Automatic Correction of Multiple-Choice Tests using Digital Cameras and Image Processing

Francisco de Assis Zampirolli and José Artur Quilici Gonzalez and Rogério Perino de Oliveira Neves
Centro de Matemática, Computação e Cognição
Universidade Federal do ABC
Santo André, Brasil 09.210-170
Emails: {fzampirolli, jose.gonzalez, rogerio.neves}@ufabc.edu.br

***Abstract*—There are currently many commercial solutions for automated scoring of Multiple-Choice Tests, usually composed of a software and a scanner, but the widespread dissemination of laptops, tablets and smartphones with built in cameras offers new possibilities for doing the same job with no need for any extra hardware. This article presents a simple and innovative method to transform captured images of answer sheets into reduced binary matrices containing answers to the questions plus some control elements, using simple morphological operations for segmentation. This methodology is applied to the real problem of automatic correction of Multiple-Choice Tests. Initially, the user positions the answer key sheet in front of the camera in order to save the image to the disk, then the image is gauged to evaluate the test type. Subsequently, student answer sheets can be read using the camera, having the test score displayed on the screen and/or saved to a file.**

Keywords: **Image Processing, Mathematical Morphology, Multiple-Choice tests, Optical Mark Recognition.**

I. Introduction

Image analysis can profit from strategies where the neighborhood relation between objects in the image, such as regions of answer sheets of multiple choice tests, can be represented simply by matrix. Furthermore, Mathematical Morphology is an elegant form to solve image-processing problems using consistent theoretical bases, that is the theory of sets [1], [2]. In Mathematical Morphology the transformations between images, which are called morphological operators, can be defined by the structuring functions [3].

The method described was first conceived to address one of the major problems in distance education. To avoid cheating and to verify the identity of the test taker, it is common to require the presence of all students in a test center in a specific date. However, for a large number of students, there might not be enough computer terminals available for an electronic test. In that case, the alternative is to apply printed Multiple-Choice Tests.

Every year, the Federal University of ABC (UFABC, São Paulo, Brazil) offers not only courses in site, but also the possibility of distance education for hundreds of students. Sometimes the number of applicants to the courses may amount to thousands. Looking for feasible ways to overcome these difficulty, a small group of teachers introduced an automated tool to cope with the limited human resources. The work here discussed is part of a long term plan for distance education which will integrate cloud computing, laptops, and smartphones to produce tailored intelligent software.

There are several popular software for generating Multiple-Choice Tests such as *Quiz-Press*, available at <www.solrobots.com/quizpress>. On the more expensive side there are also specific commercial hardware/software bundles dedicated to scoring the tests, however, there are no specific details about them in the literature regarding methodologies and techniques.

Optical Mark Recognition (OMR) is the process of discrete data acquisition by scanning predefined OMR forms, eventually detecting the presence or absence of marks in the spaces reserved for filling. This technology was pioneered in the United States in the 1950s, already being used at assessments of students, and has since become widely used in Multiple-Choice Tests. For its implementation dedicated OMR scanners or image scanners are often used. In the latter case the image is processed by an OMR software with similar efficacy <www.omrsolutions.com> but slower performance, presenting advantages such as lower cost and the possibility to employ user customizable OMR forms. In its recent use in portable devices, such as digital cameras and smartphones, the operation is susceptible to increased error and prolonged runtime, due to the manual capturing of images, given the difficulty to position the paper containing the data parallel to the camera CCD at a suitable distance and the right angle, and image quality issues, such as uneven lighting across the paper. There are several OMR and OCR (Optical Character Recognition) programs available, as well as Apps for iOS and Android phones dedicated to decoding barcodes and QRCode [4], [5], [6].

II. Methodology

Although digital imaging was introduced in the beginning of the twentieth century (1920) by the British inventors Harry G. Bartholomew and Maynard D. McFarlane, the popularization of Digital Image Processing in academia occurred only later in the same century (1970) with the cheapening of personal computers and the advent of digital cameras. Now, in the second decade of this century, we witness increasing demand for simpler and more efficient applications, primarily for use in mobile devices. With this premise in mind, this paper can be seen as a first attempt to provide a cheap alternative of an automated scoring of Multiple-Choice Tests using Image Processing.

A. Data Acquisition

We used an image database of 674 tests filled by candidates to the graduate program Specialization in Information Technology, a distance learning modality course offered at UFABC, the test being part of the entry selection process. Each test was printed on one $A4$ size paper containing both the answer sheet and 24 multiple-choice questions with five alternatives each, only one being the correct answer. Eight different sets of tests were prepared, presenting different questions and variations. The model proposed in this paper supports up to 16 different test sets. We opted to design the answer area in this format for the sake of simplicity, so that any traditional word processor can be used to generate the Multiple-Choice Tests, unlike with traditional OMR forms.

The computer used was a MacBook Pro, equipped with a $2,26GHz$ Intel Core 2 Duo processor, 2 GB of memory, operating system Mac OS X version $10.6.8$ and a graphic processor from Nvidia model GeForce $9400M$, featuring $256MB\ DDR3\ SDRAM$ shared memory. Tests and templates (containing the filled answer sheet for each different set) were later captured using the $2Mega - pixel$ built-in camera. For image processing and information display we used Matlab, version 2011 to write the source code. These requirements are necessary to validate the results presented in this article. However, a low-cost version of the software for Android smartphones is in development, intended to simplify its deployment and use.

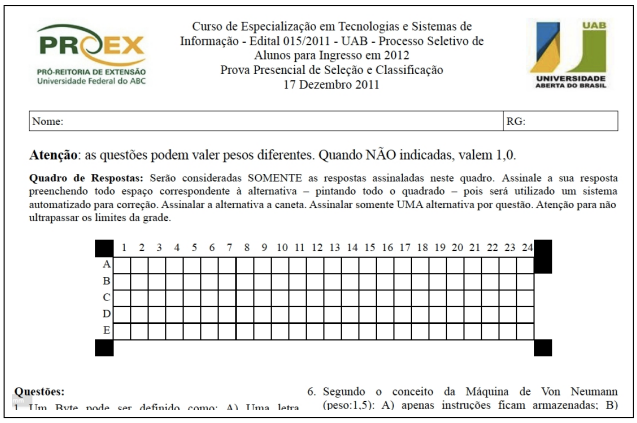
The developed code was written according to the following steps:

1) read the templates for each set;
2) save a TIFF image containing only the answer sheet area of the template;
3) save a two-dimensional matrix containing the correct answers to each set;
4) read the answer sheets from student tests;
5) save the image of each test to a separate file in TIFF format;
6) add the processed test results to a file in csv format.

B. Image Processing

In order to capture the answer sheet area, each test must be positioned in front of the notebook facing the camera, whose result is shown in Figure 1. The answer sheet must be positioned in the image in such fashion the four external bold squares fit within the displayed guidelines (the larger outer rectangle), as shown in Figure 2. Further adjustments must be made so all the 4 external squares are at least 50% contained within the yellow squares. After this adjustment, the image of the answer sheet is captured, as shown in Figure 3. Using this alignment procedure, it was unnecessary to apply rotations or scaling to the captured images, as performed by Nguyen et al. [7], this way preventing further distortions which could compromise the final analysis.

Thus, the target area is marked by four outer squares, serving as guidelines to define the image position and dimensions. For easy understanding, this area can be considered a table of 7 rows and 26 columns, or 7×26, which, excluding the borders, represents a test with 24 questions and five possible answers.


PROEX
PRÓ-REITORIA DE EXTENSÃO
Universidade Federal do ABC

Curso de Especialização em Tecnologias e Sistemas de Informação - Edital 015/2011 - UAB - Processo Seletivo de Alunos para Ingresso em 2012
Prova Presencial de Seleção e Classificação
17 Dezembro 2011

UAB
UNIVERSIDADE ABERTA DO BRASIL

Nome: RG:

Atenção: as questões podem valer pesos diferentes. Quando NÃO indicadas, valem 1,0.

Quadro de Respostas: Serão consideradas SOMENTE as respostas assinaladas neste quadro. Assinale a sua resposta preenchendo todo espaço correspondente à alternativa – pintando todo o quadrado – pois será utilizado um sistema automatizado para correção. Assinalar a alternativa a caneta. Assinalar somente UMA alternativa por questão. Atenção para não ultrapassar os limites da grade.

1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24
A
B
C
D
E

Questões:

6. Segundo o conceito da Máquina de Von Neumann (peso:1,5): A) apenas instruções ficam armazenadas; B)

Fig. 1. Example of a printed test.

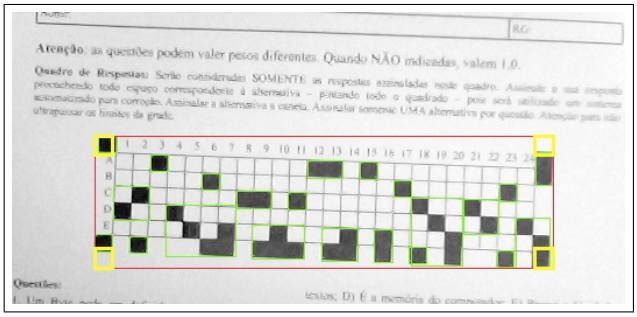


Fig. 2. Fitting the answer sheet within guidelines: The image illustrates the desired position.

The borders of the printed tests contain control elements, including the four squares used to guide the acquisition. By convention the origin $(1, 1)$ (row, column) of this matrix is the upper left corner. Thus, the four control elements for acquisition (yellow squares) from origin and clockwise are $(1, 1)$, $(1, 26)$, $(7, 26)$ and $(7, 1)$.

The matrix element $(6, 26)$ when filled (in this case, with black ink) indicates a test template, in which case an array of 7×26 of zeroes (white) and ones (black) must be written to the disk. Figure 3 shows an example of an image that generates a template in the disk.

The matrix elements $(2, 26)$, $(3, 26)$, $(4, 26)$ and $(5, 26)$ represent the type of test in binary representation, in that order of significance. Thus, 4 array elements (bits) allow up to 16 different types of tests. Figure 3 shows a type 1 test.

Images for all templates and student tests are saved to the disk in TIFF format for further improvements using image processing procedures. Figure 4 shows the program output in which the operator can verify the correctness of the data being

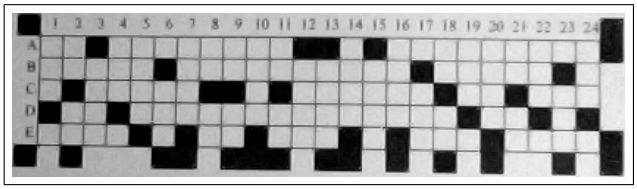


Fig. 3. Image captured with the camera, to be processed for detection of the given answers.

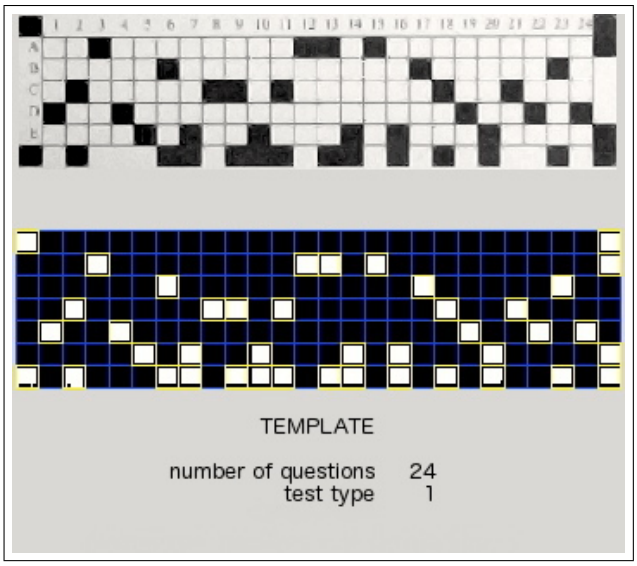


Fig. 4. Program output displayed to the user, considering a template.

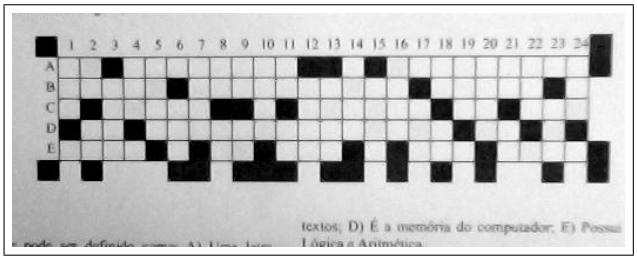


Fig. 5. Input image after capturing.

saved, in this case, a template of the test type 1. The details of image segmentation applied to achieve this result will be presented in the next section. When a student test is properly scored by the system, the program will read this template from the disk and make comparisons to generate the score.

C. Image Segmentation

To segment [8] the answer sheet area of the test, presented in the previous section, each frame was captured using the command

```
colorImage = getsnapshot(vidobj);
```

the variable `vidobj` being defined in Matlab with the command `videoinput`. The color image is then converted to gray levels, by retaining only the first band, corresponding to the red component, through the command

```
grayScaleImage = imadjust(colorImage(:,:,1));
```

in which the function `imadjust` was used to improve contrast (for more details see the help in Matlab). Figure 5 illustrates the input image in gray scale `grayScaleImage`.

After this step, we used a threshold transformation [9], as shown by the next steps:

```
T = graythresh(grayScaleImage);
binaryImage = 1-im2bw(grayScaleImage,T);
```

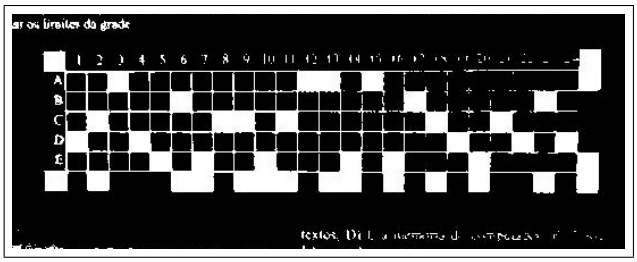


Fig. 6. Image after applying the threshold technique.

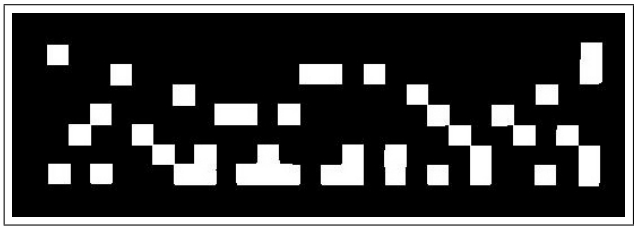


Fig. 7. Image after applying opening and clearing the border techniques.

The resulting black and white image `binaryImage` is a binary image with black squares receiving value 1 and white squares 0 in the represented 26×7 array. See Figure 6.

Next, a morphological opening operation [3] is performed, using a square structuring element of size 10 to remove noises, including the rows and columns of the answer sheet area:

```
binaryImage2=imopen(...
binaryImage,strel('square',10));
```

Finally, we eliminated the connected components touching the edge (See Figure 7):

```
binaryImage3=imclearborder(binaryImage2,4);
```

The image `binaryImage3` is the input used to determine the 26×7 matrix by using the Matlab function `regionprops`, where each marked square in the image represents at least one connected element. The marked squares are highlighted in Figure 8, where the outer red square is defined by the four yellow squares at the corners. In this case, all the 4 external squares are at least 50% contained within the yellow squares, and the process of image capture ends. Now begins the process of image segmentation to determine each square marked on the answer sheet area.

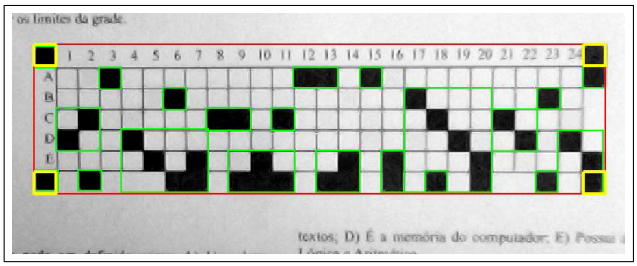


Fig. 8. Image illustrating the recognition of green squares using the `regionprops` function on the labeling of the image shown in Figure 7.

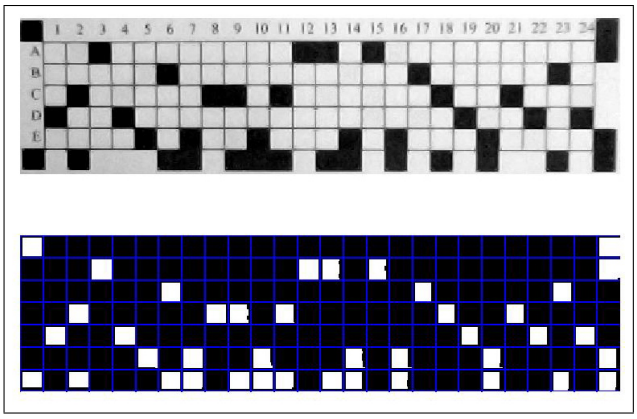


Fig. 9. Image showing the blue outline of rows and columns according to the number of questions and answers in the test.

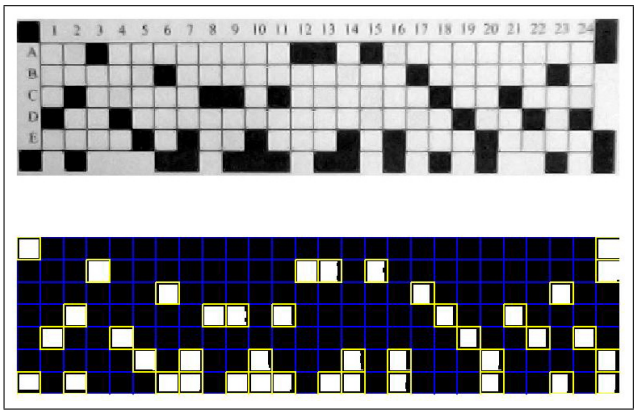


Fig. 10. Image illustrating the detection of filled squares, highlighted in yellow.

Then, the image in Figure 7 is partitioned in columns and rows, according to the number of questions and answers, zeroing the remaining spaces, as illustrated in Figure 9, with separators for rows and columns shown in blue.

The next step is to identify each filled square. This step is performed again using the `regionprops` function, with results highlighted in yellow in Figure 10.

Finally, we check for invalid responses, like questions containing none or more than one answer. In case of templates, a 7×26 matrix of zeros and ones is saved to disk, ones for marked squares and zero otherwise, as illustrated in Figure 10. When scanning a student test, the test matrix is compared with the matching template found on the disk, as shown in Figure 11. The control elements (blue squares) in the last row of the image indicate questions with increased value, in this case, questions worth 50% more. Moreover, the green square means that response was correctly marked, and red square means where would be the correct answer on a question incorrectly marked.

The video presented in [10] shows part of the correction process, conducted in one afternoon, for a total of 674 tests. The process of correcting a single test spends in average 10 seconds.

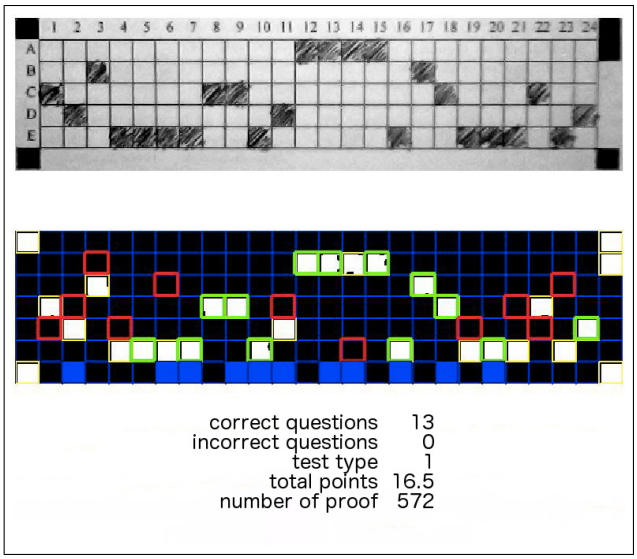


Fig. 11. The first gray scale image shows a captured student test. The bottom image shows the test scoring, highlighting the correct answers in green and the wrong answers in yellow. The red squares highlight unmarked correct answers. Control elements are located in the borders, like the outer squares in the first and last column, which contains the four corner guides and the coded test type. The marked squares in the bottom line indicate questions with increased value. Finally, a summary is shown below the image displaying the number of correct answers, invalid answers, the test type, the final score and the test number.

III. Results and Discussion

When a question has none or more than one answer or was incorrectly marked (for example, with a "X" instead of a filled square), it is eliminated and marked with a vertical yellow rectangle spanning the whole column, as shown in Figure 12, 13, 14 and 15, allowing for the operator to evaluate whether the mistake was made by the student or by the system in detecting the mark.

For 674 tests of eight different types, only nine scoring problems were detected, four of them shown in figures 12, 13, 14 and 15, resulting in error rates as low as 1.3%. The remaining five error cases were similar to those in the presented figures, each test having only one misinterpreted question. The nine problematic cases occurred because students did not follow the instructions. For the scores obtained using the presented methodology, in spite of instructions in the test, which clearly explained that the squares should be completely filled with ink for proper recognition, invalid answers were detected in only 6.7% of all 674 tests (42 tests), the operator being able to visually identify them and alter the score when necessary.

Whereas state of the art solutions for correction Multiple-Choice Tests may offer as low as 0% error rates, this paper presents a cheap alternative, able to popularize automatic corrections without the need for additional hardware.

A. Future Works

Using image processing techniques, such as region growing and other filters, image segmentation can be further improved in the algorithm to consider valid answers similar to those

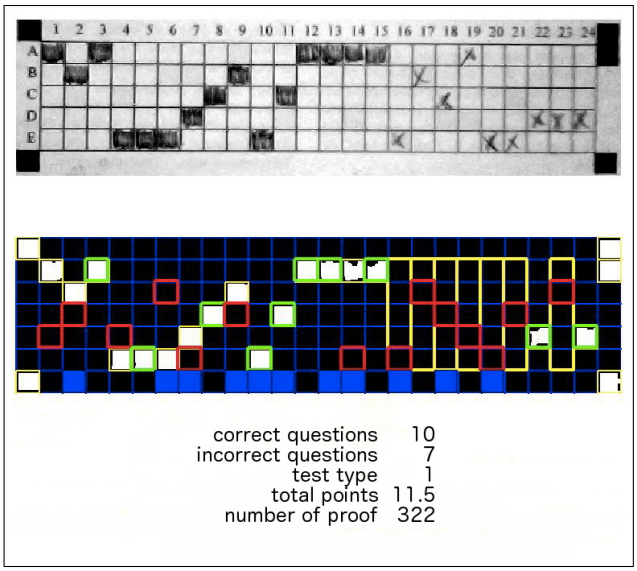


Fig. 12. Image highlighting in yellow questions containing squares incorrectly filled in test number 322.

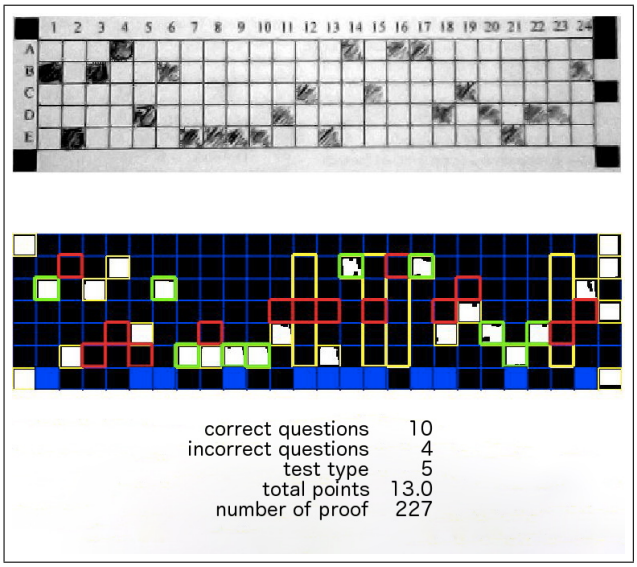


Fig. 13. Image highlighting in yellow questions containing squares incorrectly filled in test number 227.

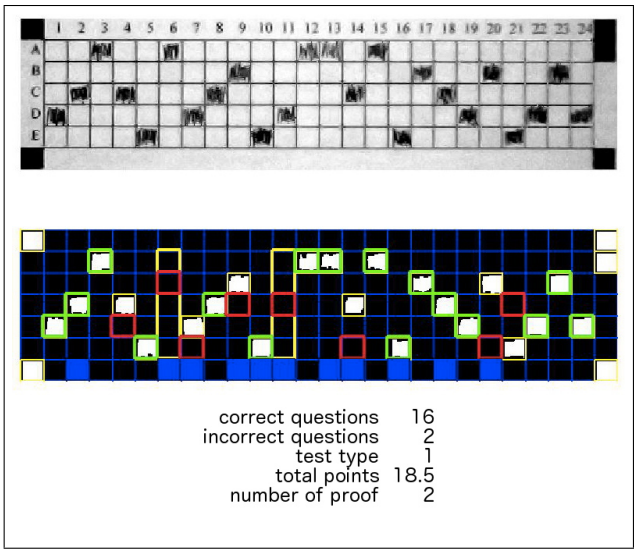


Fig. 14. Image highlighting in yellow questions containing squares incorrectly filled in test number 2.

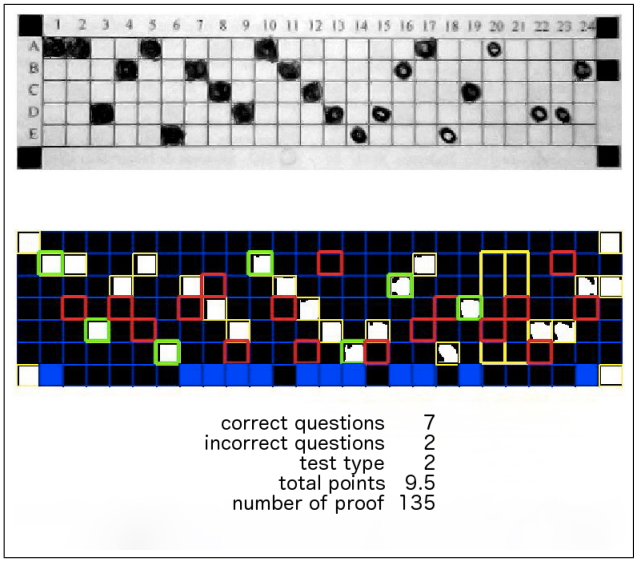


Fig. 15. Image highlighting in yellow questions containing squares incorrectly filled in test number 135.

presented in Figures 12, 13, 14 and 15 [1].

A similar application is currently being ported to mobile devices, with a few improvements which will enable it to run stand-alone on smartphones and tablets running Android OS. Early results explored the capability of transmitting images from mobile devices to desktop computers for further analysis. As an example, Figure 16 shows an image acquired from a 3 mega-pixel camera-phone (LG Model LGP350), in a resolution of 2048×1536 pixels. The acquired images was transmitted via Bluetooth and stored in a specific folder in a Desktop. The code was modified so the line responsible for acquiring the image from the computer camera now reads the files from the specified location. The remaining computations are analog to the procedures presented in this paper.

Another possible source of images is a scanner, with capability to batch digitize hundreds of tests and send the images straight to a computer folder through the network. Despite the fact that it would enable further automatization of the process, effectively eliminating the whole video capture section, it would make the software less inclusive, considering the need for a scanner in order to use it.

IV. Conclusion

In this paper we have presented a system which uses techniques from mathematical morphology, capable of acquiring images from computer cameras (and eventually a myriad of alternative devices) to perform automated scoring of a Multiple-Choice Tests, presenting high accuracy in the

[1] All images used in this article are available on <http://professor.ufabc.edu.br/~fzampirolli/MCTest> for future works by the community.

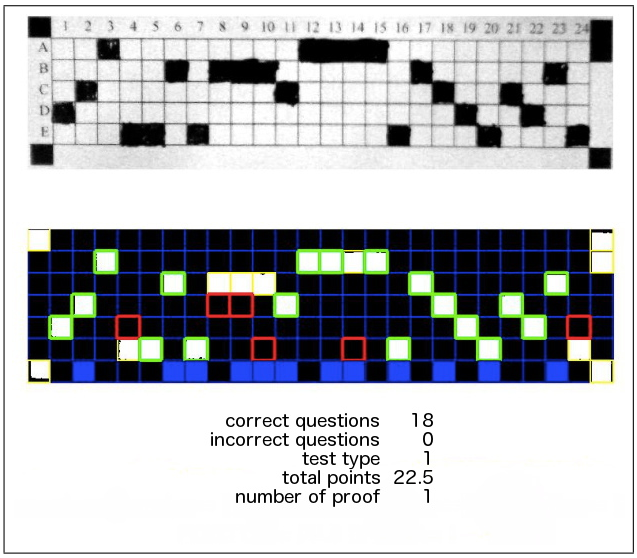


Fig. 16. Image captured by a cell phone and transmitted via Bluetooth to a PC, which was responsible for executing the same procedure presented on the stored image.

results. The steps performed by the algorithm were explained in detail (accompanied by the respective Matlab commands) and illustrated by its employment in a real case scenario.

The software is currently being ported to smartphones, what will allow teachers to travel to test centers carrying only printed tests with no need for additional resources, apply the tests, capture images and calculate scores in loco, being able to discuss the test results with the students on the same day.

REFERENCES

[1] J. Serra, *Image analysis and mathematical morphology* (Academic Press, London, 1982)

[2] J. Serra, *Image analysis and mathematical morphology - Volume {II}: theoretical advances* (Academic Press, London, 1988)

[3] H. Heijmans, *Morphological image operators* (Academic Press, Boston, 1994)

[4] D.J. Sen, R.N. Patel, U.Y. Patel, International Journal of Pharmaceutical and Applied Sciences **1**(2), 56 (2010)

[5] A.F. Mollah, N. Majumder, S. Basu, M. Nasipuri, International Journal of Computer Science Issues **8**(1) (2011). URL http://arxiv.org/abs/1109.3317v1

[6] K. Chinnasarn, Y. Rangsanseri, in *Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series*, *Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series*, vol. 3808, ed. by A.G. Tescher (1999), *Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series*, vol. 3808, pp. 702–708

[7] T.D. Nguyen, Q.H. Manh, P.B. Minh, L.N. Thanh, T.M. Hoang, in *International Conference on Advanced Technologies for Communications (ATC 2011)* (2011), pp. 268–271

[8] F. Meyer, International Journal of Pattern Recognition and Artificial Intelligence **15**(7), 1089 (1996)

[9] R.C. Gonzalez, R.E. Woods, *Digital Image Processing*, 2nd edn. (Addison-Wesley Publishing Company, 2002)

[10] F. de Assis Zampirolli, J.A.Q. Gonzalez, R.P. de Oliveira Neves. Automatic corrections of Multiple-Choice Tests. YouTube <http://youtu.be/rhp4gfwrdf0> (2012)