), whereas highly textured images (Goldhill) should produce a lower value, when the same degree of noise is added.

4.2 Experiment 2

The second experiment is focused on how the different matching algorithms and the neighborhood (η) used in the matchings, affect the measurements obtained from the metrics. The images used in this experiment were the classical test image Lena and two additional images obtained from a real application of microbiological image analysis. The first image (Cells) is a 489×187 image of a cluster of cells greatly affected by noise. The second (Cells2) is a 768×576 image of another cluster of cells with very poor illumination.

The three images were processed by different implementations of the Canny edge detector (see Section 4.3 for a detailed description of the implementations) and then a comparison between the output of each implementation and the standard (unmodified) Canny algorithm was measured with the same four metrics of Experiment 1. Since the purpose of this experiment is to study the performance of the metrics according to the error in the location of the edges, the metrics PCM and CDM have both been used with neighborhoods 1 and 2, in order to study the quality of the matching process. The different implementations of the Canny algorithm used in this experiment produce five sets of images:

- IMPL 1 GL: Gray level (GL) images obtained by the application of the first implementation of the Canny algorithm (IMPL 1).
- IMPL 1 BIN: The IMPL 1 GL images after binarization.
- IMPL 2 GL: Gray level (GL) images obtained by the application of the second implementation of the Canny algorithm (IMPL 2).
- IMPL 2 BIN: The IMPL 2 GL images after binarization.
- MATLAB BIN: Binary images obtained by the version of the Canny algorithm included in Matlab.

4.3 Canny Algorithm Implementations

The Canny operator can be divided into three stages [14]. The first stage smooths the original image and enhances edges by computing its convolution with the derivative of a Gaussian mask. Then, the nonmaximal suppression stage finds local gradient maxima to extract potential edges and, finally, the last stage of the algorithm performs a hysteresis thresholding over the image to eliminate nonrelevant edges.

The two implementations [15] considered in this report modify the Canny algorithm for a more suitable hardware implementation. An efficient implementation of this algorithm may separate the process of the convolution with a two-dimensional mask into two convolutions (horizontally and vertically) with one-dimensional masks and then combine the two gradients (G_h, G_v) to give a resultant gradient:

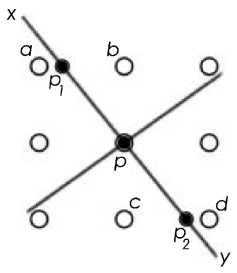

Fig. 1. Gradient interpolation.

$$G(x,y) = \sqrt{G_h(x,y)^2 + G_v(x,y)^2}. \tag{7}$$

In hardware implementations, the use of floating point values is very expensive in terms of resources. For this reason, the first implementation (IMPL 1) approximates a Gaussian derivative mask (of 11 elements) with integer elements instead of floating-point:

$$\frac{1}{16}(2 \quad 4 \quad 7 \quad 8 \quad 6 \quad 0 \quad -6 \quad -8 \quad -7 \quad -4 \quad -2).$$

The second implementation (IMPL 2) used in the experiments also starts by applying an integer derivative mask in the horizontal and vertical convolution of the image, similarly to IMPL 1. Then, the combination of the two components is performed using

$$G(x,y) = \max\Big(|G_h(x,y)|, |G_v(x,y)|\Big) + \frac{\min\Big(|G_h(x,y)|, |G_v(x,y)|\Big)}{4}, \tag{8}$$

instead of (7).

Another change introduced by IMPL 2 refers to the nonmaximal suppression stage of the algorithm. The original algorithm requires the calculation of the magnitude along a line perpendicular to the direction of the gradient. Fig. 1 illustrates the two values (p_1 and p_2) that have to be interpolated in order to know if the magnitude of the current point (p) is a local maximal along the x-y line. The algorithm states that the magnitude in the center (p) ought to be compared with the interpolation of the magnitude between the two pairs of points (a, b and c, d). This calculation, however, is very expensive in hardware and, so, IMPL 2 substitutes the average of each pair of points for the interpolations.

In terms of the position of the edges, IMPL 2 introduces a much greater error in the localization of potential edges than IMPL 1. Furthermore, the more textured an image is, the greater will be the error of IMPL 2 in the correct identification of real edges and their location.

5 Results

This section includes the results obtained in the experiments presented in Section 4 and analyzes in detail the robustness of the discrepancy measures under study. Note that PCM, CDM, and FOM are measured as a percentage and $PSNR$ is expressed in dB.

TABLE 1
PCM, CDM, FOM, and $PSNR$ Evaluation of the Edge Maps of Four Test Images under Different Kinds of Noise

		Gaussian			*Salt & Pepper*			*Speckle*		
IMAGES		0.005	0.01	0.015	0.025	0.05	0.075	0.02	0.04	0.06
Lena	PCM_2	89.9	86.8	84.7	86.8	83.1	79.0	90.5	86.8	83.5
	CDM_2	78.8	76.8	74.5	75.1	72.7	68.8	79.7	76.0	73.2
	FOM	84.2	83.1	82.8	81.3	79.3	75.5	85.3	82.9	79.0
	PSNR	18.6	17.3	16.2	18.0	16.1	15.1	19.5	17.6	16.7
Peppers	PCM_2	89.5	86.1	83.6	87.8	81.8	79.5	89.2	85.2	82.2
	CDM_2	78.7	75.9	74.2	76.0	71.3	69.0	79.4	75.7	73.2
	FOM	87.7	85.8	84.3	85.1	81.3	77.7	88.5	84.8	78.6
	PSNR	18.8	17.3	16.3	18.4	16.2	15.3	19.0	17.3	16.2
Goldhill	PCM_2	88.2	84.8	81.9	83.3	79.7	75.2	88.3	84.6	81.6
	CDM_2	79.3	76.9	74.2	74.3	70.5	68.3	79.2	75.7	73.9
	FOM	82.3	80.9	77.4	64.7	64.5	55.1	75.6	71.8	68.3
	PSNR	19.5	17.8	16.8	18.8	16.7	16.1	20.2	18.5	17.4
Disc	PCM_2	92.8	88.0	84.3	85.5	82.6	69.3	92.7	89.5	87.2
	CDM_2	78.2	73.0	71.7	71.8	67.3	59.3	78.2	76.0	73.9
	FOM	94.6	86.4	81.3	77.8	81.1	57.9	91.1	90.6	88.4
	PSNR	25.1	22.7	21.5	23.3	21.6	18.8	25.9	23.8	22.3

5.1 Results from Experiment 1

The results presented in Table 1 correspond to the measurements of the errors between the edge images obtained after applying the Canny operator to the images with the different kinds of noise and the original images (before the noise was introduced), as described in Section 4.1.

The first clear observation that can be made from Table 1 is that the results obtained by PCM, CDM, and $PSNR$ decreased in all the cases in which the added noise had been increased. This consistency, however, was not shared by FOM as it failed to reflect this fact in some cases. For example, the measured values of the Goldhill image when *Salt & Pepper* noise was added at densities 0.025 and 0.05 were almost identical and, when this same noise was added to the Disc image, the measurements stated that the edge detection algorithm performed better on the image with a noise of density 0.05, which is evidently incorrect.

The second observation is with regard to the relationship between the texture of the images and the error of the edges. CDM and $PSNR$ gave very similar results for any of the three classical test images (Lena, Peppers, and Goldhill) when the same type of noise was added to the images. For example, when *Speckle* noise with variance 0.04 was added to the three images, CDM gave a result of 75.8 ± 0.17 (mean = 75.8, standard deviation = 0.17) and $PSNR$ 17.8 ± 0.62. In most of the cases, the maximum difference between the values registered by CDM was less than 2 percent and by $PSNR$ was less than 1 dB. On the other hand, PCM registered a difference of up to 4.5 percent, a difference which, although not very large, shows that PCM has the characteristics discussed in Section 4.1 for it to be considered a robust metric.

One last point to consider in this table is the values obtained for the Disc image. Since the original image has no texture, the edges are well-identified in most of the cases, giving results higher than the other three images. Nevertheless, when *Salt & Pepper* noise was added at the highest density, because of the nature of the noise, the Canny edge detector was unable to produce a good identification of the correct edges and this was reflected in the measurements obtained by the four discrepancy measures.

TABLE 2
PCM, CDM, FOM, and $PSNR$ Evaluation of the Images Produced by Three Different Implementations of the Canny Edge Detector

Images		Impl 1 GL	Impl 1 BIN	Impl 2 GL	Impl 2 BIN	Matlab BIN
Lena	PCM_1	99.4	98.8	94.7	89.1	87.6
	PCM_2	99.6	99.4	95.8	92.9	91.4
	CDM_1	98.5	97.2	88.1	78.4	69.8
	CDM_2	96.2	95.5	83.0	76.9	64.2
	FOM	-	99.4	-	96.9	81.2
	$PSNR$	35.0	26.2	22.4	15.4	14.1
Cells	PCM_1	99.8	99.4	95.8	86.9	84.7
	PCM_2	99.8	99.7	96.4	90.1	87.6
	CDM_1	98.7	97.5	91.1	75.7	63.4
	CDM_2	97.7	96.5	86.9	74.0	56.7
	FOM	-	99.5	-	93.3	77.4
	$PSNR$	39.9	30.9	27.5	17.9	16.7
Cells2	PCM_1	99.7	98.7	96.5	83.4	87.6
	PCM_2	99.8	99.1	97.2	88.0	90.9
	CDM_1	99.1	96.9	93.4	73.3	67.4
	CDM_2	97.8	95.7	89.5	73.1	62.2
	FOM	-	99.7	-	93.0	78.7
	$PSNR$	43.0	33.5	31.3	21.9	21.7

5.2 Results from Experiment 2

As explained in Section 4.2, different implementations of the Canny algorithm were applied to three images and the resulting edge images were compared with the edge map obtained by the original (unmodified) Canny algorithm. The comparisons performed with the four discrepancy measures can be found in Table 2.

As mentioned in Section 4.2, the edge images obtained from the different implementations used in this experiment differ mainly in the location of the edges. This circumstance affects the measurements taken by the metrics in several respects.

1. *Gray-Level and Binary Images*. First of all, there are two points in which the four metrics agreed: that IMPL 2 gives worse results than IMPL 1 and that the error increases when the images are binarized (GL $\rightarrow$ BIN). The first observation confirms that, as expected, the second implementation introduces more error in the identification and location of the edges. The second observation, however, requires a more detailed analysis.

   In the case when the images are binarized, all the edges receive a value of 255. Then, the error of a match between two pixels (e.g., $140 \leftrightarrow 150 : error = 10$) becomes zero when both pixels adopt the new strength ($255 \leftrightarrow 255 : error = 0$), thus reducing the total error between the images. On the other hand, since the value of the unmatched pixels is added to the total error, when these pixels become 255, then the error to be added is also increased. In this way, the more false

pixels the image contains, as happens in IMPL 2 compared to IMPL 1, the greater difference will there be between the measurements obtained using gray-level and binary images. *PCM* and *CDM* reflected this fact by showing a disparity of only ~ 1 percent between the gray-level and binary results in IMPL 1, but up to 13 percent (with *PCM*) and 20 percent (with *CDM*) in IMPL 2.

2. *Using Different Images*. In relation to the size of the images, the fact that Cells2 is more than twice the size of the test images does not seem to affect the measurements taken by the *PCM*, *CDM*, or *FOM*. On the other hand, the measurements obtained by *PSNR* show a clear relation between the size of the image and the value of these measurements. The reason is that, in order to estimate the similarity between two images, *PSNR* takes into account all the pixels of the images instead of considering only the pixels which are not zero. Since the images Cells and Cells2 have approximately the same number of edges as the test images but distributed over a bigger area, the amount of black pixels in the images is very high. In the test images, where the edges represent approximately 10 percent of the total, *PSNR* detects that at least 90 percent of the pixels are identical. In the Cells and Cells2 images, where the edges represent less than 5 percent of the total, the *PSNR* detects a minimum of 95 percent of identical pixels between the images, thus showing an increase of the measurements by up to 8 dB.
3. *Using Different Neighborhoods*. Intuitively, when the neighborhood of the matching is increased, the metrics should return higher similarity values. The reason for this is that, with a bigger neighborhood, a matching with lower cost may exist and, therefore, the estimate of the error between the images would decrease. If a matching with a lower cost did not exist, the method should at least be able to find the matching found with the previous neighborhood and return the same similarity value.

In Table 2, when the neighborhood was increased, the similarity values obtained by *PCM* increased in most cases (except in one case in which it remained the same). In the first implementation, the pixels have not moved more than one pixel of distance from the original position and, therefore, the optimal matchings do not significantly differ from those already found with neighborhood one. In this case, the similarity values differ in just the decimals or do not differ at all. On the other hand, the second implementation does introduce movement between the edges of more than one pixel of distance and, therefore, there exist better matchings which decrease the error between the images by up to 4 percent. The reason why this effect is greater with binary images is that an optimal match in a binary image is not necessarily optimal in the gray-level one as it is easier to find better matchings when all the edges have the same value.

By contrast, when the neighborhood was enlarged in CDM, there was an increase of the error in all the results, instead of the decrease expected. This loss of precision was more accentuated in the cases where the pixels were more displaced from their original position and cases in which the ratio of false edges was much higher. In this way, the loss of precision of the method was only about 2 percent for the first implementation, but it went up to 5 percent for the second implementation and up to 7 percent for the Matlab implementation of Canny. In this last case, the result obtained by *CDM* might lead one to think that the Matlab implementation of Canny is extremely different from the implementation used as a reference, when this is not the case. We have performed other experiments not included in this report which indicate that the loss of precision of this method is even more accentuated when the neighborhood is further increased.

It is also interesting to note that most of the results from *PCM* gave higher values than *CDM*, as was expected due to the more accurate search for an optimal matching in *PCM*.

6 Conclusions

This paper has analyzed the difficulties in the use of traditional metrics, such as *PSNRI*, for comparing edge images. The main problem is that these methods are not capable of accounting for small movement between the edges produced by different algorithms and, therefore, they are not able to correctly assess the similarity between the images. This is a very common problem in the performance evaluation of edge detection algorithms and several methods aimed at overcoming this problem have been proposed in the past. In this paper, we have included a description of the operation of some of these methods and proposed a new theoretical method, named *PCM*, which takes into account not only displacement of edge positions, but also edge strength in the estimation of similarity between edge images.

In the first experiment included in this paper, the *PCM* has been tested on the edge maps of images under different kinds of noise. The results have proven the new metric to be sensitive and consistent. This experiment has also illustrated cases in which *FOM* has performed inconsistently.

The second experiment included in this paper was designed to study the performance of the different discrepancy measures when the edge images present a certain degree of movement between their edges. The edge images were used in both gray-level and binary format.

The ratio by which the overall error increases with the binarization of the images depends on how many pixels have been displaced by a small distance and how many false edge pixels have appeared. Ideally, by increasing the neighborhood of the matching, it should be possible to find matchings with less error (depending on how significant is the movement between edge positions), therefore increasing the similarity estimation value between the images. The results obtained by the *PCM* show an improvement in the accuracy of the approximation to the optimal matching when the neighborhood is increased. By contrast, *CDM* demonstrated an inconsistent dependency on neighborhood size.

Summarizing, the *PSNR* or any metric unable to detect movement between pixels should not be used for the comparison of edge images, as this might lead to misleading measurements. Results obtained by metrics using closest-distance matching or similar methods should be carefully studied and more accurate methods to find the matching, such as the *PCM*, should be used when more precision in the

measurements is required. We also remark that the method used by the *PCM* for the matching of pixels can be used in other applications in which the spatial relation between edges is important, such as object segmentation in stereo images or detection of moving objects in video analysis.

ACKNOWLEDGMENTS

The authors gratefully acknowledge the support of AXEON Limited for this work and the anonymous reviewers for helpful comments.

REFERENCES

[1] Y.J. Zhang, "A Survey on Evaluation Methods for Image Segmentation," *Pattern Recognition,* vol. 29, no. 8, pp. 1335-1346, 1996.
[2] L. Kitchen and A. Rosenfeld, "Edge Evaluation Using Local Edge Coherence," *IEEE Trans. Systems, Man, and Cybernetics,* vol. 11, no. 9, pp. 597-605, Sept. 1981.
[3] A.J. Baddeley, "Errors in Binary Images and an L^p Version of the Hausdorff Metric," *Nieuw Archief voor Wiskunde,* vol. 10, pp. 157-183, 1992.
[4] Q. Zhu, "Improving Edge Detection by an Objective Edge Evaluation," *Proc. 1992 ACM/SIGAPP Symp. Applied Computing,* pp. 459-468, 1992.
[5] M.D. Heath, S. Sarkar, T. Sanocki, and K.W. Bowyer, "A Robust Visual Method for Assessing the Relative Performance of Edge-Detection Algorithms," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 19, no. 12, pp. 1338-1359, Dec. 1997.
[6] T.B. Nguyen and D. Ziou, "Contextual and Non-Contextual Performance Evaluation of Edge Detectors," *Pattern Recognition Letters,* vol. 21, no. 8, pp. 805-816, 2000.
[7] G. Liu and R.M. Haralick, "Optimal Matching Problem in Detection and Recognition Performance Evaluation," *Pattern Recognition,* vol. 35, pp. 2125-2139, 2002.
[8] W. K. Pratt, *Digital Image Processing.* New York: Wiley-Interscience 1978
[9] K.W. Bowyer, C. Kranenburg, and S. Dougherty, "Edge Detector Evaluation Using Empirical ROC Curves," *Computer Vision and Image Understanding,* vol. 84, pp. 77-103, 2001.
[10] H.W. Kuhn, "The Hungarian Method for the Assignment Problem," *Naval Research Logistics Quarterly,* vol. 3, pp. 253-258, Oct. 1956.
[11] H.A.B. Saip and C.L. Lucchesi, "Matching Algorithms for Bipartite Graphs," Technical Report, IMECC, Universidade Estadual de Campinas, Brazil, 1993.
[12] H.N. Gabow and R.E. Tarjan, "Faster Scaling Algorithms for Network Problems," *SIAM J. Computing,* vol. 18, no. 5, pp. 1013-1036, Oct. 1989.
[13] "Test Images for Image Processing Purposes,"Hungarian Association for Image Analysis and Pattern Recognition (KEPAF), Available at: http://www.inf.u-szeged.hu/~kepaf/TestImages/. 2003.
[14] J. Canny, "A Computational Approach to Edge Detection," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 8, no. 6, pp. 679-698, 1986.
[15] T.M. Sheen, "Tools for Portable Parallel Image Processing," PhD dissertation, Univ. of Aberdeen, 1999.

Miguel Segui Prieto received the degree in computer science from the Universitat Autonoma de Barcelona in 1999. The final year project was completed at the University of Ulster, under the Erasmus European exchange program. He then collaborated on a project for the study of the Prisoner's Dilemma on Dynamic Small-World Networks at the Universitat Politecnica de Catalunya in 2000. His main interest now is to research techniques for the efficient implementation of embedded machine vision applications.

Alastair R. Allen received the BSc and DPhil degrees in physics from the Universities of Birmingham and Sussex, respectively. He lectured in physics from 1978 to 1984. Since then, he has lectured in information technology at the University of Aberdeen, where he is currently a senior lecturer in the School of Engineering & Physical Sciences jointly with the Department of Bio-Medical Physics & Bio-Engineering. His research interests in the area of computational imaging include multiscale techniques, the optimization of imaging algorithms on programmable hardware, and watermarking. His other research includes the development of concurrent computer architectures, especially for embedded imaging and instrumentation. He is a member of the IEEE.