Cite this paper as: Martín-Rodríguez F., Vázquez-Fernández E., Dacal-Nieto Á., Formella A., Álvarez-Valado V., González-Jorge H. (2010) Digital Instrumentation Calibration Using Computer Vision. In: Campilho A., Kamel M. (eds) Image Analysis and Recognition. ICIAR 2010. Lecture Notes in Computer Science, vol 6112. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-642-13775-4_34

Digital Instrumentation Calibration Using Computer Vision

Fernando Martín-Rodríguez[1], Esteban Vázquez-Fernández[1,2], Ángel Dacal-Nieto[2], Arno Formella[3], Víctor Álvarez-Valado[2], Higinio González-Jorge[2],

[1] Communications and Signal Theory Department, University of Vigo,
[2] Laboratorio Oficial de Metroloxía de Galicia,
[3] Computer Science Department, University of Vigo,
Address: ETSET, C/ Maxwell S/N (Ciudad Universitaria), 36310 Vigo, Spain
fmartin@tsc.uvigo.es, evazquez@lomg.net

Abstract. This paper describes a computer vision system designed to automatically read the displays of digital instrumentation. The system is used in calibration sessions where many measurements have to be made and where we are interested in getting the whole numerical series downloaded on a host computer. Before our system was running, a human operator had to inspect the instruments at the right times (required by the calibration procedure) and to write down all the results. Note that we are speaking of very simple and sometimes old instruments that usually do not provide a digital interface or a removable memory (and if they do, we do not have a standard interface accepted by all the manufacturers). Results show the benefits of this system, obtaining a success rate higher than 99% in display recognition

Keywords: computer vision, text segmentation, character recognition, digital instrumentation.

1 Introduction

1.1 Purposes of Development

The computer vision system was designed to automatically read digital instrumentation measurements avoiding a time-consuming work and minimizing errors due to tiredness or forgetfulness. The application was first implemented at the "Laboratorio Oficial de Metroloxía de Galicia" (LOMG: www.lomg.es) for digital thermometer calibration (Fig. 1). Nevertheless, our application will be useful with all types of instruments that exhibit a numerical display.

The process starts with a photograph of the instrument displaying a stable measurement. Then we use standard image processing techniques to segment the image characters. Finally, we will see how we recognize the digits with a new approach that combines two different classifiers.

Fig. 1. Examples of different instruments, some of them are showing display defects (bubbles in the first one, stripes in the third one).

1.2 Related Work

Our system is an example of character recognition in scene images like that in [1]. We have taken some ideas from there such as the interpolated threshold in non uniformly illuminated images.

Our system also shares ideas with car plate recognition systems [2], [3]. In such plate recognition systems, plate has to be located automatically but a fixed character font is assumed. In contrast, for digital instrumentation, we have to take into account multiple fonts (Fig. 2) which is, as we will see, a source of problems. As the operator must set up the calibrating experiment, we started with a system where the region of interest (ROI) is user supplied.

Fig. 2. Examples of different displays using different character fonts. Our system deals with all of them.

As we will describe in the following, image preprocessing and segmentation are based on standard image processing techniques adequately adjusted to our problem. The recognition stage combines two methods: first a classical one based on feature extraction followed by 1-NN classification; second an original method especially suited to recognize instrumentation digits (as we will see is somewhat inspired by the classical 7-segment display). The fusion of both recognizers is also an original contribution where we use the second one to correct possible errors of the first one.

2 Image Capturing and Preprocessing

2.1 Image Capturing

Capturing is perhaps the most important part of the whole system. A good capture will make recognition easy while a bad one would make it impossible.

Due to some in-site restrictions at LOMG, we cannot modify the environment illumination. We use a C-Cam BCi4 camera with 1280x1024 resolution (Fig. 3). In most of the cases, we use a 25 mm lens (able to focus from 15 cm to 1 m). However, sometimes the physical conditions oblige to the use of a 75 mm lens with a focal distance from 1.5 m to 10 m.

To help in capturing, we designed a mechanical arrangement that can be used with most instruments to get always the same capturing conditions (Fig. 3).

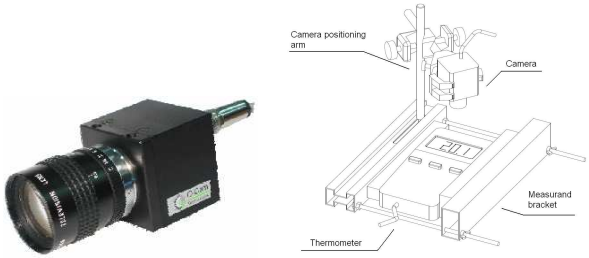

Fig. 3. System camera and mechanical arrangement for stable capture.

As the observed instrument is not moving during image capturing (remember we are calibrating the instrument, not using it dairy), we decided to rely on user to extract the ROI (Fig. 4). The region will have to be marked only once for all the series. Afterwards, we developed an automatic extraction system using the methods described in [3]. This new subsystem gets correct location in more than 99% of the test images [4] (Fig. 5) but it is not integrated with the rest of the system yet.

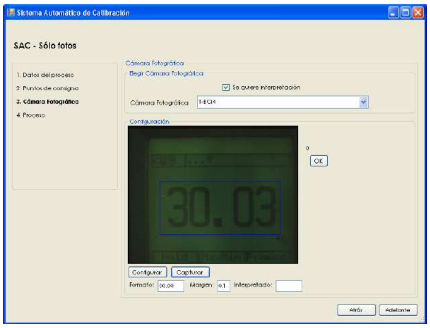

Fig. 4. User selection of Region of Interest (ROI).

Fig. 5. Automatic location [4]. Still not used at LOMG.

2.2 Binarization

Binarization is the process that converts a grayscale or color image into a binary one with only two levels. We start by converting our colored images to grayscale using the ITU-R BT.709 recommendation (gray=0.2125R+0.7154G+0.0721B). We use this equation because instrument displays are often green or, at least, dominated by the green channel. As expected, the resulting gray level distribution shows a bimodal histogram (two main peaks, see Fig. 6).

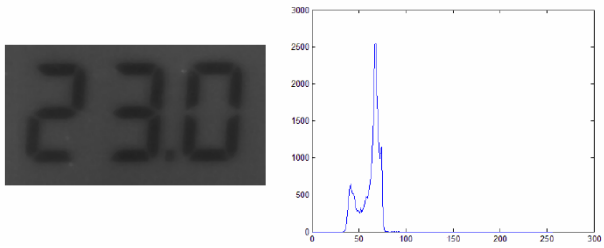

Fig. 6. Grayscale display image and its gray histogram.

We use a combination of the well known Otsu method [5], implementing it via an approximate iterative version found in [6] and the peak detection method [7] (based on searching histogram peaks and locating thresholds on the minima between them).

As can be seen in Fig. 7, Otsu method can create some segmentation problems due to the thicker characters it produces. We also experienced some problems with images with important illumination gradients (Fig. 8). In these cases, a global threshold is not enough. This can be solved splitting the image into sub-images and applying interpolated thresholds [1] (see results in Fig. 9).

Fig. 7. Left: Peak detection method. Right: iterative Otsu method.

Fig. 8. Left: Image with illumination gradient (not very evident, more visible in an equalized image: middle). Right: binarization with a single threshold.

Fig. 9. Images binarized with interpolated thresholds. Left: 8x8 sub-images. Right: 4x3 sub-images.

The final solution consists on applying first the peak detection method and then measuring threshold quality using the histogram area in the threshold neighborhood. If that area is bigger than usual the threshold is considered incorrect and we switch to an interpolated threshold with 12 sub-images (4x3 sub-image grid). We use Otsu threshold on each piece.

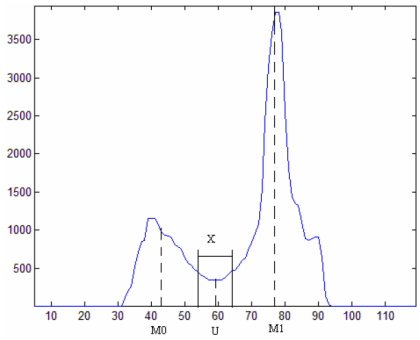

Fig. 10. Threshold quality test based on area in the threshold neighborhood. If this area exceeds 2.5% of total histogram area, threshold is not good.

2.3 Skew Angle Correction

To correct a possible skew angle, we estimate the upper contour of the characters and compute the slope of the resulting straight line (Fig. 11).

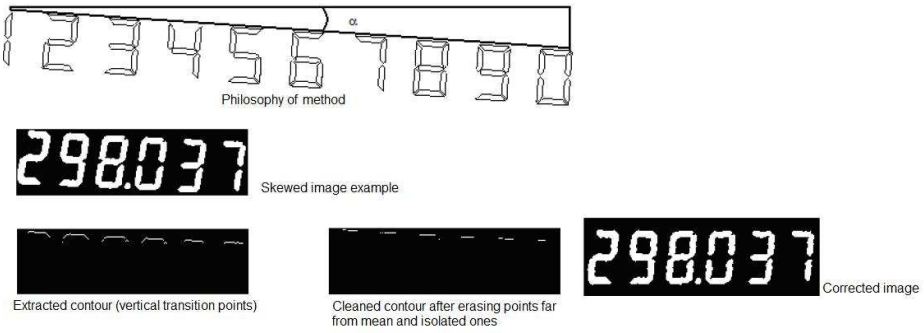

Fig. 11. Skew angle correction.

2.4 Extracting the Character Row (Presegmentation)

Extracting the character row is the same as removing the blank lines above and below characters and also to the left and to the right of them. As shown in Fig. 12, this is an easy process using horizontal and vertical image projections.

On each projection, we detect the region of interest beginning and ending by searching for large gradients, first from left to right (top to bottom) and afterwards in the opposite direction.

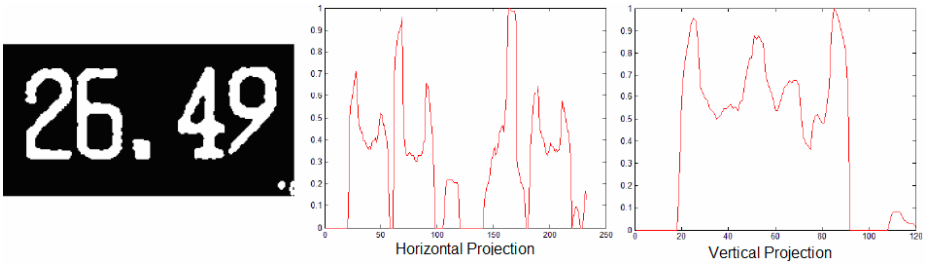

Fig. 12. Binary display image and both projections.

3 Character Segmentation

To isolate the different characters in a preprocessed row, we use what we call "enhanced projections". The enhanced horizontal projection of an image is the vector that contains in position i the dot product ($< x, y >= \sum x_j y_j$) between the $(i-1)^{th}$ and $(i+1)^{th}$ column. With this kind of projection, minima that mark character transitions are deeper than with standard projections.

The main procedure consists of searching the horizontal projection from right to left while applying a kind of hysteresis process. The right to left direction is chosen because digit ending is usually more evident than its beginning (most digits end by a

vertical line, i.e., a strong projection gradient). We use the word hysteresis because the threshold to detect a character beginning is different (bigger) to the threshold used to detect an ending (Fig. 13).

To detect segmentation errors (linked characters), we compute the aspect ratio of all segmented items (R=height/width). For a ratio R of less than 1.2 we decide that we rather have two characters. In this case, we first compute the local maxima and minima of the projection (using a sliding window procedure as described in [8]). The deeper minimum that is between two peaks is selected as the optimum breaking point.

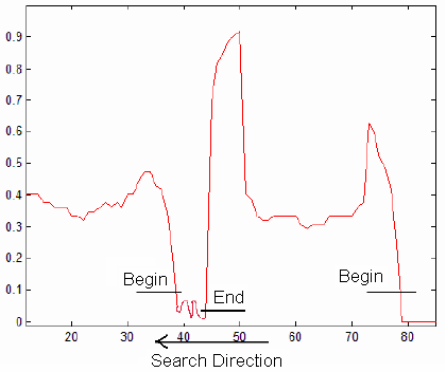

Fig. 13. Detection with hysteresis.

4 Digit Recognition

First, we normalize the extracted characters by scaling them to a fixed size of 16x16 points. Note that we do not maintain the aspect ratio. We started by keeping that ratio (getting a character with 16 points height and less than 16 points width) followed by centering with vertical lines [8]. Nevertheless, we discovered that, in the end, we yielded slightly worse results. Distorting input characters is not a problem as long as we also distort the patterns as well.

4.1 Feature Extraction

We extract features in two ways. First, we use horizontal and vertical projections of the individual characters. As different characters may have almost the same projection (Fig. 14), we split each character into two halves (upper and lower) and then we compute 4 vectors: upper horizontal (16 values), upper vertical (8 values), lower horizontal (16 values), and lower vertical (8 values). Final feature vector has 48 values.

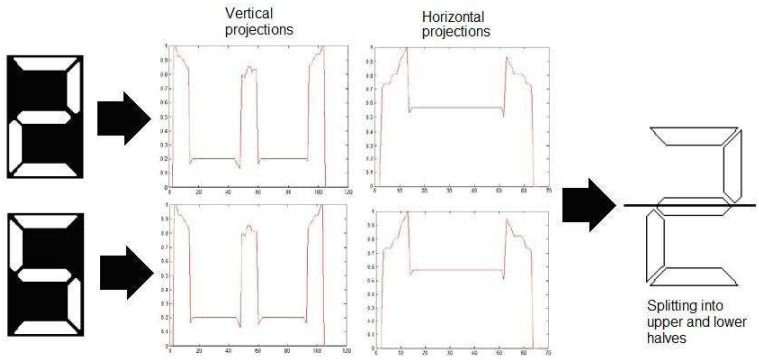

Fig. 14. Characters with almost identical projections.

Second, we use features based on Kirsch gradients [6], [8]. The Kirsch operator computes a first order derivative (similar to operators from Prewitt, Sobel, Canny… [9]). Our purpose is to compute image components along four directional axes: horizontal, vertical, right diagonal and left diagonal. For example, the horizontal component is computed via a vertical gradient (being always perpendicular to the desired direction).

Eventually, we obtain four local feature maps as 16x16 images. Using all of them as a feature vector would result in a 1024 length vector. In [8] it is suggested to decimate the 16x16 images to size 4x4. However, in our particular system we obtained better results leaving the 4 directional components in its original size (Fig. 15).

We combine the two feature vectors, i.e., projections and Kirsch, yielding a total vector of length 1024+48=1072.

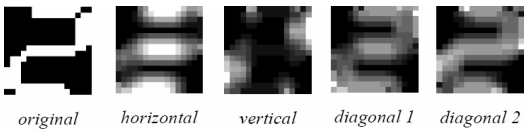

Fig. 15. Example of directional components.

4.2 Classification

We tried various classifiers (like probabilistic neural networks, Gaussian classifiers and k-NN). Best results were for the nearest neighbor algorithm (1-NN). This is not very surprising as it is explained in [10]. This system has to deal with several different character types (7-segment, graphical fonts, skewed, not skewed, etc.). In this multi-font situation, there exists sometimes more variance between the samples of the same character in different fonts than between different characters in the same font (intra-class variance greater than inter-class variance).

As patterns for the 1-NN classifier we chose perfect ones (obtained from the different fonts). We tried to use patterns from segmented input digits but the 1-NN got better results for the artificial, perfect ones.

4.3 Visual Inspection and Fusion with 1-NN Classification

Visual inspection is an intuitive method. We developed it studying the reasoning that people express when they describe how they recognize characters. For example, no matter which font, number **'2'** has always two openings: one in the upper left part and the other in the lower right one.

We implemented a complete recognizer [11] based on a template that defines the regions of interest (Fig. 16). We check whether each region is active or not by majority voting between foreground and background pixels. See that the template is a 7-segment digit with no corners (sharp in some fonts and rounded in others).

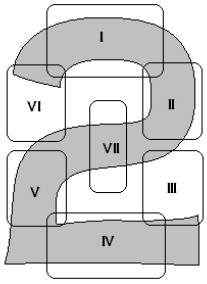

Fig. 16. . Regions of interest tested (a 7-segment like template).

To benefit from both classification schemes, we run them in parallel and combine their results in the following manner:

- We run the feature extraction and compute the norm-1 distance to every pattern yielding a distance vector.
- We run the visual inspection algorithm yielding an estimate for the digit to be recognized. This is coded as a binary vector of length 10 where an active bit at a position corresponds to recognition of that class.
- We reduce the distances that correspond to the class that was recognized by visual inspection by 20%, empirical.
- We apply 1-NN and minimum distance wins.

5 Results

Our test set consisted of 16 image sequences, with a total of 448 images. The system obtained the correct values 445 times, id. est: 99.33% recognition rate (measured on display images, not on individual digits). We have tried samples from all accessible fonts: 7-segment, skewed, not skewed, graphic display… In routine work of the LOMG, 7-segment displays (easier to recognize) are the most common ones.

Average execution time is 25 milliseconds per image (Intel Core Duo, 2.53 GHz). Our results are similar to other found in literature like [12], although in this publication they only consider 7-segment displays.

6 Conclusions and Future Lines

We have designed and implemented a useful system able to read almost any display of digital instrumentation devices.

We have employed standard image processing techniques adapted to this problem. We also designed a hybrid recognizer, which combines a classical classifier with a visual inspection algorithm. Final recognition rate suggests that we have solved the problem despite the intra-class variance due to the presence of multiple fonts.

As future work lines, we emphasize on the following: integrating the automatic ROI location into the industrial system, optimizing the feature vector trying to detect the principal components and, finally, using knowledge from previous images (for instance the font type) when recognizing subsequent images in a sequence.

References

1. Ohya, J., Shio, A., Akamatsu, S.:.Recognizing Characters in Scene Images. IEEE Transactions of Pattern Analysis and Machine Intelligence, Vol. 16, Nº 2, 214-220 (1994).
2. Cowell, J. R.: Syntactic Pattern Recognizer for Vehicle Identification Numbers. Image & Vision Computing (1995).
3. Fernández-Hermida, X. et al.: Automatic and Real Time Recognition of V.L.P.'s (Vehicle License Plates). LNCS. Nº 1311, Vol 2, 552-559. Springer, Heidelberg (1997).
4. Martín-Rodríguez, F. et al.: Localización de Caracteres en Imágenes de Instrumentación Digital. Proceedings of URSI-2009 (National Meeting of the International Scientific Radio Union). Santander, Spain (2009).
5. Otsu, N.: A Threshold Selection Method for Gray Level Histograms. IEEE Transactions on System, Man and Cybernetics (1979).
6. González, R.C., Woods, R.E.: Digital Image Processing (3rd edition). Prentice Hall, Upper Saddle River, U.S.A. (2008).
7. Martín-Rodríguez, F.: Analysis Tools for Gray Level Histograms. Proceedings of SPPRA-2003 (Signal Processing Pattern Recognition and Applications, www.iasted.org). Rhodes, Greece (2002).
8. Proceedings of the IEEE (Special Isue on O.C.R.'s), Vol 80, Nº 7 (1992).
9. Jain, A.K.: Fundamentals of Digital Image Processing. Prentice Hall. Englewood Cliffs, U.S.A. (1989).
10. Blue, J. L. et al.: Evaluation of Pattern Classifiers for Fingerprint and OCR Applications. Pattern Recognition (Pergamon Press), Vol 27, Nº 4, 485-501 (1994).
11. Vázquez-Fernández, E. et al: Human Visual Perception as a Complementary Method for Digit Recognition. Proceedings of VIIP-2009 (Visualization, Imaging and Image Processing, www.iasted.org). Palma de Mallorca, Spain (2009).
12. Corrêa Alegria, F, Cruz Serra, A.: Automatic Calibration of Analog and Digital Measuring Instruments Using Computer Vision. IEEE Transactions on Intrumentation and Measurement, Vol. 49, Nº. 1, 94-99 (2000).