Mobile Robot Trolley Using Hand Gesture Recognition Based Image Processing

1st Yudi Arrasyid
Electronics Engineering Major
State Polytechnic Of Cilacap
Cilacap, Indonesia
yudiarrasyid12@gmail.com

2nd Arif Sumardiono
Electronics Engineering Major
State Polytechnic Of Cilacap
Cilacap, Indonesia
arifsumardiono@pnc.ac.id

3rd Erna Alimudin
Electronics Engineering Major
State Polytechnic Of Cilacap
Cilacap, Indonesia
ernaalimudin@gmail.com

***Abstract*— A supermarket trolley is a device used in convenience stores or supermarkets to transport shopping items. The use of trolleys greatly assists in carrying a large quantity of items. However, as the number of purchased items increases, more force is required to manually push or pull the shopping trolley. This can lead to shoppers having to limit the amount of items they purchase to avoid excessive weight. To address this issue, the development of an automatic trolley that can track the user's hand movements is necessary, eliminating the need for manual pushing. In the developed system, image processing is performed using a camera capable of detecting hand gestures. Hand gesture detection has been programmed using the Python programming language and machine learning techniques via Mediapipe. Testing has shown that this trolley successfully follows hand gesture commands, both without load and with a load. The average speed of the trolley is 17.48 cm/s without a load and 7.47 cm/s with a maximum load of 50 kg. The trolley cannot exceed this maximum limit. The battery life of the trolley, with a discharge limit of 10.5V, is approximately 50 minutes without a load and 47 minutes with a 50 kg load. In the use of this automatic trolley, there is a maximum distance of 650 cm between objects and the trolley. For optimal hand gesture detection, the ideal camera usage distance is between 50 and 150 cm.**

Keywords—trolley, hand gesture, camera, machine learning, mediapipe

I. INTRODUCTION

Purchasing food, beverages, cleaning products, and other household items is an essential necessity in daily life. Supermarkets generally offer a wide range of choices in terms of products, brands, and variations at competitive prices. When shopping, it is important for us to have tools that can help carry our groceries to avoid feeling tired. Some available options at supermarkets for transporting groceries include shopping baskets and shopping carts.

Supermarket trolleys are devices used to transport groceries while shopping at supermarkets or stores that provide a variety of products. Typically, these trolleys are made of metal or plastic materials and are equipped with wheels for easy maneuverability within the store[1]. The use of trolleys facilitates humans in carrying and moving large quantities of items. Usually, commonly used trolleys require users to push them to assist in their movement[2].

Users typically use their physical strength to push or pull the trolley when carrying their shopping items. The more items they have, the more effort is required to move the shopping trolley[3]. This can lead to fatigue, especially if users have to walk long distances or use the trolley to carry heavy loads. As a result, shoppers may consider reducing the amount of items they purchase, ultimately reducing their total expenses[4]. Therefore, there is a need for a trolley robot that can follow the user's hand movements, eliminating the need for manual pushing.

In previous research conducted by Fera Sopiana, an automated trolley using color recognition for human tracking with a camera was successfully developed. However, there were some limitations in that study, such as the vulnerability of the camera to detect surrounding colors, which could cause the trolley to move autonomously if the supermarket had colors similar to those used by the user of the trolley. Additionally, the trolley control's inability to move straight also requires improvement to ensure the success and alignment with the planned objectives of the research[5].

The current study aims to focus on improving the technology of the previous invention by replacing the detection method with hand gesture recognition to address the limitations mentioned earlier. The study also aims to enhance the quality and breakthroughs in automatic trolley technology, particularly in achieving straight movement. Significant efforts have been made to develop an intelligent interface between users and computer-based systems based on body movements. Gestures provide an intuitive interface for humans and computers. Thus, such gesture-based interfaces can not only replace conventional interfaces but also be exploited to expand their functionalities. It is hoped that this further research can improve the movement of automated trolleys and contribute to the field of control systems and microcontrollers

II. LITERATURE REVIEW

A. *Shopping Trolley*

A shopping cart is a tool commonly used to carry a large quantity of groceries or food items that will be purchased. By using a shopping cart, we no longer need to carry heavy loads while shopping. Sylvan Nathan Goldman, a store owner in America, invented the shopping cart or trolley on June 4, 1937. The shopping cart is designed as a basket with four wheels and a push handle at the back[6]. In this research, the shopping cart is used as a carrier for goods and to accommodate various components.

B. *Image Processing*

An image or picture is a two-dimensional representation of objects in the visual world. Disciplines related to images include art, human vision, and other fields. An image consists of a collection of pixels or color points that form a two-dimensional picture[7].

Image processing is a technique used to manipulate and analyze images with the goal of altering information or extracting specific features[8]. In this study, image processing is used to process the images of detected hand gestures, enabling the gestures to control the trolley's movement.

C. Computer Vision

Computer vision is a combination of image processing and pattern recognition. It leverages the human visual system's ability to gather information[9]. Computer vision uses digital images from cameras as input and employs neural networks to detect, classify, and identify objects observed by computers[10]. In this final project, computer vision is used to process the visual input from the camera, enabling the recognition of hand objects.

2.1 Skeleton Hand Gesture Landmark Recognition

Skeleton-based recognition is a pattern recognition technique that utilizes a framework of Convolutional Neural Network (CNN) models, which falls under the umbrella of image processing. Various representations of hand datasets can be used to create models for classification purposes[11].

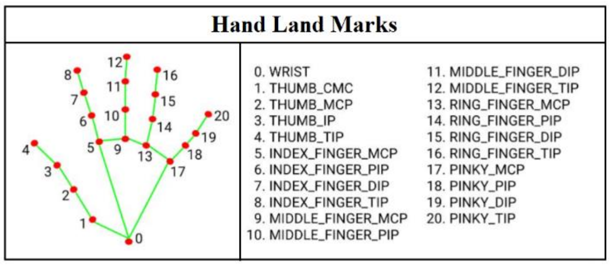


Figure 1. Skeleton Hand Landmark

When an image directly matches the framework model's landmarks, the image is classified according to the given dataset[12]. This method is similar to using deep learning techniques.

2.2 Mediapipe

Mediapipe is a Python library module that provides users with a wide range of functions and algorithms to process image or video data[13]. One of the key strengths of Mediapipe is its ability to efficiently handle time series data and perform pipeline configurations, which facilitates parallel multi-processing. This means that users can leverage the capabilities of Mediapipe to process and analyze large amounts of data in a highly efficient and concurrent manner. By utilizing Mediapipe, developers can easily incorporate complex computer vision and machine learning algorithms into their applications, allowing for advanced data processing and analysis tasks.

Figure 2. MediaPipe

A large and diverse collection of Google data has been used to train a significant number of human body detection and tracking algorithms in MediaPipe[14]. The proposed model employs a framework of nodes and landmarks to track important joints in various aspects of the human body[15]. The resulting 3D shape is then used in a Tensorflow Lite model created by Google developers to establish an information pipeline flow[16].

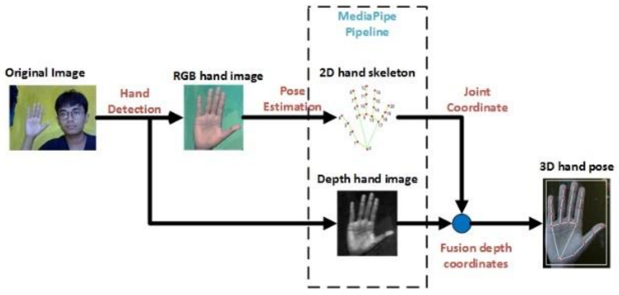
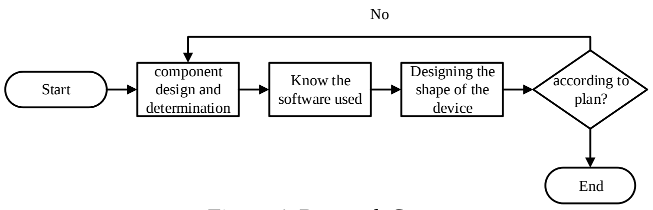


Figure 3. MediaPipe ML Pipeline

III. Method

The research begins with planning the system and selecting the sensors to be used, followed by creating a hardware prototype and developing the design program and circuit assembly. The flowchart depicting the stages of the research is shown in Figure 4.


Start
component design and determination
Know the software used
Designing the shape of the device
according to plan?
No
End
Figure 4. Research Stage

A. Tools and Materials

The design is implemented using a Shopping Cart Trolley, Arduino Mega 2560, Raspberry PI 4B, ToF VL53L0X Sensor, Ultrasonic Sensor HC-SR04, Logitech C270 Webcam, LCD I2C 20x4, Power Window Motor, Servo Motor SG90, Stepdown Module XL4015, Motor Driver BTS7960, MPU6050.

B. Block Diagram

The block diagram of the hand gesture recognition-based moving shopping cart robot is shown in Figure 5.

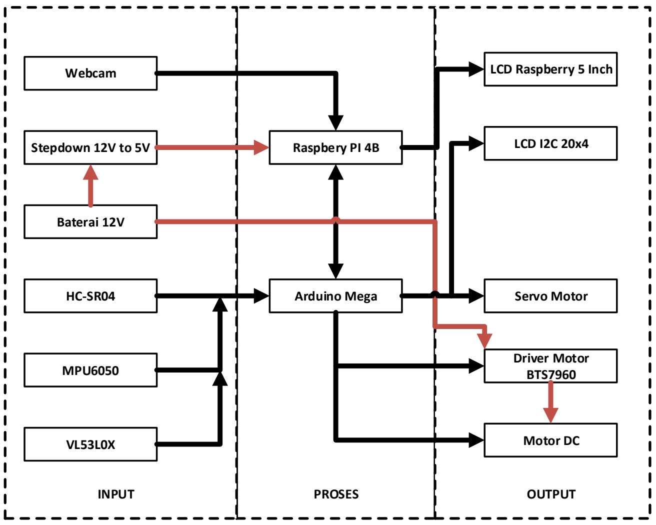


Figure 5. System Block Diagram

In the initial stage, the system operates when a 12VDC battery input voltage is supplied to the XL4015 stepdown module and the BTS7960 Motor Driver. The 12VDC voltage from the stepdown module is reduced to supply power to the Raspberry Pi, which in turn powers the Arduino and the webcam camera. The Arduino is responsible for providing power to all input devices such as sensors and the MPU6050. The Raspberry Pi automatically starts a Python program to initiate the system, and once the program is running, the camera display appears. The Raspberry Pi then sends commands to the Arduino Mega, which performs the control process.

C. Design of Control System

The shopping cart robot uses the Proportional and Derivative principles without using integral. It employs the control system shown in Figure 6.

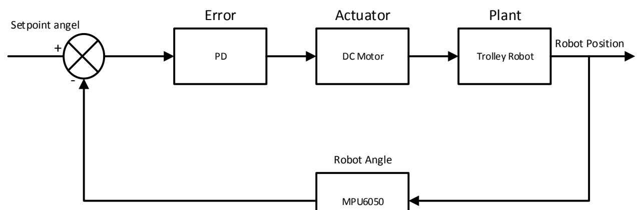


Figure 6. Block Diagram of PD Control System

In Figure 10, when the robot is about to move, the Arduino initially sets the target setpoint variable. Then, the PD control system processes the control to drive the actuator, which is the DC motor, causing the DC motor to move straight. However, if the robot deviates from the setpoint angle due to an error, the MPU6050 sensor detects the current angle of the robot to perform correction using the PD algorithm for the robot. This ensures that the robot stays at the setpoint position. The principle can be summarized by the following formula:

$P - Proporsional$
$D - Derivative$
$(Proporsional * Kp) + (Derivative * Kd) = Angle\ Steering\ Output$

D. Electrical Design

The overall circuit of the Trolley robot system can be seen in Figure 7.

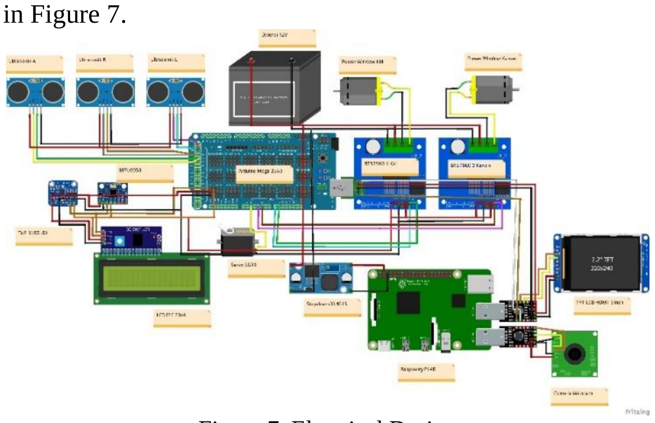


Figure 7. Electrical Design

The electronic circuit in this robot troli is designed to connect and communicate between the Raspberry Pi 4B mini PC and the Arduino Mega microcontroller. Both components are interconnected and communicate through a UART serial connection, which utilizes a USB cable as the data transmission medium

E. Mechanical Design

The mechanical design includes the creation of a framework made of iron, acrylic, and plywood. Iron is chosen as the material for constructing the troli due to its strength, allowing it to support the equipment's weight effectively and provide durability. The troli used has a capacity of 22 liters and is equipped with four wheels for mobility. Acrylic with a thickness of 3mm is selected as the material for constructing the box to house the components and electronic devices.

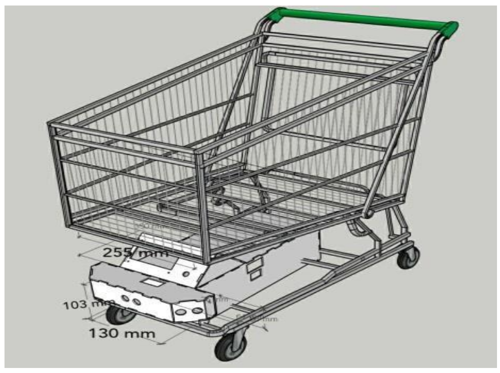


Figure 8. Mechanical Design

F. Flowchart

Flowchart or flow diagram is a standard method used to depict the process of a system. Each step in the system is represented by symbols, and the sequence of steps is indicated by arrows. At this stage, a system plan is created that includes inputs and outputs, which are representations of the data being processed and the information being generated.

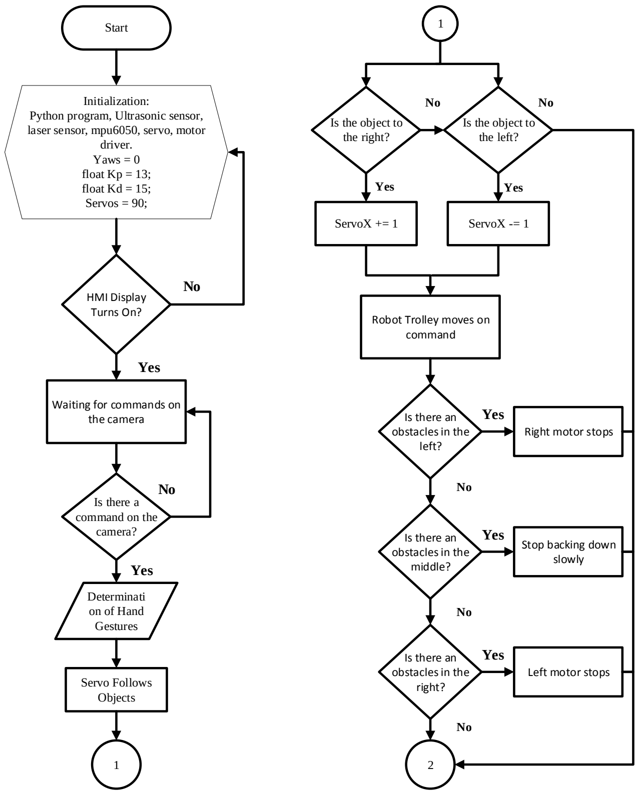
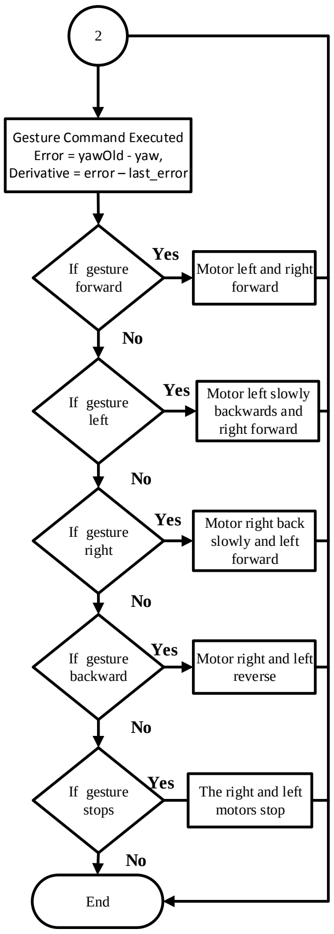

Start
Initialization:
Python program, Ultrasonic sensor, laser sensor, mpu6050, servo, motor driver.
Yaws = 0
float Kp = 13;
float Kd = 15;
Servos = 90;
HMI Display Turns On?
No
Yes
Waiting for commands on the camera
Is there a command on the camera?
No
Yes
Determination of Hand Gestures
Servo Follows Objects
1
1
Is the object to the right?
No
Is the object to the left?
No
Yes
Yes
ServoX += 1
ServoX -= 1
Robot Trolley moves on command
Is there an obstacles in the left?
Yes
Right motor stops
No
Is there an obstacles in the middle?
Yes
Stop backing down slowly
No
Is there an obstacles in the right?
Yes
Left motor stops
No
2

Figure 9. Flowchart System

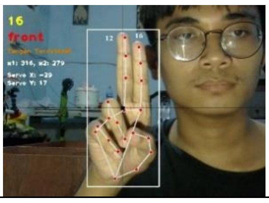
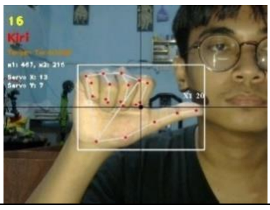
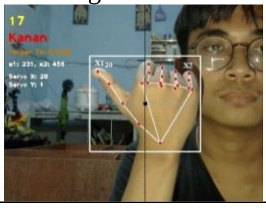
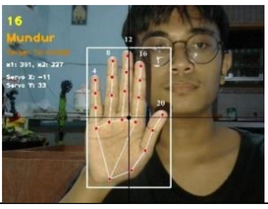
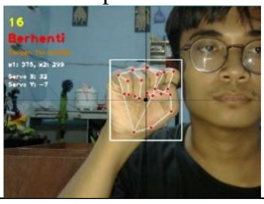

2
Gesture Command Executed
Error = yawOld - yaw,
Derivative = error – last_error
If gesture forward
Yes
Motor left and right forward
No
If gesture left
Yes
Motor left slowly backwards and right forward
No
If gesture right
Yes
Motor right back slowly and left forward
No
If gesture backward
Yes
Motor right and left reverse
No
If gesture stops
Yes
The right and left motors stop
No
End
Figure 10. Flowchart Hand Gesture Control

The Raspberry Pi automatically starts the pre-programmed Python program and is ready to receive input. When the camera detects a hand input, the servo locks the camera at the center of the hand detection box and follows the motion of the hand within the box. When the robot turns, it records the last yaw position until it stops. If the robot detects an obstacle in front of it using the ultrasonic sensor, it tries to avoid the obstacle until the desired condition is achieved.

G. *Design of Trolley Hand Gesture Control*

The commands for the movement of the trolley have been set in the Python programming to detect specific patterns in hand movements, some of which utilize the points of interest within the hand landmarks. In the programming of this trolley robot, a keras training model is not used. However, it is capable of recognizing specific patterns without the need for a training dataset. Several predefined patterns have been programmed and are displayed in Table 1

Table 1. Hand Gesture Pattern

Hand Gesture Pattern	Detection Step
Forward Pattern 	Forward Pattern uses coordinate recognition index_tip(8) && middle_tip(12) > thumb_tip(4) && ring_tip(8) && pinky_tip(20)
Left Pattern 	Left Pattern uses coordinate recognition thumb_tip(4) > pinky_tip(20)
Right Pattern 	Right Pattern uses coordinate recognition thumb_tip(4/X1) < pinky_tip(20/X2)
Backward Pattern 	Backward Pattern uses coordinate recognition index_tip(8) && middle_tip(12) && thumb_tip(4) && ring_tip(8) && pinky_tip(20) > wrist(0)
Stop Pattern 	Stop Pattern uses coordinate recognition index_tip(8) && middle_tip(12) && thumb_tip(4) && ring_tip(8) && pinky_tip(20) == 0

IV. Results

The results of the completed tool are shown in Figure 11 and Figure 12 below.

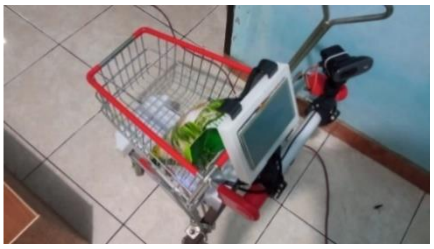


Figure 11. The Finished Tool

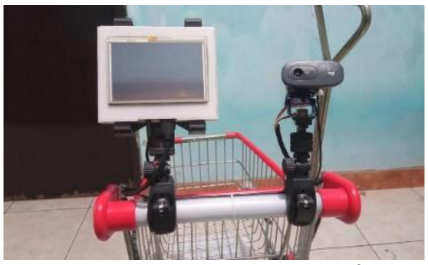


Figure 12. Human Machine Interface

A. *Trolley Movement Based on Commands From Hand Gestures Test*

The hand gesture detection testing is performed by detecting the palm of the hand using the Mediapipe library with the OpenCV module. The testing is repeated multiple times to gather samples for the accuracy of detection, to determine if the hand gesture detection can be successfully performed. The programming is done in Python using the

OpenCV library module. The results of the hand gesture detection using the OpenCV library module can be seen in Figure 13. The detected hand object is indicated by the appearance of a white box located in the area around the detected hand.

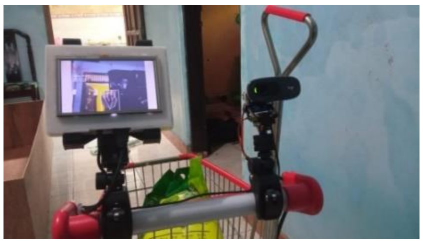
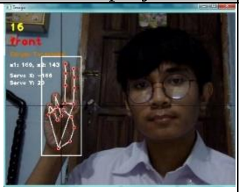
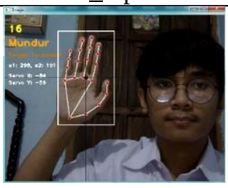
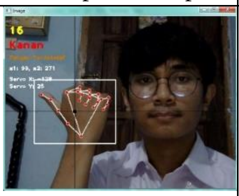
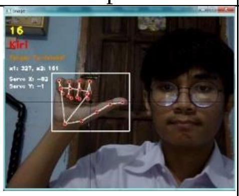
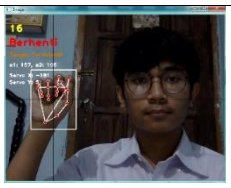


Figure 13. Motion Pattern Detection

The system is ready when the LCD display has opened the camera as shown in Figure 13, we can immediately give orders to the trolley. When we give a hand gesture forward, the trolley will move straight. When we give the hand gesture backwards, the trolley will move backwards. When we give a left hand gesture, the trolley will turn left. When we give the right hand gesture, the trolley will turn right. Meanwhile, if we give a silent hand gesture, the trolley will stop.

Table 2. Testing Trolley Movement Based on Commands From Hand Gestures

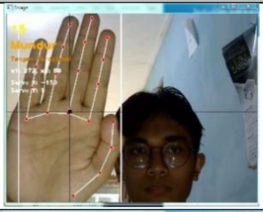
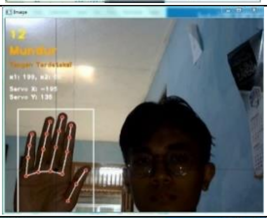
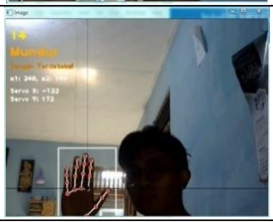
Condition	Display	Movement
Forward	 Index_tip>index_dip Middle_tip>middle_dip	Forward
Backward	 all_tip > all_dip	Backward
Right	 thumb_tip<pinky_tip	Right
Left	 Thumb_tip>pinky_tip	Left
Stop	 all_tip<all_dip	Stop

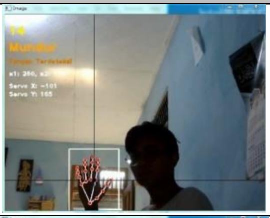
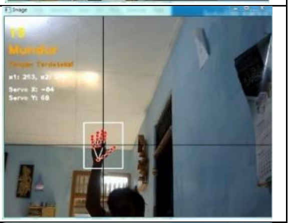
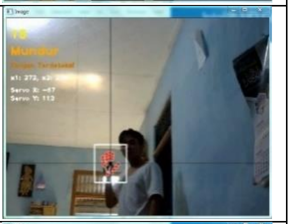
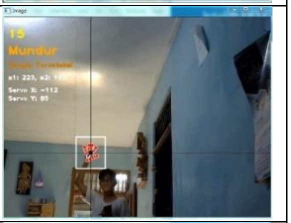
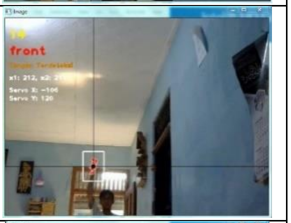
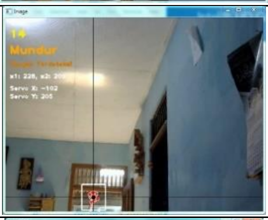
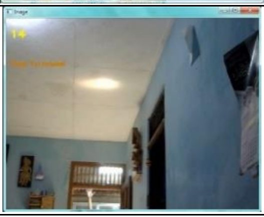
The test results according to table 10 as a whole show that the trolley has the ability to move forward, turn right or left, and rotate following the direction of object movement based on hand pattern movements. The webcam used managed to read the hand gesture object well, so that the trolley can follow the hand gesture command. The sensors and motors used operate according to the program being executed, ensuring that the trolley can move smoothly according to instructions. However, it should be noted that unstable light intensity may interfere with the webcam's ability to read objects, resulting in loss of object detection and causing the trolley to not move effectively.

B. *Hand Gesture Detection Distance on the Camera Test*

This test was carried out with the aim of finding the optimal distance between the camera and hand gesture objects, so as to produce clear and focused images. In this test, the camera is placed at different distances from the same object, and the results are carefully observed to determine the best distance that can produce clear and sharp images. Furthermore, the results of the camera distance testing are carefully recorded and analyzed in depth to determine the optimal distance required. The results of the analysis are then disclosed in detail in Table 3 in the form of a table of camera distances to objects

Table 3. Camera Distance Against Objects

No.	Camera Distance	Detection	Detection Results
1.	25 cm	Detected	
2.	50 cm	Detected	
3.	100 cm	Detected	

No.	Camera Distance	Detection	Detection Results
4.	150 cm	Detected	
5.	200 cm	Detected	
6.	300 cm	Detected	
7.	400 cm	Detected	
8.	500 cm	Detected	
9.	650 cm	Detected	
10.	700 cm	Not Detected	

In Table 3, it can be seen that the camera has the ability to detect objects at a minimum distance of 25 cm to a maximum distance of 700 cm. However, after passing a distance of 650 cm, the camera is no longer able to recognize the object in hand. This shows that the farther the hand is detected by the camera, the more difficult it is to recognize the shape of the hand. Therefore, when the object is too far away, the shape of the joints in the hand cannot be identified clearly. Thus, it can be seen that the ideal distance for detecting hand objects is in the range of 50 to 150 cm. In this range, the camera has a good ability to recognize and identify hand shapes with sufficient accuracy.

C. *Servo Motor Object Tracking Against Hand Gesture Objects Test*

Tests were carried out to determine the direction of motion of the servo motor when following a hand object, testing was carried out using a servo motor which has a maximum angle of 180°. In the initial position, the servo motor is set to face forward at an angle of 90°. When an object is detected, the servo will move to follow the object, when the object is on the left, the servo will reduce its degree, so that the left servo angle has an angle range of 0-90°. Meanwhile, when the object is to the right, the servo will increase its angle, so that the right servo angle has an angle range of 90-180°. This test aims to measure the range of angles when the object is left and right to see if the servo can adjust the object. Object tracking is tested using a protractor to match the position of the object with the actual angle of the servo. The test results are shown in Table 4. and Figure 14.

$$\%\text{error} = \frac{\text{actual value-reference value}}{\text{reference value}} \times 100\%.................(4)$$

This error test aims to measure the ability of the sensor to receive stimuli and the response of the servo when there is an object. The parameters used are the angular position of the object as the reference value and the actual angle as the actual value by looking at the angle on the LCD.

Table 4. Response Tracking Test Results

Try	Angel Position Object	Actual Angel	Deviation	Error(%)
1	0°	0°	0°	0%
2	30°	28°	2°	6,6%
3	45°	47°	2°	4,4%
4	60°	61°	1°	1,6%
5	90°	90°	0°	0%
6	120°	117°	2°	2,5%
7	135°	134°	1°	0,74%
8	150°	152°	2°	1,3%
9	165°	166°	1°	0,6%
10	180°	180°	0°	0%
	Average error (%)			1,77%

Based on the test results listed in Table 4, the following conclusion can be drawn: when the angle of the servo detects a hand gesture object less than 90°, the servo will decrease its angle to less than 90°. Conversely, if the object's angle exceeds 90°, the servo will increase the angle to more than 90°. Throughout the object detection process, the average servo error response was recorded at 1,77%. This occurs because the servo continuously adjusts its angle by following commands while observing the object. As a result, the servo attempts to correct and align its position to match the detected object, leading to the observed error of 1,77%.

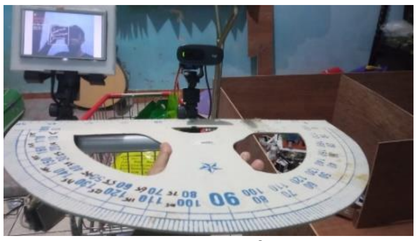


Figure 14. Initial Position of Angle Servo 90°

D. Trolley Straight Motion Test

The straight motion testing of the trolley, also known as the PD error testing, is conducted to determine if the trolley can move in a straight line or not. The testing is performed using a reference arc degree value and a measuring tape. The trolley is tested to move along a predefined path with a distance of 150cm. The percentage of error and the angle of error when the trolley deviates from the intended path are measured. The testing results are then recorded in Table 5.

To calculate the Proportional-Derivative straight motion, the following formula is used:

$derivative = error - last\_error$................................(2)

$(error \times Kp) + (derivative \times Kd) = angle$ ....(3)

Table 5. Derivative Proportional Straight Motion Testing

Try-	Distance	Angel Setpoint	Actual Angel	Deviation	error
1	150 cm	90°	85°	5°	5,5%
2	150 cm	90°	90°	0°	0%
3	150 cm	90°	92°	2°	2,2%
4	150 cm	90°	90°	0°	0%
5	150 cm	90°	95°	5°	5,5%
6	150 cm	90°	90°	0°	0%
7	150 cm	90°	89°	1°	1,1%
8	150 cm	90°	85°	5°	5,5%
9	150 cm	90°	90°	0°	0%
10	150 cm	90°	90°	0°	0%
Average error				89,6°	1,98%

Based on the testing, it was found that the average actual angle of the trolley is 89.6°, and the cause of this deviation is the inability of the front wheels to maneuver properly. The front wheels sometimes fail to rotate, resulting in a slight change in the trolley's angle when reaching the destination. However, the trolley remains within the intended path. As a result, the use of PD (Proportional-Derivative) control in the trolley greatly influences its movement. The trolley only has a small error of 1,98% in terms of the actual angle.

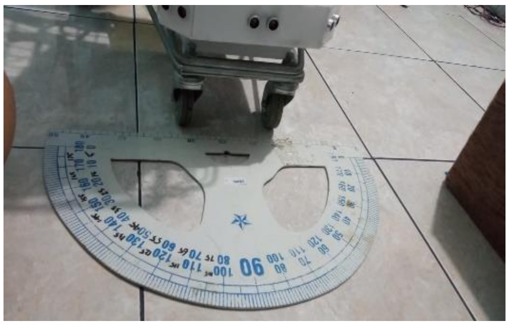


Figure 15. Angle Straight Motion Error Test 90°

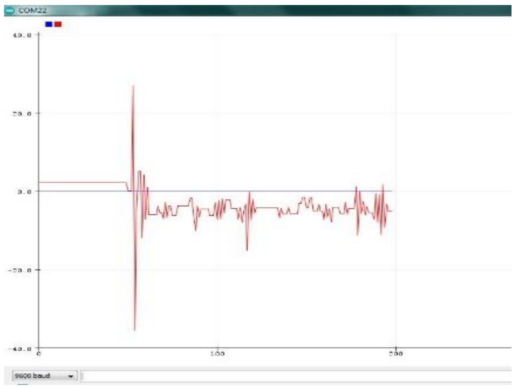


Figure 16. Proportional And Derivative Motion Graphs

Figure 16 shows the relationship between the movement of the trolley and the target setpoint, where the blue line pointing in the direction of 0.0 is the target setpoint or the robot's straight line while the red line is the angle value of the trolley robot when it moves forward, if the red graph is downwards it indicates the robot is moving to the right, whereas if the red graph is up then the robot moves left.

E. No-Load Trolley Speed Test

The testing is conducted to measure the speed of the trolley when there is no load placed on it. In this condition, the trolley only carries its own weight and the components used to construct it. Information regarding the total weight of the trolley and its components can be found in Table 6.

Table 6. Trolley Component Weight

No.	Materials	Amount	Weight
1.	Shopping Trolley	1	8 Kg
2.	Webcam Logitech C270	1	0,3 Kg
3.	Raspberry Pi 4B	1	0,65 Kg
4.	Arduino Mega 2560	1	0,037 Kg
5.	Ultrasonic Sensor HCSR04	3	0,03 Kg
6.	SG995 Servo Motor	1	0,06 Kg
8.	BTS7960 Motor Driver	2	0,1 Kg
9.	Stepdown Module XL4015	1	0,02 Kg
10.	LCD Raspberry 5 Inch	1	0.125 Kg
11.	MPU6050 Module	1	0.01 Kg
12.	Accu Battery	1	2 Kg
13.	LCD Frame	1	0.013 Kg
14.	Servo Frame	1	0.01 Kg
15.	LCD and Servo Clamp	2	0.6 Kg
16.	Acrylic Box	2	2 Kg
Total Amount			13,955 Kg

Based on the provided data, it appears that the total weight of all the components on the trolley is 13.955 kg. The testing is conducted three times to measure the speed of the trolley when it moves a distance of 1.5 meters. During this distance, the time taken by the trolley to cover the 1.5-meter distance is measured. The resulting time data will be used in the appropriate physics formula.

$v = \frac{s}{t}$..(1)

The physics formula applicable to this testing states that distance (s) is directly proportional to time (t), and velocity (v) is the ratio of distance to time. The PWM speed of the DC motor used is 153.5. To find the average speed of the trolley, the testing results can be observed in Table 7.

Table 7. Speed Test Trolley No-Load

Try	Distance	Time	Speed
1	150 cm	8,32 s	18,32 cm/s
2	150 cm	9,06 s	16,55 cm/s
3	150 cm	8,36 s	17,96 cm/s
Average Speed		8,58 s	17,61 cm/s

During the testing, the speed of the trolley varied due to several factors. One of them was the instability of the lighting conditions, which caused inconsistent hand detection readings and hindered the movement of the trolley's motor. For this reason, three experiments were conducted to find the average speed of the unloaded trolley. In the end, an average speed of 17.61 cm/s was determined.

Table 8. Speed Trolley With-Load

No.	Weight Load	Distance	Time	Speed
1	5 Kg	150 cm	9,26 s	16,19 cm/s
2	10 Kg	150 cm	9,77 s	15.35 cm/s
3	15 Kg	150 cm	11,15 s	13,45 cm/s
4	20 Kg	150 cm	13,12 s	11,43 cm/s
5	30 Kg	150 cm	15,64 s	9,59 cm/s
6	50 Kg	150 cm	20,08 s	7,47 cm/s

Figure 17. Testing Speed Trolley Load 30Kg

The testing results indicate that there is variation in the speed of the trolley under different loads. As the load carried by the trolley becomes heavier, the speed of the trolley decreases. The instability of the lighting conditions during the testing also caused the trolley to occasionally stop detecting hands, which affected the recorded speed in Table 8.

F. Battery Life Test

The battery runtime testing is conducted to determine how long the battery can last when the trolley is in use. The testing is performed from the fully charged state of the battery until it is completely drained. The battery runtime depends on the power consumed by the load from the battery. Table 9 shows the power usage by the load from the battery.

Table 9. Component Power

No.	Component	Current (A)	Voltage (V)	Power (Watt)
1.	Raspberry Pi 4B	0,9	5,02	4
2.	Arduino Mega	0,3	5,02	2
3.	LCD Raspberry 5 inch	0,46	5,02	2,3
4.	LCD I2C 20x4	0,2	5,02	1
5.	Motor Servo MG90	0,063	5,02	0,35
6.	Sensor Ultrasonic 1	0,002	5,02	0,01
7.	Sensor Ultrasonic 2	0,002	5,02	0,01
8.	Sensor Ultrasonic 3	0,002	5,02	0,01
9.	Sensor VL53L0X	0,004	5,02	0,014
10.	Driver BTS7960 1	1,1	13,2	14,52
11.	Driver BTS7960 2	1,1	13,2	14,52

1. Total power while stationary =4 W + 2 W + 2,3 W + 1 W + 0,35 W + 0,01 W + 0,01 W + 0,01 W + 0,014 W =9,6 W

2. Total power while moving =4 W + 2 W + 2,3 W + 1 W + 0,35 W + 0,01 W + 0,01 W + 0,01 W + 0,014 W + 14,52 W + 14,52 W =38,73 W

2. Battery Capacity =V x I
=12V x 7,5 Ah
=90 Wh

3. DoD Battery = Battery Capacity x 50%
= 90 x 50%
= 45 Wh

4. Time of use without moving $= \frac{DoD\ Battery}{Total\ Power} = \frac{45}{9,6} = 4,7\ Hour$

5. Usage time while moving $=\frac{\text{DoD Battery}}{\text{Total Power}} = \frac{45}{38,73} = 1,16\ \text{Hour}$

Table 10. The Battery Runtime Without Load

No.	Time Use	Battery Voltage	Battery Percentage
1.	1 minute	13,3V	99%
2.	10 minute	12,7V	86,4%
3.	25 minute	11,8V	73,8%
4.	35 minute	11,2V	63,8%
5.	45 minute	10,9V	56,7%
6.	55 minute	10,5V	31%

Table 11. The Battery Runtime With Load

No.	Time Use	Battery Voltage	Battery Percentage
1.	1 minute	13,2V	99%
2.	10 minute	12,6V	84,2%
3.	25 minute	11,6V	62%
4.	35 minute	11,8V	46,8%
5.	47 minute	10,5V	31%

Based on the calculations, it is found that the battery can last for 1.16 hours with a Depth of Discharge (DoD) of 50%.

In addition to the calculations, the battery is also tested directly from a fully charged state until it is completely depleted when used on the unloaded trolley. In the first experiment with the unloaded trolley, it was observed that the battery could last for 55 minutes when the trolley was moving without a load and 47 minutes when the trolley was moving with a load.

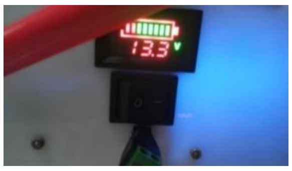


Figure 18. Battery Voltage When Full

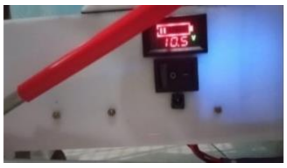


Figure 19. Battery Voltage When Empty

G. *Ultrasonic Sensors and VL53L0X ToF Sensors Test*

The testing is conducted to determine at what distance the trolley will stop when encountering an obstacle. The ultrasonic sensors and a laser sensor are used in this experiment, with three ultrasonic sensors and one laser sensor placed underneath the trolley. These sensors are utilized to stop and reverse the trolley when an obstacle is detected within a distance of less than 20cm from the trolley's position. Programming is implemented for both the ultrasonic and laser sensors. This distance threshold applies to all the ultrasonic and laser sensors. The results of the ultrasonic and laser sensor testing can be observed in Figure 20 and Table 12

Table 12. Distance of Ultrasonic Sensors and Lasers on Obstacles

Actual Distance	Detected Distance Sensor				Detection Results
	Ultrasonic A	Ultrasonic B	Ultrasonik C	Time Of Flight	
10 cm	10	10	11	15	Terdeteksi
20 cm	20	21	20	23	Terdeteksi
30 cm	31	31	30	32	Terdeteksi
40 cm	40	41	40	41	Terdeteksi

Figure 20. Sensor Distance Display on LCD

In Figure 20, there are several indicator displays showing the results of the ultrasonic and laser sensor testing. The label "distance" is used on these indicators to indicate the measured distance between the trolley and the object detected by the sensor. On the LCD screen, there are numbers 26, 32, 58, and 99, indicating the ultrasonic sensor's distance to an object on the right side at a distance of 26 cm, on the front side at a distance of 32 cm, on the left side at a distance of 58 cm, and on the back side from the laser sensor at a distance of 99 cm. These three distance values are used to make the trolley avoid obstacles while moving.

V. CONCLUSION

Based on the research conducted, the trolley robot can perform several movements such as going forward, turning right and left, and rotating according to the commands given and detected. The trolley robot also has a maximum load limit that can be transported of 50 Kg, and the average speed of the trolley when there is no load is 17.61 cm/s. Experiments show that a battery with a discharge limit of 10.5V can last for 50 minutes without load and 47 minutes with 5Kg load. The trolley robot system that uses a camera and Raspberry Pi to recognize hand movements has proven to be effective according to the plan that had been prepared previously. This system can accurately detect various hand gestures such as forward, backward, right, left, and stop with an adequate degree of accuracy. However, there are several factors that affect object detection, such as light intensity and object distance to the webcam. The webcam used managed to read the hand gesture object well, so that the trolley can follow the hand gesture command.

Hand gesture detection also has the ability to detect hand gesture objects at a minimum distance of 25 cm to a maximum distance of 700 cm. However, after passing a distance of 650 cm, the camera is no longer able to recognize the object in hand. This shows that the farther the hand is detected by the camera, the more difficult it is to recognize the shape of the hand. so the ideal distance to detect hand objects is in the range of 50 to 150. when the camera detects hand gesture objects and the angle of the hand gesture servo is less than 90°, the servo will reduce its angle to less than 90°. Conversely, if the object's angle exceeds 90°, the servo will increase the angle to more than 90°. During the object detection process, the average servo error response was recorded at 1,77%.

ACKNOWLEDGMENT

Thank you to the supervisors lecture I and II who have taken the time to test the tool, as well as to the two parents who support the making of the tool and someone who always prays for me. I thank you so much that this tool can be created.

REFERENCES

[1] S. Srivastava, S. Rai, S. Kumar, S. Bhuhsan, and D. Pradhan, "IoT based Human Guided Smart Shopping Cart System for Shopping Center," *Saudi J. Eng. Technol.*, vol. 5, no. 6, pp. 278–284, 2020, doi: 10.36348/sjet.2020.v05i06.004.

[2] O. J. Oyejide, M. O. Okwu, L. K. Tartibu, and O. I. Olayode, "Development of Sensor Controlled Convertible Cart-Trolley," *Procedia CIRP*, vol. 91, no. March, pp. 71–79, 2020, doi: 10.1016/j.procir.2020.03.097.

[3] D. Nouman, M. B. K. Keyani, M. F. Ullah, and M. I. H. Raza, "Development of Intelligent Shopping Cart using Pixy Sensor with Arduino Controller," in *International Conference on Contemporary Academic Research*, 2023, pp. 70–74.

[4] J. Dube, N. Lahekar, R. Bramhane, and N. Bawane, "Ergo Comfy Sheet Handling Trolley," in *International Conference on Communication and Information Processing*, 2022, pp. 1–7. doi: 10.2139/ssrn.4297120.

[5] F. S. Anggreyani and E. Alimudin, "Shopping Trolley Automatically Move Following Color With Camera," *Politeknik Negeri Cilacap*, 2022.

[6] A. D. Pratiwi and T. S. Pambudi, "Design of an

Independent Shopping Trolley for Weekly Shopping Needs at Modern Markets (Case Study: Batununggal Indah Modern Market).," *e-Proceeding Art Des.*, vol. 7, no. 2, pp. 5673–5682, 2020.

[7] J. Jumadi, Y. Yupianti, and D. Sartika, "Digital Image Processing for Object Identification Using Hierarchical Agglomerative Clustering Methods" *JST(Journal of Sci and Tec)*, vol. 10, no. 2, pp. 148–156, 2021, doi: 10.23887/jstundiksha.v10i2.33636.

[8] D. D. Affifah, Y. Permanasari, and R. Respitawulan, "Convolution Techniques in Deep Learning for Image Processing," *Bandung Conference Series Mathematics*, vol. 2, no. 2, pp. 103–112, 2022.

[9] A. A. Chellsya *et al.*, "Implementation of Computer Vision in Detecting Non-Helmet Violations in Motorists" *e-Proceeding Applied Science*, vol. 9, no. 1, pp. 55–71, 2023.

[10] M. A. Masril and D. P. Caniago, "Optimizing Computer Vision Technology in Industrial Robots as Object Movers Based on Color" *ELKOMIKA: Journal of Electrical Energy Engineering, Telecommunication Engineering, and Electronics Engineering*, vol. 11, no. 1, pp. 46–57, 2023.

[11] Oguntimilehin and Olatunji, "A Review of Sign Language Recognition Techniques," *International Journal of Information Systems and Computer Sciences*, vol. 10, no. 6, p. 2021, 2021.

[12] M. Oudah, A. Al-Naji, and J. Chahl, "Hand Gesture Recognition Based on Computer Vision: A Review of Techniques," *Journal Imaging*, vol. 6, no. 8, pp. 2–29, 2020, doi: 10.3390/JIMAGING6080073.

[13] S. Budiman, S. Lestanti, S. M. Evandri, and R. K. Putri, "Recognition of finger gestures to control volume on a computer using the OpenCV library and MediaPipe" *Journal of Information Technology Science*, vol. 16, no. 2, pp. 223–232, 2022.

[14] F. Zhang *et al.*, "MediaPipe Hands: On-device Real-time Hand Tracking," 2020, [Online]. Available: http://arxiv.org/abs/2006.10214

[15] M. J. Hussain *et al.*, "Intelligent Sign Language Recognition System for E-Learning Context," *Computers, Materials & Continu.*, vol. 72, no. 3, pp. 5327–5343, 2022, doi: 10.32604/cmc.2022.025953.

[16] R. Kate, P. Brahmabhatt, and S. Dhopte, "Hand Gesture Recognition System Using Holistic Mediapipe," *IRJET*, vol. 9, no. 6, pp. 862–865, 2022.