COMPUTER VISION IN INDUSTRY

Dobrivoje Đurić*,
Vladimir Matić

Singidunum University,
Belgrade, Serbia

Abstract:
This paper presents a system of computer vision designed to be used in industry for product classification. The system is based on the TensorFlow and Keras software libraries and the Python programming language. In the production process, uniform cylindrical parts reach the conveyor designed so that each part can be in one of two possible positions. The computer vision system detects the exact position of the part, or its orientation, and this information is further used during the transportation process. The purpose of the procedure is for all parts to take the same orientation in relation to the production line.

Keywords:
industry, computer vision, machine learning, Keras, TensorFlow, Python, RaspberryPi.

1. INTRODUCTION

The classification presented in this paper refers to one type of product. In this case, the same type and size of the metal screws were used. The aim of the classification is to determine the orientation of each part in relation to the production line. In this case, the part can be in one of two possible positions and for that reason, two classes are used. One position implies that the top of the screw is pointed in the direction of production, and the other position means that the top faces the opposite direction. The result of the classification is used so that all the parts take the same orientation. In addition, there is also a third class related to all the cases when the object is not properly detected. For example, when the production part is not in the required position or if more than one part is present at a time.

The solution of the computer vision system is based on the programming language Python and the libraries Keras and TensorFlow. This paper also introduces the testing equipment used to verify the functionality of computer vision in real conditions. The testing equipment includes the mini-computer Raspberry Pi with a corresponding camera and a transport system prototype, which is used to shift the parts in order to

Correspondence:
Dobrivoje Đurić

e-mail:
dobrivoje.djuric.17@singimail.rs

DOI: 10.15308/Sinteza-2019-439-444

test the accuracy and reproducibility of the classification. In order to reduce the classification time, a simple algorithm is designed and executed fast enough on the testing equipment.

By executing the program on the testing equipment, the machine learning algorithm, which was trained on another computer, is loaded first. Following this, a program loop continuously reads the output of the camera and performs the classification. This part of the program also includes functions related to the automatic control of motors, which are used for transportation of the detected elements.

The low-cost solar tracking system [1], based on an open source hardware and machine learning algorithm, was an inspiration to use the Raspberry Pi.

The used literature can be divided into two sections, one dealing with the computer vision [2], [3] and [4], while the other section deals with artificial neural networks on which the machine learning algorithm is based [5] and [6].

2. PROCESS AND TEST EQUIPMENT DESCRIPTION

During the production process, uniform cylindrical parts reach the appropriate position one at a time in such a way that they take one of two possible orientations. The computer vision system detects the exact orientation of the part and this information is further used to rotate and shift it to the appropriate location. The aim of the procedure is for all the parts to assume the same orientation in relation to the production line. Instead of the real parts, metal screws were used for the purpose of testing.

The position that a free falling screw assumes can be compared to the V-profile considering its appearance. Due to its physical design, the metal screw always assumes a horizontal position, whereby its head can be turned left or right when observed from the position of the operator. The answer to the question of the direction that the head of the screw can face can be obtained using the computer vision.

A camera is placed above the position used for the rotation of the production part. During the image processing, the classification indicates whether the head of the screw is faced to the left or to the right. Based on the classification results, the screw can rotate 180°, if necessary. The aim is for all the parts to assume the same orientation, followed by transport according to the established procedures of manufacturing. The following illustration shows the testing equipment. In order to reduce the background influence, the V-profile is in black. Using these simple images the classification is more accurate, which is important because of the simple machine learning algorithm. The motor (M1) is used to rotate the parts and the motor (M2) for their transfer to the next position. Incremental encoders E1 and E2 are used for positioning.

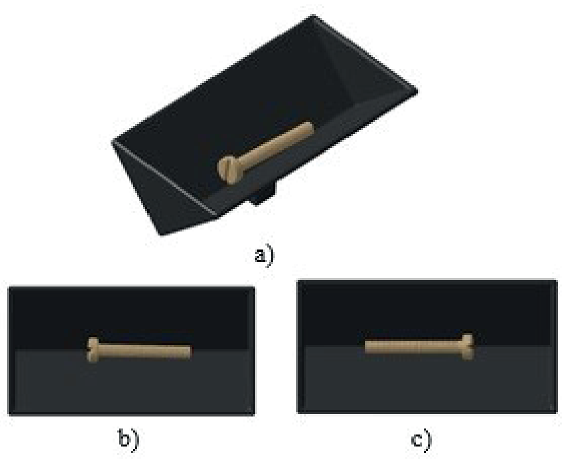

Figure 1. Illustration of product handler: a) isometric view, b) part is oriented to the right, c) part is oriented to the left in relation to the observer

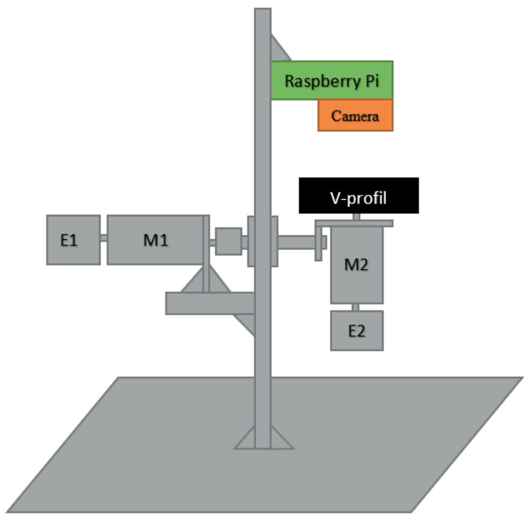

Figure 2. Illustration of testing equipment

Raspberry Pi computer is equipped with a camera and placed just above the position where the production part is located. In addition to the computer vision, Raspberry Pi is also used to control the motors M1 and M2. It is equipped with a motor control card placed in the same casing as the computer. This testing equipment creates the images of the data set used in developing the machine learning algorithm.

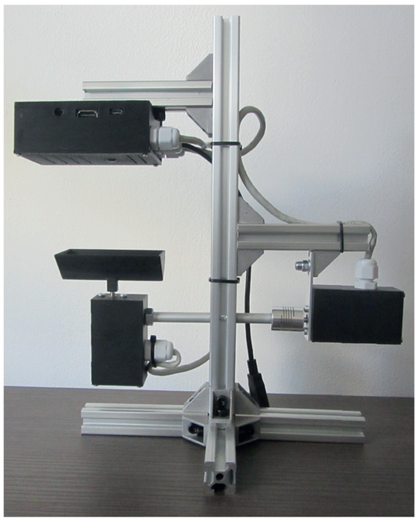

Figure 3. Test equipment

3. CLASSIFICATION PROCEDURE

The classification procedure and material transport are fully automated. The program runs on Raspberry Pi, which is an integral part of the testing equipment. When executing the program, a pre-trained model of the artificial neural network is loaded, and then a program loop starts in which the classification procedure is continuously executed. If the result of the classification indicates that the production part is not well oriented, it has to be rotated 180° and then moved to the next position. Following this, the V-profile is ready to accept the new part. If the result of the classification indicates that the products are well oriented, they move to the next position without rotation.

When the object is not detected, a message is displayed as a warning. In this case it is necessary to manually execute the repositioning of the part or transfer it to the next position. The entire procedure is performed under usual room lighting and can be described through the following steps:

1. The production part reaches the position for inspection. In our case, the metal screw is placed manually.
2. The classification indicates whether the part is well-oriented or it needs to be rotated 180°.
3. The rotation of the production part, if necessary, is achieved using the M1 motor, shown in Figure 2.
4. The product is moved to the next position by changing the slope of the V-profile using the M2 motor. When a certain slope is reached, a free-falling production part leaves the V-profile.
5. At the end of the process, the V-profile returns to its basic position.

4. MACHINE LEARNING ALGORITHM

The machine learning algorithm is based on a convolutional neural network. The input data are color images with a 96 x 96 pixels resolution. Initially, Google's MobileNet algorithm was used. Good results were achieved in terms of accuracy, but due to the complexity of the algorithm and low processing capabilities, it was necessary to create a simpler algorithm.

A TensorFlow library was used to create a new algorithm. TensorFlow was released by Google and it can be used directly or by using wrapper libraries such as Keras. Keras is a library also developed by Google in order to simplify the development of deep learning models [5].

The algorithm architecture is defined in Table 1.

Table 1. Algorithm architecture

Layer (activation)	Filter Shape	Output Size
Conv2D ('relu')	3 x 3 x 32	94 x 94 x 32
Conv2D ('relu')	3 x 3 x 32	92 x 92 x 32
Conv2D ('relu')	3 x 3 x 64	90 x 90 x 64
Average Pooling 2D	2 x 2	45 x 45 x 64
Dropout	0.2	45 x 45 x 64
Flatten		129600
Dense ('relu')		12
Dense ('softmax')	Classifier	3

The accuracy and loss achieved during the training are shown in Figure 4 and Figure 5.

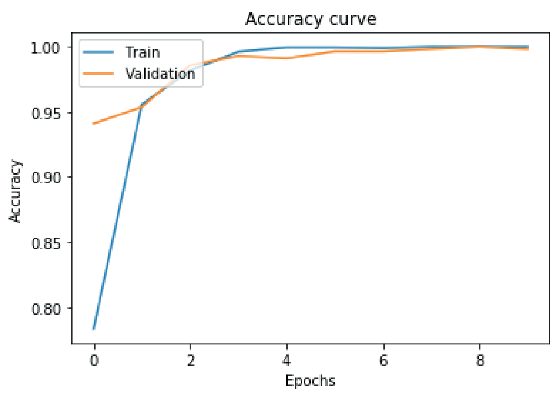

Figure 4. Model accuracy

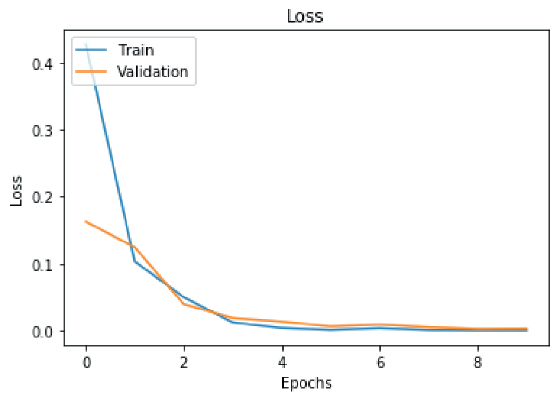

Figure 5. Model loss

Figure 6. Showing a sample of dataset images used for algorithm training.

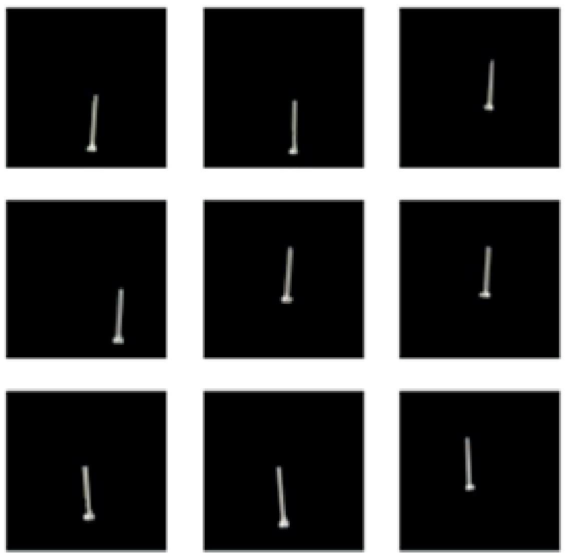

Figure 6. Sample of dataset images

5. IMAGE PREPROCESSING

The testing equipment initially made a small number of images for the data set, which were then multiplied by Keras to get more images for the training. Keras supports the *ImageDataGenerator* feature, which can create new modified images from the original ones. The changes include random rotations at an angle within a defined range, zooming, and shifting in horizontal and vertical direction. In this way, the initial small number of images produced a sufficiently large data set for model training. The class which indicates that an object is not recognized, or undefined, contains images of the empty V-profile.

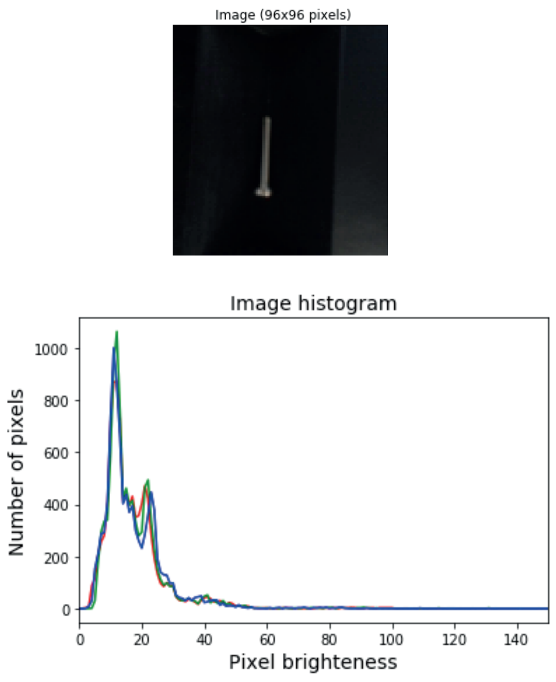

Figure 7. Screw (up) and its histogram (down)

In order to remove the background influence, which is in dark color, the pixel values are changed. All pixels values below the defined threshold are set to zero. The threshold is defined using a color histogram. Since all the images are similar, by analyzing the histogram for one image we can determine the common threshold for the entire data set.

The histogram in Figure 7. shows that the largest number of pixels has the value of RGB components below 50. This pertains to the dark background.

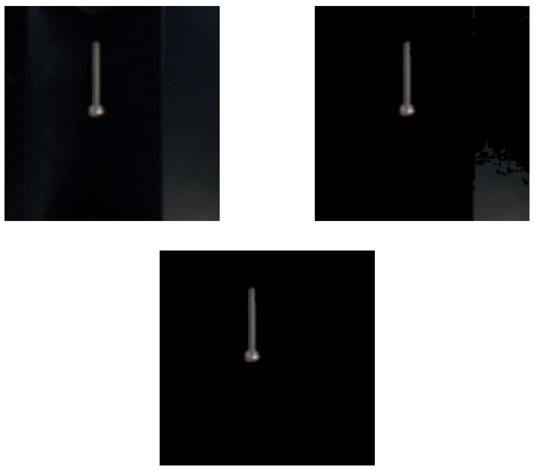

Figure 8. Image preprocessing

Figure 8. shows the image preprocessing results with the following explanation: a) original image, b) the result of the operation in which the pixel values are set to zero, if they were previously below the defined threshold, and c) the result of masking the image region. This kind of image preprocessing can be applied in industrially controlled conditions.

6. AUTOMATIC MATERIAL TRANSPORT

Manufacturing processes at modern industrial plants are based on a high level of automation. These processes are controlled by PLC (Programmable Logic Controller) which, according to the information obtained from sensors and according to the operating parameters, generates reference values used to control the actuators.

In order to increase the level of automation, computer vision is used to provide additional information about the process. In our case, the testing equipment does not include the PLC. Two motors with encoders are used for the transport of materials. Motors are controlled by the same computer, which is also used for computer vision.

A control card, located in the same casing with the Raspberry Pi computer and connected to it via the I2C interface, is used to drive the motors.

Two identical motors with gearboxes and incremental encoders are in use. Each incremental encoder is connected to the appropriate GPIO inputs of the Raspberry Pi computer, which are configured to initiate a program interrupt when detecting the front edge of the encoder pulse. The interruption executes a program code detecting the direction of movement and calculating the current position. The positioning is based on the number of encoder pulses.

The classification applies to the part of the program that is executed in an infinite loop. If three consecutive classifications are the same, the transport procedure is initiated. The procedure has two steps, the first is rotating 180° if necessary, and then shifting to the next position. The program changes the direction of the rotation in order to minimize the positioning error. If the screw has been rotated 180° in the clockwise direction, the next time the rotation will be counterclockwise. If the rotation is performed only in one direction, there is a possibility that small errors will accumulate over time, which would increase positioning errors after a large number of cycles.

7. CONCLUSION

The testing of the model in real time using the testing equipment achieved a satisfying accuracy and repeatability of the classification. Color images were used initially, and the results did not differ significantly from the gray scale images. Pre-processed color images have a dark background and a bright object, practically a small difference compared to gray scale images. The algorithm execution time is about 250 ms.

Image preprocessing in real time has a positive impact on the accuracy of the classification. By preprocessing the image, the background details are removed, which makes the image simpler. Because of using images in small resolution, program execution does not consume much time.

Lighting has a big impact, especially in terms of reflection. During the testing, the best results were achieved using regular room lighting. When the testing equipment was directly exposed to the source of light, there were intensive reflections on the surface of the object and the surrounding plastic used in the manufacture of the testing equipment. This considerably changed the results and in some cases, the classification could not be performed at all. On the other hand, the impact of the shadow was not observed. This is primarily because of the specific design of the device and the dark background.

In order to reduce the problem of illumination, images for the data set were taken under different levels of illumination. The intention was to resolve the issue by training the model in different conditions, which did provide certain improvement, but the problem was not resolved completely. The algorithm itself was also limiting because of its simplicity. When using a data set with complex images, more complex algorithms should be used.

Simple hardware produces satisfactory results. This was achieved using simple images and a small number of classes. If the number of classes is larger or images become more complex, more complex algorithms that require larger hardware capabilities would have to be used.

The tests have shown that more than one class for undefined states would be preferable. When one class includes different objects, it becomes complex. The best results were achieved when the class included only the images with the background of the object. In order to recognize the other irregular states, new classes must be added.

REFERENCES

[1] J. A. Carballo, J. Bonilla, L. Roca and M. Berenguel, "Low-cost solar tracking system based on open source hardware," *Solar Energy*, vol. 174, pp. 826-836, 2018.

[2] J. Brownlee, Deep Learning With Python, Develop Deep Learning Models on Theano and TensorFlow Using Keras, 2018.

[3] J. Minichino and J. Howse, Learning OpenCV 3 Computer Vision with Python - Second Edition, Packt Publishing, 2015.

[4] I. Zafar, G. Tzanidou, R. Burton and N. Patel, Hands-On Convolutional Neural Networks with TensorFlow, Birmingham: Packt Publishing Ltd, 2018.

[5] K. Dawson-Howe, A Practical Introduction to Computer Vision with OpenCV, Chichester: Wiley, 2014.

[6] L. Shapiro and G. Stockman, Computer Vision, Upper Saddle River, NJ: Prentice Hall, 2000.