



## Image Normalization Algorithms

The objective of an image normalization algorithm is to reduce the impact of image intensity variations caused by changes in the environmental conditions. For example, when measurements of electrical capacitance are used, local variations in the finger humidity or changes in the humidity between sessions may cause significant offsets in the corresponding intensity distributions. Similarly, pressure differences may result in changes of the local intensity. The normalization algorithm is applied in a preprocessing step, prior to both the enrolment and verification procedures to compensate for such effects.

The image variations can be categorized depending on whether they occur in space or time:

- **Intra-session variations:** Image variations occurring within a single image acquired at one specific occasion. For example, local variations in humidity or pressure may cause intra-session intensity variations.
- **Inter-session variations:** Image variations occurring between images acquired at different points in time, for example when the humidity of a finger changes from one occasion to another.

The aim of the image normalization algorithm is to compensate for both intra-session and inter-session variations. Two easy-to-implement normalization methods are proposed below.

### Local Mean Adjustment

A straightforward way to perform the normalization is to adjust the pixel values using the local mean value of each neighborhood. That is, the pixel values are normalized by subtracting the mean value of a local neighborhood and adding a fixed mean value (typically 128 when operating on eight-bit grey scale images). Of course, intensity values outside the range (e.g. [0, 255]) need to be truncated. More formally, pixel value  $I_{ij}$  is re-assigned as follows:

$$I'_{ij} = I_{ij} - \Omega_{ij} + 128$$

where  $\Omega_{ij}$  is the mean intensity value of the local neighborhood. The truncation is carried out as follows:

$$I''_{ij} = \begin{cases} 0, & \text{if } I'_{ij} < 0 \\ I'_{ij}, & \text{if } 0 \leq I'_{ij} \leq 255 \\ 255, & \text{if } I'_{ij} > 255 \end{cases}$$

We are using as size of the local neighborhood an 8x8 pixel area. Note that at the image borders, one has to skip preprocessing, or mirror the image. Note also that the newly assigned pixel value cannot be changed in the original image until all filtering of the given pixel position has been performed. Thus, the processing will require a certain amount of RAM memory.

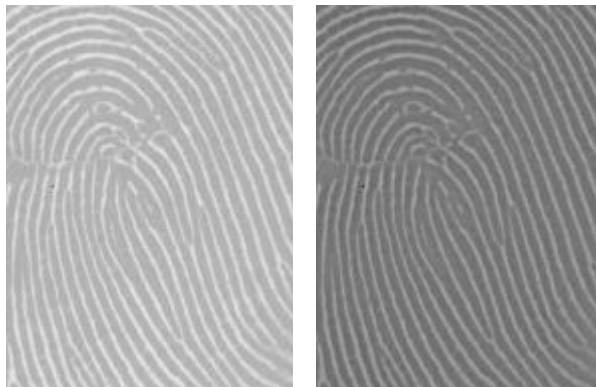


## FINGERPRINTS

The following figures shows two captured images before and after processing:



*Figure 1, Before and after local mean adjustment.*



*Figure 2, Before and after local mean adjustment.*

The following section includes a high level description for performing the local mean normalization process (without an image mirroring process):

1. *Step through each pixel position in the image, (skipping any pixel close to the image border).*
2. *For a given pixel position, (with value  $I$ ), calculate a mean value  $\Omega$  of the image in an  $8 \times 8$  pixel neighborhood centered around the given pixel.*
3. *Calculate a new value of the given pixel position as  $I' = I - \Omega + 128$ . Make sure the value is no larger than 255 and no smaller than 0. Store this value in a separate image memory area.*
4. *Repeat step 1-3 until all pixels have been processed.*
5. *Now, all pixels have been processed, and the mean value adjusted image is complete.*



### Global Histogram Stretching

A method for further enhancing the visual appearance of an image is to apply a global histogram stretching of the image. This is particularly useful for images with a low initial contrast. The local mean-value adjustment detailed in the previous section estimates and compensates for local offsets. However, a global histogram stretching algorithm also adjusts the scale or dynamic range of the intensity distributions. The latter normalization step is detailed in this section.

The basic idea of the method is to make sure that the histogram of the image makes full use of the available dynamic range. If we start with an image where the darkest pixels are dark gray, and the lightest pixels are light gray, this algorithm will rescale the pixel values so that the darkest pixels become all black, and the lightest pixels become all white. The pixel values in between will be scaled linearly between the max and the min values.

Taking this method one step further, we can align the bottom and top  $n$  percent of the distribution with the range limits. Of course, this generalization introduces non-linearities in the normalization algorithm.

More formally, the pixel value  $I_{ij}$  is re-mapped as follows:

$$I'_{ij} = \frac{I_{ij} - I_{\min}}{I_{\max} - I_{\min}} \cdot 255$$

where  $I_{\min}$  and  $I_{\max}$  denote the minimum and maximum intensity values, respectively. For the generalised method,  $I_{\min}$  and  $I_{\max}$  are the intensity values corresponding to the  $n^{\text{th}}$  and  $(1-n)^{\text{th}}$  percentiles of the distribution. In this case, the mapping is followed by a truncation:

$$I''_{ij} = \begin{cases} 0 & \text{if } I'_{ij} < 0 \\ 255 & \text{if } I'_{ij} > 255 \\ I'_{ij} & \text{otherwise} \end{cases}$$

Note that the implementation of this stretching requires no extra RAM memory, and that it can be performed by a single pass through the pixel values, (assuming the histogram is known). Figure 3 shows an example of how an original image is processed with first local mean adjustment, and finally global histogram stretching. The resulting image is clearly enhanced compared to its original.



## FINGERPRINTS



Figure 3: Original image, processed with local mean adjustment, and a global histogram adjustment.

The following section includes a high level description for performing the global histogram normalization process:

1. Determine the minimum and maximum pixel values ( $I_{min}$  and  $I_{max}$ ) of the image.
2. Step through each pixel position in the image.
3. Calculate a new value of the given pixel position as  $I' = 255 * (I - I_{min}) / (I_{max} - I_{min})$ . Make sure the value is no larger than 255 and no smaller than 0.
4. Repeat step 2-3 until all pixels have been processed.
5. Now, all pixels have been processed, and the histogram stretched image is complete.

### Possible Extension

To further improve the preprocessing of captured images, a *local* histogram stretch method could be employed. This means that different  $I_{min}$  and  $I_{max}$  values are used in different parts of the image. For example, there is no need to perform stretching in the parts of the image where there is no finger present, and in areas which contains large amount of moisture, it might be necessary to apply a less strong rescaling than for dry fingerprints, for the image to look better. This process requires a segmentation of the image into different regions prior to the rescaling, and the implementation is therefore more complex. The segmentation method can be implemented in a variety of ways, given the amount of processing power available.

### Summary

The methods for image normalization suggested in this document are intended to make the visual appearance of a captured image better. Note that it is difficult to judge how good the quality of an image is without an objective measurement method. A suitable measurement method for fingerprint images is to process the images through a fingerprint verification system. If the error rates for a large set of fingerprint images improves for a given image processing method, this constitutes a measurable image quality improvement.

A proper image preprocessing can be crucial for verification performance, and it is very often highly dependent on the algorithm being used. A good preprocessing method should make any captured image follow a pre-defined “normalized” form, for which the verification algorithm is optimized.