Controlling A Robotic Arm Using Handwritten Digit Recognition Software

Ali Cetinkaya*‡, Onur Ozturk**, Ali Okatan***

* Technology Transfer Office, Istanbul Gelisim University, Avcılar, Istanbul, Turkey.

** School of Management, Faculty of Engineering, University College London (UCL), Euston, London, UK.

*** Department of Computer Engineering, Faculty of Engineering, Istanbul Gelisim University, Avcılar, Istanbul, Turkey.

(alcetinkaya@gelisim.edu.tr, onur.ozturk.16@ucl.ac.uk, aokatan@gelisim.edu.tr)

‡ Corresponding Author: Ali Cetinkaya, Technology Transfer Office, Istanbul Gelisim University, Avcılar, Istanbul, Turkey. Tel: +90 212 422 70 00 / 7187. alcetinkaya@gelisim.edu.tr

Received: 21.09.2018 Accepted:30.1.2019

Abstract- Repetitive tasks in the manufacturing industry is becoming more and more commonplace. The ability to write down a number set and operate the robot using that number set could increase the productivity in the manufacturing industry. For this purpose, our team came up with a robotic application which uses MNIST data set provided by Tensor flow to employ deep learning to identify handwritten digits.

The system is equipped with a robotic arm, where an electromagnet is placed on top of the robotic arm. The movement of the robotic arm is triggered via the recognition of handwritten digits using the MNIST data set. The real time image is captured via an external webcam. This robot was designed as a prototype to reduce repetitive tasks conducted by humans.

Keywords MNIST Handwritten Digit Recognition, Deep Learning, Embedded System Robotic Arm Control

1. Introduction

The MNIST dataset was created using two datasets from the US National Institute of Standards and Technology (NIST). Training data set includes handwritten digits from approximately 250 people, where half of these people are high school students and the other half is the employees of the Census Bureau. The data set consists of 60,000 training digits and 10,000 test digits. Having such a huge number of data allows the software to identify handwritten digits of many types of handwriting. Furthermore, this allows our system to be used by many people due to the inclusiveness of the training and test data sets [1].

Keras is a high-level neural networks API, written in Python and capable of running on top of TensorFlow, CNTK, or Theano [2]. Keras was developed with ease of experimentation and speed in mind, therefore it is highly favoured by researchers. In our system, we used Keras API to create a 7-layer Convolution Neural Network (CNN) [3][4]. The layers were convolution, pooling, convolution, convolution, pooling, activation and identification respectively. The compilation of 15 epochs, which gives out 99.4% accuracy, takes around 40 minutes per epoch on a CPU-only computer.

Today, developments in robotics are concentrated in a number of areas. These areas are mainly about the imaging systems of robots, artificial intelligence and machine learning. In robotic imaging systems, it is the process of capturing and defining the images of objects and finding the coordinates of the specified objects. This is the process of performing the action of the robot according to coordinates after the defined movement [10, 12]. In terms of human health, there are situations where it is not possible to work in dangerous environments. For this purpose, the robot arm was operated by the sensors placed on the human arm [11]. In the studies developed on image processing, image classification and image extraction are the most important processes. At this point, that accuracy affects the success of the study [13]. The images taken from the camera define the color and shape of the object. The system applies the center-based calculation, filtering and color segmentation algorithm to locate the target and the position of the robot arm [14].

The image recognition software was designed in OpenCV3, whereas the embedded system was designed in Arduino. The hardware system is shown in Fig. 1.

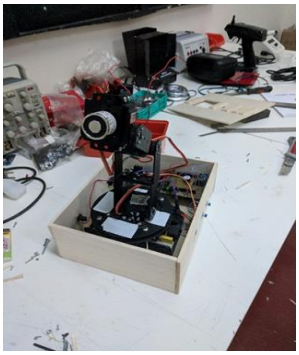

Fig. 1. Hardware layout of the system

Robot Kinematics is a geometrical study of the structure of a robot and independent of dynamic effects such as force and torque. The results of these kinematic investigations are obtained from the robots regarding the position, speed and acceleration of the joints and the final limb. The analysis of the robot requires knowledge of many branches of science, such as Mathematics, Mechanics and electronics [15, 16].

The robotic system is capable of operating for pre-defined actions in Arduino. There are four actions defined in Arduino being: moving the arm forwards, turning on the electromagnet, moving the arm backwards and turning off the electromagnet. The numbers to be identified to get into action for each operation is 2, 3, 4 and 5 respectively.

2. Hardware

Power source 12V is the required voltage for the system to work. The city grid provides the robot with 220V of electricity, therefore the power source is responsible for converting the 220V to 12V for the robot to work.

Servo engines present in the system to accurately control the robotic arm. These servo engines are capable of moving between 0 – 180 degrees however, during this experiment, the angles never exceeded 30 and 150 respectively, in order not to damage the servos. Furthermore, having three servos present in the robot alleviates the need for using the servo engines at their maximum capacity; using three servos in parallel gives the arm extended movement space.

The servo engine controller receives the movement signals from the Arduino Mega present in the system. Furthermore, servo engine controller receives 5V from regulator in order to move the servos. Both the signals from Arduino Mega and regulator are then used to control the robotic arm.

The purpose of the relay in the system to control the status of the electromagnet. This is supported by the signals received from Arduino Mega according to user input. The embedded timer in Arduino Mega excels as compared to many other Arduino boards. The system runs with three servo engines therefore the timing between the servo engines to move them to the desired angles was important and the library of Arduino Mega was the most adequate for this operation.

As previously mentioned, the system receives 12V of energy, however servos require 5V of energy to work. The regulator is responsible for converting 12V coming into the system to 5V to feed them into the servos for operation.

The electromagnet works with 12V of energy. Given the size of this prototype robot, an electromagnet with a maximum pulling force of 50N (5 kilograms) was used in order to avoid the robot from tipping over. The electromagnet can be seen in Fig. 2. along with the closed system.

Fig. 2. Fully working, closed system.

3. Embedded System and Software Algorithm's

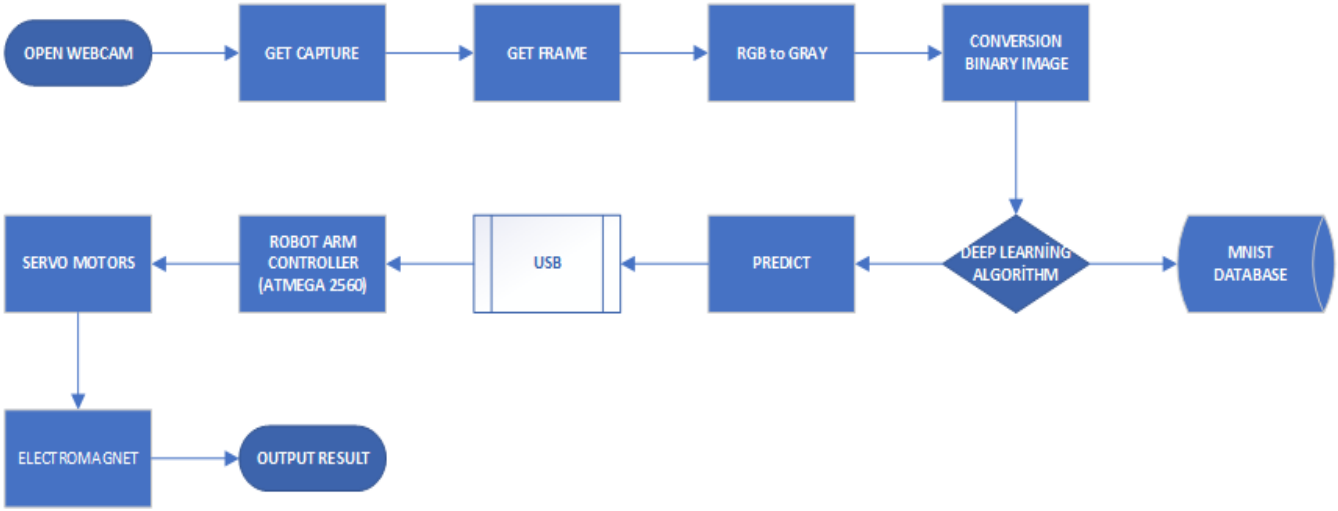

Fig. 3 Flow diagram of algorithm step.

Defining mathematical figures is an important problem. In this study using the deep learning algorithm and MNIST data set consists of 5 main stages. These are: pre-processing on the image, feature extraction, deep learning algorithm, MNIST data set and microcontroller control. The handwritten robotic arm control architecture is shown in Fig. 3.

3.1. Embedded System Control

The embedded system was developed with atmel atmega2560 microcontroller using embedded c for programming. The embedded system was designed Arduino Mega code was produced in the Arduino IDE. The Arduino software is responsible for operating the robot. This is done via data received from the MNIST Handwritten Digit Recognition Software, which is written in Python. Arduino was chosen language due to the availability robotic libraries for ease of control [5].

In the first segment of the Arduino code, the variables were declared. In this segment, there are pin declarations, servo declarations and servo angle declarations.

In the setup () segment of the code, the servos were connected to their respective pins. Furthermore, the initiation angles of the three servos were declared. The initiation angles are 90 degrees for each servo. These values help the user understand that the system has initiated because there is no user input that can keep the servos at 90 degrees. After setting the servos to their angles, the serial port connection was opened. The user can understand the connection has been established via the three-note music played by the buzzer present in the system. This serial connection allows the Arduino code to interact with the signals received from the Python code.

Finally, in the loop () section of the code, a switch case was created. The Python code sends off letters ranging from ‘a’ to ‘e’, and each letter is assigned to an action in the robot. The switch case is responsible for controlling which action is triggered according to the input received from Arduino.

3.2. MNIST Handwritten Digit Recognition Software

The Python digit recognition software works using OpenCV and Serial Port [6]. OpenCV proves rather useful when working with computer vision and image recognition due to wide variety of supported libraries and conducted experiments. “OpenCV has more than 47 thousand people user community and estimated number of downloads exceeding 14 million. Usage ranges from interactive art, to mines inspection, stitching maps on the web or through advanced robotics.” [6].

When the software is initiated, a video stream from the camera of the PC (or webcam, in this system) is recorded and presented on the screen. Even though the entire video stream is presented on the screen, the part of the screen which processes the information is marked with a blue square. Instead of detecting the entire video stream, the blue square was chosen to be area of interest due to efficiency purposes. The program works more efficiently in a smaller area. The user must present his/her handwritten in this square for the software to process the information. Once the handwritten is presented in this square, the software is responsible for counting the empty spaces between the handwritten digits. For example, if the software is counting 3 defects, there would be 4 handwritten present in the square, which would trigger an action to the servos.

The MNIST database consists of 70,000 samples of handwritten digits. Each of them is grayscale image of size 28px x 28px. The software initiates by capturing the presented handwritten digit from the external camera. Furthermore, a label on the top right of this extracted digit is placed to indicate which number is predicted for this extracted image. Then, the extracted image is translated into grayscale. The purpose of having a greyscale image is because OpenCV has built in libraries that work with grayscale image [6].

The process is to predict the handwritten digit in this grayscale image. This grayscale image is then fed into the CNN. The first layer type of layer, which is convolution, is responsible for filtering out the images to increase processing speed. The second type of layer is pooling, which is present to reduce the risks of overfitting. This type of layer reduces the parameters to be learnt and in return reduces noise within the image. The third type of layer is activation, which is where the CNN learns the properties of the images. Our system works with ReLU activation function, which is chosen for its benefits when it comes to representing a large range of numeric values. The final layer is the fully-connected layer, which represents the functionality of an Artificial Neural Network. Each node in the previous layer is connected to each other node in the upcoming layer. This process helps the CNN compare features from the inputted image with the training data set to predict the outcome of the handwritten digit [7-9].

The Python code can be seen alongside the closed system can be seen in Fig. 4.

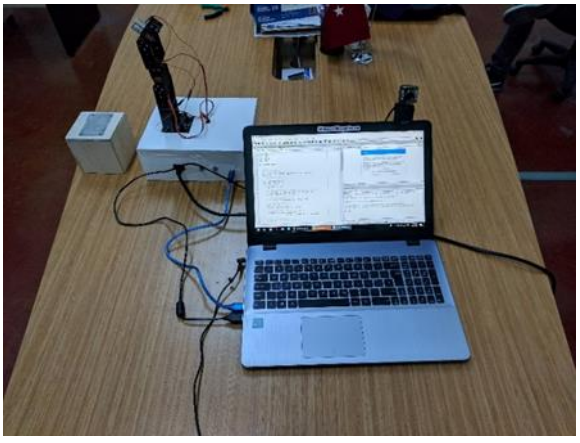

Fig. 4. Closed system along with the software.

4. Kinematic Analysis

Kinematic analysis generates kinematic equations describing robot motion geometry. Using the mechanical properties of the robot, forward kinematic analysis is required

in order to hold an object with electromagnet and leave it to a desired target. In doing these movements, inverse kinematic analysis is required to find the angles to which the joints should be found.

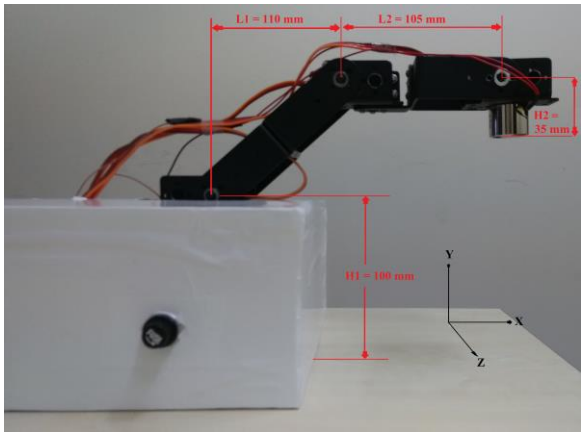

Fig. 5. Length and axis information of the robot arm.

Fig. 5. shows the part lengths and axis information of the robot arm.

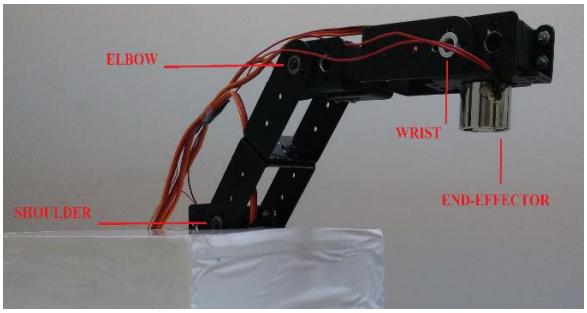

Fig. 6. Length and joint definitions of robot arm limbs

Fig. 6. shows the length of the shoulder, base, shoulder, elbow and wrist joints are formed. In addition, the robot arm has a holder end used to grasp objects.

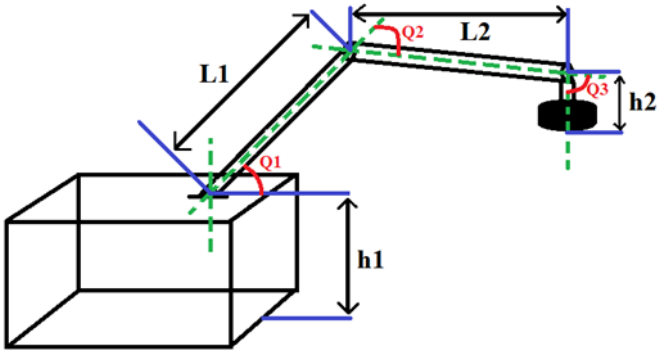

Fig. 7. Design details of robot arm parts.

As shown on Fig. 7. the robot arm lengths are L1 and L2. Height h1 between robot base and L1 arm, the height between the holder electromagnatis and the wrist on the L2 handle is known to be H2. Forward kinematic analysis of the robot through this data;

$$x = L_1*\cos\theta_1+L_2*\cos(\theta_1+\theta_2)+L_3*\cos(\theta_1+\theta_2+\theta_3) \quad (1)$$

$$y = L_1*\sin\theta_1+L_2*\sin(\theta_1+\theta_2)+L_3*\sin(\theta_1+\theta_2+\theta_3) \quad (2)$$

$$\theta = \theta_1+\theta_2+\theta_3 \quad (3)$$

We'II find out where the electromagnet on the robot arm is in the work area. If the robot arm is taken derivative of the above equations according to time to find the speed of operation,

$$x' = -L_1*\theta_1'*\sin\theta_1-L_2*(\theta_1'+\theta_2')*\sin(\theta_1+\theta_2)-L_3*\cos(\theta_1'+\theta_2'+\theta_3')*\sin(\theta_1+\theta_2+\theta_3) \quad (4)$$

$$y' = L_1*\theta_1'*\cos\theta_1+L_2*(\theta_1'+\theta_2')*\cos(\theta_1+\theta_2)+L_3*\cos(\theta_1'+\theta_2'+\theta_3')*\cos(\theta_1+\theta_2+\theta_3) \quad (5)$$

$$\theta' = \theta_1'+\theta_2'+\theta_3' \quad (6)$$

According to the data above, speed equations (4), (5) and (6) are found.

The Robot calculates the angle θ1 and θ2 where the joints should be located with Inverse Kinematics while making their movements with forward kinematics. These angle equations are given in equation (7) and (8).

$$\theta_1 = tan^{-1}\frac{y}{x} + tan^{-1}\frac{L_2*\sin\theta_2}{L_1+L_2*\cos\theta_2} \quad (7)$$

$$\theta_2 = -cos^{-1}\frac{x^2+y^2-L_1^2-L_2^2}{2*L_1*L_2} \quad (8)$$

Given above (1), (2), (3), (7) and (8) equations of the robot arm X, Y coordinates and θ angles are calculated. The angles θ1 and θ2 of the servo motors on the robot arm are calculated.

Fig. 8. shows four working positions of the robot arm. These locations show the points needed to perform the robot's tasks.

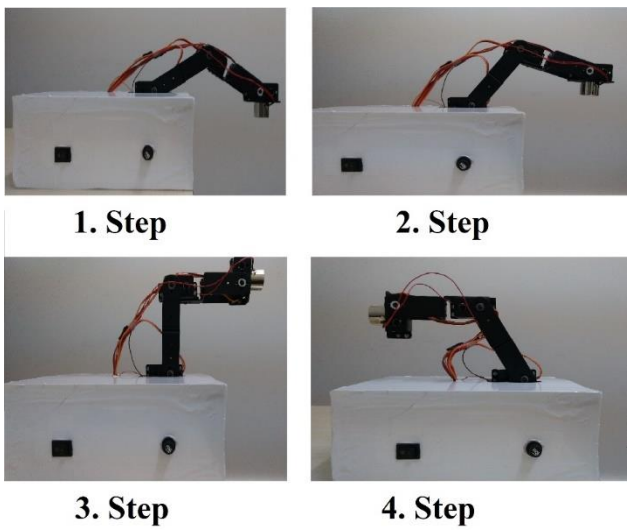

Fig. 8. 4 different motion results of the robot arm view.

5. Experiments

In the test environment, a snapshot of the numbers on the white background is taken and the operations are performed. Tests have been repeated by changing the distance between the texts and the display device. The test environment is given in Fig. 9.

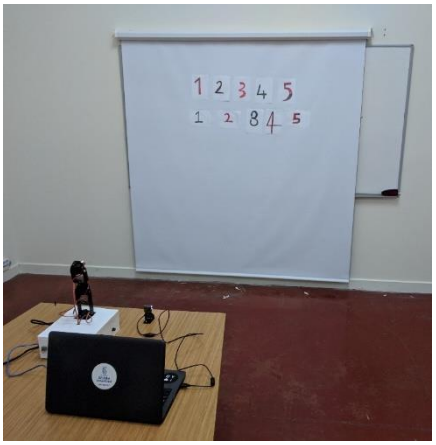

Fig. 9. Showcase of the system

In this experiment, the distance between the background and the camera was 30 centimeters. The purpose of this experiment was to test whether the software is capable of recognizing digits from a relatively short distance. The handwriting figures used in the experiment are written by the authors of this article and shown on Fig. 10. The handwritten digits for Onur Ozturk and Ali Cetinkaya respectively are presented below.

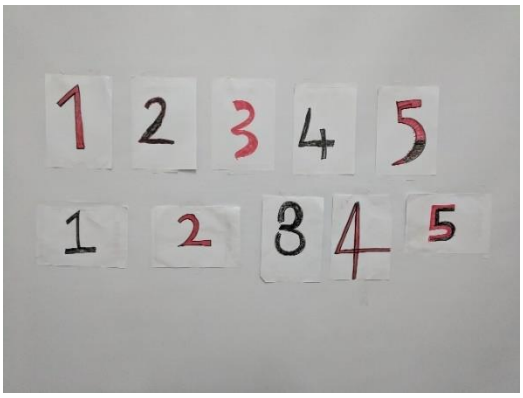

Fig. 10. Testing data set

Recognizing digits from a distance of 30cm between the background and camera.

The handwritten digits presented above were tested individually and the results received from these tests are presented below. Some figures resulted in incorrect predictions, however the majority of the predictions are correct.

The screenshots of the experiments performed between Figure 11 and Figure 26 are given.

Fig. 11. Incorrectly predicted ‘1’.

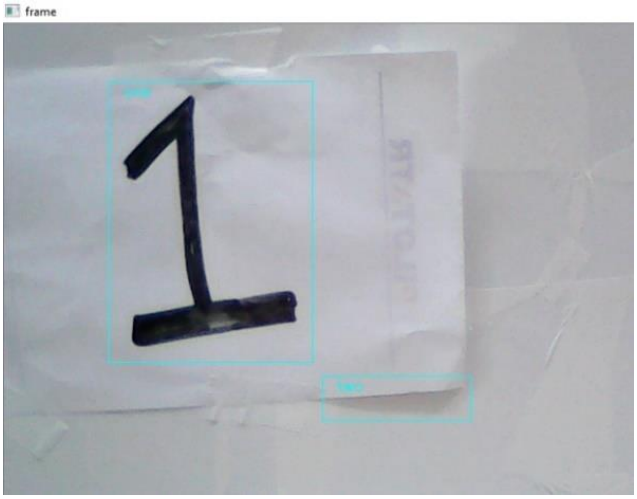

Fig. 12. Correctly predicted ‘1’.

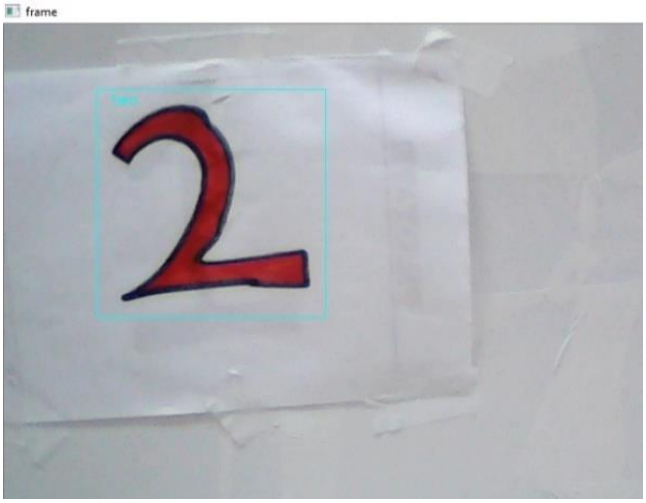

Fig. 13. Correctly predicted ‘2’.

Fig. 14. Correctly predicted '3'.

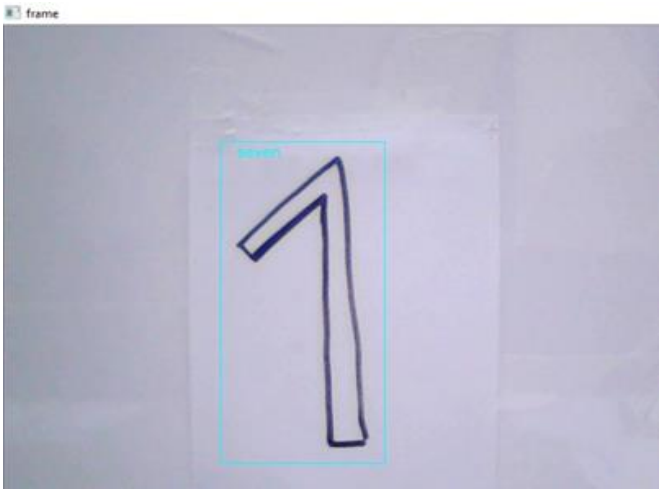

Fig. 17. Incorrectly predicted '1'.

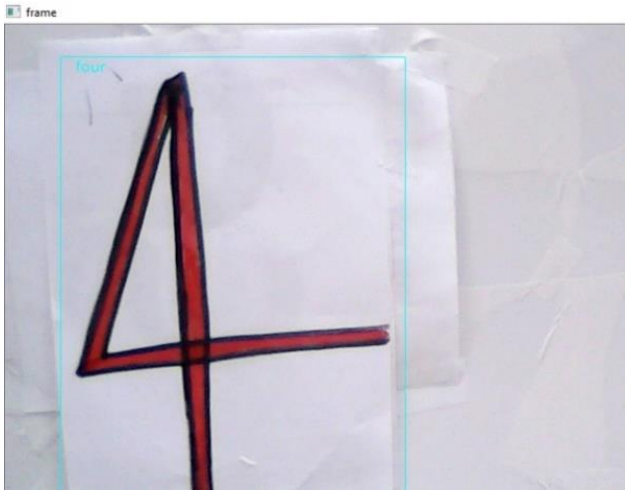

Fig. 15. Correctly predicted '4'.

Fig. 18. Correctly predicted '2'.

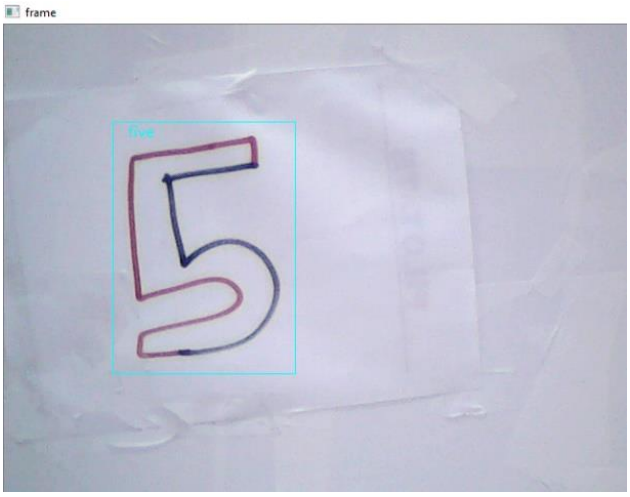

Fig. 16. Correctly predicted '5'.

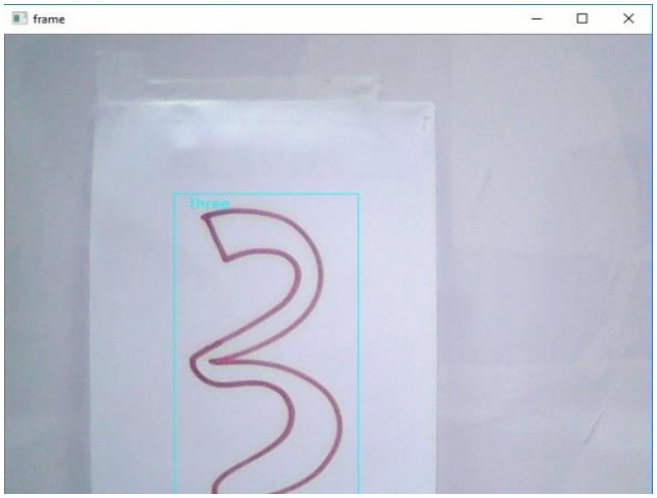

Fig. 19. Correctly predicted '3'.

Fig. 20. Incorrectly predicted ‘4’.

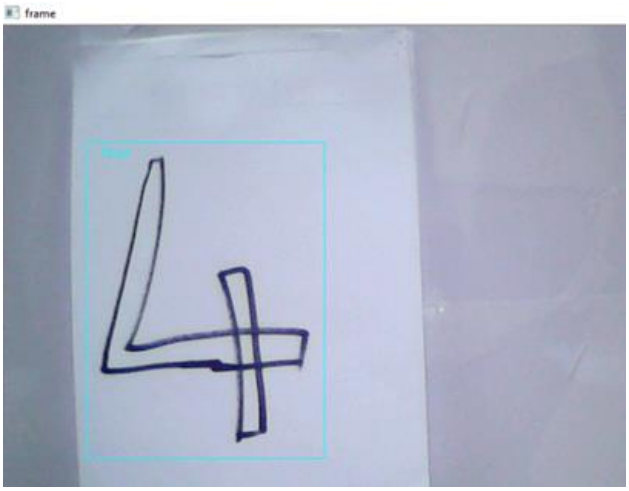

Fig. 21. Correctly predicted ‘4’.

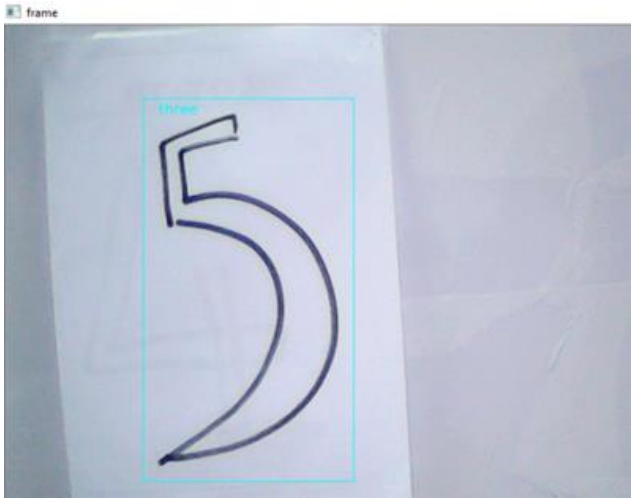

Fig. 22. Incorrectly predicted ‘5’.

In this experiment, the distance between the background and the camera was 2 meters. The purpose of this experiment was to test whether the software is capable of recognizing digits from a long distance. The test data set was presented in the previous section.

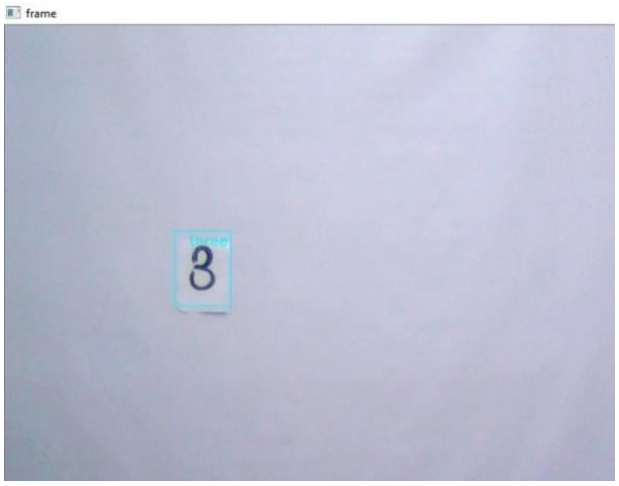

Fig. 23. Correctly predicted ‘3’.

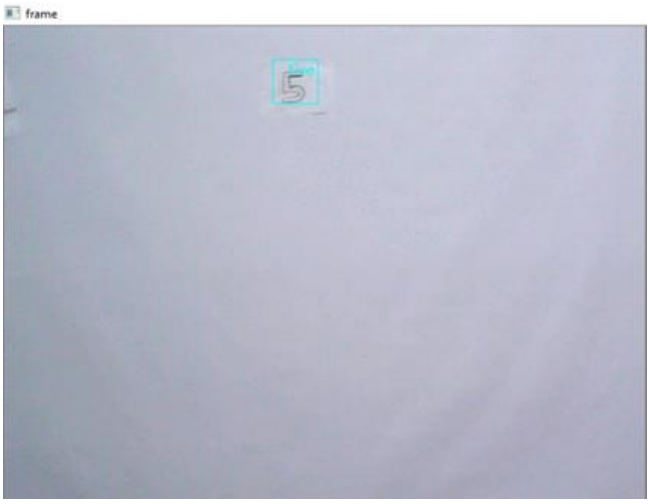

Fig. 24. Correctly predicted ‘5’.

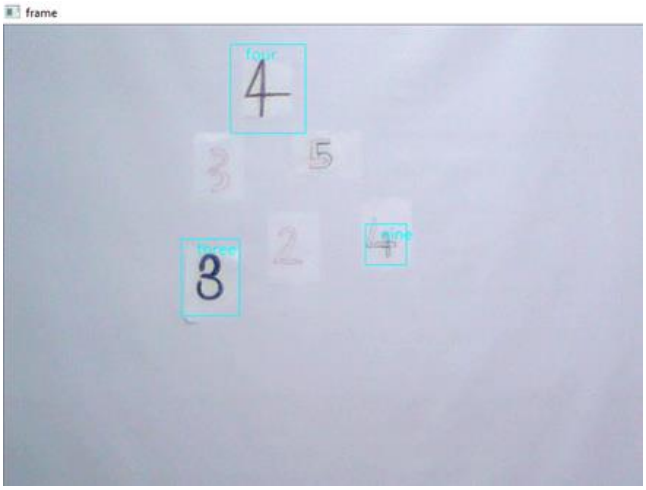

Fig. 24. Showcasing the importance of boldness from a distance of 2m.

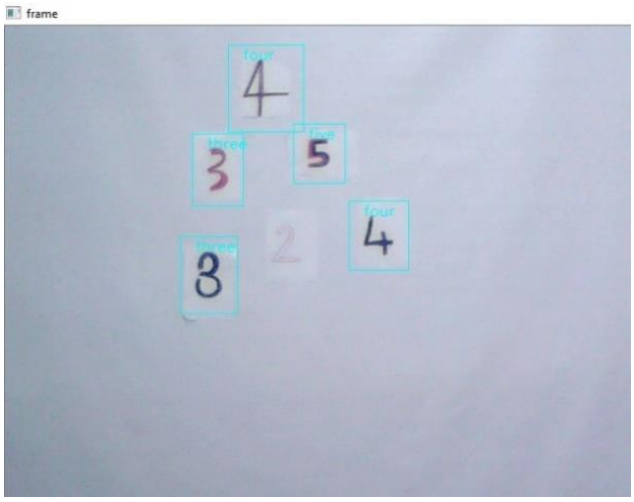

Fig. 25. Further showcasing the importance of boldness from a distance of 2m.

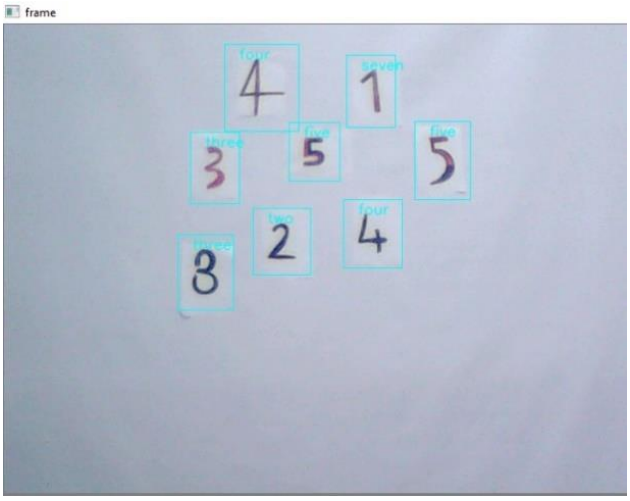

Fig. 26. Results achieved from a mixed test set.

The first issue arises from the distance at which the software is capable of recognizing the handwritten digits. From a short distance, writing the numbers with a black or red board marker was sufficient. However, as the distance between the board on which the digits were written and the camera increases, the software had difficulty recognizing the numbers. The nature of the problem is caused by the thickness of the digits written. From a short distance, the thickness of the board marker was sufficient however from a long distance, it looked thin therefore the camera could not recognize the digits.

The second issue arises from having an unstable board and camera. When conducting the tests for the system, the team realized that if either the camera or the board was shaking, the software could not identify the digits. To solve this issue, we fixed the camera to a book and the handwritten digits were stuck to the board with tape. Furthermore, if the paper with the digit written on it is not stuck to the background completely where some part of the paper is lifting off, the shadows left by the lifting paper is recognized by the software as handwritten digits. Therefore, the paper with the digit must be completely stuck to the background where no piece is lifting off the background.

The third issue arises from not having the handwritten digit completely within the frame. For example, during the testing phase when number four were shown to the camera whilst not being within the frame fully, the software recognized this number as six. In order to achieve correct recognition, the most optimal position for the handwritten digit is the middle of the frame. This issue is presented in the Experiment and Results section in Figures 9 and 10.

The final issue arises from the similarity of the digit ‘1’ and ‘7’. During our tests, different members of the team wrote different ‘1’s, where the horizontal line on the bottom was not present in some handwritten digits. The MNIST data set trained ‘1’s with the horizontal line present. Therefore, when a person did not include the horizontal line below ‘1’, the software recognized the digit as ‘7’. This issue is presented in the Experiment and Results section in Figures 9.7.

6. Conclusion

The results indicated that as the distance between the background and the camera increases, the boldness of the handwritten digits must increase in order for the camera to capture the image presented on the background. Furthermore, a special case for number ‘1’ and ‘7’ exists. The user must include the horizontal line below the ‘1’ in order for the software to predict the number correctly as ‘1’. This is most likely caused by the training data set including the horizontal line below ‘1’s.

In the results obtained, it was observed that the robot performs the movements given in Figure 8 according to the order of identification. Because there are no lines below 1 characters created for testing, MNIST could not find them in the data set and incorrectly confused them with 7 characters.

7. Discussion

In this study, a system has been developed with an electromagnet placed on a robotic arm to perform repetitive processes in the manufacturing industry. For this purpose, the numbers of "1, 2, 3, 4, 5" written by hand on the MNIST dataset were detected and recognized through the camera.

During the experiments, 17 samples were taken from the images taken from the camera. In these experiments, 12 samples had correct results, 5 samples had false results and 2 samples had incomplete readings.

One of wrong reading "1" when there is no horizontal line under a character "7" has been observed to interfere with the character.

The figures used in the study were created by Onur OZTURK and Ali CETINKAYA. As the distance between the background and the camera increases in the experiments, it is concluded that the numbers need to be thickened so that the software can capture the image. To correct this error, the reading resolution and accuracy can be increased by increasing the camera resolution.

Acknowledgements

Many thanks to Istanbul Gelisim University and Technology Transfer Office for their support.

References

[1] Y. Lecun, C. Cortes, C.J.C. Burges, MNIST handwritten digit database, http://yann.lecun.com/exdb/mnist/

[2] K. Sato, N. Shimoda, Build your own machine-learning-powered robot arm using tensorflow and google cloud | Google Cloud blog, 2017.

[3] Keras documentation, https://keras.io/

[4] TensorFlow, https://www.tensorflow.org/

[5] A. Elfasakhany, E. Yanez, K. Baylon, R. Salgado, Design and development of a competitive low-cost robot arm with four degrees of freedom, Modern Mechanical Engineering, pp.47-55.

[6] OpenCV library document, https://opencv.org/

[7] K. Simonyan, A. Zisserman, Very deep convolutional networks for large-scale image recognition. Arxiv - Computer Vision and Pattern Recognition .

[8] S. Raschka, V. Mirajalili, Python machine learning (pp. 341-385).

[9] P. Bezak, P. Bozek, Y. Nikitin, Advanced robotic grasping system using deep learning, Procedia Engineering, 96, pp.10-20., 2014.

[10] A. Dhawan, A. Bhat, S. Sharma, H. K. Kaura, Automated robot with object recognition and handling features, International Journal of Electronics and Computer Science Engineering, ISSN- 2277-1956.

[11] E. B. Mathew, D. Khanduja, B. Sapra, B. Bhushan, Robotic arm control through human arm movement detection using potentiometers. 2015 International Conference on Recent Developments in Control, Automation and Power Engineering (RDCAPE), 2015.

[12] B.Iscimen, H. Atasoy, Y. Kutlu, S. Yildirim, E. Yildirim, Smart robot arm motion using computer vision, 2015.

[13] M.A. Jayaram, H. Fleyeh, Convex Hulls in Image Processing, A Scoping Review, American Journal of Intelligent Systems, 2016.

[14] N. Rai, B. Rai, P. Rai, Computer vision approach for controlling educational robotic arm based on object properties, 2nd International Conference on Emerging Technology Trends in Electronics, Communication and Networking. 2014

[15] T. S. Tonbul, M. Sarıtas, Beş eksenli bir edubot robot kolunda ters kinematic hesaplamalar ve yörünge planlaması. J. Fac. Eng. Arch. Gazi Univ. Vol 18, No 1, 145-167, 2013

[16] A. B. Rehiara, Kinematics of adeptthree robot arm, Robot Arms, ISBN: 978-953-307-160-2, 2011.