



Research Article

Test and Development of a Low-Cost Solution for Automatic Inspection of PET Bottle Codification

Gabriel Araujo de Souza^{*} , Pedro Rafael Fernandes 

Department of Chemical Engineering, Faculty of Engineering, Federal University of Rio Grande do Sul Luiz, Englert Street, Building 12204 - Porto Alegre, RS, Brazil
E-mail: gabrielaraujodesouza98@gmail.com

Received: 5 May 2022; **Revised:** 27 June 2022; **Accepted:** 15 July 2022

Abstract: Quality of packaging is a key factor in consumer choice for a particular product. Besides the package appearance, legal requirements demand that certain information is presented to the consumer. The main goal of this work is to evaluate the technical feasibility of using a Raspberry Pi board as a platform for automatic quality inspection for the expiration date and batch information in polyethylene terephthalate (PET) bottles. The system consists in capturing digital images using a Raspberry Pi Camera, processing the image obtained and performing optical character recognition in order to determine if the coding is compliant. Two models of bottles were used, and 24 images of each were captured, 15 of which were examples of compliant bottles and 9 were defective. The tests were divided into two sequences of processing methods. The best sequence achieved 100% specificity, 91.7% accuracy for model A bottle and 83.3% for model B.

Keywords: image processing, automatic inspection, codification quality, Raspberry Pi

1. Introduction

Legal requirements demand certain information to be available to the consumer on packaging in several countries. When dealing with products packaged in polyethylene terephthalate (PET) bottles, aspects such as quality of labeling, encapsulation, readability of the expiration date and bottle formation are part of the quality control of many industries. The coding of PET bottles generally exhibits batch identification information and expiration date for food products. The presentation of this information is mandatory in Brazil according to Resolution-RDC No. 259 of ANVISA [1], which approves the technical regulation on labeling of packaged foods. This resolution defines the formats allowed and emphasizes that all information must be presented clearly and legibly on all products. Non-compliance with any specifications of the resolution is subject to the sanctions of Federal Law No. 6437 [2], which defines violations of federal sanitary legislation. For companies, this information is essential for the product tracking process. Through the batch number, it is possible to trace the entire production chain of an item, and thus to take the necessary measures if there is a problem that requires recall of the batch, or carry out investigations of anomalies found through customer's services. Therefore, the periodic verification of the coding quality of the products is a step adopted as part of the routine in several factories.

In industries that do not have an automatic system, an acceptable time interval is defined for visual inspection of the bottle codification, in order to ensure that it is not blurred or even shows missing characters and with the correct

expiration date [3]. If any non-conformity is detected, production is stopped from the moment of the last compliant check until the moment when the abnormality is solved. This production must be reprocessed or discarded. If the failure was the absence of coding, it will be necessary to put the bottles back on the production line so that they go through the coding machine again. If the failure is blurred, illegible, or incorrect expiration date, it will be necessary to manually erase the defective coding, and then re-run the bottles through the production line. The company must define whether to carry out this rework or discard the non-conforming products since rework requires manpower, physical space, and time.

Currently, there are methods based on computer vision to determine whether a certain aspect of a product is compliant or not [4-9]. These solutions transform a qualitative analysis into a quantitative one, they avoid human error in the analysis and can be carried out in production lines with high speed, at intervals determined by the needs of the companies or by the capacity of the equipment available. Cap and label quality inspections, as well as product filling levels, are examples of applications of computer vision systems consolidated in the field. Inspection of coding quality on PET bottles presents some associated difficulties. Among these difficulties, we can mention: the cylindrical/conical curvature of the printing area, which makes character recognition harder; the nature of the PET material, which can reflect light and cause some characters to blur; the coding position, which is sometimes below the liquid level, decreasing the contrast between letters and liquid, depending on its color; and the formation of foam, bubbles or drops on the walls of the bottle, which can make the identification of the characters difficult.

Considering the importance of coding quality control in the industry and seeking to improve the reliability of this verification, and thus reduce the costs involved in case of failures, this work aims to investigate the technical feasibility of an automatic coding quality inspection system for PET beverage bottles employing a commercially available low-cost microprocessor board and camera. Tests have been designed to simulate industrial plant operations. The tests allow one to determine if the devices are capable of providing the necessary accuracy for the process. Therefore, in this paper, a sequence of suitable digital image processing methods is defined in order to improve the efficiency of the optical character recognition system (OCR) employed, as well as the most suitable OCR tools for the task. Section 2 presents the main bibliographic references used in this study. Section 3 presents the materials and experimental setup, bottle models, and acceptance criteria for the algorithm classification. Section 4 presents the individual methods for image processing as well as the sequence of methods proposed. Section 5 presents the results and a detailed discussion about the main reasons behind misclassifications, and Section 6 presents a summarization of the study and provides insights for better results in future works.

2. Bibliographic references

The following topics will present information related to the subject of this study and theoretical references for the applied methods, and also similar research regarding the subject.

2.1 Fundamental concepts about digital images

An image can be defined as a continuous function of luminous intensity, denoted $f(x,y)$, whose value or amplitude in coordinates (x,y) gives the intensity or image brightness at that point. Since most image analysis techniques are performed through computational processing, the function $f(x,y)$ must be converted to a discrete form. To obtain a digital image, two steps are necessary: sampling and quantization [10].

Sampling is responsible for discretizing the image definition domain in the x and y directions, generating a sample matrix of size $M \times N$, in which each element of this matrix is called pixel. The generated matrix can be expressed as in Equation (1), where M and N represent the number of columns and lines of the matrix:

$$f(x, y) = \begin{bmatrix} f(0,0) & \cdots & f(M-1,0) \\ \vdots & \ddots & \vdots \\ f(0,N-1) & \cdots & f(M-1,N-1) \end{bmatrix} \quad (1)$$

Quantization is responsible for determining the integer number L of gray levels, considering a monochrome image, allowed for each image point. The interval between L_{min} and L_{max} is called grayscale. By convention, black is assigned to the darkest gray level, and white is assigned to the lightest gray level. It is usual in the digital processing of images to assume that the image dimensions and the number of gray levels integer powers of 2. In the case where the number of gray levels is equal to 2, the image is called binary. These images take up less storage space and are easily manipulated by computers, without requiring large processing power [10].

On the other hand, a multispectral image can be represented as a sequence of monochromatic images, such that each image is known as a band. Most visible colors by the human eye can be represented as a combination of the bands of primary colors red, blue, and green, which are commonly used to represent color images [10].

2.2 The concept of kernel or mask

In image processing, there is a need to perform filtering operations to extract information of interest from the image. These operations can be performed in both the space and frequency domains. Filtering in the spatial domain means that the operations are performed directly on the set of pixels that compose the image. That is usually done using matrices called “kernels” or “masks”. The simplest kernel has a size of 3×3 , and each position is associated with a numerical value, called weight or coefficient, as shown in Figure 1. The application of the mask centered on the coordinate (x,y) consists of replacing the pixel value at position (x,y) with a new value, which depends on the neighboring pixel values and kernel weights. The coefficients of the filter are multiplied by the gray levels of the corresponding pixels and then summed, replacing the gray level of the central pixel, according to Equation (2), where z represents the pixel position and w the weight of that position. The mask is moved to the next pixel position in the image, and the process is repeated until the mask passes through all the positions of the image. Applying this filter to each pixel of the image is called correlation.

w_1	w_2	w_3
w_4	w_5	w_6
w_7	w_8	w_9

Figure 1. Kernel of 3×3 pixels with arbitrary weights

$$R = w_1 z_1 + w_2 z_2 + \dots + w_9 z_9 = \sum_{i=1}^9 w_i z_i \quad (2)$$

2.3 Image acquisition processes

Digital cameras typically use arrays of sensors in a two-dimensional (2D) matrix format for image acquisition. Energy from a light source is reflected in objects in the scene and then goes through the first part of the image acquisition system, which consists of an optical lens that projects the scene onto the focal plane of the lens. The arrangement of sensors, located in the focal plane, produces an electrical voltage proportional to the integral of the energy received at each sensor. Digital and analog circuits convert this voltage into an analog signal, which is then digitized by another part of the signal acquisition system. The final result is the acquisition of a digital image [11].

The exposure time is determined by the shutter speed and can be defined as the amount of time the shutter remains open while shooting a picture. So, the use of higher shutter speeds results in shorter exposure times and vice versa [12]. In the rolling shutter method, the process of stopping the accumulation of energy in the sensors is not done simultaneously, but line by line in the arrangement, rather than simultaneously as in the global shutter method. This way of reading the accumulated charges of the sensors can cause the rolling shutter effect if the captured object is in motion and the exposure time is not short enough to ensure that the image freezes. Figure 2 shows an example of this effect.

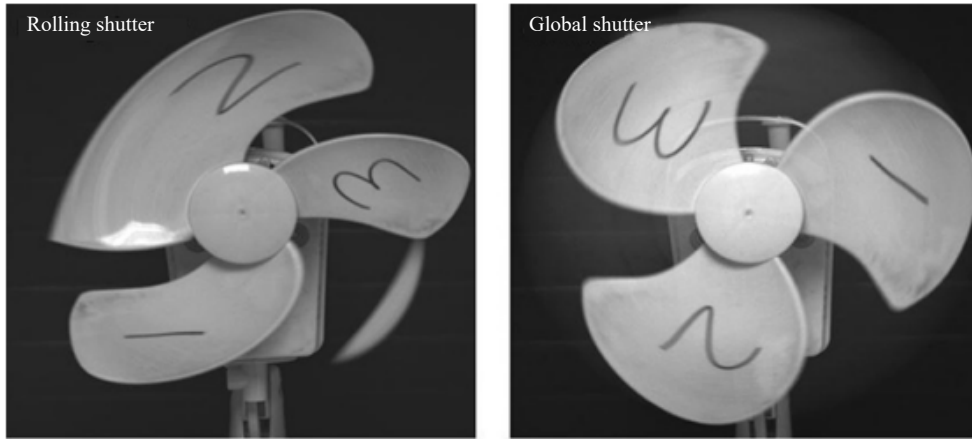


Figure 2. Example of rolling shutter and global shutter effects for the same exposure time. Adapted from [12]

The research developed by Silva [13] shows the influence of adjustment parameters of a semi-professional camera in the acquired images. Among the parameters tested, it was evaluated the influence of shutter speed in capturing images of still objects, and it has been observed that the increase in shutter speed causes a darkening in the acquired images, as the time for capturing photons is shorter. Furthermore, higher speeds contribute to image stabilization, helping to minimize shakes and blurs.

Another scenario tested in the research involves differences in the nature of incident light. It has been observed that the distance between the light source and the object changes the type of shadow generated. The closer the source is to the object, the sharper the shadow. Finally, it was observed that, by using a sheet of aluminum foil as a screen, it is possible to direct photons to regions of the object that do not receive direct lighting, generating the effect of diffuse lighting, which is responsible for causing a decrease in the obtained shadow size or even cast no shadow at all. Figure 3 exemplifies the effects of variation of shutter speed and variation in the forms of incidence of light on the object.

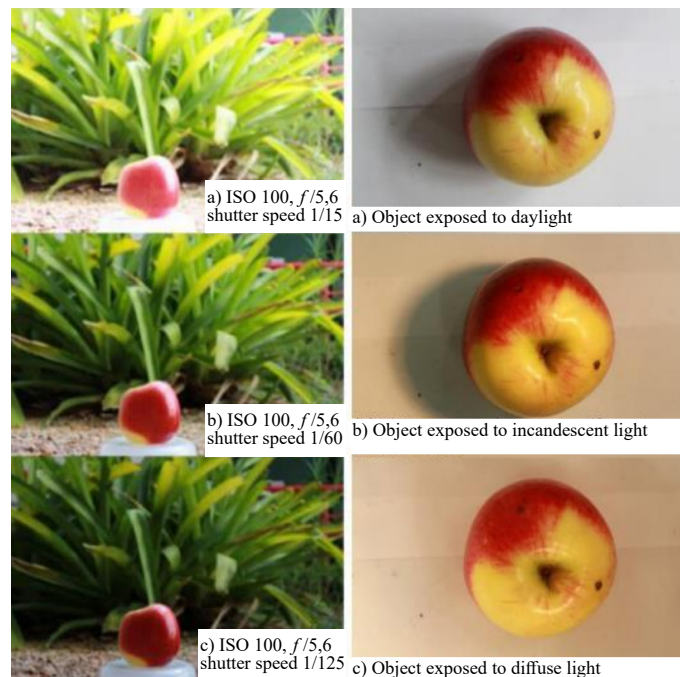


Figure 3. Effect of the tested variations. Adapted from [13]

2.4 Concepts about text detection and recognition and similar research

Identifying text in images typically involves two steps. The first one is detecting the region where the text is contained, obtaining the coordinates of the image that contains the text, commonly called a bounding box. The second stage consists of recognizing the characters of this region, and for that, OCR algorithms are utilized. For both stages, it is common to use algorithms based on neural networks, trained for each of the objectives.

The study developed by Zaafouri et al. [14] proposes the steps to build a vision system to recognize the expiration date characters of industrial products. The model was tested on a variety of products, with different formats, with an exclusive focus on identifying expiration dates. Four processing stages are proposed: image pre-processing, character segmentation, feature extraction and character recognition. The pre-processing step consists of converting the image from the red, green and blue (RGB) color space to the grayscale space. Then, the angle correction, the thresholding operation and application of the morphological erosion operation are performed, responsible for decreasing the thickness of the digits, seeking to disconnect characters. Angle correction and erosion thresholding operations are shown in Figure 4.

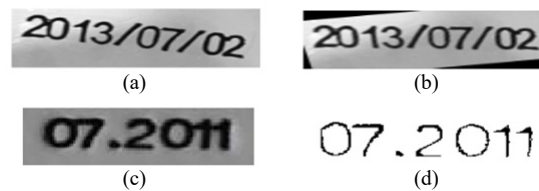
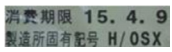
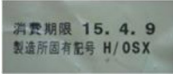



Figure 4. (a) Original image; (b) Angle correction; (c) Connected characters; (d) After binary threshold and erosion. Adapted from [14]

In Hosozawa et al. [15], an application for use on smartphones is developed in order to identify expiration dates printed on industrialized food packaging. The authors propose that application users use their smartphone to capture a photo of the expiration date and the item in question before storing it in a refrigerator for perishables items. After capturing the images, the recognized expiration date and the captured photo are stored on a server, which will be responsible for managing the food stock contained therein. Three open-source OCR algorithms were tested, and the most suitable for the application was chosen. Based on operating system compatibility, recognized character types, and available documentation, OCR Pytesseract was chosen. The influence of the size of the region of interest on the character recognition ability was studied. As shown in Figure 5(a), the authors showed that the existence of space around the area with the text of interest leads to recognition errors, even in images with good contrast and lighting. The influences of contrast and brightness were also analyzed, and it was observed that, for contrast and brightness values of -40%, 0% (default) and +40%, the results obtained are identical, and the characters are correctly recognized, as shown in Figure 5(b) and (c).

According to the authors, no expiration date with dotted characters was recognized correctly, as well as in the case where the characters are in a position other than horizontal. The number of packages tested with this type of source is not mentioned. Expiration dates printed on PET and glass bottles have been tested, and it is mentioned that on bottles less than 5 cm in diameter the characters are not recognized correctly, due to the natural distortion caused by the shape of the bottles. As for the influence of light, the authors mention that even small amounts of light reflected by the surface of the package can cause errors in character recognition. Figure 6 demonstrates the case of the dotted source and the influence of light reflection. Table 1 presents a compilation of problems encountered and solutions proposed by the authors.

Blank space	Little	Half of image	Two-thirds of image
Input image			
Results	Correctly recognized	Misrecognized	Misrecognized

(a)

Contrast	-40%	0%	+40%
Input image			
Results	Correctly recognized	Correctly recognized	Misrecognized

(b)

Brightness	-40%	0%	+40%
Input image			
Results	Correctly recognized	Correctly recognized	Misrecognized

(c)

Figure 5. (a) Influence of blank spaces in OCR performance; (b) Influence of contrast; (c) Influence of brightness. Adapted from [15]

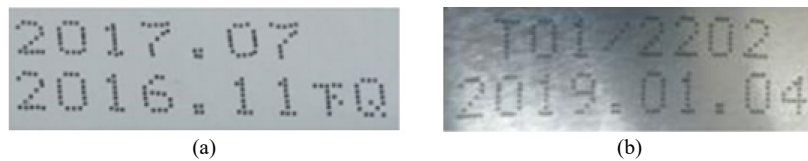


Figure 6. (a) Expiry date with dotted matrix characters, misrecognized; (b) Expiry date subjected to light reflexes, misrecognized. Adapted from [15]

Table 1. Relation between problems encountered and proposed solutions. Adapted from [15]

Problem	Proposed solution
(1) Images including wide non-character space are misrecognized	(a) Crop the image to include only the space containing characters
(2) All characters which are not taken horizontally by a camera are misrecognized	(b) Rotate characters horizontally
(3) Some background and character colors were misrecognized	(c) Convert background and character colors to white and black, respectively
(4) Dot matrix characters are misrecognized	(d) Connect dots
(5) Thin line fonts are misrecognized	(e) Thicken character lines
(6) Very distorted characters are misrecognized	
(7) Images including reflection are misrecognized	

Considering that the research proposal is the development of an Android application, items (1) and (2) are resolved directly in the application, where it is possible to rotate and crop the image as the user wishes. Problem (3) is solved with thresholding, and problems (4) and (5) with the dilation method, as shown in Figure 7. The authors did not propose

a solution to problems (6) and (7).

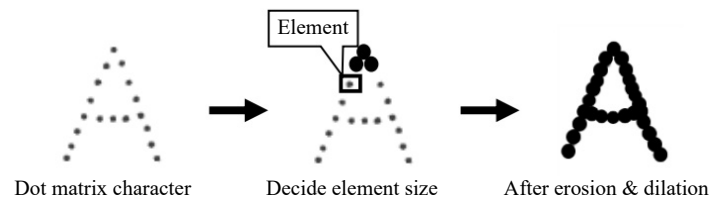


Figure 7. Demonstration of the dilate technique to connect the dots. Adapted from [15]

The work developed by Gong et al. [16] aims to identify the expiration dates of industrial packaging through the steps of detecting the location of the text and subsequent use of OCR algorithms for recognition. The images used have text in different orientations, the packages are varied and, therefore, have different coding standards, changing the way it is presented, as well as the colors of the letters and background colors. As for the detection of the expiration date region, an accuracy rate of 98% was obtained over the 240 test images. This was achieved using the Efficient and Accurate Scene Text (EAST) [17] neural network, after fine-tuning with 800 image samples of expirations dates. For character recognition, the Pytesseract library was used. The authors found that recognition errors can occur in cases where the text is blurred, and data regarding the accuracy of this recognition were not presented.

In Cunha [18], 18 OCR tools available on the market were analyzed. Based on availability criteria, support for programming languages, accepted image formats and costs, nine were selected to be used in the tests proposed by the study: Tesseract, Google Cloud Vision Application Programming Interface (API), GOCR, ABBYY Cloud OCR SDK, OCR Document Haven On Demand, OCR.Space API, New OCR API, Microsoft Vision API, OCR Web Service, REST API, SemaMediaData OCR and Cloudmersive. The study measures the accuracy of the nine OCR tools applied to five images, namely: an electronic invoice, a public commitment note, an Ordinance from the Ministry of Education, an image of the lyrics of the National Anthem and an image of a poem by Fernando Pessoa. Each image was tested at 100, 300 and 600 dpi resolutions, and a photo version. For the samples in the photo version, it is noted that the OCR Web Service and Google Cloud Vision API tools were consistently the ones with the best accuracy.

2.5 Similar research regarding quality control with moving samples

In Machado [19], the development and evaluation of a system for inspecting bottles in the production line was carried out, using a Raspberry Pi 3 Model B board. Two models of bottles were evaluated, one made of white high-density polyethylene (HDPE) material with a blue cap, and another made of green PET with a green cap. The goal of the proposed system was to detect the non-conformity of the absence of a cap, and it was evaluated based on the speed of the production line of a filling machine for sanitizing products. The direct lighting method with light-emitting diode (LED) lamps was used. The camera used was the Raspberry Pi Camera V2. Altogether, 313 images of white vials and 125 images of green vials were captured and analyzed, and the images were divided between vials with and without caps. The nominal speed of the line was 8160 vials/hour. The exposure time used was 1500 μ s, with a resolution of 1280x720 pixels. The author mentions that, due to the limitations imposed by the lighting system, it was not possible to use exposure time values below 1500 μ s, which caused some distortions in the images, such as the motion blur effect and the rolling shutter effect. The author also mentions that the captured images presented small variations in the lateral position of the flask, caused by delays in the inspection system and the high speed of the conveyor. All threshold operations performed used the Otsu method, which consists of calculating a global threshold value that maximizes the contrasts for the image. The results obtained had an accuracy of at least 98%, and some methods employed got 100% accuracy. The average processing time varied between 312 ms and 2.07 s, depending on the method employed.

The method proposed by Sharma et al. [20] performs an inspection on PET bottles in order to identify if the bottle is filled, the radius of the base and the top, the fill level, and if the cap was properly placed. The distinction between full

and empty vials is made on the basis of noise removal and augmentation methods. For top and bottom inspections, circle search methods were applied. The presence of cap and fill levels were defined according to the distance between two detected lines and a reference line, which is localized between the two lines. The distance between the top and center lines represents the height of the cap, and for the filling level, the distance between the center and bottom lines must be equal to a predetermined value, which is considered the correct fill level. The authors varied the lighting conditions of the experiments, noting that inadequate lighting can lead to detection failures. The accuracy obtained was 100% in inspections of fill level and presence of a cap in scenarios with adequate lighting, and 88% of accuracy with inadequate lighting. The authors also mention that it is necessary to ensure that the bottle is in the center of the captured image. Therefore, detection of vertical lines is performed, and the image will be processed only if the detected lines are in the proper region. The distance between the bottle and the camera is approximately 30 cm, and a white background screen has been placed for level and cap quality inspections. Conveyor speed or image processing time are not mentioned.

From the references presented, we can confirm that there is a high interest in developing visual inspection methods for quality control from affordable materials. Studies were presented that apply digital processing techniques and technologies with embedded systems. However, expiration date inspection was done with stationary samples, and studies with moving samples identified characteristics such as liquid level and cap placement, but not coding quality. This work proposes to investigate the feasibility of using an embedded system to identify coding quality in moving samples, thus innovating by introducing a new challenge to the technology. The choices for image processing, OCR tools, and set-up equipment for image acquisition are based on the presented literature recommendations.

3. Materials and methods

The system used for the tests consists of a Raspberry Pi Model 3B microprocessor board, with a Rascamera V2 camera linked via camera serial interface (CSI) cable and with an infrared sensor connected to the Raspberry pins through jumper cables. This camera allows the adjustment of parameters such as shutter speed, brightness, resolution, image format, and saturation, among others. The cost of the inspection system is around US\$62, considering US\$35 for the Raspberry Pi, US\$25 for the PiCamera V2 and \$2 for the infrared sensor [21]. We consider this system to be low-cost because the cost is much lower compared to professional machine vision inspection systems, where only the cameras cost at least US\$3000, and a complete solution, at least US\$10000. These estimates were made considering prices available on the internet, and may not represent the average market prices, as many suppliers do not provide values without a quote request.

To simulate the production line, a domestic conveyor was used. Bottles are manually placed at the beginning of the conveyor and the equipment with the sensor is positioned at the end. When the sensor detects the presence of a bottle, a photo is captured and processed. The distance between the camera and the bottle on the conveyor is approximately 5 cm. Two lamps aligned to the camera position were used, behind and in front of the bottle. The rear lamp is of type LED with a luminous flux of 2250 lm and the front lamp is fluorescent with 2040 lm. The assembled system can be seen in Figure 8. The conveyor speed used for all samples was 0.40 m/s.



Figure 8. Test apparatus

3.1 Image processing algorithm

In order to capture the images, an algorithm was developed for the acquisition of the images, with the parameters being the shutter speed, brightness and resolution. The shutter speed was set to 800 μ s because values higher than that caused the rolling shutter effect, which causes distortions in the codification characters. This algorithm runs on the Raspberry Pi, and the image capture is triggered by the presence of a bottle as detected by the infrared sensor. The sensor has an emitter and a receiver. When the receiver senses that the emitted frequency has rebounded, the sensor returns a value of zero. Therefore, it means the bottle has been detected. The pseudocode is presented in Figure 9.

```
import RPi.GPIO as GPIO
from picamera import picamera

GPIO.setup(infraredPinNumber, GPIO.IN)
camera ← PiCamera()
camera.resolution ← (640,480)
camera.brightness ← 70
camera.shutter_speed ← 800
camera.start_preview()

while True:
    sensor ← GPIO.input(infraredPinNumber)
    if sensor == 0:
        camera.capture(PathToStoreImage)
        break
```

Figure 9. Pseudocode for algorithm in Raspberry Pi

After acquiring the images, they were extracted from the Raspberry Pi and downloaded to a computer where the images were processed and the quality of the coding evaluated. The steps used in the treatment of the photos were determined based on the existing recommendations in the literature, such as similar research [4-10] and theoretical reference on digital image acquisition and processing [11, 12]. The text detection and recognition steps were performed with the help of OCR tools, while the photo treatments used the OpenCV library. All algorithms developed used Python programming language in version 3.8.0.

3.2 Experimental setup

The inspection system was tested on two different PET bottle models. Model A presents the coding in the region just above the label, with no curvature in the text. Model B presents the coding close to the cap and with a slight curvature in the text. The models are presented in Figure 10.



Figure 10. Bottle codification types: (a) A; (b) B

Five bottles of each model from different batches were employed, to ensure greater variability in the presented characters. Each bottle was put through the conveyor three times, totalizing 15 compliant samples of each model. In

addition to these, three defective bottles of each model were used with some of the following non-conformities: lack of coding, lack of character, and blurred letter. Each of these bottles was put through the system three times, totalizing nine non-compliant samples. Bottles were classified into two categories, “compliant” (positive) and “non-compliant” (negative). The results were additionally labeled as correct (true) or wrong (false). The results were expressed using confusion matrices and the metrics defined in Equations (3) to (5), where “TP” stands for True Positive, meaning that a compliant bottle sample was correctly identified as compliant by the algorithm; “TN” stands for True Negative, meaning that a non-compliant bottle was correctly identified as non-compliant; “FP” stands for False Positive, meaning that a non-compliant bottle was incorrectly identified as compliant; and “FN” stands for False Negative, meaning that a compliant bottle was incorrectly identified as non-compliant.

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}} \tag{3}$$

$$\text{Sensitivity} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{4}$$

$$\text{Specificity} = \frac{\text{TN}}{\text{TN} + \text{FP}} \tag{5}$$

3.3 Coding quality criteria

To evaluate the encoding quality, an algorithm was developed to check if an adequate number of letters and numbers were found, as well as if the mandatory characters were found. The fixed and variable characters of each model are as follows:

- Model A: the sequences “VAL”, “PET-PCR” and “L:PS” are fixed, while all digits are variable.
- Model B: the sequences “VAL” and “FAB” are fixed, while the last three characters of the first line are variable, and all digits are variable.

For the common characters to all bottles of the same model, conditional criteria were added to accept the equivalence of some characters that are similar and can be difficult to distinguish by OCR, which could be unnecessarily detrimental to the method. The equivalences allowed by the algorithm are shown in Table 2.

Table 2. Allowed equivalences

Allowed equivalence
V, v, U, u, W, N
A, R
P, F
B, E
S, 8, 9, 3
C, O

4. Image processing

In this work, two sequences of image processing methods were performed. The first sequence had an exploratory goal, serving as the basis for the construction of the final algorithm. From the results of the first sequence, a second sequence was proposed, aiming to improve some of the flaws found in the former. The sequences of treatments used are explained below.

4.1 Initial method

The first sequence tested applied the methods described in Figure 11. The sharpen method was used to increase the contrast of the digital image, applying the kernel in Figure 12 to the image. Figure 13 shows the result of using this technique. Note that the contrast between the characters and the background becomes more evident in the image (b). Then, a transformation is applied to map the image from the blue, green and red (BGR) color space to the grayscale, a step that significantly reduces the processing time required, as it reduces the amount of information that the image features. To perform the transformation, the “COLOR_BGR2GRAY” method available in the OpenCV library was used. This method uses Equation (6) to calculate the intensity of each grayscale pixel, according to the intensity of the BGR matrices in the original image.

$$\text{BGR to Gray: } Y = 0.114B + 0.587G + 0.299R \tag{6}$$

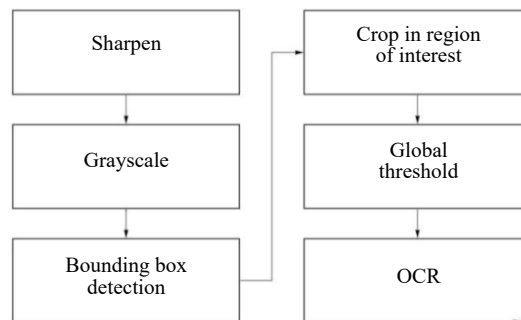


Figure 11. Initial sequence proposed

-1	-1	-1
-1	9	-1
-1	-1	-1

Figure 12. Kernel utilized for sharpen method

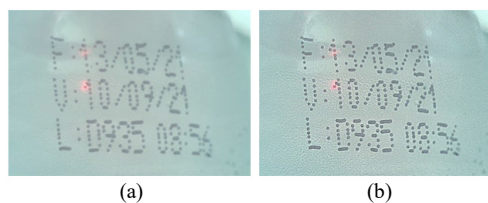


Figure 13. Result of the sharpen method: (a) Pre-processing; (b) Post-processing

Afterwards, a text detection operation is performed, aiming to obtain the coordinates of the bounding box surrounding the text. The performed tests showed that the method is able to correctly detect the location of the text, however, in some cases, more than one bounding box is found, or the coordinates found are very close to the characters at the edges, giving rise to problems in the text recognition step afterward.

With the coordinates of the text of interest, cropping is performed in this region, followed by the thresholding

operation, that is, the transformation of the image to black and white color space, seeking to obtain the maximum possible contrast between characters and image background. The tests performed in this sequence used a fixed threshold value (T) for the entire image, therefore the method is called “global threshold”, as exemplified in Figure 14 for three different values of T.

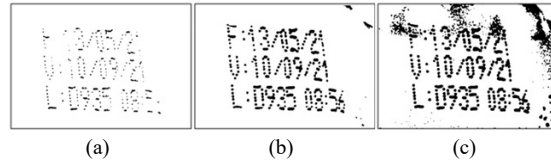


Figure 14. Effect of the threshold value in processing the image: (a) T = 140; (b) T = 160; (c) T = 180

When testing the quality of OCR of the images in Figure 5, the tests with T = 160 resulted in 100% of the characters being recognized correctly, while the tests with T = 140 failed to recognize some characters, such as the digits “1” and “6”. The tests with T = 180 failed to identify the characters “2” and “F” due to interference from dark regions. Therefore, for this sequence, the value of T = 160 was used. The results provided 33.3% accuracy for model A and 20.8% for model B. It was observed that, when using a fixed value of T in the samples, small variations of lighting caused large variations in text recognition, due to the presence of shadows and reflections of light by the bottle material, which resulted in low accuracy. Figure 15 exemplifies some cases in which it was not possible to correctly identify some of the characters.

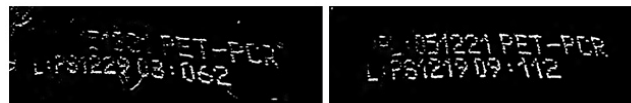


Figure 15. Some cases in which a fixed T value was not adequate to distinguish characters from the background

4.2 Improved method

The second sequence, displayed in Figure 16, maintains the use of sharpen methods and the transformation to grayscale. In addition to these, the operation of background removal, in order to improve the image quality in low-contrast regions, and denoise, responsible for eliminating noise based on the intensity of the neighboring pixels, were added. The success of both methods depends on the specified kernel size. For background removal, it must be consistent with the size of the text area in the image, and for denoise, with the size of the spaces normally found between characters. The effects of these four treatments can be seen in Figure 17.

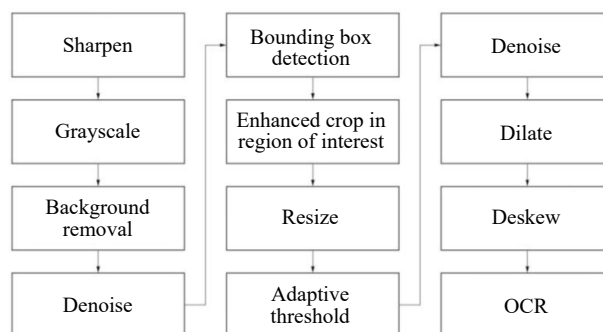


Figure 16. Improved sequence proposed

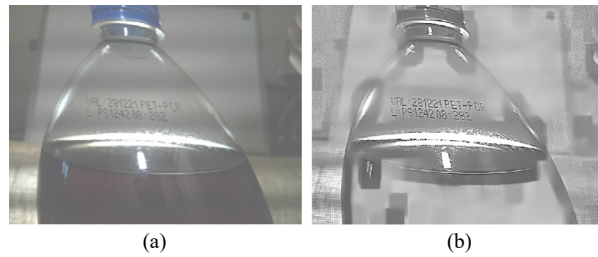


Figure 17. Result of the first four methods used: (a) Pre-processing (b) Processed with sharpen, grayscale, background removal and denoise

The cropping operation was adjusted to include a safety margin for the coordinates found, namely, 12 pixels for the vertical direction and 25 pixels for the horizontal direction. In addition, some optional parameters were used in the bounding box detection method, such as the minimum distance between two bounding boxes such that they are considered the same. With these parameters and refinements in the method, it was possible to obtain a single bounding box that correctly encompasses the text area in all images, as exemplified in Figure 18, where the rectangle in green is formed by the 4 coordinates found by the detect text method.

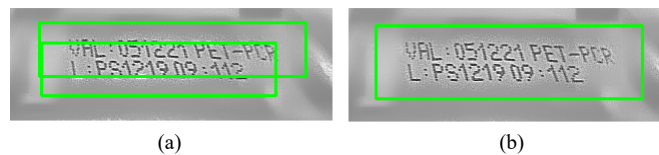


Figure 18. Result of the bounding box method: (a) No specified minimum distance between boxes; (b) With minimum distance between boxes

On the basis of text location, cropping is performed on the image, followed by a rescaling operation. This operation is employed because the size of the characters affects the precision of OCR methods. This effect can be seen in Figure 19, where the characters in green represent the values recognized by OCR Pytesseract. In the case where the original size image is employed, no characters are correctly identified. In the enlarged image, it is possible to identify 22 out of 25 characters correctly. This effect was especially important for Pytesseract, although the EasyOCR and Google Vision API OCRs also showed better results with rescaled images.

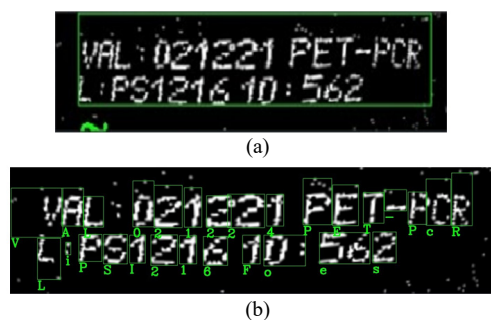


Figure 19. Effect of image size on text recognition: (a) Pre-processing; (b) 4x enlargement

The next treatment is performed solely on model B bottles, since the image containing the text is slightly arc-shaped, which reduces the performance of the OCR algorithms. Therefore, an arc distortion treatment is performed, and the correction angle was set as 30° in the distortion method. Figure 20 shows an example of a model B bottle after arc distortion.



Figure 20. Effect of arc distortion method: (a) Pre-processing; (b) Post-processing, distortion angle of 30°

From the achieved image, centered on the region of interest and with the size increased by a factor of 4, the adaptive threshold processing is performed. This treatment allows small changes in lighting, such as reflections on the bottle surface, or the occurrence of drops in the coding region, to have a lesser effect on the OCR result. In this method, a specific T value is calculated in order to maximize the contrast for each region of the image, according to the defined kernel size.

The low shutter speed required to capture the image without blurring causes some undesirable effects in all images, such as dark horizontal stripes throughout the picture. These regions are darker at the edges of the image, because of less incidence of light, since the lighting used was focused on the center of the photos. The use of adaptive thresholding also has an advantage regarding this effect, because if one of these dark stripes occurs directly over the coding region, separating the text of interest from the background becomes very difficult when using a single T value for the entire image. Figure 21(a) exemplifies the effect of the adaptive threshold. Finally, a denoise operation is performed again, as shown in Figure 21(b).

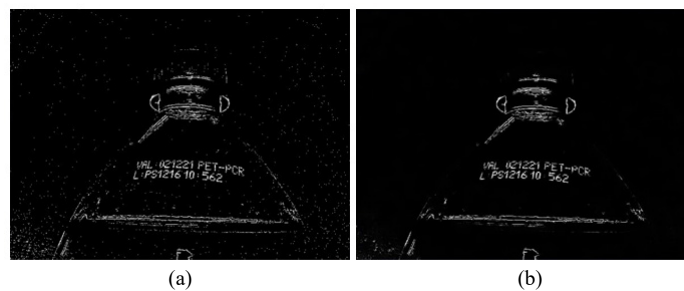


Figure 21. Application of adaptive threshold and denoise methods: (a) Adaptive threshold solely; (b) Adaptive threshold and denoise

Afterwards, the morphological dilation operation is performed. This operation is responsible for making some parts of the characters thicker, in addition to allowing the points that form the characters to connect to each other, thus improving the performance of the OCR algorithms. The effect of this operation is exemplified in Figure 22. Note that with the 5x5 kernel, parts of the characters begin to overlap, which negatively affects character recognition, as described in Zaafouri et al. [14].

Finally, angle correction is performed, as shown in Figure 23. The angle is calculated as the arc-tangent of the slope obtained by the height and width difference between the leftmost white pixel of the image and the rightmost white pixel. The pseudocode for all the operations presented in the improved method is shown in Figure 24, where all the OpenCV methods employed are visible.

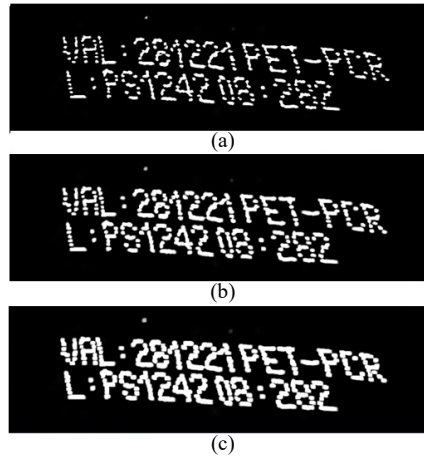


Figure 22. Example of the dilation method: (a) Pre-processing; (b) 3x3 kernel; (c) 5x5 kernel

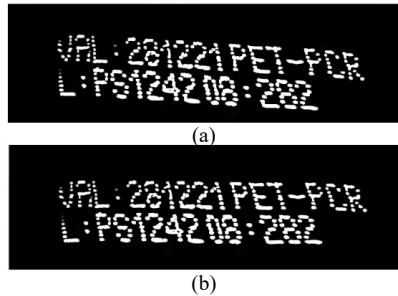


Figure 23. Example of the angle-correction method: (a) Pre-processing; (b) Corrected distortion

```

import easyocr
import cv2
from google_cloud import vision

img_original ← cv2.imread(pathToStoredImage)
img_sharpened ← cv2.filter2D(img_original, SharpenKernel)
img_grayscale ← cv2.cvtColor(img_sharpened, cv2.COLOR_BGR2GRAY)
structuring_element ← cv2.getStructuringElement(cv2.MORPH_RECT, (40,50))
background ← cv2.morphologyEx(img_grayscale, cv2.MORPH_DILATE, structuring_element)
img_filtered_background ← cv2.divide(img_grayscale, background, scale=255)
img_denoised ← cv2.fastNlMeansDenoising(img_filtered_background)

# Initializes the BoundingBox detection with EasyOCR
reader ← easyocr.Reader
result_set ← reader.readText(denoised_img)
x1, x2, y1, y2 ← result_set.getBoundingBox()
img_crop ← img_denoised[(y1-10):(y2+10), (x1-25):(x2+25)]
img_resized ← cv2.resize(img_crop, fx=4, fy=4, interpolation = cv2.INTER_CUBIC)
img_threshold ← cv2.adaptiveThreshold(img_resized, 255,
                                     cv2.ADAPTIVE_THRESH_GAUSSIAN_C, cv2.THRESH_BINARY_INV)
img_denoised ← cv2.fastNlMeansDenoising(img_threshold)
img_dilated ← cv2.dilate(img_denoised, kernel3x3, iterations = 1)
img_deskewed ← deskew(img_dilated)
cv2.imwrite(PathToStore, img_deskewed)

# Initializes the character recognition with Google Cloud API
image ← vision.Image(PathToImageDeskewed)
text ← text_detection(image)

```

Figure 24. Pseudocode for the improved method. The methods beginning with “cv2” are the ones from the OpenCV library

4.3 Real-time inspection

The experiments presented in the earlier sections were done in two separate parts. First, the bottles passed through the system and the images were collected, and after, the images were downloaded to a personal computer for

experimenting with the image processing techniques, where the sequences in 4.1 and 4.2 were developed. Starting from the algorithm that runs on the Raspberry, responsible for image acquisition and the algorithm that runs on the personal computer, a modified version of both was developed to include the use of web sockets to connect the Raspberry Pi (client) to the personal computer (server). The modifications only concern the use of web sockets, nothing else changed. The final result can be seen as pseudocode in Figure 25. The goal was to test how long would it take to process each image in real-time. The processing took 2 seconds on average, considering all the samples utilized in the study, corresponding to 30 bottles per minute. In a real industry, this system would not be capable of analyzing every bottle, due to its high processing time. However, processing time could be reduced if the OCR step were built into the main processing algorithm, instead of calling an external API. Furthermore, codification problems usually keep happening until someone takes action manually, to replenish ink or clean the printer die that may be clogged. As long as the non-compliance is not a one-off, the system can still be useful, even not evaluating all bottles.

Client side (Raspberry Pi)	Server side (personal computer)
	<pre>import socket server ← socket.socket() server.bind((IP, PORT)) server.listen() BUFFER_SIZE = 4096 While true: client ← server.accept()</pre>
<pre>import socket client ← socket.socket() client.connect((IP, PORT)) BUFFER_SIZE = 4096 While true: <proceeds with image capture code> file_data ← File.read(PathToCapturedImage, BUFFER_SIZE) While file_data: client.send(file_data) file_data ← File.read(BUFFER_SIZE)</pre>	
	<pre>recv_data ← client.recv(BUFFER_SIZE) While recv_data: File.write(recv_data) recv_data ← client.recv(BUFFER_SIZE) <proceeds with processing code> text ← text_detect(PathToStoredImage) conformity ← verifyConformity(text) msg ← conformity client.send(msg)</pre>
recv_data ← client.recv(BUFFER_SIZE)	

Figure 25. Pseudocode for real-time inspection. The columns represent on which side each code is running, and each line presents the sequence in which each action occurs

5. Results and discussion

According to Figure 26, It can be seen that the classification of model A was more accurate than that of model B, and both obtained a specificity of 100%, that is, no non-conforming bottle was classified as conforming. This is the main goal of product line inspection, since false negative results may not issue an immediate command to stop the line, but trigger further analysis instead. The incorrect classification of the two bottles of model A was due to the non-detection of the characters positioned at the end of the first line. Ambient lighting was not enough to provide adequate

contrast for these characters, which made the adaptive threshold method, combined with the subsequent denoise method consider part of these characters as the background of the image. Figure 27 shows the images of model A bottles that were incorrectly classified by the algorithm. Note that in (b) it was not possible to recognize the character “R”, even allowing its equivalence to “P”. This problem concerning the characters at the edges could be solved by using a camera positioned at a greater distance. However, for this, superior image quality would also be necessary, because the greater the distance between the camera and the text, the lower the resolution of the area of interest. Another solution is to improve lighting conditions, reduce the occurrence of shadows, and use diffuse light, as suggested by Silva [13].

		Model A		Model B	
		Expected classification		Expected classification	
		P	N	P	N
Performed classification	P	20	0	11	0
	N	2	9	4	9

Accuracy	91,7%
Sensitivity	87%
Specificity	100%

Accuracy	83,3%
Sensitivity	73%
Specificity	100%

Figure 26. Results obtained for both bottle models

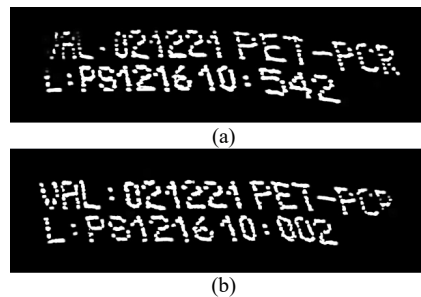


Figure 27. Examples of misclassification in the algorithm: (a) Unrecognized “V”; (b) Unrecognized “R”

All the non-compliance scenarios were correctly classified as “negative”. The OCR algorithms proved to be extremely sensitive to small changes in angle and lighting. Since all characters in model A bottle are fixed, any slight blur is sufficient to classify as non-conforming. In the set of images of this model, no digit was recognized erroneously, although there is a chance that a bad coding will cause a false positive, that is, the OCR does not recognize the correct digit, but recognizes any other digit, thus satisfying the compliance conditions, since all digits are variable. Nevertheless, additional criteria, such as the progression of expiration dates, could be used.

Misclassified images of bottle model B had similar causes, as well as occurrences where a digit was recognized as a character. The proposed rules allowed the exchange of some characters for others, including digits, but the exchange of any digits for characters was not allowed. In the second line of text of model B, which has 10 digits, a digit “0” was recognized as the letter “Q”, as well as the digit “7” by the letter “T”, as shown in Figure 28.



Figure 28. Examples of misclassification in the algorithm: (a) Exchange of “0” by “Q”; (b) Exchange of “7” by “T”

In addition to these, two other images of model B were incorrectly classified, both from the same bottle, as shown in Figure 29. Note that the letter “F” is not complete, as well as the characters “1” and “L” that appear in sequence. This case clearly shows the influence of the chosen kernel size. The distance between the top of two characters “L” or between the bottom of two digits “1” is greater than between other pairs of characters. Therefore, a possible solution is to increase the size of the denoise kernel, which would increase the distance required between pixels for them to be considered noise.



Figure 29. Examples of misclassification in the algorithm

The acceptance of the equivalence of some characters proved to be fundamental, especially the exchange of “V” for “v”, as well as the angle correction, proposed by Zaafouri et al. [14]. The dilation technique proposed by Hosozawa et al. [15] was also very helpful to achieve acceptable results. The adaptive threshold combined with denoise proved to be a reasonable solution to the thresholding problem.

Despite the position of the camera and sensor being fixed, the captured images presented small variations in the position of the bottle. This may have been caused by the low processing power of the Raspberry Pi used, as discussed in Machado [19]. It was observed that in cases where the bottle is not captured at the center of the image, the first or last character of the coding is misrecognized more often, probably due to the lower lighting available in these regions. A solution that minimizes this problem is to develop a routine that checks the position of the bottle before processing, as suggested by Sharma et al. [20] and create a cabin where diffuse light surrounds the bottle [13]. The detection of bounding boxes was successfully performed on all images, even when the bottle is not centered, both for the EasyOCR tool and for the Google Cloud Vision API.

6. Conclusions

Tests with a low-cost apparatus for online coding inspection were carried out with two models of PET bottles, with specimens that present non-conformities. Two treatment sequence proposals were defined, the first one obtaining 33.3% and 20.8% accuracy for models A and B, respectively, and the second (improved) proposal obtaining 91.7% and 83.3%, respectively. In addition, a rule for conformity assessment was proposed, which allows the equivalence of some specific characters, and an algorithm capable of processing images in real-time using web sockets was developed.

Although the accuracy was not excellent, the proposed low-cost system is able to obtain a specificity of 100% in the tested speed of the bottles. Thus, this system could help to reduce the chances that a coding problem is noticed late, causing non-quality costs, or that this bottle reaches the market, being liable to pose risks to the consumer, generating fines for the company, and damage the company public image. However, some adjustments are necessary for the

sequence of treatment steps, which may even depend on the type of bottle. In addition, the tested bottle speeds are below those practiced normally in the industry, which is an important factor in the functioning of the quality inspection system. Future works might use a higher quality camera and try to replicate the same experiment presented, but with higher conveyor speeds, as well as use a more robust microprocessor board with better technical specifications, since most of the issues were caused by the low image quality. A cabin for proper lighting could also improve the results. As it is, this system could only be employed in industries where the conveyor speeds are equal to or below 0.40 m/s. The limiting factor for the conveyor speed was the low incidence of light when using less than 800 μ s for the shutter speed, which is necessary to avoid the blurring effect in the acquired images.

The sequence of image processing methods developed in this study could also be useful to identify other types of information on packages, not just the expiry date and batch numbers, and can be tested on other types of packaging as well. The development of a build-in OCR algorithm could reduce the processing time, which was on average 2 s.

In order to avoid human verification having to be triggered frequently by the automatic inspection system, it would be possible to implement resilience algorithms against false negatives, that is, if there is a detection of non-compliance, one could wait for the confirmation of the non-compliance situation in subsequent images until an alert is triggered. In addition, several other types of batch analysis would be possible, for example, if the coding model adopted records the packaging time on the bottles, verification of the time sequence could be analyzed, etc.

Conflict of interest statement

There is no conflict of interest for this study.

References

- [1] National Health Surveillance Agency. Resolution-RDC No. 259. *Technical regulation for labeling packaged foods*. Brazil: Ministry of Health; 2002.
- [2] Presidency of the Republic. Law No. 6437. *Configures infringements to the federal health legislation, establishes the respective sanctions, and gives other provisions*. Brazil: Presidency of the Republic; 1977.
- [3] Ferreira MC. *Quality systems in the production of soft drinks based on consumers' satisfaction*. Master thesis. Federal University of Rio de Janeiro and Uniminas; 2010.
- [4] KEYENCE. *Vision Systems*. <http://www.keyence.com.br/products/vision/vision-sys/> [Accessed 8th May 2022].
- [5] KRONES. *Inspection Technology*. https://www.krones.com/en/products/machines/inspection-technology.php?page=1&searchtext=&filter%5B4%5D%5B4_6%5D=4_6&filter%5B1%5D%5B%5D=all&filter%5B5%5D%5B%5D=all&searchtext=&searchtextold= [Accessed 25th April 2022].
- [6] COGNEX. *Date/Lot Code Inspection*. <https://www.cognex.com/industries/food-and-beverage/packaging-inspection/date-lot-code-inspection> [Accessed 25th April 2022].
- [7] Equipamentos McPack. *Camera Inspection System*. <https://www.mcpack.com.br/sistema-de-inspecao-por-camera/> [Accessed 10th May 2022].
- [8] KPM Analytics. *Vision & Thermal*. <https://www.kpmanalytics.com/product-categories/vision-thermal> [Accessed 10th May 2022].
- [9] Smart Vison Lights. *Food/Beverage*. <https://smartvisionlights.com/resources/applications/food-beverage/> [Accessed 12th May 2022].
- [10] Pedrini H, Schwartz WR. *Digital image analysis: principles, algorithms, and applications*. São Paulo, Brazil: Thomson Learning Edições Ltda; 2007.
- [11] Gonzalez RC, Woods RE. *Digital image processing*. 3rd ed. London, United Kingdom: Pearson; 2008.
- [12] Coates C, Juvan-Beaulieu I. *Rolling shutter vs global shutter sCMOS camera mode*. <https://andor.oxinst.com/learning/view/article/rolling-and-global-shutter> [Accessed 28th July 2022].
- [13] Silva AJDJ. Parameter adjustment methodology for acquiring good quality images using a commercial semi-professional camera. *Revista Sítio Novo*. 2020; 4(4): 202-216. <https://sitionovo.ifto.edu.br/index.php/sitionovo/article/download/723/262>
- [14] Zaafouri A, Sayadi M, Fnaiech F. A vision approach for expiry date recognition using stretched gabor features. *The International Arab Journal of Information Technology*. 2015; 12(5): 448-455. <https://ccis2k.org/iajit/PDF/Vol%20>

12,%20No.%205/6700.pdf

- [15] Hosozawa K, Wijaya RH, Linh TD, Seya H, Arai M, Maekawa T, et al. Recognition of expiration dates written on food packages with open source OCR. *International Journal of Computer Theory and Engineering*. 2018; 10(5): 170-174. <http://www.ijcte.org/vol10/1220-CM2001.pdf>
- [16] Gong L, Yu M, Duan W, Ye X, Gudmundsson K, Swainson M. A novel camera based approach for automatic expiry date detection and recognition on food packages. In: Iliadis L, Maglogiannis I, Plagianakos V. (eds.) *Artificial intelligence applications and innovations. ALAI 2018. IFIP advances in information and communication technology, vol 519*. Cham: Springer; 2018. p.133-142.
- [17] Zhou X, Yao C, Wen H, Wang Y, Zhou S, He W, et al. EAST: An efficient and accurate scene text detector. In: O'Conner L. (ed.) *30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*. Piscataway, New Jersey: IEEE Computer Society; 2017. p.2642-2651.
- [18] Cunha RC. *Comparative study on optical character recognition tools*. Undergraduate thesis. Instituto Federal de Minas Gerais; 2018.
- [19] Machado UT. *Automatic inspection of bottles using computer vision in embedded system*. Undergraduate thesis. University of Caxias do Sul; 2019.
- [20] Sharma S, Krupa KV, Gandhi R, Jain A, Shah N. Empty and filled bottle inspection system. *ICTACT Journal on Image and Video Processing*. 2015; 6(2): 1122-1126. http://ictactjournals.in/paper/IJIVP_V6_I2_paper_3_1122_1126.pdf
- [21] Upton E. *Raspberry Pi 4 on sale now from \$35*. <https://www.raspberrypi.com/news/raspberry-pi-4-on-sale-now-from-35/> [Accessed 4th June 2022].