Edge Detection Techniques of Thermal Images

Szabolcs Serény
Ábuda University/Institute of Software Engineering, Budapest, Hungary
sergyan.szabolcs@nik.uni-obuda.hu

Abstract—Selecting the right parameter is a vital issue in many image processing algorithms. Normally this is done after experimenting with a limited number of pictures, and then one just hopes that the parameters will work adequately with all the other images. Experimenting with the parameters, however, consumes precious human time.

Edge detection of thermal images is a special issue of edge finding. The thermal images have some special properties, and their edge detection methods these may be taken into account.

In this paper a technique is presented that aims at finding the right parametrization of frequently used edge detection algorithms by comparing several approaches and deciding on the optimal one automatically.

I. INTRODUCTION

Most image processing algorithms use a handful of parameters [1]. For example, edge detection algorithms are based on determining a suitable threshold of the image gradient matrix [2]. The threshold, however, must be a parameter and the resulting edge matrix significantly depends on the selection of the aforesaid parameter [3].

As an example, consider the edges of a thermal image (Fig. 1) in Fig. 2 - 8 which were obtained using different parameters of the same edge detection algorithms. The use approach was the Sobel filtering using threshold values as 150, 100, 80, 60, 50, 40 and 20.

Parameters are typically tuned, based on a good deal of experiments, to the given image and the task to be performed on it in order to get an optimal result. However, if the task is image retrieval in a large image database, then it is impossible to tune the parameters to each database image individually. In such a case the typical solution is to use a parameter that delivers a relatively good result to most images [4]. This can be enhanced, though, by trying to automatically find a more suitable parameter to each image. The first experimental results of this kind were published in [5], [6], [7]. With one of my colleague we published an algorithm for finding the optimal parameters of homogeneous region detection in gray-scale images. The crucial idea was to create the region decompositions of the image using several parametrizations and then determine which decomposition is closest to most other decompositions. This approach made it necessary to define decomposition distance. In this paper we performed experiments to prove that the developed method is powerful to find the parametrization most fit for the given image if we use thermal images.

In the current paper we automatically detect the threshold used with edge detection in greyscale images. The crucial idea is similar to that of used with region detection, namely the
edge matrices found with different thresholds are compared to find the edge matrix that is similar to most other matrices. What is new here is a modification of earlier use automatic parametrization algorithm that is presented in Section 2. The algorithm has been tested with thermal images using two edge detection methods. See Section 3, for a comparison of our approach to another automatic threshold optimization method. The test results are presented in Section 4. A conclusion of results, as well as further development plans are summarised in Section 5.

II. DESCRIPTION OF THE ALGORITHM

The goal is to find an edge matrix of a gray-scale image that fits best to the given image. Best fitting is, unfortunately, somewhat subjective (see also Section 3.).

An often used method to find an edge matrix of a gray-scale image is to compute the gradient vector to each pixel. Then we say that a pixel is an edge pixel if the norm of its gradient exceeds a given threshold $\tau$. The difference in edge detection algorithms lies in the interpretation of the gradient and in the selection of the threshold [8], [9].

In our algorithm threshold selection has been made automatically according to the following:

1) In the first step the noise reduction is our main goal [10]. We have to select a noise reduction algorithm which does not significantly blur the edges (significant changes) from the image. As pre-processing the Symmetric Nearest Neighbour filter [11] or the Kuwahara filter [12] were applied.

At Symmetric Nearest Neighbour filter four sets are generated from the neighbours of the examined pixel $(P(x,y))$

\[ S_1 = \{P(x-1,y-1), P(x+1,y+1)\} \]
\[ S_2 = \{P(x-1,y), P(x+1,y)\} \]
\[ S_3 = \{P(x-1,y+1), P(x+1,y-1)\} \]
\[ S_4 = \{P(x,y-1), P(x,y+1)\} \]

See the set of neighbours in Fig. 9.

Let be $cl(S_i)$ that pixel in $S_i$ whose gray-scale value is the closest to the gray-scale value of $P(x,y)$. In the result image replace $P(x,y)$ by the following mean value:

\[ \frac{1}{4} \sum_{i=1}^{4} cl(S_i) \] (1)

Using the Kuwahara filter four overlapped rectangle regions ($R_1, R_2, R_3, R_4$) of the neighbours are applied, whose intersection is the examined pixel $(P(x,y))$. See the used set of neighbours in Fig. 10. Then the variance of the gray-scale values are determined for each region, and that region is selected whose variance is the minimal. Then in the result image $P(x,y)$ is replaced by the average of the gray-scale values from that region.
2) Using \( N \) different thresholds we have created the edge matrices of the image which are denoted by \( E^i \), where \( i \in \{1, 2, \ldots, N\} \).

3) We determine the number of edge pixels of each edge matrix \((|E^i|)\), and compute their mean:

\[
\text{average}(|E^i|) = \frac{\sum_{i=1}^{N} |E^i|}{N}.
\]

4) In the following we only consider the edge matrices for which

\[
|E^i| \geq \text{average}(|E^i|).
\]

In the sequel these will denoted by \( E^j \), where \( j \in \{1, 2, \ldots, M\} \) and \( M \leq N \) is the number of such edge matrices.

5) We perform a \( 3 \times 3 \) dilatation on each of the remaining edge matrices. In this step we can enhance edges and increase the width of ideal edge lines [13]. The results will be denoted by \( DE^j \).

6) At this point the main question is how can we compare two edge matrices, so two binary sets [14].

Before we define the distance of two edge matrices we need to clarify what we mean by the intersection and union of two dilated edge matrices. The intersection of the dilated edge matrices \( DE^{ik} \) and \( DE^a \) is a number, namely the number of those pixels that are edges in both dilated edge matrices. This number will be denoted by \( DE^{ik} \cap DE^a \). Similarly, the union of the dilated edge matrices \( DE^{ik} \) and \( DE^a \) is the number of those pixels that are edges in at least one of the dilated edge matrices. This number will be denoted by \( DE^{ik} \cup DE^a \).

Now we are ready to define the distance of two dilated edge matrices as follows:

\[
d(DE^{ik}, DE^a) = 1 - \frac{DE^{ik} \cap DE^a}{DE^{ik} \cup DE^a}.
\]

It is clear that the above distance will be zero if the two dilated edge matrices coincide, and the distance will be 1 if the two dilated edge matrices have no common pixel. It is an open question whether this "distance" is really a distance in mathematically point of view. Clearly the triangle inequality needs to be checked. However, this is of no concern to us as the "distance" fulfils its task of measuring similarity well enough.

7) Not let us compute the matrix \( D \) of size \( M \times M \) as follows:

\[
D_{ij} = d(DE^i, DE^j).
\]

Clearly matrix \( D \) is symmetric.

8) Let \( k \) denote the index such that the column sum of the \( k \)th column of \( D \) is minimal. Then the parametrization used for computing the dilated edge matrix \( DE^{ik} \) will be considered optimal for the original image.

The algorithm first selects the parametrizations (thresholds) that result in edge matrices containing not too few pixels and then that one is selected that is closest to most others.

III. EXPERIMENTS

A. Applied edge detection algorithms

For testing the Sobel [15] and Prewitt [16] edge detection algorithms were used.

The Sobel algorithm uses the mask

\[
M_x = \begin{bmatrix}
-1 & -2 & -1 \\
0 & 0 & 0 \\
1 & 2 & 1
\end{bmatrix}
\]

for determining the \( x \)-direction gradient \( G_x(f) \) of the image \( f \), and the mask

\[
M_y = \begin{bmatrix}
-1 & 0 & 1 \\
-2 & 0 & 2 \\
-1 & 0 & 1
\end{bmatrix}
\]

for determining the gradient \( G_y(f) \) along the horizontal direction. Then the edge matrix of \( f \) contains those pixels for
where $\tau$ is a preselected threshold. The principal goal in the automatic parametrization algorithm is to determine the most suitable value of this threshold.

The Prewitt algorithm differs from Sobel only in the selection of the masks:

$$M_x = \begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix}$$  \hspace{1cm} (9)

and

$$M_y = \begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix}$$  \hspace{1cm} (10)

B. The test database

For testing we have used thermal images which were taken indoor and outdoor as well. Some of these images you can see in Fig. 11 - 13.

IV. RESULTS

A crucial issue regarding testing is to decide whether a computed edge matrix is good enough. To this aim the generated edge matrices are compared to one another.

The edge function in the MATLAB Image Processing Toolbox can automatically set a threshold by taking the mean of the norms of the gradients.

We have found that in 31% and 34.2% of all cases of the Sobel and Prewitt algorithms, respectively, the edge function and our method differs in the optimal threshold with a value of less than 10%, so we can say that they have found roughly the same optimal threshold.

To decide which way is better for optimal threshold selection we have compared the edge matrices produced by the two methods (MATLAB and ours) by declaring those matrices better in which the edges were more "continuous" and in which the important edges of the original image really appeared. Using this criterion we have found in the Sobel case that our method is better in 47.6%, MATLAB is better in 12.8%, and the methods produce roughly similar results in 14.6% of all cases. In the Prewitt case the relevant percentages are 42.4%, 12.6% and 17.0%, respectively.

V. CONCLUSIONS AND FURTHER WORKS

We have found, based on the experiments, that our method seems to better than some other widespread solutions. However, we also have to point out that the running time of our algorithm is considerably higher as we compute image matrices by several parametrizations which have to be compared as well, while other methods, such as the above, derives the threshold directly from the image. On the other hand, in image database applications where the database image parameters must be determined only once and it is the search that happens often this delay is not a great disadvantage.

We are planning to decrease subjectivity in deciding which image is better by involving other people to make the decisions and perform a statistical analysis thereof. Other plan is to test the algorithm in a special type of database[17].

ACKNOWLEDGEMENT

The project was realized through the support of the European Union, with co-financing of the European Social Fund TÁMOP-4.2.1.B-11/2/KMR-2011-0001.

REFERENCES


