Buletin Ştiinţific - Universitatea din Piteşti
Seria Matematică şi Informatică, Nr. **14** (2008), pg.1-N

Fast K-Means Image Quantization Algorithm and Its Application to Iris Segmentation

Nicolaie POPESCU-BODORIN*

June, 2008

Abstract

The main algorithm presented in this paper (Fast K-Means Image Quantization) is an adaptation of the generic k-means algorithm, tuned for fast chromatic clustering of both 24-bit RGB and Grayscale continuous-tone still images. Here we show that, in some specific conditions, the automatic discovery of the area morphology can be done through a non-deterministic algorithm mainly controlled by the image itself. The most important properties of the Fast K-Means Image Quantization Algorithm are also presented here in association with some newly proposed principles of Optimal K-Means Downsampling which have been proven to be closely related to the algorithm. Relevant results in Iris Segmentation are presented as a practical application of the Fast K-Means Image Quantization Algorithm.

2000 Mathematics Subject Classification: Primary: 68U10, 62H35, 68T45; Secondary: 62P30, 68W99;

Key words and phrases: iris segmentation, k-means, chromatic clustering, image quantization, k-means quantization, k-means clustering, k-median filter, downsampling, equipotential chromatic map, optimal k-means downsampling, fast k-means image quantization algorithm;

*Spiru Haret University, Dept. of Mathematics and Computer Science, 13 Ion Ghica Street, Bucharest 3, România; Email: bodorin [a] ieee . org;

1 Problem statement

One fundamental question in the present paper is *how simple can the image segmentation be?* Assuming that the segmentation is meant to discover area morphology, we might try to acquire both this kind of knowledge and the image itself, at the same time, in the same process. It is generally accepted that image acquisition is mainly a quantization process. Therefore we can rephrase our question as follows: *is there an image quantization suitable for enhancing area morphology?*

To avoid any confusion we will remind the reader that the terms downsampling / downsizing / subsampling (and sometimes resampling) are used in image processing to identify *a procedure by which the spatial resolution of an image is reduced* [1]. At first sight, there is no connection between downsampling and quantization defined as being the process in which *the amplitude of a signal is compared to a set of decision levels* [2]. But, in fact, downsampling decreases the storage of an image, creating (and ignoring) gaps in the original image, gaps that will be further reconstructed by some median filter. Consequently, downsampling/upsampling procedures retain in the image storage other values than those originally being stored by the initial quantization of the image. This fact, which implicitly links together the two concepts discussed, enables us to formulate the following problem: *for a given image, is it possible to identify a set of reconstruction values and a set of neighborhoods, both of the same length, ensuring that the mean of each neighborhood is the corresponding reconstruction value?*

Further in this paper we will show that the Fast K-Means Algorithm defines a quantization procedure and an equivalent median filtering mechanism, both suitable for enhancing area morphology in the processed image.

2 Image quantization, downsampling and median filtering

The classical approach to downsampling is to consider the minimum coded units as being blocks, or much more generally neighborhoods, sometimes mutually exclusive, sometimes not, but small enough to minimize the visual effects of some median filter applied to them. In this

way, the initial image is viewed as being covered by a uniformly spaced grid defining the blocks, or by a uniformly shaped collection of neighborhoods. A lucky guess of such a cover ensures that chromatic variation across these neighborhoods is smaller than a certain threshold. Otherwise, an important (noise-like) local chromatic variation corresponding to an essential detail may produce a strong unwanted visual effect.

Downsampling can also be done adaptively to the amplitude of the high frequency components [3] as well as by varying the threshold and the type of neighborhoods according to local chromatic variation. By doing any of these, we negotiate between making a more accurate numerical representation of any particular processed image and preserving enough regularity in the neighborhoods shape to ensure computational efficiency.

Any median filtering on imposed uniformly shaped neighborhoods does not solve the following issues:

- The encoding of one fixed chromatic value depends on the local chromatic medians computed across those neighborhoods containing that value, instead of depending on the value itself. In this way, we are reducing the entropy in each of the minimum coded units but we are introducing in the encoded image an entropy that wasn't really there at the beginning (an intrinsic noise of the median filter that has been applied). For an arbitrary median filter the existence of an equivalent image quantization is not guaranteed.

- No matter how strong the visual effects are, median filtering on imposed uniformly shaped neighborhoods will always produce new edges placed on each border of the filtered neighborhood. This is also another noise, intrinsic to median filters, whose visibility increases with the regularity of the neighborhood collection covering the image and also with the extent of these neighborhoods. In other words, this kind of median filtering tends to overwrite the area morphology originally stored by initial image quantization.

3 Principles of k-means optimal downsampling

This paper is an outcome of several experimental studies during which we identified k-means algorithms as being a viable solution for defining an image quantization and a median filter that are equivalent to each other. There are some new principles of k-means optimal downsampling

underlying this approach; these principles were suggested step by step by our practice and set out as follows:

1. Any image is a self-described package of information, each chromatic layer being an array of pixels with random chromatic values for which the location is not important (when we are downsampling an image, in fact we are truncating the histogram to get more redundancy in order to reduce the entropy). This principle ensures that computed centroids do not depend on the area morphology of the processed image, but only on the image histogram.

2. During the downsampling process it is mandatory to maintain a functional correspondence between the chromatic values within the original image and those within the processed image. This is the minimal requirement which downsampling (and/or median filters) must meet in order to match a quantization (non-functional graphs couldn't match an image quantization). Doing otherwise is against the first principle above and also, from a practical point of view, it means implicitly adding to the downsampled image a description of some facts that never existed in the original image.

3. In the k-means context, the set of optimal reconstruction values (of a quantization at any given number of chromatic levels) and also the shape, dimensions and extent of the elements within the neighborhood set covering the image are all intrinsic properties of the given image and consequently need not be imposed during the computation. The morphology of the image is self-explanatory.

4. Any robust efficient implementation of the k-means algorithm designed to process at least a subclass of 8-bit/channel images must support uint8[1] acceleration over that subclass, or in other words, the normalized double representation of the processed image must be a stability point of a computational mechanism tolerating at least round-off error perturbations, meaning that for a given image and for a given choice of an initial set of centroids, if we substitute the uint8 representation for the double normalized one, the algorithm must converge nearly to the same solution (to a nearly optimal solution).

[1]Matlab naming convention for 8-bit unsigned integer data type

4 Fast k-means image quantization algorithm

The formulation of the problem had suggested from the beginning that the k-means algorithm is the first candidate for answering the questions raised in the first section of this paper. A generic representation of the k-means algorithm is the following:

K-Means Procedure;
INPUT: dataset to clusterize, desired number of clusters, eventually other custom data (such as the initial choice of centroids);
    While the termination condition isn't satisfied:
        For each element in the dataset:
            Find the closest centroid (ambiguity must also be treated);
            Mark the inclusion of the element within the corresponding cluster;
        EndFor;
        If none of the elements in the dataset changes its cluster then:
            Fulfill the termination condition;
        EndIf;
        For each of the computed clusters:
            Recompute cluster centroid as being the mean of contained elements;
        EndFor;
    EndWhile;
OUTPUT: Computed centroids; Computed clusters;

The main algorithm presented in this paper is an adaptation of the generic k-means algorithm, tuned for fast chromatic clustering in Matlab. It takes all computational advantage of working with the uint8 representation of 24-bit images:

-Collapsing memory storage by avoiding chromatic values redundancy: as can be seen in the generic k-means algorithm, there is no need to keep track of the position of the pixels in the image. This enables one to exploit the major redundancy of chromatic values which will make the image storage to collapse. In the k-means context, there are only two

important data to retain from each luma-chroma plane of the input image: what unique chromatic values it contains and how many times they appear. Consequently, the exact histogram of the given chromatic layer is the proper storage to work with (is the shortest container for the two required pieces of data). All the information we need from the original chromatic layer can be contained in a maximal 2x256 array, no matter how big the original image.

-Collapsing memory storage by avoiding inclusion index redundancy: initially, the index memorizing the inclusion of each pixel to one specific cluster is a two-dimensional array as wide as the original image, but again, its value depends on the chromatic values of the corresponding pixels instead of depending on the pixels position in the image. Therefore, the initial inclusion index can also be efficiently stored in a maximal 2x256 uint8 array containing duplicate data of the first part of the exact histogram (maximum 256 unique values) and corresponding inclusion index of each chromatic value.

The algorithm also makes the most of Matlab acceleration mechanisms such as vectorized optimized loops and efficient handling of large spreaded array subsets (loops elimination) through the use of logical indexing. At first sight, these optimizations seem to add an unnecessary (or even unacceptable) particularization of the algorithm, but all of these mechanisms are replicable, even with better time performances, in other programming languages.

Taking into account that later in this paper we will be interested in estimating the algorithms performance, isolating the true computational kernel from input, output and collapsing operations is the proper way to formulate the algorithm:

function RCP = ResampleKmeansUint8(CP, NC);
$[EH, OC]$ = ExactHistogram(CP);
% deflates chroma plane in its exact histogram;
$[C, CI]$ = KernelKmeansUint8(EH, OC, NC);
RCP = ParseOutput(CP, CI, EH, C);
% inflates EH and CI to reconstruct chroma plane;

function $[C, CI]$ = KernelKmeansUint8(EH, OC, NC);
CI = uint8(zeros(size(EH‘))); C = uint8([]);
CC = uint8($[1 : NC]$); T = false;
while $\neg T$
C = GetNewCentroids(C, EH, OC, CI, NC, CC);
$[T, CI, CC]$ = RecomputeClusters(EH, OC, C, CI, T);
end;

where the meaning of each variable is as follows: RCP - requantized chroma plane array; CP - chroma plane array; NC - desired number of clusters; EH - array of values appearing in exact histogram of CP; OC - array of occurrences accounted for each of the values of EH; C - array of computed centroids; CI - collapsed cluster index; T - termination flag; CC - index of changed clusters

5 Properties of the Fast K-Means Image Quantization Algorithm

5.1 Speed

The main improvement against the generic k-means algorithm is the fact that, no matter how big the initial image, the computational kernel here is working only with a maximal set of 1024 values for each luma-chroma layer: maximum 3x256 uint8 values for EH, CI and C (but usually NC is small and consequently C is also shorter than EH), and maximum 256 unsigned integer values (uint32 for very large images such as 65536x65536 images) for OC. Therefore, the computational complexity of the computational kernel *does not depend on the size of the processed image*. This is why the computational kernel of the Fast K-Means Image Quantization Algorithm truly does have an incredible speed.

Figure 1 shows the correspondence between the average times spent by each function within the main algorithm and the desired number of clusters in each of the Cb, Cr chroma layers. The average times are computed running the algorithm for 84 images (24-bit, 512x512, RGB).

It can be seen in Figure 1 that the use of logical indexing breaks the initial exponential slope of the ParseOutput function. Even so, by comparison with all other functions, it still seems to be far from a highly optimized variant. But taking into account that the time complexity on that slope is O(kHW), with k being the desired number of clusters and H,W being the dimensions of the image, we can conclude that it is a fairly optimized variant because the initial exponential trend of the time complexity line is given by exponential growth of k values.

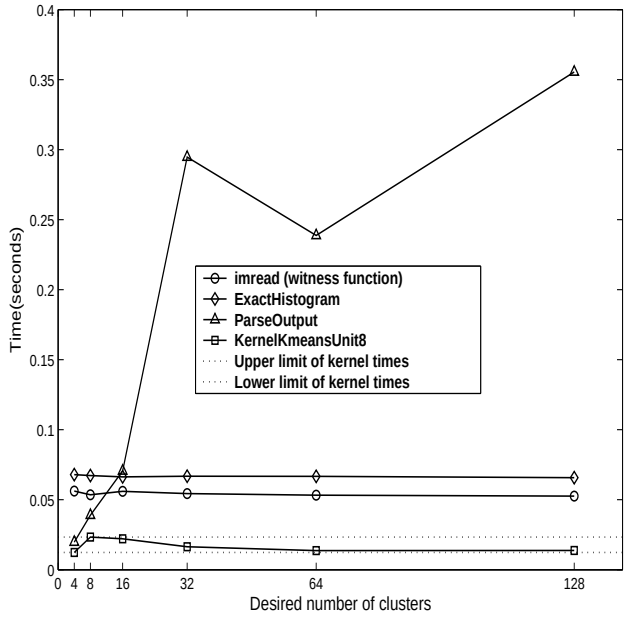

Figure 1: Speed of the computational kernel of the Fast K-Means Image Quantization Algorithm

Figure 1 also shows that the computational kernel of the Fast K-Means Image Quantization Algorithm runs very fast indeed and its time complexity line is tightly bounded in a narrow band (0.013-0.025 seconds when executed on a 2.8GHz Pentium IV Prescott processor), far below any polynomial or superpolynomial time [4] . All the data in the Figure 1 are extracted from Matlab Profiler.

5.2 K-means image quantization

In each chroma layer, each chromatic value is replaced by the centroid of the cluster containing it, hence the fast k-means algorithm formulated above requantizes the original chroma plane using the centroids of computed clusters as reconstruction values.

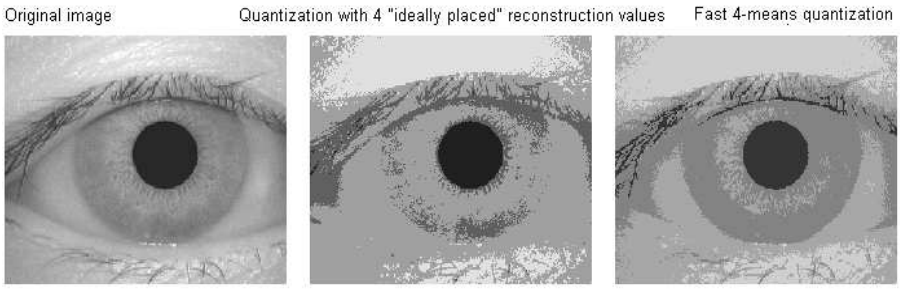

Figure 2: A 4-means quantization example

Being a quantization, the k-means algorithm is implicitly an approximate reading operation on the original data, something that median filters are usually not.

It is generally accepted that under ideal conditions [2] (almost never satisfied by the k-means clustering mechanism) the optimal choice for the reconstruction values of a scalar quantization is to place them in the middle of each decision interval. Our k-means requantization certainly doesn't do that at all. The difference between the k-means computed centroids and those "ideally placed" is a measure of how ideal the real images aren't, and also for how unnatural some ideal statistic criteria are to the human eye and to the image itself. A telling example can be seen in Figure 2 suggesting that, at least in chromatic clustering, a good strategy is to assume that an image is a collection of pixels with no specific properties, and to keep considering that until the moment when the image itself tells us something else. Taking any other way means overwriting the initially available information within the original image hoping to obtain better results from an ideal hypothesis rather than from an approximate reading.

5.3 K-means equipotential chromatic map

Let us consider a grayscale image viewed as a continuous 3-D geographic map M, in which each point is placed at a certain height according to its chromatic value. By projecting k-means clusters onto the vertical axis, we obtain the decision levels of that implicit image quantization defined by k-means. Conversely, each cluster is a geographic area (a 3-D surface) situated between certain heights. But in the requandized image, each chromatic cluster is replaced by its (chromatic) centroid, meaning that each point within the map M is projected onto a horizontal plane corresponding to its nearest centroid. Hence the requantized map M', which is also the algorithm's final result, is still a 3-D map but it contains only horizontal "flatted segments" (only 2-D surfaces) situated at certain heights defined by computed centroids. All pixels within the same cluster in M share the same reconstruction value in M'; in other words, they share the same potential in M'. Therefore, *k-means equipotential chromatic map* seems to be a proper name for the result of the algorithm.

The map M' resulting from the k-means algorithm can also be viewed as a *3-D histogram* with k (very unusual) beans but we found this name to be highly confusing and irrelevant to the context. By projecting M' onto the horizontal reference plane we obtain the requantized image R, a contour plot of M' which also preserves the chromatic segments of M' (see Figure 3 or the third image in the Figure 2). This is why, despite the dimensional ambiguity, even the requantized image can share the same name with M' (requantized image R is a 2-D container for all 3-D information within M').

5.4 K-means median filter

The case when an image quantization is also a median filter is very rare indeed. But k-means defines a special median filter. Usually, median filtering is done over an image partition which presents some type of regularity. It is generally accepted that the image partition must possess some kind of regularity in order to ensure computational efficiency. Addressing chunks of memory data having a variable shape and extent could be a very problematic issue indeed. The solution here is *the collapsed cluster inclusion index*: we access the pixels by chromatic value

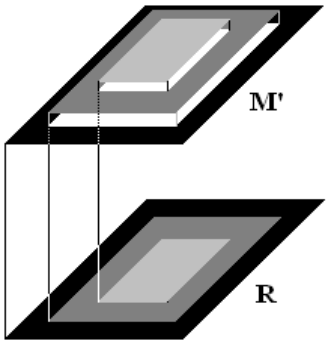

Figure 3: An equipotential chromatic map

using logical indexing instead of accessing them by position. This enables us to work efficiently with such irregular and unusual image partitions as the k-means chromatic clusters are.

The *k-means equipotential chromatic map* defines the image partition on which k-means quantization acts exactly like a median filter.

5.5 K-means chromatic segmentation

Another important aspect is that the Fast K-Means Image Quantization Algorithm tends to 'recognize' segments (clusters) that are also meaningful to the human understanding of the picture. This is because the algorithm tends to ignore relatively constant chromatic areas in direct relation to their extent and to their position in their cluster: a relatively small area with a relatively small chromatic variation doesn't usually have an important contribution to the mean of the cluster containing it, especially when the values in that area are very different from the clusters centroid.

5.6 Geometrical meaning of k-means chromatic clustering

Taking into account all the above properties, the geometrical meaning of the k-means chromatic clustering can be set out as follows: it transforms an image into an *equipotential chromatic map* that satisfies the k-means optimality criteria. The sum of the point-to-centroid distances is min-

imized over all partitions with k chromatic levels, i.e. over all of the others *k-levels equipotential chromatic maps* we could choose. Accepting this sum as an adjacence measure between original image and its *k-means equipotential chromatic map*, we can say that the k-means quantization algorithm assign k reconstruction values in such a way that maximize the adjacence between the original image and the requantized image.

We must bear in mind that Fast K-Means Image Quantization Algorithm works mainly in the 8-bit unsigned integer domain, and therefore the computed centroids are nearly optimal.

5.7 When could it go wrong?

When used as segmentator, the k-means chromatic clustering algorithm can go wrong if the original image is just a pure constant chromatic gradient (for example, a black-to-white, left-to-right constant chromatic gradient with constant columns) and also if the original image doesn't really have any segments, or more generally, if the area morphology and chromatic segments (clusters) are strongly decorrelated. This is why the k-means image quantization algorithm couldn't be considered a robust acquisition method for generic images. But conversely, the k-means algorithm leads to very good segmentation results whenever the correlation between image morphology and chromatic variation is sufficiently strong.

6 Iris segmentation

The most important application of the Fast K-Means Image Quantization Algorithm that we have done so far is the iris segmentation. We consider the generic process of iris recognition [5] as being divided into the following parts:

(1) acquire the image;
(2) crop the acquired image to the significant area;
(3) locate the pupil and its center;
(4) perform iris segmentation;
(5) detect and reject cases of visual exposure inconsistency;
(6) compute the Key Features Array from the extracted iris area;

(7) compare the Key Features Array against those stored in some ID Keys Database.

This paper investigates in particular the localization of the pupil and its center, as well as the next two steps (i.e. 3, 4, and 5).

Initially, pupil localization was done for some images from the old CASIA V1 database (see figure 4) and this has been proved to be simple enough to achieve using the Fast K-means Image Quantization Algorithm. It converged in very few iterations to a very good approximation of the pupil area (the target signal), slightly perturbed only by some specular light and by some eyelash points.

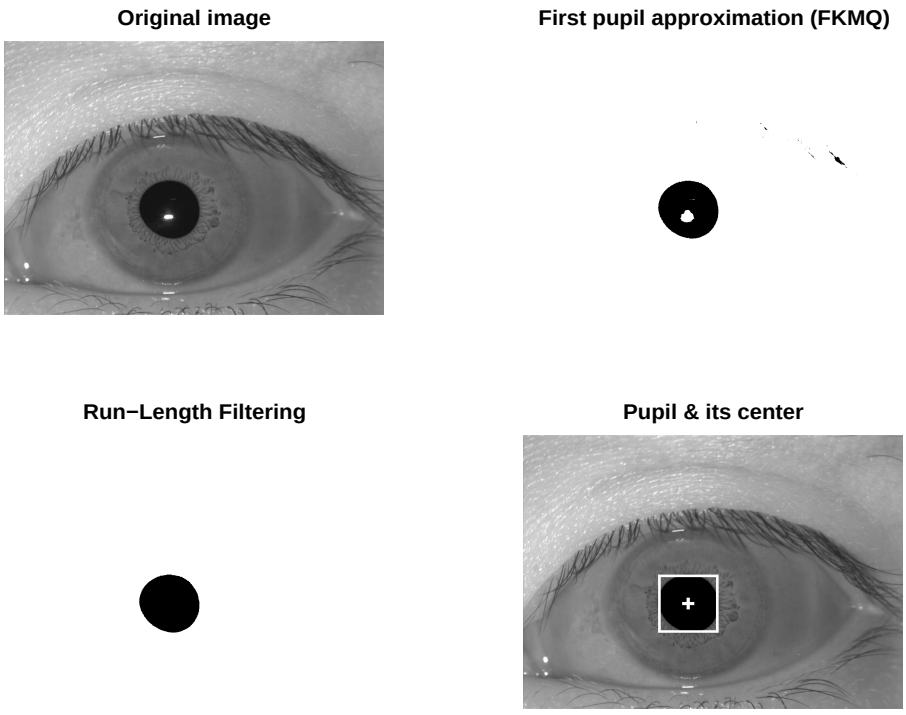

Figure 4: Fast 16-Means Iris Segmentation: Finding the pupil and its center through Fast-K-Means Quantization and Run-Length Filtering (Original image from old CASIA V1 Iris Database).

Here both specular light and eyelashes have the meaning of noise. The signal-to-noise ratio being good enough, filtering the unwanted noise was not a problem. We used Run-Length Encoding to filter the noise, deleting pixels around the pupil (erosion) and dilating the remaining cluster to fit its interior (or to close its interior when the specular light perturbed the pupils border, or both of the cases).

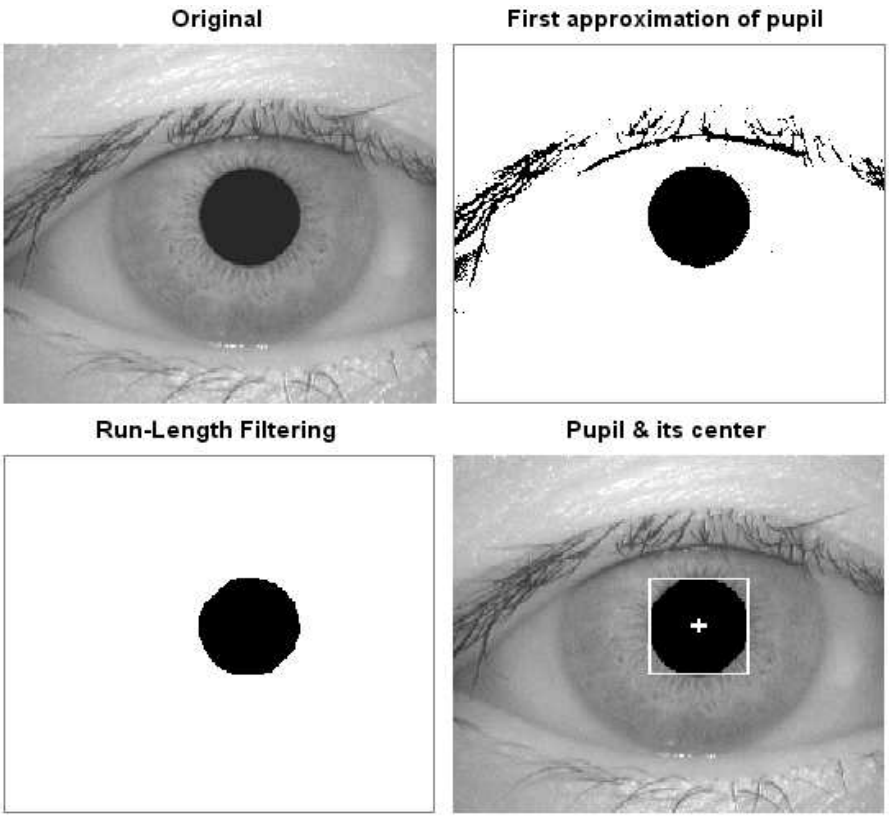

Figure 5: Fast 4-Means Iris Segmentation: image from the present CASIA V1 iris database - noisier eyelashes.

In the context of finding the location of the pupil through the k-means algorithm, the current CASIA V1 database has proved to be more challenging. This is because the images are pre-processed, meaning that the pupil is filtered and there is no specular light on the pupil. There are

two kinds of data loss involved in this operation: the specular light cant be further used as an estimation of pupil location (whenever it exists, the specular light on the pupil is the easiest area to locate in the entire image), and the second and more important, the chromatic values in the pupil area grew closer to those of the eyelashes. Therefore, the first approximation of the pupil computed through Fast K-Means Image Quantization Algorithm became noisier (figure 5) and all the segmentation procedure had to be recalibrated. All in all, the current V1 database has proved to be more demanding when it came to testing the robustness of our segmentation algorithm. For all of the reasons above, we have chosen the current CASIA V1 database for further consideration in this paper.

6.1 Iris segmentation algorithm

Our iris segmentation algorithm is the following:

Generic Iris Segmentation Procedure;
1. INPUT: current iris image and eventually other custom data (calibration variables);
2. Run the Fast 4-Means Quantization adaptively to the consistency of the first cluster to obtain the first approximation of the pupil;
3. Use Run-Length Filtering to denoise (eliminate surrounding noise and fill the gaps in the pupil, if any);
4. Identify the pupil with all remaining pixels in the pupils cluster;
5. Compute the center of the pupil and the pupil radius;
6. Filter the entire image with a central potential field of energy originating in the center of the pupil (the iris will become darker and the exterior of the iris whiter).
7. Extract the entire cluster obtained in (2) from the filtered image obtained in (6);
8. Binarize the result of (7) through Fast 2-Means Quantization;
9. Use Run-Length Filtering to denoise (if necessary) the result of (8);
10. If the result in (9) satisfies some consistency criteria then go to (11), else ask for the next input and go to (1);
11. OUTPUT: result of (9) is a binary index of a noisy iris segment.

The algorithm yields very good results but as a whole it is still experimental. Pupil localization is fully calibrated for the CASIA V1 database

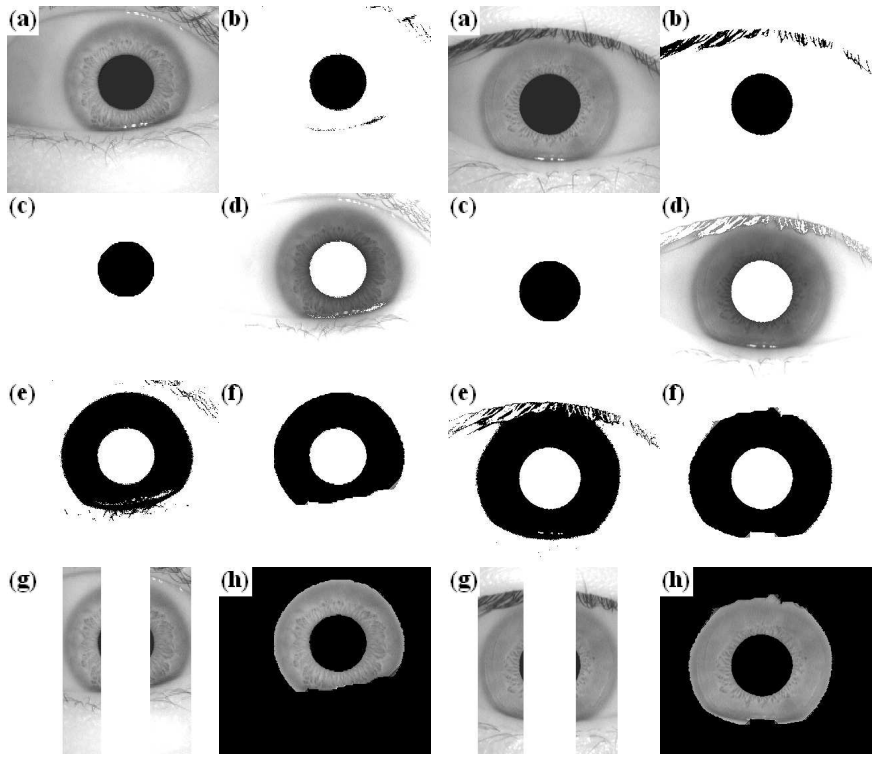

Figure 6: Iris segmentation examples
(Original images from CASIA V1 Iris Database)

and its success rate is nearly 100% on this database. It went wrong (3-pixel error for the pupil center) in a single case which has also being rejected for severe inconsistency of the iris segment (sleepy eye, very noisy eyelashes, pupil severely obstructed by the upper eyelid, all at once).

Some examples of iris segmentation results can be seen in figures 6 and 7:
(a) - eye image;
(b) - initial pupil chromatic cluster obtained in (1);
(c) - location of the pupil and its center computed in (3-5);
(d) - potential filtering (6,7);
(e) - 2-means binarization (8);

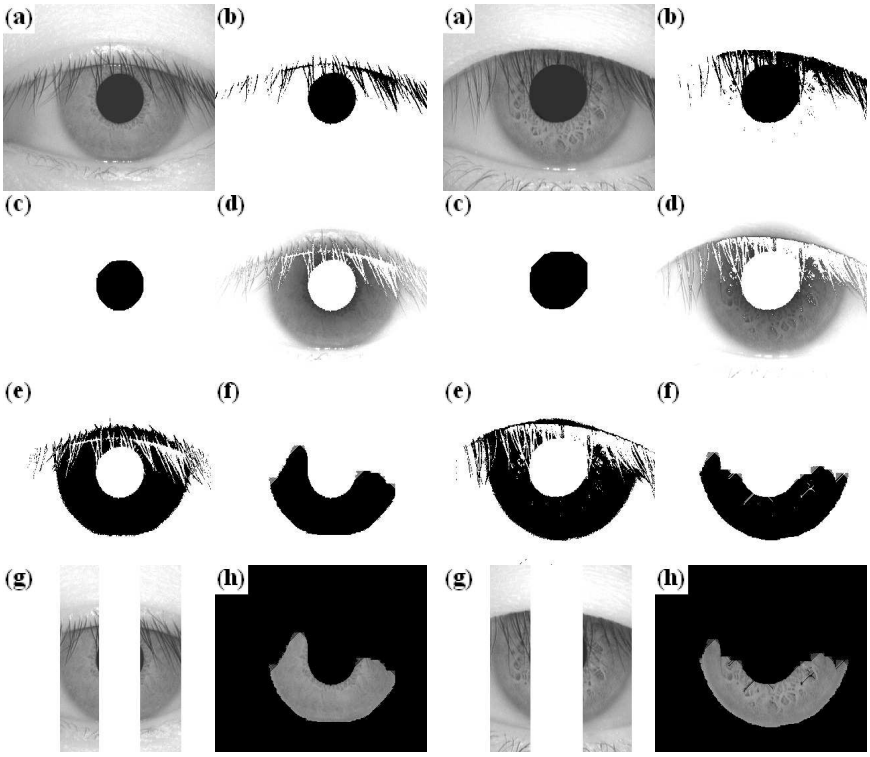

Figure 7: Iris segmentation examples: noisier cases - ocluded iris area (Original images from CASIA V1 Iris Database)

(f) - run-length filtering (9);
(g) - left-to-right extent of the iris;
(h) - first approximation of the iris segment (10,11).

7 Conclusion

This paper shows that iris segmentation can be done using the Fast K-Means Image Quantization Algorithm.

In broader terms, the Fast K-Means Image Quantization Algorithm defines a quantization procedure and an equivalent median filter, both

suitable for enhancing area morphology in the images in which there is a sufficiently strong correlation between the morphology and the chromatic variation.

By applying the Fast K-Means Image Quantization Algorithm to iris segmentation we show that there is at least one important practical problem in which segmentation is as simple as quantization, median filtering or k-means algorithm.

Third party copyrights

Parts of the research in this paper use the CASIA-IrisV1 database collected by the Chinese Academy of Sciences Institute of Automation (CASIA).

Details (including data format, copyright agreement and application procedure) of this database can be found here:
http://www.sinobiometrics.com/english/Databases.asp

References

[1] International Telecommunication Union (ITU), The International Telegraph and Telephone Consultative Committee (CCITT), International Standard ISO/IEC, 10918-1: 1993(E), CCITT Recommendation T.81, Digital compression and coding of continuous still images - Requirements and Guidelines, Section 3 (Definitions, abbreviations and symbols), 1993.

[2] William K. Pratt, Digital Image Processing, Chapter 6 (Image Quantization), John Wiley Sons, 2001, third edition.

[3] Weisi Lin, Li Dong, Adaptive Downsampling to Improve Image Compression at Low Bit Rates, IEEE Transactions on Image Processing, vol. 15, no. 9, September 2006 2513.

[4] David Arthur, Sergei Vassilvitskii, How Slow is the k-Means Method?, Stanford University, http://www.cs.duke.edu/courses/spring07/cps296.2/papers/kMeans-socg.pdf

[5] Li Ma, Tieniu Tan, Yunhong Wang, Dexin Zhang, Efficient Iris Recognition by Characterizing Key Local Variations, IEEE Transactions on image processing, vol. 13, no. 6, June 2004, 739.