

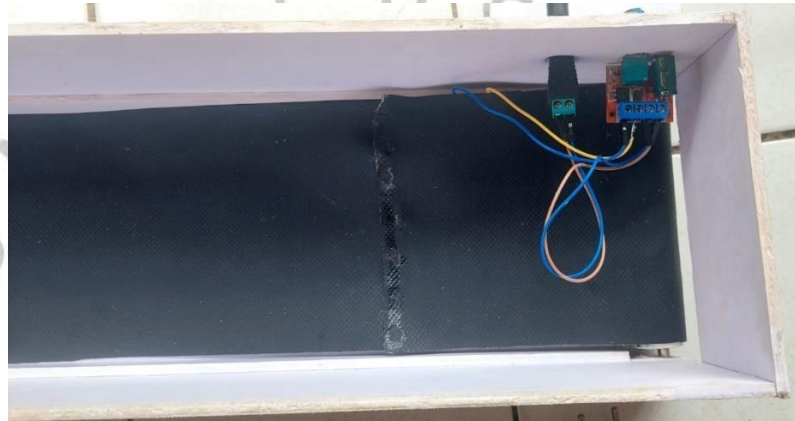
## BAB V HASIL DAN PEMBAHASAN

### 5.1 Hasil

Rancangan yang telah dibuat sebelumnya akan direalisasikan sesuai dengan spesifikasi yang telah ditetapkan. Hasil dari rancangan tersebut akan dijelaskan secara detail pada bagian berikut.

#### 5.1.1 Hasil Perakitan

Pada bagian ini merupakan hasil dari perakitan sistem yang dibuat berdasarkan kebutuhan komponen pada spesifikasi perangkat keras sebelumnya. Berikut merupakan hasil akhir dari prototipe sistem yang telah dibuat.



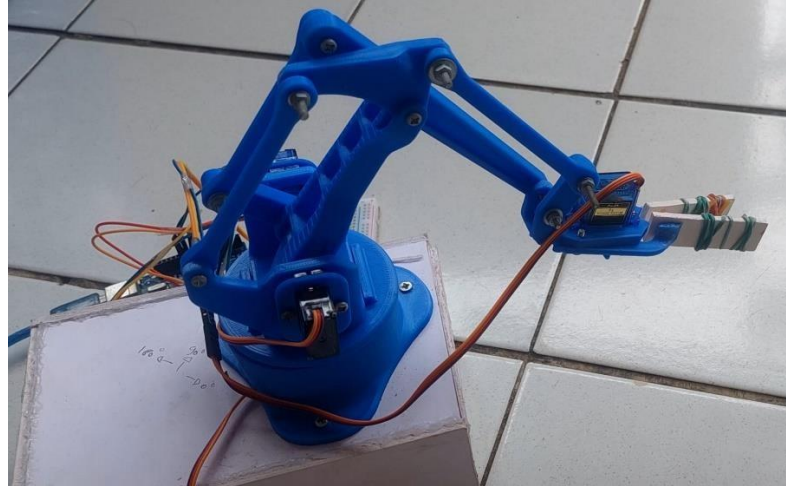
*Gambar 5. 1 Perancangan Modul L298N*

Gambar 5.1 di atas merupakan sebuah rangkaian Module L298N yang terhubung dengan Motor DC dan Addaptor 15V.



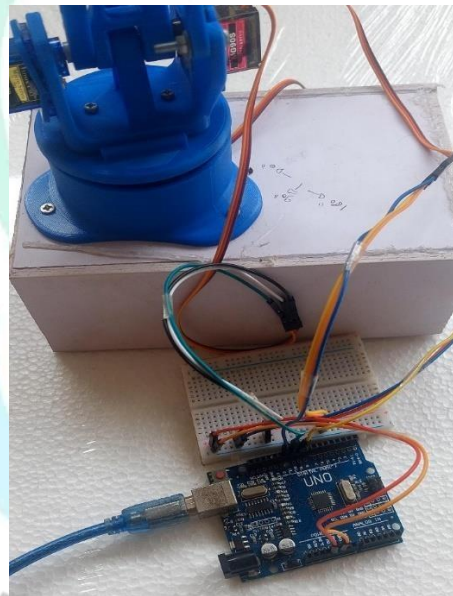
*Gambar 5. 2 Rangkaian Motor DC*

Gambar 5.2 di atas merupakan rangkaian dari Motor DC yang terhubung dengan Addaptor 15V dan Module L298N yang dikontrol oleh Potensiometer untuk mengatur kecepatan Motor DC.



Gambar 5. 3 Rancangan Lengan Robot

Gambar 5.3 di atas merupakan rancangan atau desain dari lengan robot yang dilengkapi dengan 3 Servo Motor, fungsi servo tersebut digunakan untuk menggerakkan lengan robot.



Gambar 5. 4 pengendali lengan robot dengan Arduino Uno R3

Gambar 5.4 merupakan rangkaian elektronik dari lengan robot atau 3 buah servo motor, dengan pin 9 servo pada bagian *Base*, pin 10 pada bagian *Elbow* dan pin 11 pada bagian *Gripper*.

## 5.1.2 Program

Pada pembahasan kali ini menjelaskan proses terkait pemrograman pada sistem yang telah dibuat.

### 5.1.2.1 Visi Komputer Untuk Klasifikasi

Berikut di bawah ini merupakan hasil dari pemrograman Visi Komputer untuk klasifikasi objek .

Tabel 5. 1 Program Klasifikasi Objek

Tahap ke	Kode	Penjelasan
1	<pre> # Function to parse command line arguments def parse_opt():     """Parses command-line arguments for YOLOv5 detection, setting inference options and model configurations."""     parser = argparse.ArgumentParser()     parser.add_argument("--weights", type=str, default=str(ROOT / "best.pt"), help="model path or triton URL")     parser.add_argument("--source", type=str, default="0", help="file/dir/URL/glob/screen/0 (webcam)")     parser.add_argument("--imgsz", "--img", "--img-size", nargs="+", type=int, default=[320, 240], help="inference size h,w")     parser.add_argument("--port", type=str, default="/dev/ttyUSB0", help="serial port for Arduino communication")     parser.add_argument("--device", default="cpu", help="device to run model on (e.g., 'cuda' or 'cpu')")     opt = parser.parse_args()     opt.imgsz *= 2 if len(opt.imgsz) == 1 else 1 # expand     print(vars(opt))     return opt         </pre>	Digunakan untuk membaca argument dari baris perintah untuk deteksi YoloV5, pengaturan opsi inferensi, dan konfigurasi model
2	<pre> # Load YOLOv5 model with specified weights and device device = args.device if torch.cuda.is_available() else 'cpu' model = torch.hub.load('ultralytics/yolov5', 'custom', path=args.weights, force_reload=True).to(device)         </pre>	Memuat model Yolov5 dengan bobot dan perangkat yang ditentukan
3	<pre> # Setup serial communication with Arduino ser = serial.Serial(args.port, 9600, timeout=1)         </pre>	Mengatur komunikasi serial dengan Arduino
4	<pre> # Function to send commands to robot arm via serial def control_robot_arm(object_type):     if object_type == "plastic bottle":         ser.write(b'plastic\n')     elif object_type == "glass bottle":         ser.write(b'glass\n')     elif object_type == "can":         ser.write(b'can\n')         </pre>	Untuk mengirim perintah ke lengan robot melalui serial Arduino

5	<pre># Function for object detection with processing time in ms def detect_objects(frame):     start_time = time.time() # Start time for processing time      results = model(frame)     labels, cords = results.xyxy[0][:-1].cpu().numpy(), results.xyxy[0][:-1].cpu().numpy()      detected_objects = []</pre>	<p>Untuk deteksi objek dengan waktu pemrosesan dalam ms</p>
6		<p>Digunakan untuk menghitung ambang batas kepercayaan pada jenis objek yang sudah dilatih, misalnya 0.5 merupakan ambang batas minimum Tingkat kepercayaan pada objek yang sudah dilatih.</p>
7	<pre># Draw rectangle around detected object x1, y1, x2, y2 = int(x1*frame.shape[1]), int(y1*frame.shape[0]), int(x2*frame.shape[1]), int(y2*frame.shape[0]) cv2.rectangle(frame, (x1, y1), (x2, y2), (0, 255, 0), 2)</pre>	<p>Digunakan untuk menggambar box atau kotak persegi pada objek yang terdeteksi.</p>
8	<pre># Add object name and confidence score confidence_text = f'{detected_objects[-1]}: {conf * 100:.2f}%' cv2.putText(frame, confidence_text, (x1, y1 - 10), cv2.FONT_HERSHEY_SIMPLEX, 0.6, (0, 255, 0), 2)</pre>	<p>Digunakan untuk menambahkan nama pada objek yang terdeteksi dan nilai kepercayaan objek.</p>
9	<pre># Calculate processing time in ms processing_time_ms = (time.time() - start_time) * 1000 print(f"Processing time per frame: {processing_time_ms:.2f} ms")  return detected_objects, frame</pre>	<p>Menghitung waktu pemrosesan program dalam milidetik.</p>

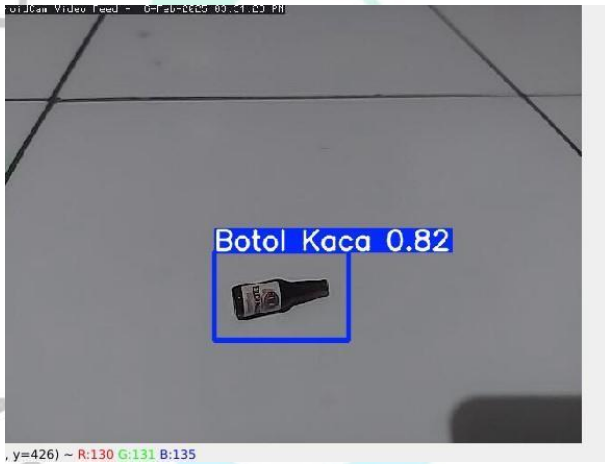

10	<pre># Initialize Webcam or Video Source cap = cv2.VideoCapture(args.source) cap.set(cv2.CAP_PROP_FRAME_WIDTH, args.imgsz[0]) # Set frame width cap.set(cv2.CAP_PROP_FRAME_HEIGHT, args.imgsz[1]) # Set frame height</pre>	<p>Menginisiasi webcam atau sumber video. Yang digunakan untuk mengatur tinggi atau lebar pada frame.</p>
11	<pre># Function to capture frames def capture_frames():     while True:         ret, frame = cap.read()         if not ret:             print("Failed to grab frame")             break         if not frame_queue.full():             frame_queue.put(frame)</pre>	<p>Digunakan untuk menangkap frame.</p>
12	<pre>def process_frames():     frame_count = 0     start_time = time.time()      while True:         if not frame_queue.empty():             frame = frame_queue.get()              # Convert frame to RGB before detection as model expects RGB input             rgb_frame = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)             detected_objects, processed_frame = detect_objects(rgb_frame)              # Calculate FPS             frame_count += 1             elapsed_time = time.time() - start_time             if elapsed_time &gt; 0:                 fps = frame_count / elapsed_time              # Print detected objects and FPS             if detected_objects:                 print(f"Detected objects: {' ', '.join(detected_objects)}")                 control_robot_arm(detected_objects[0]) # Send command for the first detected object             else:                 print("No objects detected")</pre>	<p>Digunakan untuk memproses frame dan konversi frame ke RGB sebelum deteksi.</p>
13	<pre># Calculate FPS frame_count += 1 elapsed_time = time.time() - start_time if elapsed_time &gt; 0:     fps = frame_count / elapsed_time</pre>	<p>Untuk menghitung FPS.</p>
14	<pre># Print detected objects and FPS if detected_objects:     print(f"Detected objects: {' ', '.join(detected_objects)}")     control_robot_arm(detected_objects[0]) # Send command for the first detected object else:     print("No objects detected")</pre>	<p>Untuk menampilkan objek yang terdeteksi, kecepatan FPS, dan mengirim perintah objek yang terdeteksi ke lengan robot melalui serial Arduino.</p>

15	<pre># Show FPS on the processed frame cv2.putText(processed_frame, f'FPS: {fps:.2f}', (10, 30), cv2.FONT_HERSHEY_SIMPLEX, 0.6, (255, 0, 0), 2)  # Show processed frame cv2.imshow("Object Detection", processed_frame)</pre>	<p>Untuk menampilkan FPS pada frame hasil pemrosesan dan menampilkan frame hasil pemrosesan.</p>
----	---	--

### 5.1.2.2 Hasil Pelatihan Model Yolov5

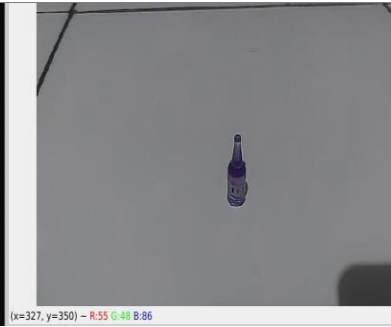
Hasil dari pelatihan model dataset sistem cerdas pemilah sampah daur ulang berbasis klasifikasi objek dan menggunakan lengan robot telah diklasifikasikan sesuai dengan tahapan yang telah dijelaskan pada bab 4 yaitu,

tabel 5. 2 Hasil Pelatihan Yolov5

NO	Hasil	Keterangan
1.		<p>Dari hasil program yang telah dijalankan sebanyak sepuluh kali, Botol Kaca terdeteksi dengan tingkat akurasi tertinggi mencapai 0.82 dari 1.00 atau setara dengan 82% dari 100%.</p>
2.		<p>Dari hasil program yang telah dijalankan sebanyak sepuluh kali, Botol Kaca terdeteksi dengan tingkat akurasi terendah mencapai 0.26 dari 1.00 atau setara dengan 26% dari 100%.</p>

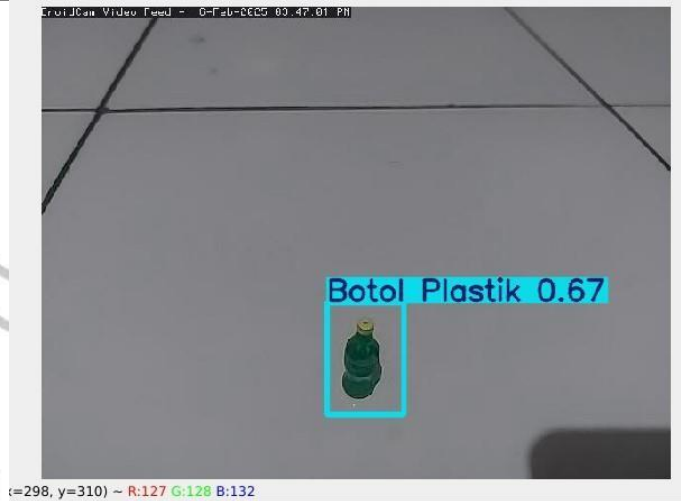
3.

```
256x320 (no detections), 192.9ms  
256x320 (no detections), 190.0ms  
256x320 (no detections), 155.5ms  
256x320 (no detections), 199.7ms  
256x320 (no detections), 204.8ms  
256x320 (no detections), 183.9ms  
256x320 (no detections), 167.8ms  
256x320 (no detections), 123.3ms  
256x320 (no detections), 153.6ms  
256x320 (no detections), 161.8ms  
256x320 (no detections), 151.3ms  
256x320 (no detections), 169.8ms  
256x320 (no detections), 163.9ms
```



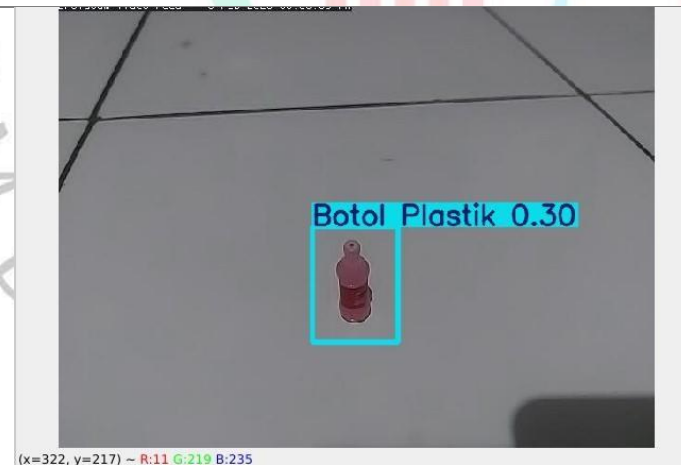
Dari hasil program yang telah dijalankan sebanyak satu botol kaca dari sembilan tidak dapat terdeteksi atau gagal dalam mendeteksi objek tersebut.

4.



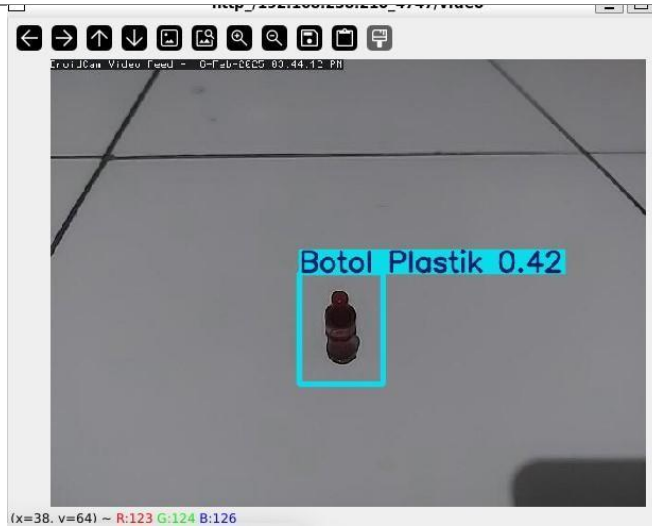
Dari hasil program yang telah dijalankan sebanyak 10 kali, 1 botol plastik terdeteksi dengan tingkat akurasi tertinggi mencapai 0.67 dari 1.00 atau setara dengan 67% dari 100% dari 9 botol plastik.

5.



Dari hasil program yang telah dijalankan 1 botol plastik dari 10 botol terdeteksi dengan tingkat akurasi terendah mencapai 0.30 dari 1.00 atau setara dengan 30% dari 100%.

6.



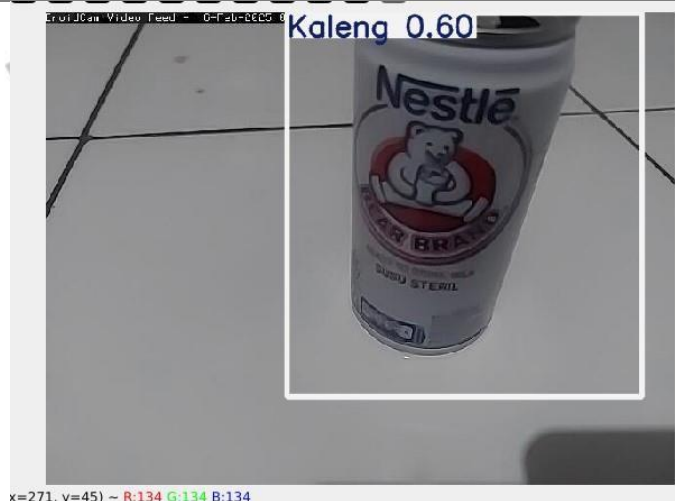
Dari hasil program yang telah dijalankan 4 botol plastik dari 10 botol plastik terdeteksi dengan tingkat akurasi rata-rata mencapai 0.42 dari 1.00 atau setara dengan 42% dari 100%.

7.



Dari hasil program yang telah dijalankan sebanyak 10 kali, 2 kaleng dari 10 kaleng dapat terdeteksi dengan tingkat akurasi tertinggi mencapai 0.70 dari 1.00 atau setara dengan 70% dari 100%.

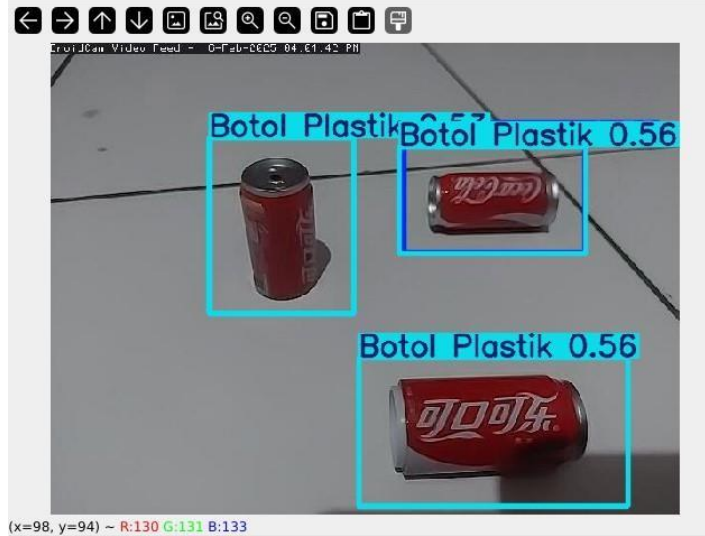
8.



Dari hasil program yang telah dijalankan sebanyak 10 kali, 5 kaleng dari 10 terdeteksi dengan tingkat akurasi rata-rata mencapai 0.60 dari 1.00 atau setara dengan 60% dari 100%.



9.



Dari hasil program yang telah dijalankan sebanyak 10 kali, 3 kaleng jenis coca cola dari 10 tidak dapat terdeteksi atau salah dalam mengklasifikasikan objek tersebut.

Tabel di atas merupakan hasil deteksi sistem cerdas pemilah sampah daur ulang berbasis klasifikasi objek dengan menggunakan algoritma YOLOv5 dan dengan tingkat akurasi atau *confidence* yang tertera pada hasil tersebut.

### 5.1.2.3 Pengendali Lengan Robot

Berikut di bawah ini merupakan hasil dari pemrograman pengendali lengan robot.

tabel 5. 3 Program Lengan Robot

Tahap ke	Kode	Penjelasan
1	<pre>Servo servoBase; // Servo untuk base rotation Servo servoElbow; // Servo untuk elbow Servo gripper; // Servo untuk gripper</pre>	Inisialisasi servo untuk setiap DOF yang digunakan.
2	<pre>void setup() {   Serial.begin(9600); // Mulai   komunikasi serial   servoBase.attach(9); //   Hubungkan servo base ke pin 9   servoElbow.attach(10); //   Hubungkan servo elbow ke pin 10   gripper.attach(11); //   Hubungkan gripper ke pin 11    // Set posisi awal   moveToPosition(0, 45); // Base   di 0° dan elbow di 45°   closeGripper(); // Gripper   di posisi tertutup (0°)   Serial.println("Robot arm is   ready."); }</pre>	Digunakan untuk mengatur pin pada servo yang akan digunakan dan menyetel posisi awal pada servo.
	<pre>void loop() {</pre>	

3	<pre> if (Serial.available()) {     String command =     Serial.readStringUntil('\n'); // Baca     perintah dari serial     Serial.println("Command received: " +     command); </pre>	<p>Digunakan untuk membaca perintah dari serial yang diterima melalui program klasifikasi.</p>
4	<pre> if (command == "plastic") {     executePickAndPlace(0,     90, 70); // Ambil plastik dan     pindahkan ke tempat plastik } else if (command == "glass") {     executePickAndPlace(0,     90, 90); // Ambil kaca dan     pindahkan ke tempat kaca } else if (command == "can") {     executePickAndPlace(0,     90, 110); // Ambil kaleng dan     pindahkan ke tempat kaleng } else if (command ==     "no_object") {     Serial.println("No object     detected. Robot arm will not     move."); } } </pre>	<p>Digunakan untuk mengeksekusi berdasarkan jenis objek yang diterima melalui serial dari program klasifikasi.</p>
5	<pre> void executePickAndPlace(int     baseAngle, int elbowPickAngle, int     basePlaceAngle) {     // Posisi pengambilan barang     moveToPosition(baseAngle,     elbowPickAngle); // Gerak ke     posisi pengambilan     delay(1000); // Stabilisasi     openGripper(); // Buka gripper     delay(500);      closeGripper(); // Tutup gripper     untuk mengambil barang     delay(1000); // Stabilisasi     moveToPosition(baseAngle, 45);     // Kembali ke posisi angkat     delay(1000); // Stabilisasi      // Posisi penempatan barang      moveToPosition(basePlaceAngle,     45); // Rotasi base ke tempat     penempatan     delay(1000);     openGripper(); // Lepaskan     barang     delay(1000); // Stabilisasi </pre>	<p>Program tersebut berfungsi untuk Gerakan <i>pick - and -place</i>. Yang Dimana ketiga servo tersebut bergerak sesuai dengan perintah atau pada posisi yang sudah ditentukan.</p>

	<pre>// Kembali ke posisi awal moveToPosition(0, 45); closeGripper();// Pastikan gripper tertutup di posisi aw</pre>	
6	<pre>void moveToPosition(int basePos, int elbowPos) {     gradualMove(servoBase, servoBase.read(), basePos, 5); // Gerakkan base secara bertahap     gradualMove(servoElbow, servoElbow.read(), elbowPos, 5);// Gerakkan elbow secara bertahap }</pre>	Digunakan untuk menggerakkan lengan robot ke arah tertentu dan secara bertahap.
7	<pre>#Fungsi untuk membuka gripper void openGripper() {     gradualMove(gripper, grripper.read(), 90, 5); // Posisi 90° untuk membuka gripper }  // Fungsi untuk menutup gripper void closeGripper() {     gradualMove(gripper, grripper.read(), 0, 5); // Posisi 0° untuk menutup gripper }</pre>	Program tersebut berfungsi untuk membuka dan menutup gripper atau penjepit. Dengan posisi 90 derajat untuk membuka dan 0 derajat untuk menutup.
8	<pre>void gradualMove(Servo &amp;servo, int startPos, int endPos, int stepDelay) {     if (startPos &lt; endPos) {         for (int pos = startPos; pos &lt;= endPos; pos++) {             servo.write(pos);             delay(stepDelay); // Mengatur kecepatan gerakan dengan delay         }     } else {         for (int pos = startPos; pos &gt;= endPos; pos--) {             servo.write(pos);             delay(stepDelay); // Mengatur kecepatan gerakan dengan delay         }     } }</pre>	Digunakan untuk mengatur kecepatan Gerakan dengan delay yang sudah ditentukan.

### 5.1.3 Hasil Pengujian Kotak Hitam

Berikut merupakan hasil dari pengujian kotak hitam sistem cerdas pemilah sampah daur ulang berbasis klasifikasi objek dan lengan robot.

tabel 5. 4 Hasil Pengujian Kotak Hitam

Tahap ke	Aktivitas	Hasil Pengujian
----------	-----------	-----------------

1	Addaptor disambungkan pada colokan	Berjalan dengan baik
2	Potensiometer digunakan untuk menggerakkan atau mengatur kecepatan motor dc.	Berjalan dengan baik
3	Motor dc bergerak dengan lancar.	Berjalan dengan baik
4	Motor Servo (Base) bergerak dengan lancar.	Berjalan dengan baik
5	Motor Servo (Elbow) bergerak dengan lancar.	Berjalan dengan baik
6	Motor Servo (Gripper) bergerak dengan lancar.	Berjalan dengan baik
7	Lengan robot dapat bergerak lancar.	Berjalan dengan baik
8	Konveyor dapat berjalan dengan lancar.	Berjalan dengan baik
9	Program mengklasifikasikan botol plastik sebanyak 10.	Berjalan dengan baik
10	Program mengklasifikasikan botol kaca sebanyak 10.	Berjalan dengan baik
11	Program mengklasifikasikan kaleng sebanyak 10.	Berjalan dengan baik
12	Lengan robot mengambil botol plastik	Berjalan dengan baik
13	Lengan robot mengambil botol kaca	Berjalan dengan baik
14	Lengan robot mengambil kaleng	Berjalan dengan baik
15	Lengan robot menempatkan 3 botol plastik ke kontainer 1 dengan konveyor	Berjalan dengan baik
16	Lengan robot menempatkan 3 botol kaca ke kontainer 2 dengan konveyor	Gagal
17	Lengan robot menempatkan 3 kaleng ke kontainer 3 dengan konveyor	Gagal
18	Lengan robot menempatkan 10 botol plastik ke kontainer 1 tanpa menggunakan konveyor	Berjalan dengan baik
19	Lengan robot menempatkan 10 botol kaca ke kontainer 2 tanpa menggunakan konveyor	Berjalan dengan baik
20	Lengan robot menempatkan 10 kaleng ke kontainer 3 tanpa menggunakan konveyor	Berjalan dengan baik

Pada tabel 5.4 di atas merupakan hasil dari pengujian kotak hitam yang dilakukan oleh peneliti. Berikut di bawah ini merupakan pembahasan dari hasil tabel di atas.

(1). Dari 20 tahapan di atas pada tahap ke 9, hasil dari klasifikasi objek jenis botol plastik sebanyak 10 dari 10 objek adalah 100% dapat terdeteksi tanpa kendala dengan tingkat akurasi yaitu, satu botol plastik berhasil mencapai tingkat akurasi tertinggi yaitu 0.67 dari 1.00 atau setara dengan 67% dari 100%, tiga botol plastik mencapai 50% - 60%, empat botol plastik atau rata – rata mencapai 35% - 40% sedangkan dua botol yang terendah mencapai 30%.

- (2). Pada tahap ke 10, botol kaca yang dapat terdeteksi adalah 90% atau 9 dari 10 objek yang terdeteksi dengan tingkat akurasi yaitu, satu botol kaca berhasil mencapai tingkat akurasi tertinggi sebanyak 82% dari 100%, dengan rata-rata 60%-70% dan akurasi terendah mencapai 26%.
- (3). Pada tahap ke 11, kaleng yang dapat terdeteksi adalah 70% dengan tingkat 30% kesalahan dalam klasifikasi atau gagal. Berikut merupakan tingkat akurasi keberhasilan dalam klasifikasi objek kaleng, dua kaleng mendapat tingkat akurasi tertinggi yaitu 70%, rata-rata 40% - 60% dan terendah salah dalam mengklasifikasikan objek dengan sesuai.
- (4). Pada tahap ke 15, lengan robot hanya bisa meletakkan 2 jenis botol plastik kedalam kontainer yang sesuai dari 3 jenis botol plastik.
- (5). Pada tahap ke 16, lengan robot gagal dalam mengambil dan meletakkan 3 objek jenis botol kaca kedalam kontainer yang sudah ditentukan.
- (6). Pada tahap ke 17, lengan robot gagal dalam mengambil dan meletakkan 3 objek jenis kaleng kedalam kontainer yang sudah ditentukan.
- (7). Pada tahap ke 18, lengan robot berhasil dalam mengambil dan meletakkan objek jenis sampah botol plastik kedalam kontainer sebanyak 10 dari total 10 sampah botol plastik. Lengan robot berhasil dikarenakan objek sampah tersebut memiliki lebar ukuran 3cm dan berat 3 gram, oleh karena itu lengan robot berhasil dalam mengambil objek tersebut, jika objek memiliki lebar lebih dari 4cm dan berat lebih dari 15 gram maka kemungkinan lengan robot tidak akan berhasil dalam memindahkan objek tersebut. tingkat keberhasilan lengan robot dalam mengambil objek botol plastik adalah 100%.
- (8.) Pada tahap ke 18, lengan robot berhasil dalam mengambil dan meletakkan objek jenis sampah botol plastik kedalam kontainer sebanyak 8 dari total 10 sampah botol kaca. Lengan robot berhasil memindahkan dikarenakan 8 objek tersebut memenuhi syarat standar dalam batas berat dan ukuran yang dapat diambil oleh lengan robot yaitu, dengan lebar ukuran 3cm dengan berat 8 gram maka lengan robot masih bisa mengambil objek tersebut, dikarenakan batas maksimal lengan robot dapat mengambil objek adalah 4cm dan berat 15 gram. Tingkat keberhasilan lengan robot mengambil objek botol kaca adalah 80%.
- (9). Pada tahap ke 18, lengan robot berhasil dalam mengambil dan meletakkan objek jenis sampah botol plastik kedalam kontainer sebanyak 5 dari total 10 sampah kaleng. Kaleng tersebut dapat diambil dikarenakan memiliki lebar 3cm dan berat 3 gram adapun kaleng yang gagal dikarenakan memiliki diameter lebar 6cm dan berat 15gram oleh karena itu lengan robot tidak dapat mengambil objek tersebut. Tingkat keberhasilan lengan robot dalam mengambil objek kaleng adalah 50%.