International Journal on Smart Material and Mechatronics
IJSMM Vol. 11 No. 2 2024 DOI. 10.20342/IJSMM.11.2. 315

p-ISSN: 2356-5314
e-ISSN: 2460-075X

Implementation of Motion Robot Arm Mitsubishi RV-2SDB (Palletizer) Using The Hierarchical Finite State Machine Method For Sorting Systems Based On Box Color and Shape

Eko Budi Utomo, Indra Adji Sulistijono, Afif Nur Rasyidin
Bachelor of Applied Science Mechatronics Engineering
Politeknik Elektronika Negeri Surabaya
Surabaya, Indonesia
ekobudi_u@pens.ac.id, indra@pens.ac.id, afifnurrasyidin86@gmail.com

***Abstract*— This research aims to develop an automation system for the palletizing process by integrating OpenCV and Siemens S7-1200 PLC. The developed system uses a camera to detect the shape and color of workpieces, which are then processed by OpenCV. The processed data is sent to the Siemens S7-1200 PLC using the Snap7 Python library to control the movement of the Mitsubishi MELFA RV-2SDB robotic arm. The sorting process is conducted based on the identified shape and color of the workpieces. This system is designed to improve production efficiency with consistent results and reduce the operator's workload through automation. Testing was conducted 72 times, showing that the system can distinguish and move workpieces with an accuracy rate of 95.9% and a failure rate of 4.1%. These results demonstrate the system's ability to effectively identify and classify workpieces. The implementation of this system is xpected to make a significant contribution to the manufacturing industry, especially in the packaging and distribution processes, by increasing efficiency, consistency, and work safety. Additionally, this research provides practical guidance for the integration of vision technology with PLC systems for robotic applications, which can serve as a reference in the development of other industrial automation systems.**

***Index Terms*— OpenCV, PLC Siemens S7-1200, Automation.**

I. INTRODUCTION

In the industrial era, automation has become an important role in improving efficiency and productivity [1]. One significant application of automation is in the palletizing process, where goods are moved and arranged systematically on pallets for easy storage and shipping. The manual palletizing process has several disadvantages, such as the risk of injury to workers due to repetitive heavy lifting, unevenness in the arrangement of goods, and low productivity. To overcome these problems, many companies are turning to the use of robot palletizers that combine robotic technology with automated control systems [2].

A robot palletizer is an industrial machine that uses a robotic arm to move goods. This robot arm is equipped with a gripper or grasping tool, so that it is able to move goods from one location to another without the need to move places. In this research, a Mitsubishi MELFA RV-2SDB robotic arm automation system is developed that is equipped with sorting capabilities based on the shape and color of the workpiece using OpenCV technology and a Siemens S7-1200 PLC. OpenCV is used to detect the shape and color of the workpiece through image processing taken by the camera [3]. The detection data is then sent to the Siemens S7-1200 PLC using the Snap7 Python library. The system is designed to improve production efficiency, ensure consistent results, and reduce operator workload through automation.

This research encompasses system design, implementation, and performance testing of the workpiece sorting system. The test results show that the system has an accuracy rate of 95.9% with a failure rate of 4.1% out of 72 trials. Thus, this system is expected to make a significant contribution to the manufacturing industry, particularly in the packaging and distribution processes.

II. BASIC THEORY

A. Digital Image Processing

Digital image processing is a discipline that studies image processing techniques. The image referred to in this research is a static image from a vision sensor in the form of a webcam. Mathematically, an image is a continuous function of light intensity on a two-dimensional plane. In order to be processed with a digital computer, the image must be presented numerically with discrete values. The representation of continuous functions into discrete values is called image digitization [4].

An image can be defined as a function f(x,y) of size M rows and N columns, with x and y as spatial coordinates, and the amplitude of f at coordinate point (x,y) is called the intensity or gray level of the image at that point. If the values of x, y, and amplitude f are finite and discrete, then the image is a digital image. Figure 1 shows the representation of a digital image in 2 dimensions.

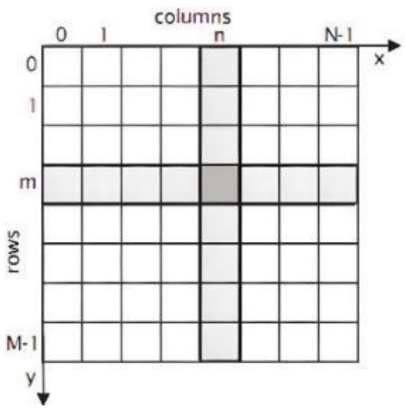


Fig. 1. Digital image representation in 2D[4]

B. Finite State Machine Method

Finite State Machine (FSM) is a mathematical model used to explain the behavior of systems that have finite states. In this model, the system is considered as a machine that can be in one of several different states at any given time. FSM consists of several elements, namely state, input, output and transition function. The state is the condition of the system at any given moment. Input is the data given to the system. Output is the result produced by the system after processing the input. While the transition function is a function that regulates the movement of states from one state to another. To simplify the explanation, a simple diagram of the FSM is shown in Figure 2.

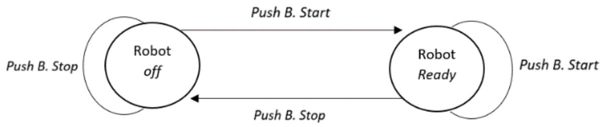


Fig. 2. Simple finite state machine.

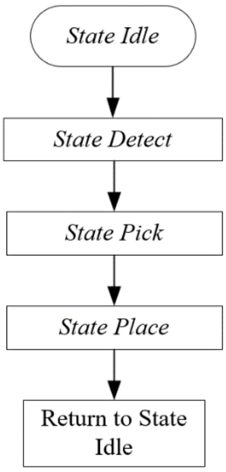


Fig. 3. Workflow fsm of robot palletizer

Figure 3 shows the workflow of the FSM for a robot palletizer. The following is an explanation of each state in the FSM :

1. Idle State (Robot Homing)
The FSM is in an idle state or the robot is in a homing position waiting for input from the photoelectric sensor or camera sensor. If the sensors detect a workpiece, the FSM moves to the detect state.

2. State Detect
In state detect the FSM identifies the workpiece using the camera sensor. If the workpiece is recognized, the FSM moves to the pick state.

3. Pick State
The PLC controller sends a command to the robot arm controller to pick the workpiece. After the workpiece is picked, the FSM moves to state place.

4. State Place
The arm robot places the workpiece on the correct or appropriate pallet. After the workpiece is placed on the pallet, the arm robot returns to the homing position or idle state.

C. Robot Station

The robot station functions to perform pick and place tasks on the workpiece. All movements at this station are carried out by the Mitsubishi MELFA RV-2SDB robot arm. At the robot station there is a conveyor that functions to distribute workpieces or workpieces. In the robot station there is a Mitsubishi MELFA RV-2SDB arm robot shown in Figure 4. Arm robots or commonly known as manipulator robots are one type of robot that is often used in industry. A robot manipulator that mimics the characteristics of a human arm is also called an articulated arm [5][6]. All robot arm joints are rotary or revolate type. However, the motion of an articulated arm is different from that of a human arm. While robot joints have fewer degrees of freedom (DOF). Robot manipulators can move freely even from a certain angle with more freedom than humans [7]. For example, the elbow of an articulated robot can bend up and down, whereas humans can only bend their elbow in one direction and the rest of the arm is straight. Figure 4 is a picture of the robot station.

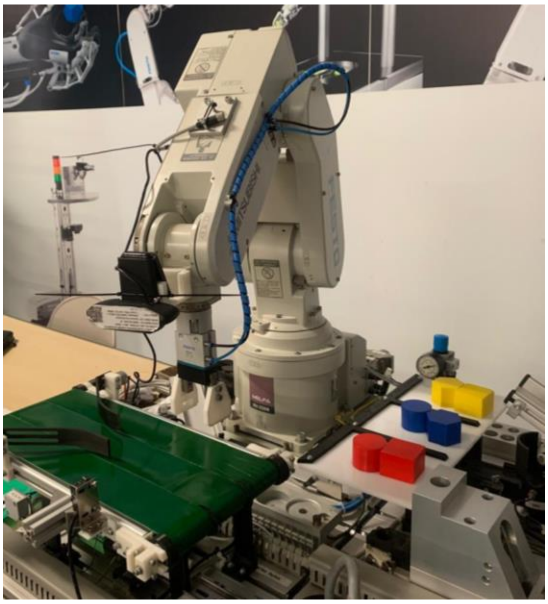


Fig. 4. Robot station

III. SYSTEM OVERVIEW

This section will discuss the overall system design which includes the system block diagram, hardware configuration, and system implementation. These steps include planning the hardware and software used in the robot arm automation system for the palletizing process.

A. Block diagram

The initial stage of the system created is input from computer vision that detects the workpiece. Computer vision data is input to the Siemens S7 1200 PLC in the form of data from the color and shape of the workpiece to carry out the sorting process. The data is then processed by the Siemens S7 1200 PLC and then sent to the CR1DA-700 controller to drive the robot arm. Furthermore, the robot arm will pick up the workpiece that is still on the conveyor, after the robot arm picks up the workpiece, it will then place the workpiece on the appropriate pallet that has been provided. In the workpiece sorting process, the conveyor is equipped with a camera for vision sensors so that it can determine the shape and color of the workpiece. The vision sensor program uses an open source library, OpenCV Python, which is used for the workpiece detection process. After obtaining data on the shape and color of the workpiece, the data is sent to the Siemens S7 1200 PLC using the Snap7 Python library. The system design in the form of a flowchart is described in Figure 5.

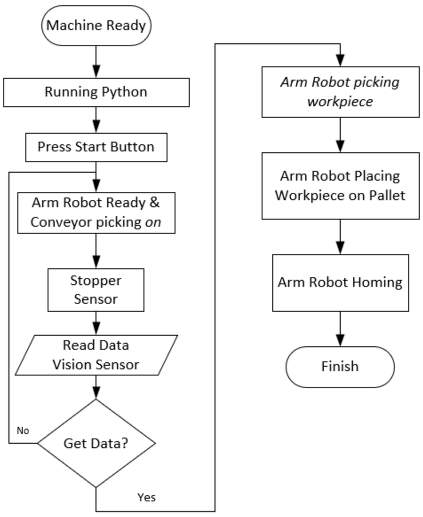


Fig. 5. Workflow diagram of robot palletizer system

B. Hardware configuration

From the hardware design in Figure 6, we can know the data flow of the entire system. The vision sensor gets data then processed by OpenCV with Python programming and then sends the processed data to the Siemens S7 1200 PLC via ethernet communication via the Snap7 Python library. Then the data is forwarded to CR1DA-700 which is used to direct the robot arm to the appropriate workpiece pallet that has been provided.

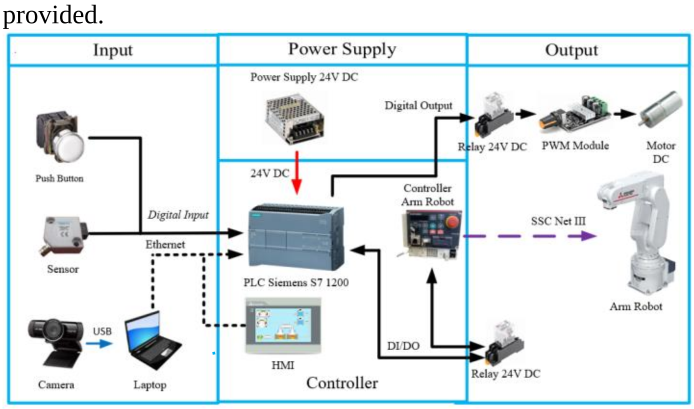


Fig. 6. Block diagram of hardware design

C. Opencv shape and color detection

Shape and color detection using OpenCV There are three main steps to detect shapes and colors in this experiment. The first one detects the object and removes the background. The second one creates the contour of the detected object. And the third detects the shape of each detected object.

1. Detect Objects and Remove Background

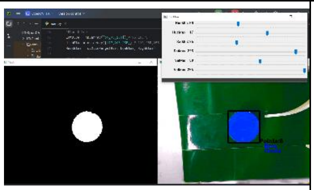
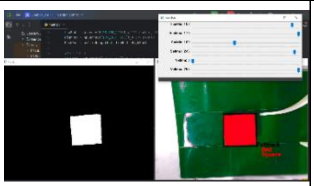
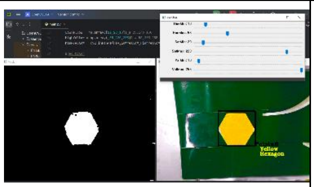
Create a trackbar to change the range of colors to detect certain colors. This research uses the Hue, Saturation, and Value method to detect various colors on the workpiece.

2. Create Contours

Contours are simply the boundaries of an object. start a while loop to work with video, so we loop frame by frame and then convert frames from BGR format to Hue, Saturation, and Value (HSV). We do this so that later we can create a mask. To convert colors from BGR to HSV format.

From the trackbar above, we will take the color range data from the workpiece. From the color range data that has been obtained, it will be defined in the Python code. The following Table 1 is an example of the yellow, red, and blue color range of the workpiece. This can be different because it is influenced by the camera used and the lighting in the room.

TABLE I. DATA RANGE COLOR

Color Range Value	Image	Description
LowBlue = ([69,95,150]) HighBlue = ([117,235,255])		Process to find the color of the blue circle shape workpiece
LowRed = ([168,100,120]) HighRed = ([179,245,255])		Process for finding the color of the red square shape workpiece
LowYellow = ([18,20,10]) HighYellow = ([55,220,255])		Process for finding the color of the yellow hexagon shape workpiece

From the color range, you can create a masking of each color. Masking can be created using the `inRange` function (`cv2.inRange`). The `inRange` function has three parameters, namely the source image in HSV or BGR format, lower or lower value threshold that determines the minimum value of the value range, and upper or upper value threshold that determines the maximum value of the value range. The `inRange` function works by checking the value of each pixel in the source image. If the pixel value is within the specified value range, then the pixel will have a value of 1 in the mask. Conversely, if the pixel value is outside the specified value range, then the pixel will have a value of 0 in the mask. As in Figure 7.

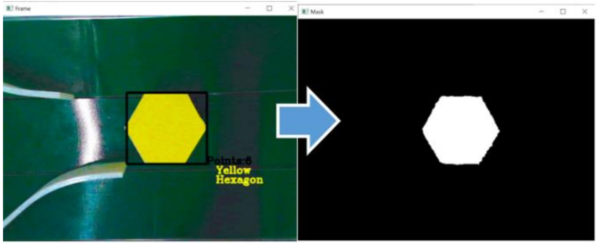


Fig. 7. Masking workpiece yellow color

From `findContours` (`cv2.findContours`). The `findContours` function has three parameters, namely, image or image source in the form of BGR format or the result of masking above, contour detection mode, and method for contour detection. The contour detection method uses the `CHAIN_APPROX_NONE` function, which returns all contour points. Contour points are points that form the contour. The `findContours` function works by checking each pixel in the source image. If the pixel value is above a certain threshold, then the pixel will be considered a contour point. As in Figure 7 above, the workpiece detected here can find contours using the OpenCV built-in function has 6 contour points, so a hexagon workpiece will be defined.

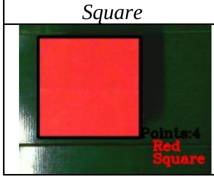
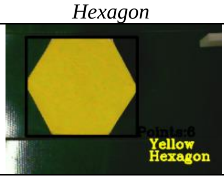
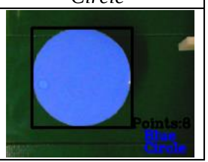
3. Shape Detection

In this research, the detection is done by calculating how many corner points the contour has. To minimize noise to get a clean contour, you can use the approximation function in OpenCV, namely `cv2.approxPolyDP`. this function works by connecting contour points adjacent to the line. The approximation function has two main parameters, the approximated contour and the epsilon value used to determine the distance between two points that are considered adjacent. To measure the distance between two points using Equation (1) the 2-dimensional euclidean distance is as follows :

$$d = \sqrt{(x2 - x1)^2 + (y2 - y1)^2} \quad (1)$$

If the distance between two points is less than or equal to epsilon, then the two points are considered close together. Smaller epsilon values will result in polygons with more points, while larger epsilon values will result in polygons with fewer points. If a contour has 4 corner points, then it is a square, 6 corner points are hexagons, and anything else is a circle, As in Table 2.

TABLE II. SHAPE AND COLOR DETECTION OF WORKPIECE

Square	*Hexagon*	*Circle*
		

D. OpenCV communication with PLC Siemens S7 1200

Communication between OpenCV and PLC, specifically the Siemens S7-1200 or S7-1500 series, is achieved using the Snap7 library. This open-source library is used for communication with PLCs. The Snap7 library allows for various operations such as reading and writing data to a Siemens PLC from an application written in the Python programming language, as in this research. The Snap7 library supports multiple programming languages, including C, C++, C#, and Python. Below are the steps for using the Snap7 library with the Python programming language.

1. Instal *library* Snap7
   Can install using pip: pip install python-snap7.

2. Making connections to the PLC
   The following is the code to connect to the PLC Siemens S7 1200 which is used to read and write data as shown in Figure 8 and Figure 9.

```
def ReadMemory(plc,byte,bit,datatype): #define read memor
    result = plc.read_area(areas['MK'],0,byte,datatype)
    if datatype==S7WLBit:
        return get_bool(result, byte_index: 0, bool_index: 1)
    elif datatype==S7WLByte or datatype==S7WLWord:
        return get_int(result, byte_index: 0)
    elif datatype==S7WLReal:
        return get_real(result, byte_index: 0)
    elif datatype==S7WLDWord:
        return get_dword(result, byte_index: 0)
    else:
        return None
```

Fig. 8. Code for reading data

```
def WriteMemory(plc, byte, bit, datatype, value): #define wr
    result = plc.read_area(areas['MK'],0,byte,datatype)
    if datatype==S7WLBit:
        set_bool(result, byte_index: 0,bit,value)
    elif datatype==S7WLByte or datatype==S7WLWord:
        set_int(result, byte_index: 0,value)
    elif datatype==S7WLReal:
        set_real(result, byte_index: 0,value)
    elif datatype==S7WLDWord:
        set_dword(result, byte_index: 0,value)
    plc.write_area(areas['MK'],0,byte,result)
```

Fig. 9. Code for writing data

3. Address configuration of PLC
   It is necessary to configure the IP address of the PLC Siemens S7 1200 correctly as shown in Figure 10. If the IP address is wrong or incorrect, there will be a communication failure between the PLC and OpenCV.

```
#Configuration PLC
IP = '192.168.0.1' #IP PLC
RACK = 0 #RACK PLC
SLOT = 1 #SLOT PLC
```

Fig. 10. Address configuration of PLC siemens s7 1200

4. Create address code to write data to the PLC
   Here is the Python code used to write data to a Siemens S7-1200 PLC. In the code below, data is written or sent to the PLC using memory word address 30 to send data for yellow workpieces, memory word address 40 to send data for red workpieces, and memory word address 50 to send data for blue workpieces, as shown in Figure 11.

p-ISSN: 2356-5314
e-ISSN: 2460-075X

```
#Write Variable to PLC
WriteMemory(PLC, byte: 30, bit: 0, S7WLWord, YellowVariable)
WriteMemory(PLC, byte: 40, bit: 0, S7WLWord, RedVariable) #
WriteMemory(PLC, byte: 50, bit: 0, S7WLWord, BlueVariable) #
```

Fig. 11. Code write data to PLC siemens s7 1200

5. Create address on PLC Siemens S7 1200

After the Python program sends or writes data to the Siemens S7-1200 PLC, the PLC program uses the same addresses as in the Python programming, namely MW30, MW40, and MW50, as shown in Figure 12.

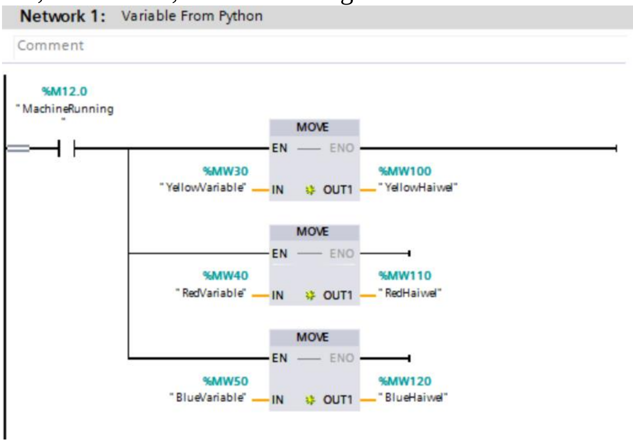


Fig. 12. Program read data on PLC siemens s7 1200

IV. Experiments

In this testing, the Mitsubishi MELFA RV-2SDB robotic arm can sort workpieces based on their shape or color and then package or group them onto the designated pallets. Below is the testing of the palletizer system.

A. *Testing the palletizer system based on shape*

Below is the testing of the palletizer system based on the shape of the workpiece. This test is conducted to observe how the robotic arm recognizes, picks, and places the workpieces. The purpose of this test is to ensure that the palletizer system on the robot functions properly.

1. State idle (robot homing)

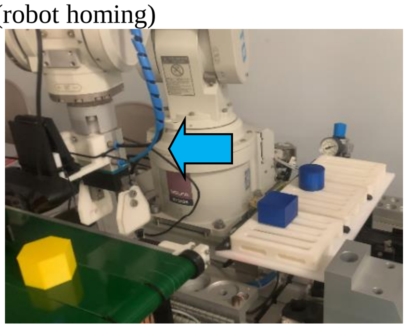


Fig. 13. Robot arm state homing position

In Figure 13 the robot arm is in the initial state of the homing position to detect the workpiece with the robot arm coordinates X = 315.100, Y = 89.360, Z = 290.470, A = 179.080, B = -0.620, C = 178.790.

2. State detect

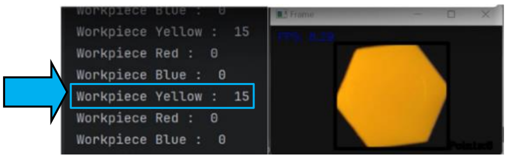


Fig. 14. Display vision sensor palletizer system by shape

In Figure 14 the robot arm is in the initial state of the homing position to detect the workpiece. When the wokrpiece reads a value of 15, it means that the workpiece shape is hexagon with robot arm coordinates X = 315.100, Y = 89.360, Z = 290.470, A = 179.080, B = -0.620, C = 178.790.

3. State pick

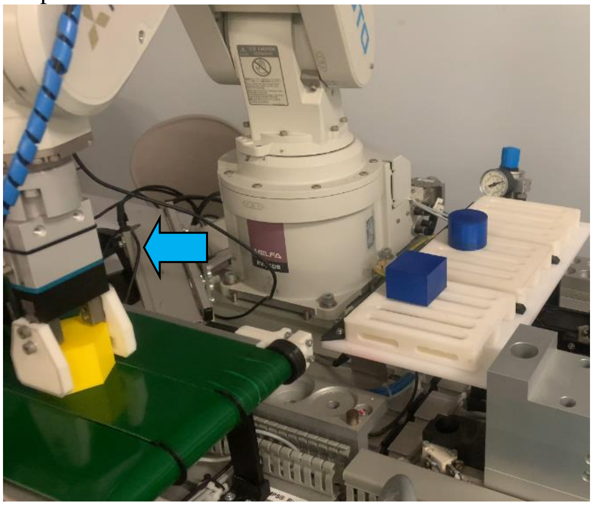


Fig. 15. Robot arm pick workpiece palletizer system based on shape

In Figure 15 the robot arm moves towards the hexagon workpiece that has been detected to pick it up with the robot arm coordinate points X = 406.390, Y = 116.760, Z = 289.450, A = 179.080, B = -0.620, C = 87.870.

4. State place

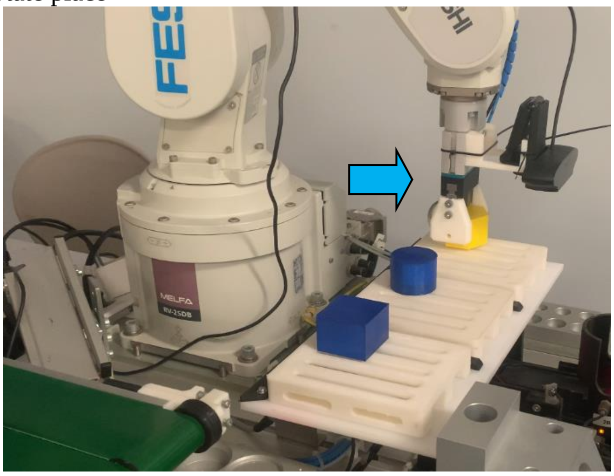


Fig. 16. Robot arm place workpiece palletizer system based on shape

In Figure 16, the workpiece is picked up to be moved to pallet number 1 with the robotic arm coordinates X = -67.440, Y = 226.540, Z = 249.570, A = 178.750, B = -0.240, C = -89.420.

From the above test, it was found that the robot can detect the shape of the workpiece it has picked up and then place the

p-ISSN: 2356-5314
e-ISSN: 2460-075X

workpiece onto pallet 1, which is designated for hexagon-shaped items. This process was repeated 4 times, requiring 4 workpieces to fully fill pallet 1. The results of the test show that the robot can perform this task accurately and efficiently. The image below shows the results of the above test.

Fig. 17. Palletizer results based on shape workpiece hexagon

Figure 17 shows the results of the above test, demonstrating the success of the system in managing and arranging workpieces optimally. The image illustrates how the robotic arm with a camera sensor is able to perform its task efficiently, showing significant potential for implementation in industrial automation. With this technology, companies can achieve higher levels of efficiency, minimize human errors, ultimately reducing production costs and improving overall productivity.

B. Testing the palletizer system based on colors

Below is the testing of the palletizer system based on the color of the workpieces. This test is conducted to observe how the robotic arm recognizes, picks, and places the workpieces. The purpose of this test is to ensure that the palletizer system on the robot functions properly.

1. State idle (robot homing)

In Figure 13 the robot arm initial state homing position to detect the workpiece with the robot arm coordinate point X = 315.100, Y = 89.360, Z = 290.470, A = 179.080, B = -0.620, C = 178.790.

2. State detect

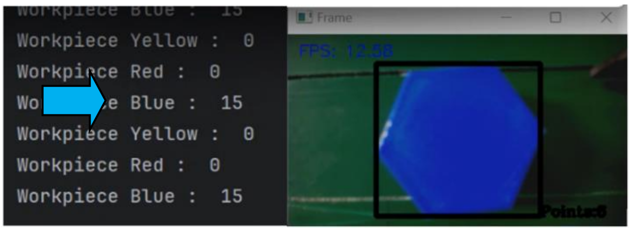


Fig. 18. Display vision sensor palletizer system by color

In Figure 18 the robot arm is in the initial state of the homing position to detect the workpiece. When the blue wokrpiece has a value, it means that the workpiece color is blue with the robot arm coordinate point X = 315.100, Y = 89.360, Z = 290.470, A = 179.080, B = -0.620, C = 178.790.

3. State pick

In Figure 19 the robot arm moves towards the workpiece that has been detected to pick it up with the robot arm coordinate points X = 406.390, Y = 116.760, Z = 289.450, A = 179.080, B = -0.620, C = 87.870.

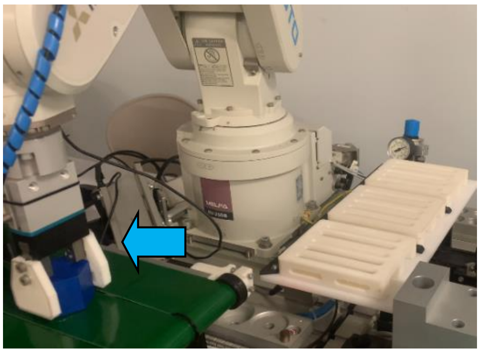


Fig. 19. Robot arm pick workpiece palletizer system based on color

4. State place

Fig. 20. Robot arm place workpiece palletizer system based on color

From Figure 20 the workpiece is picked up to be moved to pallet number 1 with the robotic arm coordinates X = -67.440, Y = 226.540, Z = 249.570, A = 178.750, B = -0.240, C = -89.420.

From the above tests, it is obtained that the robot can detect the shape of the workpiece that has been taken and then put the workpiece into pallet 1, which is a pallet specifically for blue. The process is repeated or it takes 4 workpieces to fully fill pallet 1. The results of the test show that the robot is able to carry out this task precisely and efficiently. The following image below is the result of the above test.

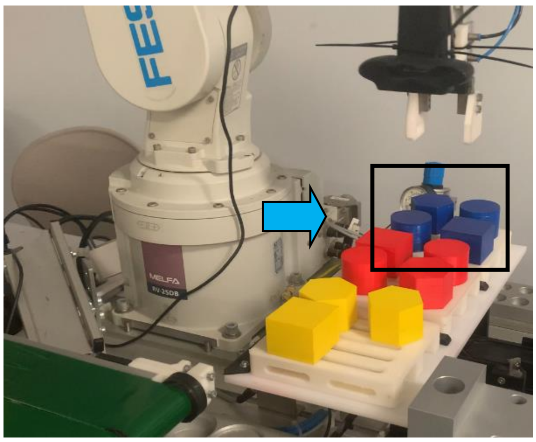


Fig. 21. Palletizer results based on workpiece color blue

From the above test, it was found that the robotic arm can sort workpieces on the conveyor. The workpieces it picks up are then packaged or grouped onto the designated pallets.

Figures 17 and 21 show the results of the above test, demonstrating the system's success in managing and organizing workpieces optimally. These images illustrate how the robotic arm with a camera sensor can efficiently perform its tasks, showing significant potential for implementation in industrial automation. With this technology, companies can achieve higher efficiency levels and minimize human errors, ultimately reducing production costs and increasing overall productivity.

C. Palletizer system reliability testing result

Reliability testing of the pick and place workpiece system uses the Inter-Rater Reliability method. This method is useful for determining the system's consistency by conducting repeated observations. From these repeated observations, failures are recorded.

1. Palletizer system testing results based on shape

This test was conducted by repeating 36 times to determine the consistency of the failure percentage. Testing the reliability of the workpiece selection system with experiments using 12 examples of workpieces consisting of square, circle, and hexagon shapes. The test results can be seen in Table 3.

TABLE III. PALLETIZER SYSTEM TESTING BASED ON SHAPE

No.	Shape *workpiece*	Error	Failure
1	*Square*	1/12	Data read more than 1 workpiece
2	*Circle*	0/12	None
3	*Hexagon*	0/12	None
Total number of trials		36	

From Table 3, the results for testing the reliability of the workpiece sorting system based on shape by taking a sample of 12 kinds of workpieces with square, circle, and hexagon shapes as below.

$$\frac{1}{36} \times 100\% = 2.77\%$$

The calculation above shows that the system has a percentage of failure of 2.77% for 36 test trials.

2. Palletizer system testing results based on colors

This test was conducted by repeating it 36 times to determine the consistency of the failure percentage. The reliability testing of the workpiece selection system involved 12 examples of workpieces, which included square, circular, and hexagonal shapes. The results of the test can be seen in Table 4.

TABLE IV. PALLETIZER SYSTEM TESTING BASED ON COLOR

No.	Color *workpiece*	Error	Failure
1	*Blue*	0/12	None
2	*Red*	2/12	Data read more than 1 workpiece
3	*Yellow*	0/12	None
Total number of trials		36	

From Table 4, the results for testing the reliability of the workpiece sorting system based on shape by taking a sample of 12 kinds of workpieces with square, circle, and hexagon shapes as below.

$$\frac{2}{36} \times 100\% = 5.55\%$$

The calculation above shows that the system has a percentage of failure of 5.55% for 36 test trials.

V. CONCLUSION

Ideal lighting conditions affect the test results. This is because the vision sensor operates by reading the difference in light intensity between the workpiece and the background. When lighting conditions are ideal, the vision sensor can more easily read these intensity differences. As a result, the data sent to the Siemens PLC will be more accurate, leading to more stable test results with minimal failures.

Based on the results of the pick and place system test on the robotic arm with two sorting modes, one based on shape and one based on color: In the color-based sorting mode, the system showed less stability with a failure rate of 8.3%. This indicates that the color-based sorting mode for workpieces still needs improvement to enhance accuracy in sorting workpieces. The failures were caused by several factors, including inadequate room lighting, incorrect scanning or reading of the workpiece by the vision sensor, or the Python program sending two workpiece data to the PLC, resulting in errors in pick and place.

This research still has potential for further development, especially in terms of workpiece sorting or handling, such as based on the size of the workpiece. Additionally, there is a need to improve communication between the robotic arm controller and other controllers, such as PLCs, using communication protocols that allow data transmission in integer form or other data types. If the robot controller can receive data in integer form, it is expected that the robotic arm will be able to move more flexibly and not only based on predefined points (point to point).

[1] REFERENCES

[2] Pranoto, B., & Firdaus, A.R "Rancang Bangun Lengan Robot dengan Sistem Kontrol Otomatis dan Human Machine Interface untuk Mesin Operasional Industri Manufaktur. Jurnal Energi dan Teknologi Manufaktur (JETM)", 2021, Doi : 10.33795/jetm.v4i01.73.

[3] Lee, Jeng-Dao, Chen-Huan Chang, En-Shuo Cheng, Chia-Chen Kuo and Chia-Ying Hsieh. "Intelligent Robotic Palletizer System." Applied Sciences (2021), doi : https://doi.org/10.3390/app112412159

[4] Zelinsky, Alexander. "Learning OpenCV-Computer Vision with the OpenCV Library" 2009 IEEE Robotics & Automation Magazine 16, ISBN 10: 0-596-51613-4 | ISBN 13: 9780596516130. p

[5] Kusumanto, R. D., N. T. Alan, "Pengolahan Citra Digital Untuk Mendeteksi Obyek Menggunakan Pengolahan Warna Model Normalisasi RGB", 2011, Semantik.

[6] C. Han, H. Ma, W. Zuo, S. Chen and X. Zhang, "A General 6-DOF Industrial Robot Arm Control System Based on Linux and FPGA," 2018 Chinese Control And Decision Conference (CCDC), Shenyang, China, 2018, pp. 1220-1225, doi: 10.1109/CCDC.2018.8407315.

[7] P. Bora and V. Nandi, "Low Cost Shadow Function Based Articulated Robotic Arm" 2015 International Conference on Energy, Power and Environment: Towards Sustainable Growth (ICEPE), Shillong, India, 2015, pp. 1-4, doi: 10.1109/EPETSG.2015.7510079

[8] Y. Zhang, X. Li, H. Zhu and Ning Tan, "Joint-Angle-Drift Remedy Of Three-Link Planar Robot Arm Performing Different Types Of End-Effector Trajectories" 2009 IEEE International Conference on Intelligent Computing and Intelligent Systems, Shanghai, China, 2009, pp. 581-585, doi: 10.1109/ICICISYS.2009.5358320.