Chapter

1 Image Thresholding

> Truly to enjoy warmth, some small part of you must be cold, for there is no quality in this world that is not what it is merely by contrast. Nothing exists in itself.
>
> — Herman Melville

1.1 Introduction

Classification of image pixels into groups sharing some common characteristics is often the very first step of automatic image interpretation. Usually we wish to segment an image into blobs representing the individual objects it contains, so that they can be subject to measurements or any other mean of inspection.

Usually trivial for the human mind, unsupervised **Image Segmentation** is far from straightforward in general case. The available methods vary in complexity and principles, taking into account various image parameters such as color, brightness, gradient, texture or motion.

In the industrial setting it is often the case that the image content can be clearly divided into background (e.g. the surface of conveyor line or inspection station) and foreground (e.g. the objects being inspected). Such simple, binary pixel classification is called **Image Thresholding**.

1.2 Global Thresholding

The simplest thresholding operator simply selects the pixels of intensity within a predefined range. If we interpret the results as a binary image with black pixels denoting the background and white pixels denoting the foreground, the operation applied to an image I computes the result as follows:

$$B[i,j] = \begin{cases} 1 & \text{if } minValue \leq I[i,j] \leq maxValue \\ 0 & \text{otherwise} \end{cases}$$

Figure 1.1 demonstrates example results of thresholding the same image with different range of foreground intensities.

Global thresholding is *global* in that it evaluates each pixel of the image using the same foreground intensity range. As such, it requires not only that the background is consistently darker (or brighter) than foreground, but also that the lightning is reasonably uniform throughout the entire image.

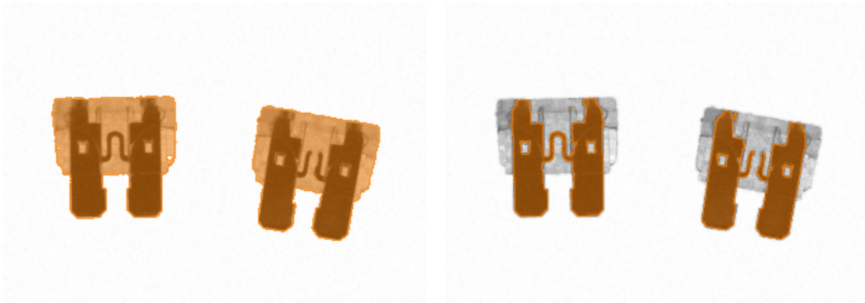

Figure 1.1: Results of global thresholding with different threshold values - pixels identified as foreground marked in orange.

The importance of uniform (in space) and constant (in time, when a series of images is analyzed) lightning in successful application of inspection system is paramount. Whenever bad lightning conditions prevent application of a technique, we should try to amend the lightning first, and only if this is not possible we should move to adjusting the algorithm.

That being said, numerous methods were developed to allow successful thresholding despite the lightning imperfections.

1.3 Threshold Selection

If the lightning is reasonably uniform throughout the image, but changes over time (which is usually the case whenever the system is not fully isolated from the sunlight) the threshold values should be adjusted accordingly. As the system should be essentially unsupervised in operation, we need to employ a technique that will allow us to determine the feasible threshold automatically, given only the image to be thresholded.

Applying such technique would also eliminate the bias introduced by manual adjustment of the threshold parameters - usually there is a range of feasible threshold values and the extracted objects appear smaller or bigger depending on the selected value.

Automatic threshold selection has been subject to extensive research and a rich set of different methods has been developed. A survey[1] by Sezgin and Sankur mentions 31 different methods of automatic selection of global thresholding values. We will demonstrate some of the most popular techniques.

The distribution of pixel intensities is an important source of information about the applicability of global thresholding and the possible threshold values. Because of that we will present histogram of pixel intensities along each example of this section.

Mean Brightness

As long as both background and foreground are consistent in brightness and occupy similar proportion of the image space, we may expect that the average image intensity will lie somewhere between the intensities of objects and background and as such would be a feasible threshold value.

In **Table 1.1** we can see an image for which this method performs correctly. Well separated background and foreground intensities appear as two significant modes in the image histogram. The modes are similar in size, which is a consequence of roughly even distribution of background and foreground in the image space.

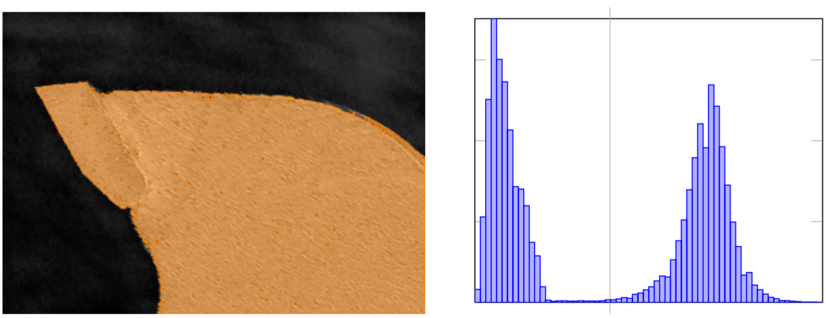

Table 1.1: Example image successfully thresholded using mean brightness as the threshold value.

Unsurprisingly, the average pixel brightness (denoted with vertical line in the histogram) fits between the two modes and allow for accurate thresholding.

Unfortunately the accuracy of this method quickly drops as the disproportion between background and foreground increases. **Table 1.2** demonstrates an example for which the method fails, even though the histogram modes are still well-separated and the range of feasible threshold values is trivial to read from the histogram.

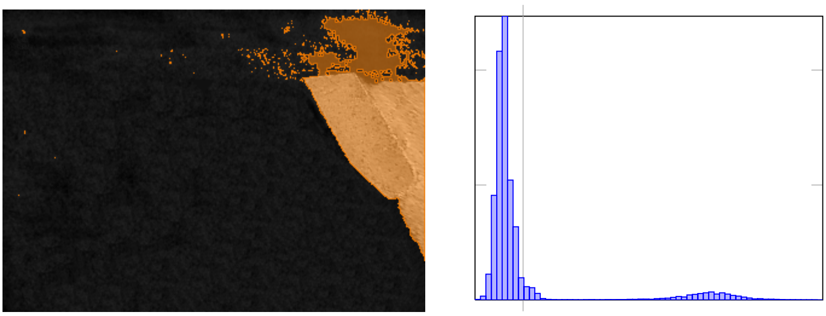

Table 1.2: Example image for which mean brightness is not a feasible threshold value.

Histogram Shape Analysis

In the previous section we have seen two examples of images having bimodal histograms with a clear valley between two modes corresponding to the range of feasible threshold values. Some of the popular threshold selection methods look for this valley algorithmically - either directly or indirectly, analyzing the shape properties of image histogram.

Intermodes

In one of the first papers[2] written on the threshold selection problem Prewitt and Mendelsohn proposed to smooth the image histogram iteratively until only two local maxima are preserved, and then select the threshold value as a mean of this two remaining maxima. **Table 1.3** presents a succesfull application of this method to the mean brightness method failed.

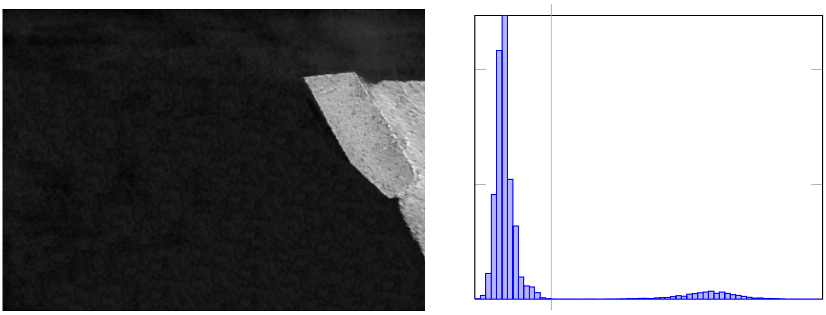

Table 1.3: Successful application of Prewitt & Mendelsohn intermodes method.

Concavities

Another method, proposed[3] by Rosenfeld and Torre computes convex hull $Hull(H)$ of the histogram H and considers local maxima of $Hull(H) - H$ (i.e. points of maximal concavity) as candidates for the threshold value; selecting the one that maximizes the product of cardinality of foreground and background pixels.

Clustering

Instead of analyzing the brightness histogram, we may formulate the threshold selection problem in terms of clustering - indeed, our aim is to divide the full intensity spectrum into two clusters, foreground and background intensities, separated by the threshold value. Second family of techniques follow this interpretation of the problem.

K-Means

Well known general-purpose K-Means algorithm iteratively computes the means of current set of K clusters, and reassigns the elements being clustered, each one to the cluster represented by the nearest of means computed in this iteration.

This idea was applied[4] to threshold selection problem by Ridler and Calvard. The algorithm maintains two clusters containing complementary parts of the intensity range. At each step mean brightness of the pixels in each cluster is computed and pixels are reassigned to the cluster of nearest mean.

It is worth noting that even though the authors proposed an iterative scheme of computation, it is perfectly feasible to perform a brute force search over all possible threshold values and select the one that fits halfway between means of the induced clusters; especially in the common case of uint8 industrial images of only 255 possible intensity levels. If we precompute the image histogram and maintain the running averages of the clusters, we may process each threshold value in constant time.

Otsu Method

Another approach was proposed[5] by Otsu. He suggested to select the threshold value that maximizes the between class scatter between background and foreground; which is:

$$|F| \cdot |B| \cdot (\overline{F} - \overline{B})^2$$

This method remains one of the most popular threshold selection method and is often referenced in the publications.

1.4 Dynamic Thresholding

When the lightning in the scene is uneven to the point where image foreground intensity in dark parts of the image is at the same level as the background intensity in bright sections - or, in other words, intensity ranges of background and foreground are overlapping - it is clear that global thresholding cannot be applied.

An example of such problem is illustrated in **Figure 1.2**. As we can see, bad lightning setup makes left bars of the barcode appear brighter than the background in the right part of the barcode. Key point in overcoming this issue lies in an observation, that the barcode, even under bad lightning, is still *locally* darker than the background in its entirety.

This is illustrated in **Figure 1.3**, where we plot the 1D profile of the barcode extracted along the scan line marked in the image and the same profile smoothed with running average operator of with 10.

Figure 1.2: .

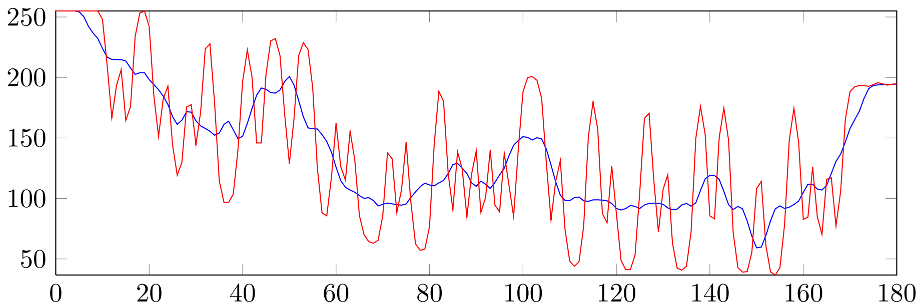

Figure 1.3: Brightness profile of the barcode image in red, smoothed profile in blue.

Therefore, if we define the threshold value in relation to mean brightness at each location, we can get accurate results despite bad lightning conditions. **Figure 1.4** demonstrates failed global thresholding attempt (with lowest threshold value that includes whole bar area in the result) and the results of dynamic thresholding of the image.

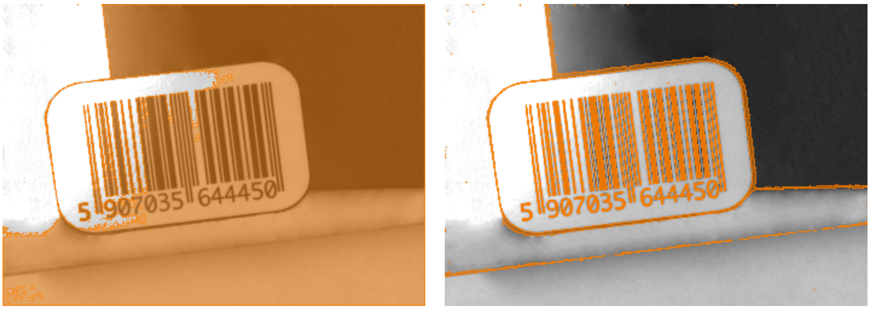

Figure 1.4: Results of global thresholding and dynamic thresholding of the barcode picture.

1.5 Hysteresis Thresholding

1.6 Color-based Thresholding

When inspection is conducted on color images it may be the case that despite a significant difference in color, the brightness of the objects is actually the same as the brightness of their neighborhood. In such case it is advisable to use Color-based Thresholding filters: ThresholdToRegion_RGB, ThresholdToRegion_HSV. The suffix denote the color space in which we define the desired pixel characteristic and not the space used in the image representation. In other words - both of these filters can be used to process standard RGB color image.

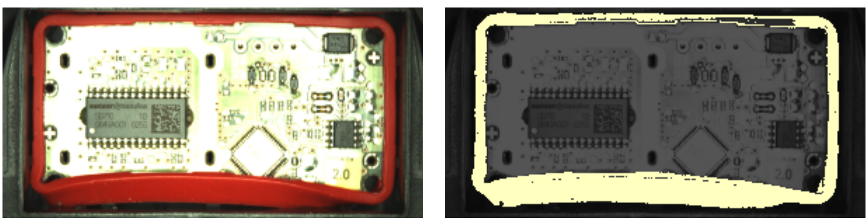

Figure 1.5: Color-Based Thresholding