An Efficient Algorithm for Optimal Multilevel Thresholding of Irregularly Sampled Histograms

Luis Rueda

Department of Computer Science, University of Windsor, 401 Sunset Ave.,
Windsor, ON, N9B 3P4, Canada
Phone: +1 519 253-3000 x 3002; Fax: +1 519 973-7093
lrueda@uwindsor.ca

Abstract. *Optimal* multilevel thresholding is a quite important problem in image segmentation and pattern recognition. Although efficient algorithms have been proposed recently, they do not address the issue of irregularly sampled histograms. A *polynomial-time* algorithm for multilevel thresholding of irregularly sampled histograms is proposed. The algorithm is polynomial not just on the number of bins of the histogram, n, but also on the number of thresholds, k, i.e. it runs in $\Theta(kn^2)$. The proposed algorithm is general enough for a wide range of thresholding and clustering criteria, and has the capability of dealing with irregularly sampled histograms. This implies important consequences on pattern recognition, since optimal clustering in the one-dimensional space can be obtained in polynomial time. Experiments on synthetic and real-life histograms show that for typical cases, the proposed algorithm can find the optimal thresholds in a fraction of a second.

1 Introduction

Multilevel thresholding is an important technique that has many applications in image processing, including segmentation, classification, clustering and object discrimination. Various parametric and non-parametric thresholding methods and criteria have been proposed, being the three most important streams the following: Otsu's method [8], which aims to maximize the separability of the classes measured by means of the sum of between-class variances, Kittler and Illingworth's criterion [4], which minimizes the error between a mixture of Gaussians and the actual histogram, and the criterion that uses information-theoretic measures to maximize the separability of the classes [3]. Variations of these criteria include a bi-level method that correlates the discrepancy between the original and thresholded image and the number of bits used to represent the segmented image [14], the use of an information-theoretic criterion based on Renyi's entropy [10], and a unified method introduced in [13]. Other criteria that have been proposed use spatial information about the pixels, also known as 2D thresholding [11,17]. All these criteria proposed efficient algorithms for bi-level thresholding.

For *optimal* multilevel thresholding, various algorithms have been proposed. A fast multilevel thresholding algorithm was proposed in [5], and efficient implementations of the three criteria were proposed in [12], in which a reduced number

N. da Vitora Lobo et al. (Eds.): SSPR&SPR 2008, LNCS 5342, pp. 602–611, 2008.

of subsets are evaluated. Other efficient, but yet *sub-optimal*, thresholding algorithms have been proposed and which use different kinds of heuristics, including a method based on genetic algorithms [15], a hybrid method that combines Gaussian curve fitting and particle swarm optimization [16], and a hybrid approach that combines particle swarm optimization and expectation maximization [2]. All these approaches suffer from the lack of either optimality or running time efficiency. Polynomial-time algorithms have been recently proposed , which solve the problem efficiently [6]. However, these algorithms are for specific thresholding methods and for regularly sampled histograms only (i.e., the bins are consecutive over a certain range). Some image segmentation problems contain sparse histograms, or in some cases only a few gray levels are available. These can be represented by means of irregularly sampled histograms, which do not necessarily have consecutive bins, and which can be sampled in other domains (e.g. real numbers). In this paper, a polynomial-time algorithm for optimal multilevel thresholding of irregularly sampled histograms is proposed. This algorithm has important applications and consequences in pattern recognition, since optimal clustering in the one-dimensional space can be efficiently obtained.

2 Exhaustive Search Multilevel Thresholding

Although various thresholding criteria and methods have been proposed, three main streams of thresholding criteria are the most-widely used: maximizing the sum of between-class variances [8], maximizing the sum of entropies of each individual class [3], and minimizing the error between the actual histogram and a mixture of Gaussians [4]. A general framework for the thresholding problem on irregularly sampled histograms is discussed here.

Definition 1 (Histogram). *An irregularly sampled histogram H is an ordered set*[1] $\{h_1, h_2, \ldots, h_{n-1}, h_n\}$, $h_1 < h_2 < \ldots < h_{n-1} < h_n$, *where h_i has a probability, p_i, associated with it.* □

If the probabilities are unknown, and given in terms of frequencies, they can be computed as follows. If $F = \{f_1, f_2, \ldots, f_n\}$ are the frequencies of a histogram H, $P = \{p_1, p_2, \ldots, p_n\}$ can be computed as $p_i = f_i/N$, where $N = \sum_{i=1}^{n} f_i$.

Definition 2 (Threshold set). *A threshold set T is defined as an ordered set $T = \{t_0, t_1, \ldots, t_k, t_{k+1}\}$, where $0 = t_0 < t_1 < \ldots < t_k < t_{k+1} = n$ and $t_i \in \{0, 1, \ldots, n\}$.* □

For an ordered subset of T, the notation $T_{i,j} = \{t_i, t_{i+1}, \ldots, t_j\}$ is used, where $i, j = 0, 1, \ldots, k, k+1$, $i < j$, and $T = T_{0,k+1}$. The problem of multilevel thresholding consists of finding a threshold set T^*, in such a way that a function $\Psi : \{1, 2, \ldots, n\}^k \times [0, 1]^n \to \mathbb{R}^+$ is maximized/minimized. Using this threshold set, H is divided into $k+1$ classes: $\zeta_1 = \{h_1, h_2, \ldots, h_{t_1}\}$, $\zeta_2 = \{h_{t_1+1}, h_{t_1+2}, \ldots, h_{t_2}\}$, $\ldots$, $\zeta_k = \{h_{t_{k-1}+1}, h_{t_{k-1}+2}, \ldots, h_{t_k}\}$, $\zeta_{k+1} = \{h_{t_k+1}, h_{t_k+2}, \ldots, h_n\}$.

[1] In a b-bit grayscale image, the corresponding histogram is $H = \{0, 1, \ldots, 2^b - 1\}$.

In the original works of Otsu [8], Kapur *et al.* [3], and Kittler-Illingworth [4], efficient algorithms for finding bi-level thresholding were proposed. For optimal multilevel thresholding, different ways of implementing exhaustive-search algorithms were proposed in [5,12], which search for combinations of $t_1 < t_2 < \ldots < t_k$. This implies a number of subsets reduced to $\binom{n}{k}$ different subsets, or equivalently $\Omega\left(\left(\frac{n}{k}\right)^k\right)$ subsets, and subsequently an *exponential* complexity on the number of thresholds.

3 A Dynamic Programming Scheme

Dynamic programming algorithms for *optimal* multilevel thresholding have been designed specifically for the between-class, minimum error and entropy-based criteria, for regularly sampled histograms [6]. A generalized algorithm and framework for a wide range of criteria and for irregularly sampled histograms are given here.

Definition 3 (function ψ). *Let $P = \{p_1, p_2, \ldots, p_n\}$ be the probabilities of a histogram $H = \{h_1, h_2, \ldots, h_n\}$. The function $\psi_{l,r}$, where $l \leq r$, is a real, positive function of $p_l, p_{l+1}, \ldots, p_r$, $\psi_{l,r} : \{1, 2, \ldots, n\}^2 \times [0,1]^{l-r+1} \to \mathbb{R}^+ \cup \{0\}$.* □

Either $\psi(l, l+1, \ldots, r, p_l, p_{l+1}, \ldots, p_r)$, $\psi(l, r, p_l, p_{l+1}, \ldots, p_r)$ or for short, $\psi_{l,r}$, will be used to denote the function ψ for $l, r, \{p_l, p_{l+1}, \ldots, p_r\}$. Also, if $r < l$, by definition $\psi_{l,r} \triangleq 0$, and if $r = l = 0$, $\psi_{0,0} \triangleq 0$.

Definition 4 (function Ψ). *Let $T = \{t_0, t_1, \ldots, t_k, t_{k+1}\}$ be a threshold set, where $k \geq 1$. The function Ψ is defined for all m, where $1 \leq m \leq k+1$, as a function $\Psi : \{1, 2, \ldots, n\}^m \times [0,1]^n \to \mathbb{R}^+$ as follows:*

$$\Psi(T_{0,m}) = \Psi(\{t_0, t_1, \ldots, t_m\}) \triangleq \sum_{j=1}^{m} \psi_{t_{j-1}+1, t_j} \,, \tag{1}$$

where $1 \leq m \leq k+1$. If $m = 0$, then $\Psi(\{t_0\}) = \psi_{t_0,t_0} = \psi_{0,0} = 0$. □

As a consequence of this, if $k = -1$, $\Psi(\{t_0\}) = \psi_{t_0,t_0} = 0$, and if $k = 0$, $\Psi = \Psi(\emptyset) = \psi_{1,n}$, where $\emptyset$ is the empty thresholding set.

The optimal solution for multilevel thresholding can be characterized in terms of solutions to sub-problems. For this, Ψ and $\psi_{l,r}$ must satisfy positivity, decomposition into sum of independent terms, and a third condition on computing $\psi_{l,r}$.

Condition 1. *For any probability set P and any threshold set T, $\Psi > 0$ and $\psi \geq 0$.* □

Condition 2. *For any integer $m = 1, 2, \ldots, k$, $\Psi(T_{0,m})$ can be expressed as follows:*

$$\Psi(T_{0,m}) = \Psi(\{t_0, t_1, \ldots, t_{m-1}\}) + \psi_{t_{m-1}+1, t_m} \,. \tag{2}$$

□

Condition 3. *If $\psi_{t_{j-1}+1,t_j}$ is known, then $\psi_{t_{j-1}+2,t_j}$ can be computed in $\Theta(1)$ time.* □

The polynomial-time algorithm for optimal thresholding resorts on characterizing the optimal solution in terms of the solutions to sub-problems. Suppose that the optimal solution for $T_{0,j-1} = \{t_0, t_1, \ldots, t_{j-1}\}$, $\Psi^*(T_{0,j-1})$, is known. Then, the optimal solution for $T_{0,j}$, $\Psi^*(T_{0,j})$, is computed as:

$$\Psi^*(T_{0,j}) = \begin{cases} 0 & \text{if } j = 0 \\ \max_{\min\{t_{j-1}\} \le t_{j-1} \le \max\{t_{j-1}\}} \Psi^*(T_{0,j-1}) + \psi_{t_{j-1}+1,t_j} & \text{if } 1 \le j \le k+1 \end{cases} \tag{3}$$

where

$$\min\{t_j\} = \begin{cases} j \text{ if } 0 \le j \le k \\ n \text{ if } j = k+1 \end{cases} \tag{4}$$

and

$$\max\{t_j\} = \begin{cases} 0 & \text{if } j = 0 \\ n-k+j-1 & \text{if } 1 \le j \le k+1 \end{cases} \tag{5}$$

The optimal solution for T is obtained by computing $\Psi^*(T) = \Psi^*(T_{0,k+1})$, and taking $T^* = \arg\max_T \Psi^*(T)$. The optimality is stated in the following theorem, whose proof follows by contradiction. Note that the above characterization of the optimal solution can also be used for a *minimization* criterion by substituting "$\max_{\min\{t_{j-1}\} \le t_{j-1} \le \max\{t_{j-1}\}}$" for "$\min_{\min\{t_{j-1}\} \le t_{j-1} \le \max\{t_{j-1}\}}$" in (3).

Theorem 1. *For any value of $t_j = j, j+1, \ldots, n$, $\Psi^*(T_{0,j})$ computed as in (3) is optimal by knowing the optimal value of Ψ for $T_{0,j-1}$, $\Psi^*(T_{0,j-1})$, for $\min\{t_{j-1}\} \le t_{j-1} \le \max\{t_{j-1}\}$.* □

Due to the recursive nature of the optimal solution for $T_{0,j}$, it is not difficult to see that a recursive algorithm can be derived by considering the base case for $j = 0$ and the general case for values of $j \ge 1$. Such an algorithm would run in the worst case in $\tau(n, j) = \sum_{i=j}^{n-1} \tau(i, j-1) + \Omega(n)$ time, where i is for computing the maximum and $\Omega(n)$ is needed to compute $\psi_{t_{j-1}+1,t_j}$ in the worst case. However, Equation (1) suggests a dynamic programming solution. This algorithm can be implemented for *any* criterion[2], whenever the objective function satisfies the underlying definitions and conditions. We show here how it can be implemented for the between-class variance criterion. The between-class variance criterion, function $\Psi_{\mathrm{BC}}(T)$, can be expressed as follows:

$$\Psi_{\mathrm{BC}}(T) = \sum_{j=1}^{k+1} \omega_j \mu_j^2 = \sum_{j=1}^{k+1} \psi_{t_{j-1}+1,t_j}\,, \tag{6}$$

where $\psi_{t_j+1,t_{j+1}} = \omega_j \mu_j^2$, $\omega_j = \sum_{i=t_{j-1}+1}^{t_j} p_i$, and $\mu_j = \frac{1}{\omega_j} \sum_{i=t_{j-1}+1}^{t_j} h_i p_i$.

As seen below, it is not difficult to show that $\Psi_{\mathrm{BC}}(T)$ as defined in (6) satisfies Conditions 1 to 3.

[2] The implementations of the entropy-based and minimum error criteria are not difficult to be derived. For the interested reader, they can be found in [9].

Condition 1: For all i, $p_i \geq 0$, and at least one $p_j > 0$, yielding $\omega_j \geq 0$, and at least one $\omega_j > 0$. Also, since at least one $\mu_j^2 > 0$, then at least one $\psi_{t_j+1,t_{j+1}} = \omega_j \mu_j^2 > 0$. This implies from (6) that $\Psi_{\mathrm{BC}}(T) > 0$. □

Condition 2: Assume that $\sum_{j=l}^{r} \omega_j \mu_j^2 = 0$ if $l > r$. Then, for $m = 1, \ldots, k$, $\Psi_{\mathrm{BC}}(T_{0,m})$ can be expressed as follows:

$$\Psi_{\mathrm{BC}}(T_{0,m}) = \left(\sum_{j=1}^{m-1} \omega_j \mu_j^2 \right) + \omega_m \mu_m^2 = \Psi_{\mathrm{BC}}(\{t_0, t_1, \ldots, t_{m-1}\}) + \psi_{t_{m-1}+1,t_m} \,. \tag{7}$$

□

Condition 3: If $\psi_{t_{j-1}+1,t_j}$ is known, then $\psi_{t_{j-1}+2,t_j}$ can be computed in $\Theta(1)$ time. From (6), and using the definitions of ω_j and μ_j, $\psi_{t_{j-1}+1,t_j}$ can be expressed as follows:

$$\psi_{t_{j-1}+1,t_j} = \omega_j \mu_j^2 = \frac{1}{\sum_{i=t_{j-1}+1}^{t_j} p_i} \left[\sum_{i=t_{j-1}+1}^{t_j} h_i p_i \right]^2 \tag{8}$$

$$= \frac{1}{p_{t_{j-1}+1} + \sum_{i=t_{j-1}+2}^{t_j} p_i} \left[h_{t_{j-1}+1} p_{t_{j-1}+1} + \sum_{i=t_{j-1}+2}^{t_j} h_i p_i \right]^2 . \tag{9}$$

Let $a \leftarrow p_{t_{j-1}+1} + \sum_{i=t_{j-1}+2}^{t_j} p_i$, and $b \leftarrow h_{t_{j-1}+1} p_{t_{j-1}+1} + \sum_{i=t_{j-1}+2}^{t_j} h_i p_i$. Since a and b are known from the previous step, then a, b and $\psi_{t_{j-1}+2,t_j}$ can be computed for the current step in $\Theta(1)$ as: $a \leftarrow a - p_{t_{j-1}+1}$, $b \leftarrow b - h_{(t_{j-1}+1)} p_{t_{j-1}+1}$, and $\psi_{t_{j-1}+2,t_j} \leftarrow \frac{b^2}{a}$. □

The algorithm, depicted in Fig. 1, starts by setting the ranges for t_0, t_1, …, t_k, t_{k+1} as in (4) and (5), which is done by invoking findThresholdRanges(k). It uses a table C to store the solution to smaller values of k, and the table is incrementally filled by column, for $k = 1, 2, 3$, and so on. Thus, $C(t_j, j)$ contains the optimal solution for $T_{0,j} = t_0, t_1, \ldots, t_j$, $\Psi^*(T_{0,j})$, for $\min\{t_j\} \leq t_j \leq \max\{t_j\}$. The cell at position $(0, 0)$ is set to 0, while the cell at position $(n, k+1)$ contains the optimal value of $\Psi^*(T)$. As stated in Condition 3, ψ_{i+1,t_j} is computed in $\Theta(1)$ by knowing the value of ψ_{i,t_j}, and stored in variable "psi". The index 'i' in the inner for loop is used to represent t_{j-1}. Similarly, table D stores the thresholds for all the optimal solutions. Thus, $D(t_j, j)$ contains the value of t_{j-1} for which $\Psi^*(T_{0,j})$ is optimal. A detailed example can be found in [9]. The worst-case time complexity of Algorithm *Multilevel_Thresholding* is $\Theta(kn^2)$. For the between-class variance and minimum error criteria, it has been shown that the worst-case time complexity can be even reduced to $\Theta(kn)$, which implies linear time on the number of bins, and this, assuming regularly sampled histograms. For the entropy-based criterion, the worst-case time complexity is still $\Theta(kn^2)$.

Algorithm *Multilevel_Thresholding*
Input: A histogram, $H = \{h_1, \ldots, h_n\}$ with $P = \{p_1, p_2, \ldots, p_n\}$. Number of thresholds, k.
Output: A threshold set, $T = \{t_0, t_1, t_2, \ldots, t_k, t_{k+1}\}$.

```
minTj, maxTj ← findThresholdRanges(k)
C(0,0) ← 0; D(0,0) ← 0
for j ← 1 to k + 1 do
  for t_j ← minTj(j) to maxTj(j) do
    C(t_j, j) ← 0;  psi ← ψ_{j,t_j}
    for i ← minTj(j − 1) to min{maxTj(j − 1), t_j − 1} do
      if C(i, j − 1) + psi > C(t_j, j) then
        C(t_j, j) ← C(i, j − 1) + psi; D(t_j, j) ← i
      end if
      psi ← Compute ψ_{i+2,t_j} from psi, h_{i+1} and p_{i+1}
    end for
  end for
end for
return findThresholds(D)
procedure findThresholdRanges(k: integer)
for j ← 0 to k + 1 do
  if j = k + 1 then minTj(j) ← n else minTj(j) ← j end if
  if j = 0 then maxTj(j) ← 0 else maxTj(j) ← n − k + j − 1 end if
end for
return minTj, maxTj
end procedure
procedure findThresholds(D: table)
T(k + 1) ← n
for j ← k downto 0 do
  T(j) ← D(T(j + 1), j + 1)
end for
return T
end procedure
```

Fig. 1. General algorithm for multilevel thresholding based on dynamic programming

4 Applications

The salient features of the proposed algorithm and general framework for optimal multilevel thresholding is that it can be applied to any criterion and irregularly sampled histograms. This provides a much more general framework than the previous approaches and algorithms. The implications of this generalization are in various aspects and applications of pattern recognition and image segmentation, and also, in general, in signal processing. Three of them are discussed below.

Segmentation of images using multilevel thresholding, in some cases, involves sparse histograms (i.e., histograms that with nonzero probabilities for a reduced number of bins, and spread out over a large domain). This is a typical case in segmentation of images representing the biofilm physical structure, and which are extracted from three-dimensional confocal laser scanning microscopy [1].

These images have a very high resolution and lead to sparse histograms on large domains.

Another important application of the proposed framework is to use it to finding the optimal number of clusters or classes, and still maintaining the optimality of the clustering and/or thresholding. Many indices of validity have been proposed, namely the I index, Davies Bouldin index, Dunns index, and the Xie-Beni index, among others [7]. It is not difficult to see that most of these indices satisfy the three conditions and can be used (i) to find the best number of clusters or classes, and (ii) as multilevel thresholding criteria on irregularly sampled histograms.

Lastly, an extremely important consequence of the proposed framework and algorithm in pattern recognition is in clustering. By modeling a one-dimensional clustering problem as an irregularly sampled histogram, the optimal clustering and optimal number of clusters can be obtained as follows. Assume that a given dataset, $D = \{x_1, x_2, \ldots, x_n\}$, is to be clustered, where $x_i \in \mathbb{R}$. By sorting D in increasing order, obtaining a new dataset, say for convenience in notation, $H = \{h_1, h_2, \ldots, h_n\}$, and assigning each probability $p_i = 1/n$, it is straightforward to see that the proposed framework and algorithm can optimally cluster D using any criteria, and even find the best (optimal, based on a certain criterion) number of clusters.

5 Experimental Results

A few experiments were performed on randomly generated histograms and on multilevel thresholding segmentation of real-life images. For the ease of notation, MTBC is used to refer to the implementation of the between-class variance criterion. The first set of experiments was performed on randomly generated histograms of various sizes and thresholded for various levels (values of k). The algorithm was implemented in Matlab and the experiments were run on a PC workstation at 3.4Ghz with 1GB of RAM. Nine different values of n, $n = 256, 512, \ldots, 65536$. For each value of n, nine different values of k were considered: $k = 2, 3, \ldots, 10$. Thus, for each value of n, nine different histograms were generated by a using a distribution that obeys a mixture of $k + 1$ Gaussians, to which white noise with a $N(0, 0.005)$ was incorporated in order to make it as real as possible. One such histogram generated with a mixture of three Gaussians ($n = 256$ and $k = 2$) is depicted in Fig. 2. The two optimal threshold levels were obtained by MTBC using only 0.005 seconds of CPU time.

The CPU time (in seconds) for all values of n and k are listed in Table 1. It can be observed that increasing n, the time taken by MTBC increases but polynomially. That is, for $2n$ bins, it takes approximately 3-4 times more than the time taken for n bins. This behavior is followed in general for all values of k. On the other hand, for a fixed value of n, increasing the value of k, again, increases the number of CPU seconds but linearly. It is interesting to see that for $n = 65536$ bins (or gray levels in image segmentation), MTBC can find $k = 10$ thresholds in only 2117 seconds, or approximately 35 minutes. Note that

Table 1. CPU times (in seconds) taken by MTBC to find k thresholds, $k = 2, 3, \ldots, 10$ on randomly generated histograms of sizes $n = 2^b$, where $b = 8, 9, \ldots, 16$

$n_{\downarrow}$ $k^{\rightarrow}$	2	3	4	5	6	7	8	9	10
256	0.005	0.007	0.011	0.015	0.020	0.022	0.025	0.029	0.033
512	0.014	0.029	0.043	0.056	0.069	0.085	0.099	0.111	0.126
1024	0.052	0.106	0.164	0.220	0.277	0.331	0.394	0.483	0.508
2048	0.214	0.419	0.655	0.873	1.112	1.340	1.583	1.815	2.064
4096	0.810	1.672	2.510	3.438	4.521	5.341	6.271	7.261	8.147
8192	3.312	6.878	10.152	19.283	17.742	20.966	25.013	29.195	32.668
16384	13.324	26.734	41.047	56.019	69.896	89.723	98.922	116.602	130.022
32768	51.376	105.440	159.379	231.875	287.749	351.783	404.136	469.089	527.862
65536	212.084	429.374	647.685	913.694	1167.983	1400.655	1628.987	1874.508	2116.873

an exhaustive-search multilevel thresholding algorithm (cf. [12]) would need to test $\frac{65536!}{65526!10!} \approx 4.02 \times 10^{41}$ different subsets of thresholds, which implies a huge number of computations. A practical comparison was done with an exhaustive search algorithm on finding 2 and 3 thresholds on a histogram of 2048 bins and took 0.616 and 185.967 seconds for $k = 2$ and 3 respectively, compared with the fractions of seconds taken by MTBC.

Other experiments were conducted with Lena and Peppers, for which the original images are depicted in Figs. 3 (a) and (d). The images segmented using the thresholds found by MTBC are depicted in Figs. 3 (b) and (e). The histogram was constructed for each of these images, and the optimal thresholds were found by using MTBC, where $k = 5, 4$ respectively. The histograms along with the optimal thresholds are depicted in Figs. 3 (c) and (f). All these images are 8-bit graylevel, and the segmentation of the images by computing the optimal thresholds were performed in 0.0224 and 0.0194 CPU seconds respectively. For all the images, an appropriate number of thresholds was found manually, and the histograms reported correspond to the best number of thresholds based on visually analyzing the histograms. However, it is not difficult to see that finding the best number of thresholds automatically is not a difficult task, as it would

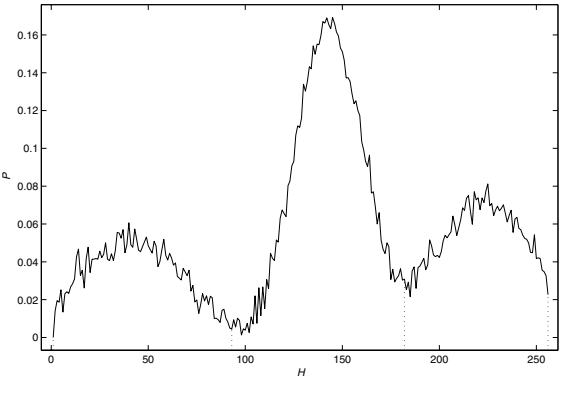

Fig. 2. An example of a randomly generated histogram using a mixture of three Gaussians. The two optimal thresholds were detected using MTBC.

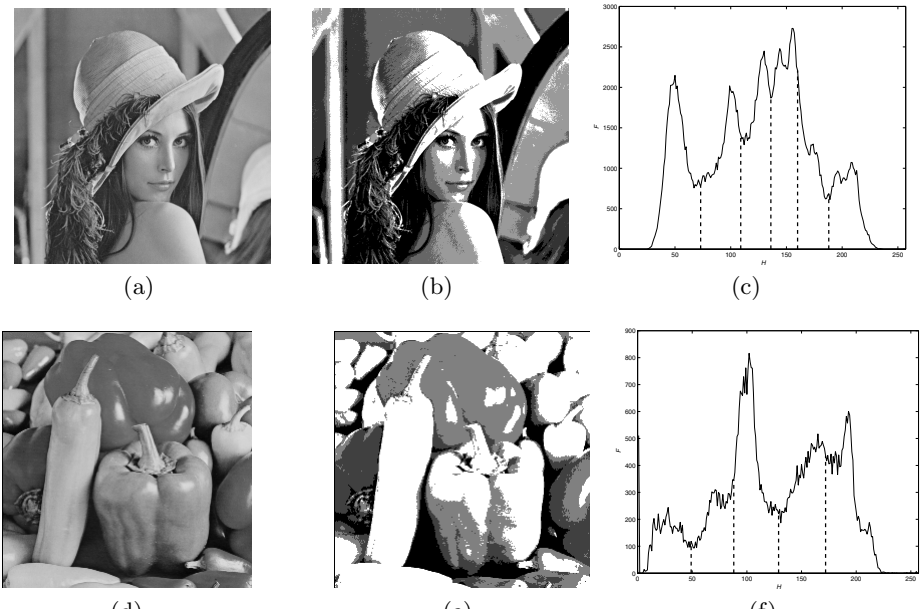

Fig. 3. Lena and Peppers segmented with the MTBC algorithm and the corresponding histograms with the optimal thresholds

imply to test an index of cluster validity after each column of table C is completed – thus, the complexity would remain the same.

6 Conclusions

An efficient algorithm for optimal multilevel thresholding of irregularly sampled histograms is proposed, which runs in polynomial time. A general algorithm based on dynamic programming is discussed, for which the optimality of the solution is also stated. Experiments have been carried out on randomly generated histograms and on segmentation of real-life grayscale images showing the efficiency of the algorithm. Applications of the proposed framework to different pattern recognition problems have been discussed, being the most important optimal one-dimensional clustering in polynomial time. As a consequence of this, clustering and thresholding problems for more than one dimension are topics that deserve further investigation.

References

1. Beyenal, H., Donovan, C., Lewandowski, Z., Harkin, G.: Three-dimensional Biofilm Structure Quantification. Journal of Microbiological Methods 59(3), 395–413 (2004)
2. Fan, S., Lin, Y.: A Multi-level Thresholding Approach Using a Hybrid Optimal Estimation Algorithm. Pattern Recognition Letters 28, 662–669 (2007)

3. Kapur, J., Sahoo, P., Wong, A.: A New Method for Gray-level Picture Thresholding Using the Entropy of the Histogram. Computer Vision Graphics and Image Processing 29, 273–285 (1985)
4. Kittler, J., Illingworth, J.: Minimum Error Thresholding. Pattern Recognition 19(1), 41–47 (1986)
5. Liao, P., Chen, T., Chung, P.: A Fast Algorithm for Multilevel Thresholding. Journal of Information Science and Engineering 17, 713–727 (2001)
6. Luessi, M., Eichmann, M., Shuster, M., Katsaggelos, A.: New results on efficient optimal multilevel image thresholding. In: Proc. of the IEEE International Conference on Image Processing, Atlanta, USA, pp. 773–776. IEEE Press, Los Alamitos (2006)
7. Maulik, U., Bandyopadhyay, S.: Performance Evaluation of Some Clustering Algorithms and Validity Indices. IEEE Trans. on Pattern Analysis and Machine Intelligence 24(12), 1650–1655 (2002)
8. Otsu, N.: A Threshold Selection Method from Gray-level Histograms. IEEE Trans. on Systems, Man and Cybernetics SMC-9, 62–66 (1979)
9. Rueda, L.: A Polynomial-time Algorithm for Optimal Multilevel Thresholding (submitted, 2008), `http://www.cs.uwindsor.ca/~lrueda/papers/PolyThresJnl.pdf`
10. Sahoo, P., Wilkins, C., Yeager, J.: Threshold Selection Using Renyi's Entropy. Pattern Recognition 30(1), 71–84 (1997)
11. Wang, S., Chung, F., Xiong, F.: A Novel Image Thresholding Method Based on Parzen Window Estimate. Pattern Recognition 41(1), 117–129 (2008)
12. Wu, B., Chen, Y., Chiu, C.: Efficient Implementation of Several Multilevel Thresholding Algorithms Using a Combinatorial Scheme. International Journal of Computers and Applications 28(3), 259–269 (2006)
13. Yan, H.: Unified Formulation of a Class of Optimal Image Thresholding Techniques. Pattern Recognition 29(12), 2025–2032 (1996)
14. Yen, J., Chang, F., Chang, S.: A New Criterion for Automatic Multilevel Thresholding. IEEE Trans. on Image Processing 4(3), 370–378 (1995)
15. Yin, P.: A Fast Scheme for Optimal Thresholding Using Genetic Algorithms. Signal Processing 72, 85–95 (1999)
16. Zahara, E., Fan, S., Tsai, D.: Optimal Multi-thresholding Using a Hybrid Optimization Approach. Pattern Recognition Letters 26, 1082–1095 (2005)
17. Zhou, Y., Yang, K.: New 2D Adaptive Image Thresholding Method Based on Within and Between Cluster Scatter. In: Proc. of the 27th International Congress on High-Speed Photography and Photonics, SPIE, vol. 6279, p. 62793N (2007)