Automatic Color Reference Target Detection

Luis E. García Capel, Jon Y. Hardeberg
The Norwegian Colour and Visual Computing Laboratory, Gjøvik University College, Gjøvik, Norway

Abstract

The use of standard color reference targets at image acquisition allows to compensate for different camera characteristics, illumination conditions and exposure times, ensuring true colors in digital photo workflows. Reliable automatic detection of reference targets makes color correction faster, and this becomes critical in mass digitization processes. The existing automatic algorithms usually assume that there is little perspective distortion and/or that the scanning resolution is known, achieving very limited results for example when the relative size of the color target is unknown. In this paper we present a preprocessing step that aims at automatically detecting a region of interest (ROI) where the reference target is located. We compare the performance of one of the available automatic tools (CCFind) with and without this preprocessing step, and show a considerable improvement in the detection of color reference targets in a new challenging dataset. In addition, a simple template matching approach is compared with the performance of CCFind. The results show that the selection of a smaller ROI complements well with the existing approaches and helps to improve detection.

Introduction

The aim of color management is to get colors to match from input to output. When a camera captures true colors, we would like our monitor or printer to display these colors accurately. This becomes a challenge with the huge amount of devices existing nowadays, since each input device interprets colors differently, and the way they are interpreted also varies with the manufacturer. In fact, even two camera models from the same manufacturer can capture colors differently. Light also comes in many different ways and colors. Depending on the time of the day, natural light can go from bluish to yellowish, and this variation is even more important for artificial indoor lighting. Our human visual system compensates for all of these different color light sources without us really noticing. However, since cameras see light literally, these subtle shifts can pose a problem.

In order to reduce the differences between the images acquired by different devices and under different illumination conditions, standard color reference targets are used. Capturing images including these standard references thus make color corrections faster, since we can for instance apply the same white balance to a group of images acquired under the same lighting conditions. Ensuring true colors is critical in any photo workflow because it provides a reference point for posterior editing.

In areas such as mass digitization environments, tools for automated quality analysis are a prerequisite for the management of technical content quality [1]. Existing systems are mostly based on reference based measurement methods, and some of the automated quality assessment (QA) algorithms proposed in the literature rely on the presence of reference targets within every digitized image file. Since the standard reference and source media could be located at any position within the digitized image, extraction of reference targets is needed prior to analysis and color correction.

This paper proposes an algorithm that aims at automatically finding a region of interest (ROI) where a standard color reference target is located, facilitating the work of currently existing automatic techniques that detect reference targets. The performance of these techniques, with and without the ROI selection step, has been evaluated.

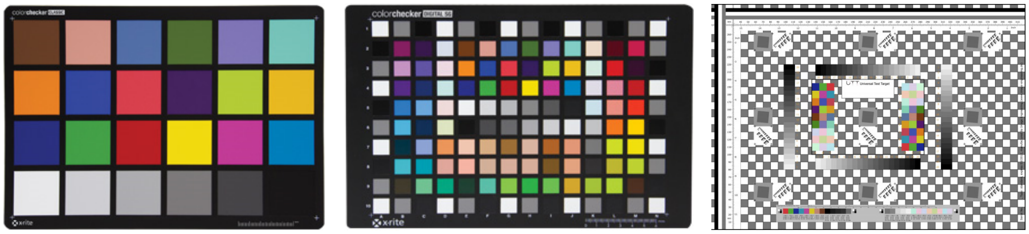
The remainder of this paper is organized as follows: *Color Reference Target Detection* presents several popular color reference targets and the existing approaches to automatically detect them within an image. *Proposed System Overview* focuses on the description of the proposed system, where the automatic ROI selection step is described. In addition, two automatic reference target finders are presented. The experimental results are described in *Experimental Results*, where we justify the selection of some parameters and we compare the performance of the proposed system with the current approaches (without any preprocessing step). Finally, *Conclusions and Future Work* are discussed.

Color Reference Target Detection

The employment of reference targets for professional digitization or photography is common nowadays. Color reference targets are therefore of central importance in any mass digitization process. Depending on their type, reference targets allow a precise quantification of many different parameters influencing the digitization. Knowing the real size and the colorimetry of the reference target, they allow to determine the scale, rotation and any distortions present in the digitized asset, as well as the color and illumination deviation and uniformity.

In the following, some of the most popular color reference targets in use today are presented:

- The ColorChecker Classic target [2] (Figure 1(a)), initially developed by McCamy et al. [3] in 1976 and commercialized as the ”Macbeth ColorChecker”. It contains 24 natural, chromatic, primary and gray scale uniformly-sized squares in a wide range of colors printed on a 8,25” × 11” cardboard. It has become the industry standard, since it is still the most common tool employed for color comparison.
- The ColorChecker Digital SG [2] (Figure 1(b)) is an enhancement of the ColorChecker Classic for digital photography. It contains an extended color palette in the form of 140 quadratic patches chosen for its location in color space to expand the color gamut, and it is printed on a 8,25” × 11” cardboard as well.
- The Universal Test Target (UTT) [4] (Figure 1(c)) is a recent open standard test target designed to provide insight into the



(a) (b) (c)

Figure 1. Examples of available color reference targets: a) ColorChecker Classic; b) ColorChecker Digital SG; c) Universal Test Target.

> complete image quality of all types of high-end cameras and scanners following the current ISO-standards. The target is available with various sizes, which makes it suitable for all kinds of digitization projects, preservation and access, carried out by libraries, archives and museums.

Very little research has been done in the area of automatic color reference target detection. Currently there exists only a few proposals addressing this problem, and they are usually based on several assumptions that limit the applicability to a set of similar images. A step in this direction was done by Tajbakhsh and Grigat [5], who introduced a semi-automatic method for reference target detection and color extraction. Their method focuses on images exhibiting a significant degree of distortion. Liu et al. [1] proposed an automatic method based on the fact that there are low perspective distortions, the scanning resolution is known and the lighting is approximately constant in the image. A more general approach was considered in the method proposed by Bianco and Cusano [6], who used color descriptors to automatically locate the color reference target, and a geometrical and appearance validation step helped to select the most feasible pose. These principles make the method robust to varying illumination conditions.

Freely available tools such as CCFind [7] and MacDuff [8] also exist. They aim at detecting the ColorChecker Classic inside an image. CCFind is implemented in Matlab (Mathworks Inc.) and returns the coordinates of the center points of the color patches. By not using color as a cue, it can be used with unconventional lighting and multispectral sensors. Its main limitation is however that smaller ColorCheckers are more difficult to find. On the other side, MacDuff is implemented in C++ and uses some code from OpenCV in its implementation. By performing geometrical operations (rotations, scaling, etc.) and by computing a distance metric between colors, the ColorChecker is finally found.

Commercial software is also capable of doing a semi-automatic color reference target detection. Examples of this are the X-Rite ColorChecker Passport Camera Calibration Software [9], which tries to perform an initial automatic detection, Imatest [10] and BabelColor PatchTool [11]. The software however usually relies on human intervention to manually mark or correct the detected reference target (by dragging the cursor), after which color correction is performed. Since manual intervention is not practical in mass digitization processes for obvious reasons of cost and speed, it is thus interesting to develop fully automatic tools for the detection of one or several color reference targets in digital images.

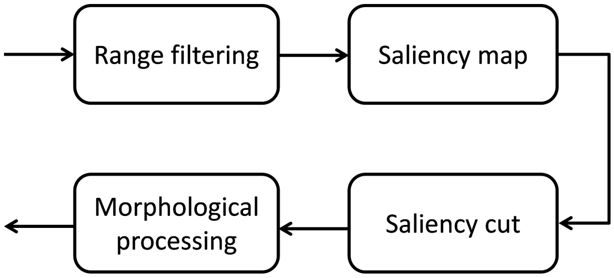


Figure 2. The main steps of the ROI selection algorithm.

Proposed System Overview

ROI Selection

As mentioned in the previous sections, this paper proposes a preprocessing step that allows existing tools (e.g. CCFind) to more easily locate a color reference target within an image. The goal of this step is thus to find the region of interest (ROI) in which a reference target is located. The algorithm has been implemented in Matlab (Mathworks Inc.) and consists of four main steps (see Figure 2) presented in detail below.

In Step 1, an input color image is resized for more efficient processing. If the largest dimension of the image is larger than 500 pixels, the image is resized to meet that value. Then, the local range of the image is computed by means of the Matlab function *rangefilt* with the default parameters. This performs a texture transformation that uses morphological processing (dilation and erosion) to determine the maximum and minimum values in a given neighborhood. The result is an image whose pixels contain the range value (maximum value − minimum value) of the 3-by-3 neighborhood around the corresponding pixel in the input image. The reason behind this transformation is that since there is a certain contrast between the color patches and the black frame, the edges of the color patches are enhanced, increasing the color contrast in the region where the reference target is located.

In Step 2, the local range is used to compute a saliency map using region and color information. The approach used is the Region-based Contrast (RC) method proposed by Cheng et al. [12], whose implementation is publicly available from the author's website. RC-maps incorporate spatial relations by first segmenting the input image into regions, and then computing color contrast at the region level. The saliency for each region is computed as the weighted sum of the region's contrasts to all other regions in the image. This approach has been shown to provide state-of-the-art saliency detection results, and our experimentation has shown its particular suitability for identifying the region of interest around a reference target, since it outperforms the re-

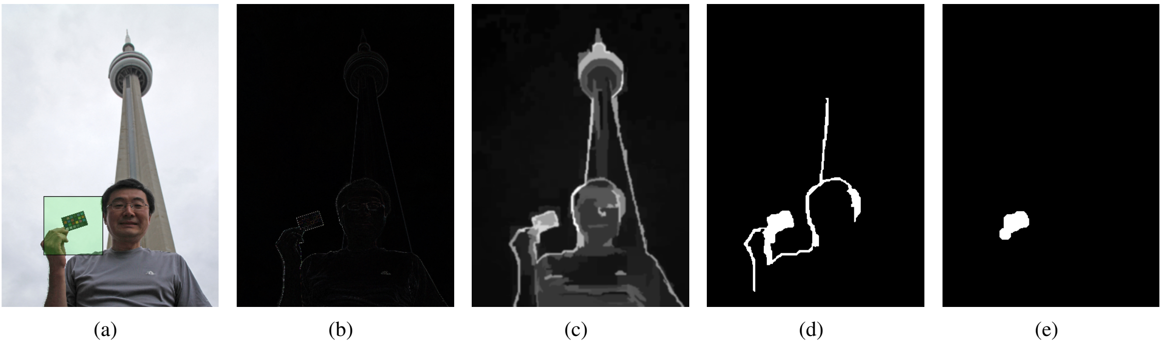


Figure 3. *The steps of the automatic ROI selection method proposed. a) Original image and ROI found (marked in green) after processing; b) Image after range filtering; c) Saliency map; d) Saliency cut; e) Result after morphological processing.*

sults obtained by the other four methods included in the author's implementation.

In Step 3, after the saliency map has been computed, saliency cut (thresholding) is done again with the implementation proposed in [12]. In this approach, GrabCut is iteratively applied to refine the segmentation result initially obtained by thresholding the saliency map. GrabCut, which by itself is an iterative process using Gaussian mixture models and graph cut, helps to refine salient object regions at each step. The importance of this step relies on the selection of the threshold parameter (see *ROI Selection Adjustment*).

In Step 4, morphological operations are applied to the saliency mask previously generated. A circular structuring element of radius 10 pixels, selected empirically, is used to perform image opening (erosion followed by dilation) in order to remove any object contours from the mask. After this, the resultant area is given by the smallest bounding box containing the mask and a certain horizontal and vertical margin. The margin value used has also been chosen empirically, and it corresponds to the 5% of the largest dimension of the original image. Since there could be cases in which the processed mask is empty (no clear salient points have been detected in the image), the way the algorithm deals with such situations is by returning the original image instead.

An example of the output images after each of these steps is shown in Figure 3.

Color Reference Target Finder

Once a smaller region (ROI) of the image has been selected to facilitate the search, the following step is to apply a technique that allows automatically finding the color reference target within it. Two approaches were used in our experiments: CCFind [7] and a template matching method that is able to find reference targets of different sizes and orientations with low perspective distortion.

CCFind

CCFind [7] aims at finding the coordinates of the central points of a ColorChecker Classic (24 squares), and this is done by searching recurring shapes inside the image. This makes it robust to illumination changes and perspective distortion. However, its main limitation is the difficulty of finding ColorCheckers that are relatively small within the image, or those that are close to the image borders.

Template Matching

A simple template matching approach inspired by [13] was implemented. It employs the normalized cross-correlation coefficient of the grayscale images, since this technique is more robust to lighting (dependent on weather conditions and time of day) and exposure variations. By changing the size of the template and its orientation with respect to the image, the combination that gives the best cross-correlation coefficient is taken as the match. In order to decrease the computational complexity, a selection of regularly spaced angles and scales is used. In addition, the early jump-out strategy described in [14] is employed, in combination with the parallel computing options included in Matlab. The main limitation of this algorithm is the computational time required. Further details about the implementation can be found in [15].

Experimental Results

The results were evaluated on a computer consisting of an Intel Core i7-2630QM CPU with 6 GB of RAM. The dataset used for the experiments contains 43 images ranging from 2 to 21 Mpx. These images were provided by Dr. Francisco Imai, and have made available as the "Colourlab Image Database: Imai's ColorCheckers (CID:ICC)" at `http://www.colourlab.no/cid`. All these images include a reduced version of the ColorChecker Classic (known as the ColorChecker Mini [2]). This dataset is challenging because the size of the ColorChecker is relatively small in most of the images. In addition, some of the ColorCheckers suffer from perspective distortion, and the images have been acquired under very distinct illumination conditions. An illustration of some of the images included in the dataset is shown in Figure 4.

ROI Selection Adjustment

The impact of changing the threshold of the saliency cut step was studied. Four parameters were measured: 1) the average size reduction between the selected ROI and the original image (the mean of the relative size of both dimensions); 2) the number of detection failures, given by the number of times the algorithm outputs an empty ROI; 3) the accuracy, given by the ratio between the number of ROI that actually contain a reference target and the total number of images in the dataset; and 4) the number of correct reference target detections using CCFind (for a maximum image size of 2,000 pixels in the largest dimension).

The results suggested that a threshold value of 100 seemed

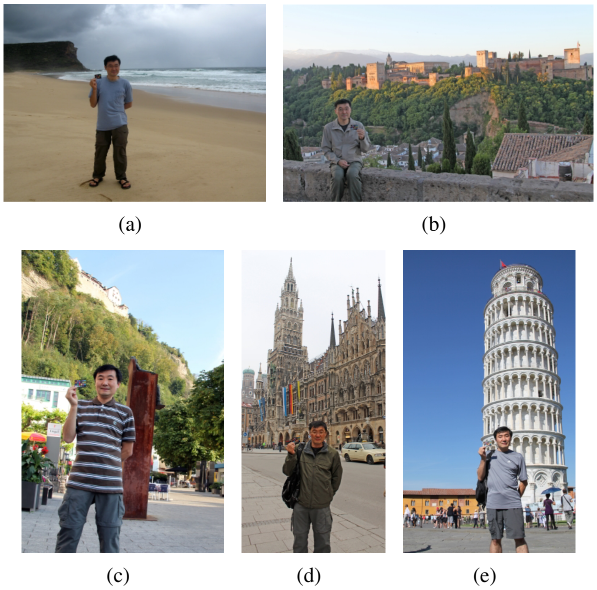


Figure 4. Some of the images included in the dataset.

to provide a good compromise between scale reduction and accuracy when the whole dataset was considered (see [15] for more details), also achieving the biggest number of correct detections. For such a threshold value, the selected ROI became empty in 4 of the images, whereas it did not include the reference target in 7 of them. The failures were mainly due to the fact that either the relative size of the reference target was too small, or that there were edges of high contrast in the image that reduced the saliency of the color patches. Some examples of correct detections are shown in Figure 5, whereas examples of detection failures (empty ROIs) and wrong detections (the reference target is not in the ROI) are shown in Figure 6. In the following sections, we consider a threshold value of 100 in all the computations.

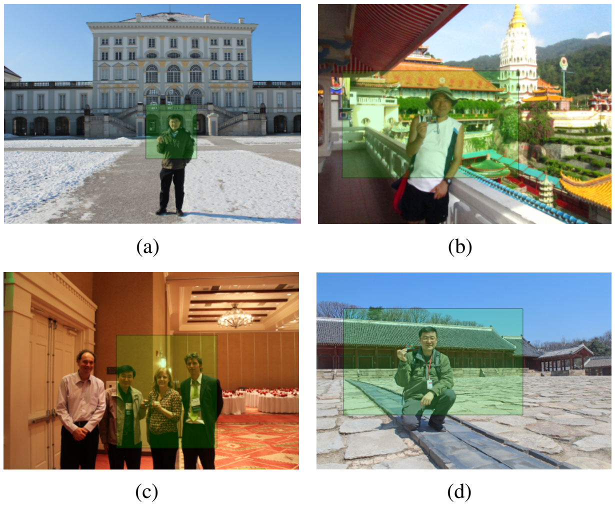


Figure 5. Examples of correct ROI selections for a saliency cut threshold value of 100.

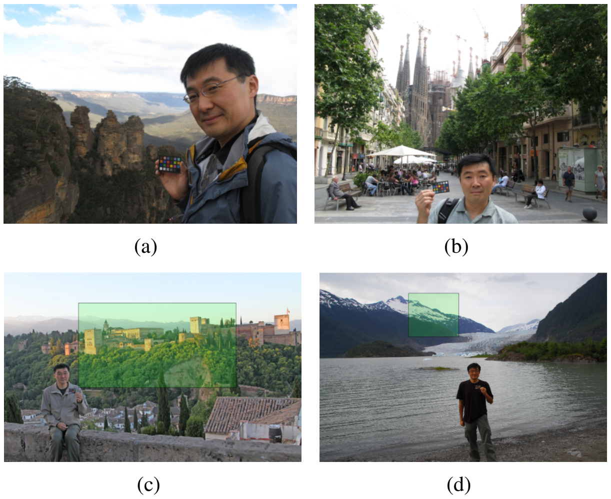


Figure 6. Examples of incorrect ROI selections. a) & b) Detection failures (empty ROIs); c) & d) Wrong detections (the target is not in the ROI).

CCFind with ROI Selection

The main problem of the existing automatic approaches for finding reference targets usually lies on adopting assumptions that make the algorithms only applicable to certain types of images. In the particular case of CCFind, its performance severely decreases when working with images in which the reference target is relatively small. Although template matching techniques can better deal with these problems by using a bigger range of template scales, this becomes very computationally costly. For this reason, in order to more efficiently use these algorithms and to achieve higher detection success rates, it is absolutely necessary to reduce the search area. In this section, we evaluate the performance of CCFind with and without the use of the automatic ROI selection algorithm proposed. As we will see, this preprocessing step can improve the detection results of this challenging dataset by about 20%, also reducing the total computation time.

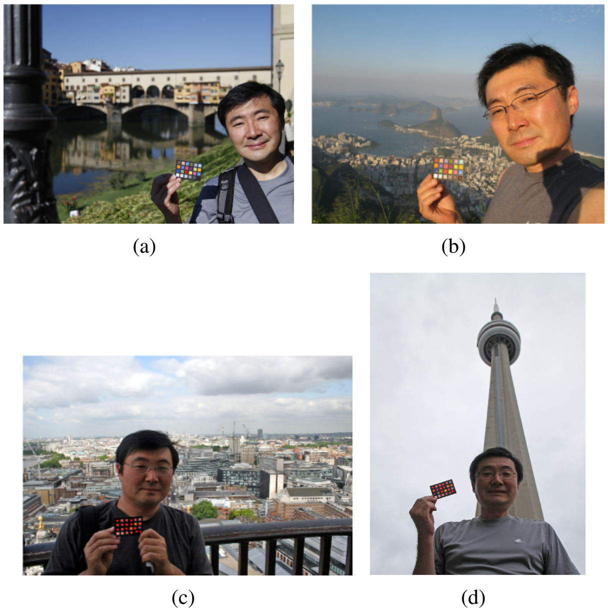
The challenge of this dataset can be clearly seen in Table 1, since without the preprocessing step the reference target can only be found in 4 of the 43 images. The results can be however improved up to 5 images by skipping the resizing step (in this experiment the largest image dimension has been limited to 2,000 pixels), although at the prize of a much higher computational cost (more than 10 hours to process all the images). The images successfully processed are shown in Figure 7.

Table 1. Detection results using CCFind.

Method	# Correct	Computation time
CCFind	4	00:33
CCFind + ROI Sel.	12	00:27

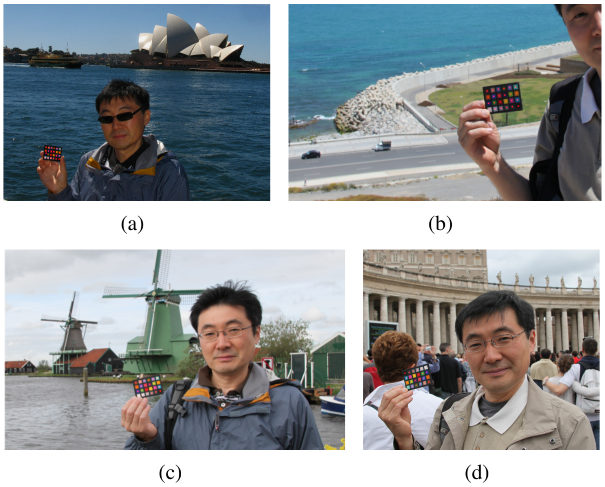
Table 1 also shows the results when the preprocessing step is applied. The results prove that the proposed method helps to outperform the standard approach (CCFind only), achieving detections even when the reference target is relatively small and also improving the total computation time. Examples of successful detections using the ROI selection step are shown in Figure 8.

Going further, the results not only indicate that detection can



(a) (b) (c) (d)

Figure 7. *The four images where the color reference target is correctly found by CCFind without the ROI selection step.*



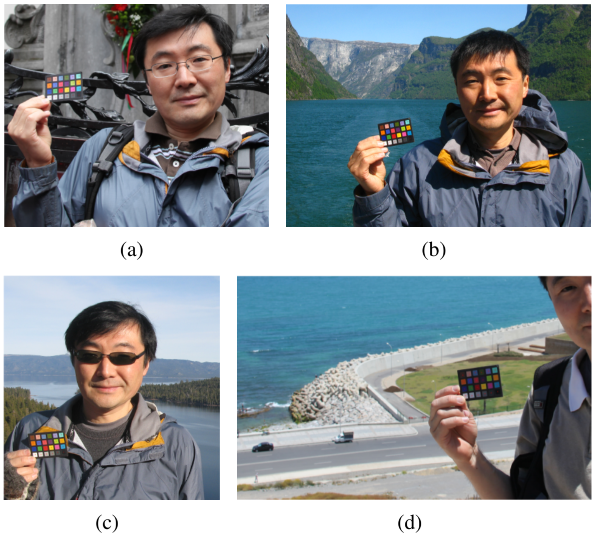
(a) (b) (c) (d)

Figure 8. *Color reference target detection examples when using the ROI selection step followed by CCFind. The ROI of the full-size images are shown for better visualization.*

be improved with the ROI selection step, but also that lower resolutions might be used to run the search (thus decreasing the total computation time). This is due to the fact that the shape of the reference target is more easily recognizable in the cropped images.

Template Matching with ROI Selection

The benefits of the proposed method are not only limited to CCFind but also to other techniques. Such is the case of template matching. In this approach, by reducing the size of the initial search area, it is possible to also reduce the number of window sizes (scales) at which the template comparisons are done. Furthermore, by selecting a smaller area where the reference target is located, false matches are bound to be reduced. Using the proposed implementation of a template matching algorithm, Table 2 indicates that without the ROI selection step only one reference target is correctly found (the same as in Figure 7(b)). The overall results are thus improved with the proposed preprocessing step. Some examples are illustrated in Figure 9.



(a) (b) (c) (d)

Figure 9. *Color reference target detection examples when using the ROI selection step followed by template matching. The ROI of the full-size images are shown for better visualization.*

Table 2. Detection results using template matching.

Method	# Correct	Computation time
Temp. Matching	1	2:15
Temp. Match. + ROI Sel.	12	1:55

Although the results obtained by CCFind and template matching show the same number of correct detections, analysis of the detected reference targets by each of the approaches allows us to see that: 1) 8 of the images are equally detected by both methods; and 2) the remaining 4 images are method-specific, and are related to the characteristics of each algorithm. In Figure 8(a) we can see one example of a reference target correctly detected by CCFind, but that template matching is unable to detect. This has two main reasons: 1) since the range of scales used during the iteration is discretized to improve the computation time, the exact size of the reference target might not match the template size, achieving false matches especially when the relative size of the reference target is small; and 2) when the reference target has perspective distortion or occlusions, the template matching approach shows some limitations, as it only considers rotation and scale variations. The other images only detected by CCFind show problems related to the appearance variability of the reference target (uneven illumination and too saturated colors), difficult to handle when using a single template. An example of the opposite case, detection only by template matching, is shown in Figure 9(a). Since CCFind relies on edge extraction to find recurring shapes inside the image, the presence of other elements with clear edges such as the black fence in the figure can make the algorithm fail. The rest of the images seem to show the same problem, elements in the image that mask the shape of the reference target. In these cases, template matching shows a clear advantage.

Conclusions and Future Work

Technology has shown an important progress in the past few years, and creating a color-managed workflow is no longer limited to just photo professionals. In order to optimize the digitization of images, a new algorithm that improves the automatic detection of color reference targets when used in combination with currently available tools has been proposed. The algorithm uses a texture transformation to better differentiate the reference target region from the rest of the image (i.e. to enhance its contrast). Then it creates a saliency map by using region and color information, and a threshold is used to create a binary mask of this saliency map. A region of interest (ROI) is finally determined by applying morphological operations to the resultant mask.

The proposed algorithm has been tested on a new challenging dataset that contains images taken under very different illumination conditions, and with reference targets of distinct relative sizes. The experiments show that the algorithm improves both detection and computation time by reducing the search area, although that further improvement is still needed for its application in a fully automated workflow. Tests done with CCFind and a proposed template matching approach show the advantages and disadvantages of each of these methods, and also the general applicability of the proposed algorithm, since it does not require any a priori information about where the reference target is located within the image. This means that it can be used as a preprocessing step prior to existing automatic reference target detection techniques.

The given implementation may lend itself to many extensions. For example, by combining the automatic ROI selection approach with other color and texture descriptors, the detection accuracy could be improved. Its applicability could also be extended to the detection of other reference targets, such as the ColorChecker Digital SG or the Universal Test Target, for which more tests and parameter adjustments would be needed. Finally, future work would also include testing the algorithm on other datasets, for which the performance may sensibly vary and a different parameter selection could be needed.

Acknowledgments

Special thanks to Dr. Francisco Imai for generously allowing his personal photo collection to be used for research purposes. This work was supported by the Erasmus Mundus Master CIMET scholarship 2012-2014.

References

[1] M. Liu, I. Konya, J. Nandzik, N. Flores-Herr, S. Eickeler, and P. Ndjiki-Nya, A New Quality Assessment and Improvement System for Print Media, EURASIP Journal on Advances in Signal Processing, pg. 117, (2012).

[2] X-Rite Photo, `http://xritephoto.com/`. Last visited on 24/8/2014.

[3] C.S. McCamy, H. Marcus, and J.G. Davidson, A Color-Rendition Chart, Journal of Applied Photographic Engineering, 2(3), pg. 9599, (1976).

[4] D. Wueller, H. van Dormolen, and V. Jansen, Universal Test Target Technical Specification, version 1.1, (2011), available from `http://www.universaltesttarget.com/`. Last visited on 27/8/2014.

[5] T. Tajbakhsh and R.R. Grigat, Semiautomatic Color Checker Detection in Distorted Images. Proceedings of the Fifth IASTED International Conference on Signal Processing, Pattern Recognition and Applications, pg. 347352, (2008).

[6] S. Bianco and C. Cusano, Color Target Localization under Varying Illumination Conditions, Computational Color Imaging, Lecture Notes in Computer Science, vol. 6626, pg. 245255, (2011).

[7] K. Hirakawa, ColorChecker Finder, `http://campus.udayton.edu/~ISSL/software/`. Last visited on 27/8/2014.

[8] R. Baumann, MacDuff ColorChecker Finder, `http://github.com/ryanfb/macduff`. Last visited on 24/8/2014.

[9] X-Rite, X-Rite ColorChecker Passport User Manual, `http://www.xrite.com/documents/manuals/en/ColorCheckerPassport_User_Manual_en.pdf`. Last visited on 24/8/2014.

[10] Imatest Master, `http://www.imatest.com/products/imatest-master/`. Last visited on 24/8/2014.

[11] BabelColor PatchTool, `http://www.babelcolor.com/`. Last visited on 24/8/2014.

[12] M-M. Cheng, G-X. Zhang, N.J. Mitra, X. Huang, and S-M. Hu, Global Contrast Based Salient Region Detection, Proceedings of the 2011 IEEE Conference on Computer Vision and Pattern Recognition, pg. 409416, (2011).

[13] N. Neverova, Automatic Recognition of Color Calibration Targets, Selected Topics in Color Imaging Project Report, Gjøvik University College, (2011).

[14] R. Brunelli, Template Matching Techniques in Computer Vision: Theory and Practice, Wiley, (2009).

[15] L.E. García Capel, Automatic Colour Target Detection, Selected Topics in Color Imaging Project Report, Gjøvik University College, (2013)

Author Biography

Luis E. García Capel received his BSc in telecommunications engineering from the Technical University of Cartagena, Spain, in 2007, and his MSc under the Erasmus Mundus program "Color in Informatics and Media Technology (CIMET)" in 2014. His industrial background comprises years of experience in the field of mobile telecommunications and a research internship at Technicolor R&D France. His research interests include applied color science, image and video processing, and computational photography.

Jon Y. Hardeberg received his PhD from the ENST in Paris in 1999. After a short industry career in the US, he joined Gjvik University College in Norway in 2001, where he is Professor of Color Imaging and member of the Norwegian Colour and Visual Computing Laboratory. His broad research interest within the field has resulted in more than 150 papers. He currently leads several large research projects, including the Initial Training Network CP7.0.