Al-Nahrain Journal for Engineering Sciences

Al-Nahrain J. Eng. Sci. 29(1) 152-173, 2026
Original Article

Automation in loading/unloading semi-knockdown vehicles in assembly plants for intelligent navigation

Adeyinka Abdulquadri Oluwo[1]●, Muhiz Olawale Ambali [2]●, Sunday Ayoola Oke[3]●, Henry Ogbemudia Omoregbee[4]●, Bayo Yemisi Ogunmola[5]●, David Tijesunimi Livingstone [6]●and Francis Onoroh[7]●

1,2,3,4,5,6,7Department of Mechanical Engineering, University of Lagos, Lagos, Nigeria

●*These authors contributed equally to this work*

DOI: http://doi.org/10.29194/NJES.29010152

Received: October 7, 2025 **Revised:** January 14, 2026 **Accepted:** January 23, 2026 **Published:** March 20, 2026

Abstract

Manual handling of semi-knockdown vehicles in assembly plants is unsafe, time-consuming, inefficient, and prone to quality irregularities. To intervene in addressing these problems, this study develops a prototype of an automated load carrier intelligent navigator. The work centre is analysed for space, material type and handling requirements. This is followed by design and testing, whereby software, hardware and mechanical engineering are integrated in the context of process optimisation. The prototype was tested on rough and smooth surfaces, for no-obstacle and obstacle avoidance conditions. On rough and smooth surfaces with no obstacles, the minimum distance considered is 0.5m, and the average speed and time determined are 0.08m/s and 6.23s, 0.17m/s and 2.97s, respectively. For the maximum distance of 3.0m, the average speeds and times determined are 0.081 m/s and 37.42s, and 0.18 m/s and 17.35s, respectively. The average distance considered for both rough and smooth surfaces is 1.75 m, and the average speed and time at each scenario are 0.081 m/s, 21.78s, and 0.17 m/s, 10.26s. The voltage of the battery drops, with a corresponding decrease in the speed of the motors. The automated carrier prototype makes the best decisions when it encounters an obstacle, giving the best outputs. This paper contributes by providing real-time intelligent navigation data and accurate regulation of the automated carrier for automotive assembly plants. Its novelty lies in conducting experimental investigations using the automated loading/unloading intelligent navigator to explore its advantages compared to manual loading/unloading in automotive assembly plants. In conclusion, building a carrier for assembly operations enhances assembly operational performance, correcting inefficient and unsafe loading and unloading processes.

Keywords: *Automated Carrier, Semi-Knockdown Vehicles, Fuzzy Logic Control Unit, Automated Carrier Prototype Design.*

Corresponding author: Provide the corresponding author information and publisher here. E-mail address: Sa_oke@yahoo.com

1. Introduction

In vehicle assembly plants, semi-knockdown vehicles are transported from the original vehicle assembly plant to locations where these are assembled before sales. After shipping the vehicles from containers to the local assembly plants where parts of these semi- knockdown vehicles are to be integrated, substantial human effort are invested by the technicians to offload these semi- knockdown vehicles to the floor where they would be assembled. However, the workers transferring these loads from heights face significant safety risks, such that lowered heavy loads improperly handled could unbalance, falling on the workers, leading to death or severe injuries. Furthermore, the ropes that suspend these semi-knockdown vehicles could experience failure, due to improper handling and material fatigue. There is need to devise alternative approaches for offloading semi knockdown vehicles to avoid deaths liabilities due to injuries and disruptions to work flow.

While manual handling of semi-knocked-down vehicles may be effective, transition to the automated carrier still requires deeper analysis. A gap exists in the inability of the present manual method of loading and unloading semi-knocked-down vehicles to quantitatively evaluate the responses of the automated carrier during operations and under obstacle avoidance conditions. The current manual transportation method of the semi-knocked-down vehicles struggles with information on smooth and rough surfaces for navigation and different distance movements. This paper develops and tests a prototype, demonstrating the performance of the prototypes on rough and smooth surfaces during operations and under obstacle avoidance conditions.

If safety must be guaranteed, liability cost eliminated, and enhance conformance of the workforce to safety rules ensured, an automated carrier needs to be designed, developed, and implemented to replace manual offloading of semi-knockdown vehicles. Moreover, an assessment of automated carrier shows a merit for assembly in an automobile workshop. This minimizes the

risks that might occur during the assembly operations causing sudden mishaps or dents on the semi knockdown vehicles, reducing any form of undesirable events. This study improves the finish phase of the semi knockdown vehicles after assembly. Moreover, there remains a large research gap between theoretically advanced models that are human aware and their real-world applicability and useability in the logistics and manufacturing real-world. Also, the real-time optimization of task allocation and intuitive communication protocols between humans and robots in shared material handling workstations is of interest to research and improve the efficiency of collaboration.

2. Literature review

This section lay outs a brief investigation of load carrier practices. Experiments and applications proposed that manual carriers pose a lot of risks such as poor ergonomics, frequent accidents in work areas, vehicle defects and miscommunication among workers. Besides, the automated carrier can bridge the gap between working effectively and efficiently with working manually on carriers due to the addition of sensors, motor drivers, control unit and so on. There are series of conclusions accessible owing to earlier researchers. Majority of researchers deduced major applications to solving the issue of manual carriers in an assembly plant with the use of automated guided vehicle. Furthermore, implementing of sensors reduces risks associated with manual handling of carriers in industries like automobiles, buildings, etc. The use of sensors for automation stands out due to their ability to detect physical changes in the working area thus, converting these changes to signals sent electrically that is readable and can be processed when operating on other systems. This assessment looks into previous literature on series of carriers used in modern applications. The analysis embodies awareness of publications by numerous authors in acknowledged journals from multiple sources, classified under the following groups: (1) Load, manual and automated carriers and (2) Methodology, characteristics, applications and experimental procedures.

Load carrier systems to lift heavy objects use actuated mechanical systems, which are normally servo motors, lead screws, or linear actuators under the control of microcontrollers to translate electrical power into controlled vertical movement which allow heavy loads to rise safely and securely. Furthermore, studies that expatiate on load carriers using experiments and applications as a reference are discussed as follows: Christian and Jochen [1] proposed a hybrid packing solution to the problem of auto-carrier loading problem on the operational level. Saikumar et al. [2] introduced a framework whereby loads of varying sizes are transported from one place to another using load carriers. Devesh et al. [3] proposed the ergonomic design of load carriers. Karthik and Rakesh [4] considered the merging of load carriers with autonomous truck NXT. Valentin et al. [5] declared that good operation of truck loading spaces can favorably affect the financial execution of automated loading during the movement of a greater volume load. The automated carrier represents a type of robotic systems designed to be autonomous in the movement of materials, to navigate dynamic indoor environments, and to carry a payload, yet all without the use of constant human control. Furthermore, papers that expatiate more on the automated carriers using experiment and applications as a backing are considered as follows: Esan and Ajilore [6] highlighted the design, fabrication, and test of a load carrier of a typical industry. Reethya [7] provided the design and development of a Plant Watering Robot using Arduino micro controller. Muazu et al. [8] was concerned with designing and creating an independent firefighting robot. Mukthar et al. [9] simulated an obstacle avoidance robot. Danladi et al. [10] presented autonomous car (AC), driverless car (DC), robotic car, or self-driving car that is capable of running on minimal or no human intervention. Muhammad [11] studied the Obstacle Avoidance Robot, capable of identifying the impediments lying on its course automatically and avoiding them. Anye et al. [12] studied a driving simulator using the HDV CF that operates in a mixed-flow traffic of three CAV control conditions (string-stable, string-unstable, and similar to conehead). Konda et al. [13] studied a self-driven car with partially automated features. Faysal et al. [14] created an advanced robotic system that would combine the development of hardware with theoretical simulation of the need to create multipurpose. Muhammad and Mukthar [15] studied obstacle Avoiding Robot with the capability of detecting obstacles in its way and helping it avoid them by altering its direction. Faikul et al. [16] developed a control system that is based on Neuro-Fuzzy such that the robot can navigate obstacles as the robot goes on to the destination. Morariu et al. [17] proposed an effective navigation algorithm of gathering tennis balls under the condition of using a visual sensor, namely. Nuntachai and Phichitphon [18] designed and built an electric wheelchair that moves in any direction and speed required with the help of the joystick controller. Akhtar et al. [19] presented a system comprising a slave robotic arm and a master-wearable devices with two-way connection of robotic arm and the operator (master wearable device). Kolapo et al. [20] introduced an obstacle detection and avoidance system of an unmanned Lawnmower. Chidvilasini et al. [21] proposed an approximate distance measuring device built to measure the distance of any object within distance of 50 cm and 500 cm (5m). The manual carriers are mechanical or semi-mechanical equipment used to lift, carry, transport or reposition heavy or bulky loads using completely human physical effort without electric, hydraulic or pneumatic power. These devices enhance human strength via mechanical advantage allowing material to be handled safely and efficiently in situations where automation is not feasible, unavailable, or unnecessary. Furthermore, papers that looked more on the manual carriers using experiments and applications as a consideration are as follows: Wadea et al. [22] aimed to quantify the effectiveness of various assemblies teaching format. Ruimin and Yanjiao [23] worked on carriers used in the car assembly workshop, car assembly and car parts delivery. Amirreza [24] considered, on a partstation-by-part basis, one policy per part-station pair that will minimize its operating cost (annual) and maintain capacity, space, and flow constraints. Prathamesh et al. [26] designed and installed Smart Manual Assembly Table (SMAT), which converts smart assistance technologies (SAT) into productive industries. Ish [27] indicated that the origin of the contract of carriage is the contracting carrier who is the party engaging in the contract with the passenger.

Moreover, the following aspects contain a rundown of previous research by scholars regarding the design of an automated carrier prototype for moving semi knockdown vehicles across workstations in an assembly plant. It broadly talk over the advantages and downside associated with numerous methods. An outstanding portion of the literature spotlights the development of an Arduino based obstacle avoidance robotic system which makes use of two sensors (ultrasonic and infrared). Among the excess of available carriers, the Arduino micro controller is mostly emphasized on in the literature. This fondness comes from the assurance that Arduino micro controllers tends to it's low barrier to entry by giving an open-source platform for micro controllers, easy-to-use, and budget friendly. Tables (1) to (3) show the summary of the relevant literature. In Tables 1, 2 and 3, the thematic categories of the literature have been given in terms of manual carriers, automated carriers, and robotic applications. Moreover, unlike the majority of reported studies that are biased to words viewing studies from the lens of logistics and human-aware system perceptions, the diverging lens of visualisation from the physical, implementable, experimentally-proven autonomous load carrier is adopted in the present study. Furthermore, the actual manufacturing limitations are considered where additive manufacturing replaces the subtractive manufacturing scheme to produce the parts for the auto carrier. Specifically, considering manual carriers, the literature treats ergonomic aids in manual terms, while the present study completely diverges to autonomous load transportation. Regarding previous works on automated carriers, there appears to be an over-reliance on simulation. However, the present study corrects this weakness by introducing a whole physical prototyping and testing. Robotics applications considered in the review also have the weakness of relying heavily on simulation, but the focus on physical prototyping, which this article showcases, overcomes this weakness.

Table 1. Manual and load carriers

S/N	Authors	Methodology	Configuration (Characteristics)	Experimental or Application	Key Findings
1.	Manuel et al. [25]	An overview of literature and various case studies	Human-based processes, sensors, and an embedded approach producing real-time data.	An application	Improves efficiency, productivity and quality
2.	Christian and Jochen [1]	Mixes a mixed-integer programming with a Multi-Start Local Search Heuristic	Methodologies to estimate realistic load factors of auto carriers.	An application	The effective solution of a hybrid model with a mixed-integer quadratic assignment problem and a two-dimensional nesting problem
3.	Devesh et al. [3]	Literature review	Concerned with ergonomics	A literature review	Anthropometrically designed systems ease physical demands
4..	Saikumar et al. [2]	Minimizing manual handling of material	Described as having untethered magnetic actuation, excellent multimodal locomotion.	Is both applied and experimental.	Manages a broad spectrum of machines including small scale carts and heavy industrial machines.
5.	Valentin et al. [5]	Return on investment and the break-even period of the automated loading/unloading technologies	Explores opportunities and advantages of applying unmanned loading technologies overall.	An application	Increased cargo volume shipped per trip is effective in improving the financial performance of automated systems
6.	Wadea et al. [22]	Ultra-careful experimental strategy	Compared the effect of various forms of instructions with text and no text on human performance rates.	An experiment	Adding colour and text to assembly instructions produce a small influence on performance on simple tasks.
7.	Prathamesh et al. [26]	Investigational evaluation methodology	A Smart Manual Assembly Table	An application	High productivity, error reduction and efficiency of the operator.
8.	Ish [27]	Comparative legal methodology and doctrinal methodology.	Explains the legal differences and relations between contracting carriers and actual carriers in transportation of cargo or passengers.	An application	Contractual carrier causes the transport contract. The service is carried out by the actual carrier.
9.	Lois [28].	Multi-stage sampling technique	Studied the incidence of musculoskeletal disorders in male workers who are involved in manual material handling in markets	An application	The lower back is the most widely affected area.
10.	Khan et al. [29]	Observation of operational data of ships	Study of a refrigerated cargo carrier and its trim work on an ocean trip.	An application	Weather conditions are real-time factors affecting the optimum trim of a refrigerated cargo carrier.
11.	Gernot et al. [30]	Literature review, technical specifications provided by manufacturers, and case studies.	Reviews new technologies in mobile skyline cable yarding carriers and carriages, the elements of the cable yarding system that transfer the load.	An application	Carriers become technology platforms with built-in geospatial and camera systems

Table 2 Automated carriers

S/N	Authors	Methodology	Configuration (Characteristics)	Experimental or Application	Key Findings
1.	Reethya [7]	As its main framework agile development methodology or a Waterfall Model.	A microcontroller, a soil moisture sensor, a water pump, and a water tank	An application	A fully operational arduino system could efficiently and automatically irrigate plants
2.	Andrea et al. [31]	An experimental, project-based learning methodology that is consistent with constructionism principles of learning to support applied mechatronics learning.	Its educational intent, where students combine elements to teach themselves about microcontroller programming, circuit design, mechanical modeling, and navigation.	Both an experiment and an application	Automated guided vehicle construction is a useful practical method to learn mechatronics.
3.	Muazu et al. [8]	Design-based procedure	A mechanical design, some electrical hardware, and software	Both experiment and application	The systems are able to successfully detect the fire and then automatically move to the fire and extinguish it with a water pump.
4.	Danladi et al. [10]	Data analysis, case studies, expert interviews or literature review.		A critical study.	Technological and safety constraints, extreme cost, huge economic and social turbulence, complicated legal and ethical issues, and public perception issues.
5.	Anye et al. [12]	Running experiments with alternative connected and autonomous vehicle control conditions, statistically analyzing the data, and using car-following model tuning	They are irradiated with a programmable car-following logic that can be set to different car-following behaviours to explore human driver responses.	An experiment	How human drivers will act when they co-exist with Connected and Automated Vehicles on the road.
6.	Konda et al. [13]	With an Rpi camera module and OpenCV library to process images	Has a sensor and algorithm-based approach to creating a map of the environment in 3D.	An application	A suggested system that deploys an expensive camera and an OpenCV library with a Raspberry Pi to detect traffic lights

Table 3. Robotics applications

S/N	Authors	Methodology	Configuration (Characteristics)	Experimental or Application	Key Findings
1.	Thibault et al. [32]	Human-aware optimization models in logistics and manufacturing systems	Human-robot and human-automation interactions.	A survey and a literature overview	There is a gap between the theoretical human-conscious models and their field applicability and usability in industrial settings.
2.	Thibault et al. [33]	Detailed analysis, and categorization of mathematical models and methods	The overall nature of collaborative robots and human-robot interaction in general, anthropometric measuring systems	A review article	Automation, the value of planning, and information systems as a way to be efficient
3.	Simon and Munashe [34]	Qualitative analysis and case study.	Human/robot capability to work together, portability, scalability and resistance to failure.	An application	High costs and hold-ups.
4.	Esan and Ajilore [6]	Infrared sensors, arduino UNO, infrared sensors, and mechanical drive on three wheels with a steerable front wheel to make a line-following robot to handle materials.	Obstacle sensors, one unit load carrier to move objects and a real-time communication system with facility control systems to get instructions and updates.	Both an experiment and an application.	The artificial AGV was successfully used as a material handling system
5.	Muhammad [11]	A hybrid of Generalized Dynamic Fuzzy Neural Networks (GDFNN)	Moves freely over obstacles with an ultrasonic sensor as its distance gauge.	Both application and experimentation	The robot can improve its learning abilities and the robot can perform the tasks prescribed in a complex environment.
6.	Amirreza [24]	Unified Mixed-Integer Linear Program (MILP)	A robot with technical capabilities such as power, speed, force and load weight, but which is operated collaboratively (also known as a cobot).	An application	Hybrid material delivery systems are usually better than the pure use of one approach.
7.	Kazemi and Gentes [35]	Robotic systems engineering	Their integrated CFD and VOF boiling solver is able to model the dynamic RPT (Rapid Phase Transition) of leaking LPG through a broken carrier by resolving density and temperature fields throughout the process.	An application	Add the effective establishment of a mobile robot platform with special decontamination and measurement tools, the ability to work in the teleoperation mode and semi-autonomous mode, the application of new sensing, A.I., and validation techniques to provide safe and accurate work under hazardous conditions, which will reduce the risk of human harm to a minimum and increase efficiency.

3. Methodology

3.1. Software design

Software for the project was designed involving programs and writing commends that the sensor data will understand and the actuators can also run. Instructions were given for example, it can command the prototype to move clockwise or otherwise. The hardware design needs a schematic for the connection and the design of components and interconnection include the following: motor driver, ball converter (step and voltage converter), two sensors which are ultrasonic sensor, actuators and servo motor. The configuration of the chip is that it was used with the board and the declaration of each pin function was done.

Recall that the principal aim of this section is to design and program the automated carrier. To achieve this, four principal steps were undertaken. The first involves supplying a 5-volt power source to the chip. This step is necessary to prevent the Arduino from being supplied with more than the maximum voltage it can absorb. Usually, 15 volts is available, but it has to be channelled through the buck converter, which steps it down to 5 volts. The next step is to declare the pins (output and input pins) which are connected to the device. It is necessary to follow this step so that the pins transferring signals can be defined. Moreover, the mapping task is done to the different spots so that the Arduino can receive and transfer signals effectively. Furthermore, the declaration of the types of pins is made. The essence of this step is to declare which of the pins receives signals distinct from other pins that transmit signals.

This piece is made up of a software component that houses the Arduino UNO where the codes are written in the form of commands that can be comprehended and executed by the sensor data and actuators (see appendix). Adjustment of the codes and commands of which direction the load carrier was to be headed to.

3.2. Hardware setup

The hardware part will involve the chassis, TT motors, mecanum wheels, Arduino UNO, L298N motor driver, buck converter, servo motors, ultrasonic sensors, breadboard, and a switch as shown in the following diagram. L298N H-bridge is a motor driver and it is fitted to the mecanum wheels, buck converter that assists in reducing the voltage, ultrasonic sensors that act as the eyes of the load carrier that sends and receives signals in the form of waves, TT motors which is the direct current motor of the mecanum wheels.

The chip can be configured without a regular board but in the process of this work it was designed with a regular board (Arduino UNO) and stated each pin. This specification involves the declaration of each pin functionality in the case of the digital pins, it will be used to operate the mecanum wheels which will be clearly stated in the code that this specific pin was used to operate the wheels. In the analog pin whereby, they have also been employed to control the ultrasonic sensors which were also indicated in the codes clearly. Every pins has its purpose, to control the speed, to control the rotation of the servo motors etc. Chips are the programmer where codes are being transmitted to with the maximum capacity of 2mb.

Figure 1 helps in understanding the targets for each step towards the completion of task for the development of the automated carrier. By this scheme of research, a clear view is provided to researchers or anyone interested in implementing the procedures such that repetitiveness of task in the precise suggested approach is accomplished. It becomes a useful tool for designers to provide technicians width such that with little supervision much can be accomplished in the fabrication of parts and process integration for the development of the automated carrier. With this information, arguments that usually arise between the designers and technicians that are given the assignment to fabricate the parts will be avoided. Compared with Khan et al. [29], which emphasized the significance of establishing the environments, choosing the appropriate hardware, determining the logic of design, implementing and testing the experiment, the presence study is validated since it concurs with this mentioned literature (Figure 1). Figure 1 illustrates the steps taken in the research process undertaken on automated carrier development. The process involves eleven main steps, which could be broadly grouped into preliminary investigative steps involving literature review, objective formulation and materials. The next phase is the prototype development phase, which entails the development of a prototype and setting up wires and connecting them to the Arduino. The next phase is the software aspect, which involves coding. The last phase involves the connection of a battery and a buck converter. Acrylic material is fed to the laser cutting machine, which is cut to the desired shape. The motors are then connected to the wires; they are then attached to the acrylic material, which forms the chassis.

3.3.CAD designs on each component

The following CAD designs (Figures (2) to (8)) were obtained based on the proposed dimensions using Solid works and Laser works. Figure (2) is the mounting platform for all the assembled mechanical parts, withstanding dynamic loads and stresses and offering the structural rigidity to maintain the carrier's shape. It is made of acrylic material, easy to machine, durable, and cost-effective. Figure (2), which shows the CAD model of the chassis, details the geometric relationships of the chassis. Figure (3) is a CAD model for the servo motor. The servo motor enables the rotation of the ultrasonic sensor(0 degrees – 180 degrees). During the design phase, the two complementary servo motors are used avoid signal interference as the prototype scans the environment. The CAD model, which shows the isometric, plan, side, and front views, provides details on the mounting interfaces, such as size, location and whole patterns. It assists in the mechanical assembly with other assembly components. Figure (4) shows the CAD model for the buck converter. The buck converter was used to step down the 16 volts from the battery to the 5 volts required by the Arduino. The CAD model digitally represents the functional characteristics evaluation criteria and the circuit's physical implementation. The TT motor provides the rotational motion required by the mecanum wheels. It was taken into account at the design stage because it is of the lowest cost, simple to use, and small. The gear ratio that was selected is the 1:220 gear ratio, so as to provide the required torque to move the prototype. Figure (5) shows the geometry and physical dimensions that aid the design integration. The CAD drawing also reviews the pressure dimension and the mounting interfaces of the TT motor. The L298N motor driver, located between the Arduino and the TT motor, provides control over the speed and the direction of where the TT motor rotates. The prototype carrier was tested at varied speeds on

different surfaces, so the L298N motor enables this to be carried out. The CAD model for the L298N motor driver is shown in Figure (6) which offers the electrical as well as physical representation of the L298N motor driver. The mecanum wheels enable omnidirectional movement of the wheels (forward, backwards, strafe left, strafe right, diagonal, rotate at a point). The CAD model for the wheels is shown in Figure (7). This is presented as a complex design that eliminates the need for the typical steering system. The ultrasonic sensor acts as the eyes for the prototype carrier, whereby one part of the cone transmits, and the other part receives signals. Figure (8) shows the functional description of the carrier, where the physical dimensions, including height, length, and width, as well as the housing shape, are specified.

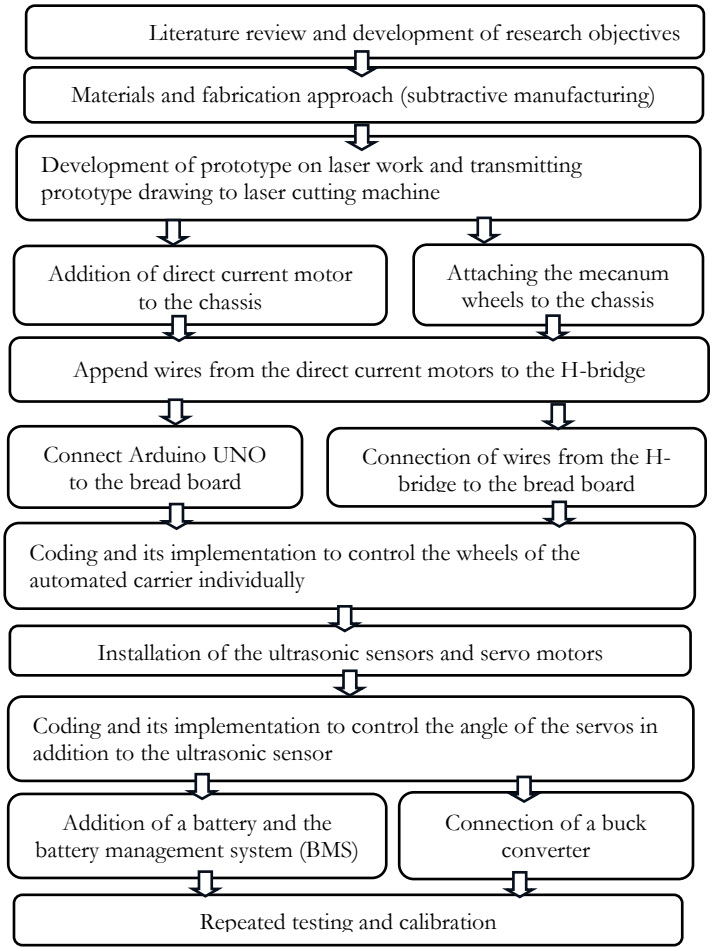

Figure 1. Scheme of the present study

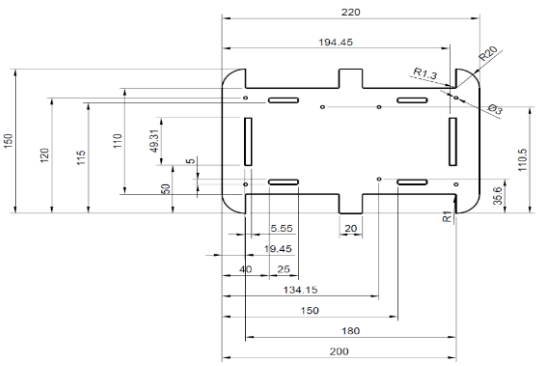

Figure 2. CAD model of the chassis

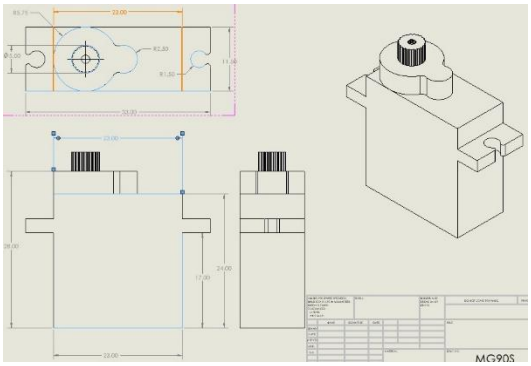

Figure 3. CAD model of the servo motor

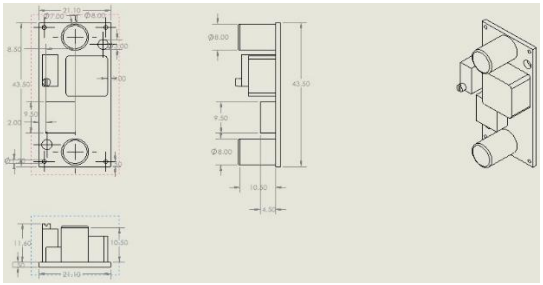

Figure 4. CAD model of the buck converter

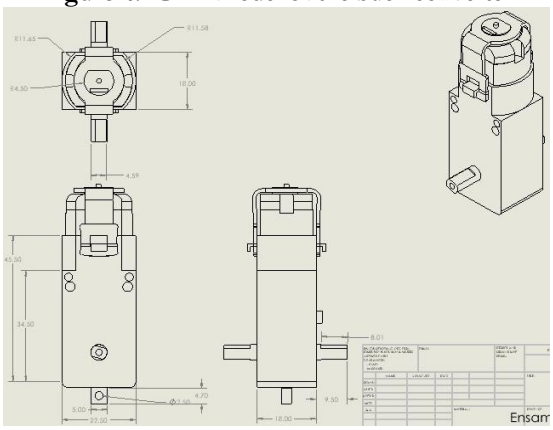

Figure 5. CAD model of the TT motor

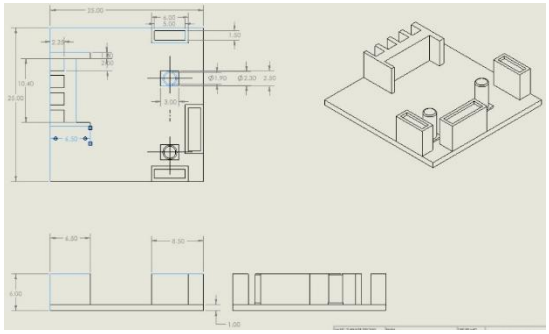

Figure 6. CAD model of the L298N motor driver

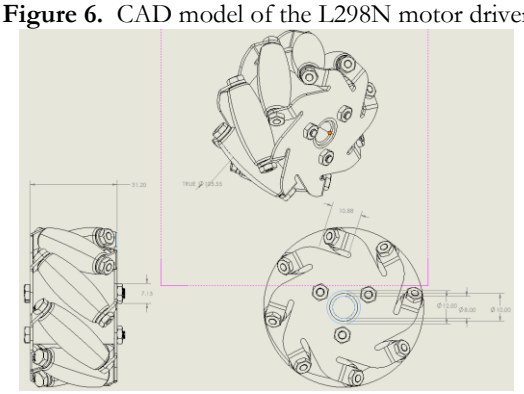

Figure 7. CAD model of the mecanum wheels

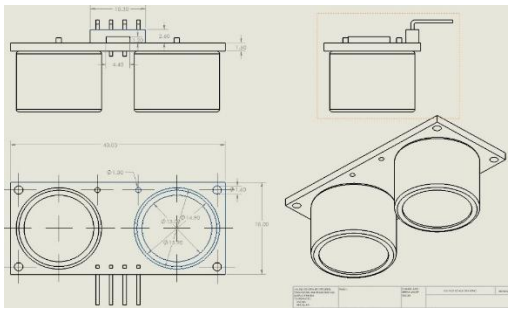

Figure 8. CAD model of the ultrasonic sensor

3.4. Prototype fabrication and assembly process

Prototype fabrication

- Putting the acrylic sheet on a laser cutting machine to cut out the required chassis required for the prototype giving an output of a collection of specially-shaped acrylic elements. Putting the acrylic sheet on a laser cutting machine to cut out the required chassis required for the prototype, giving an output of a collection of specially-shaped acrylic elements. The reason why this material was taken into consideration is that of its light weight, cost-effectiveness, durability and easy machinability. It must be shaped to the necessary form since prototype designs are allocated certain geometries that allow other components to fit easily on them.

Assembly process

- Soldering wires on the TT motors to obtain a motor spinning in both clockwise and anticlockwise direction. Soldering wires on the TT motors, whereby specific wires are assigned to the negative and positive terminals of the TT motors, to obtain a motor spinning in clockwise and anticlockwise directions.
- Mount the TT motors on the chassis for structural stability and support as well as for efficient transmission of power. Mounting the TT motors to the chassis as they are needed to offer solid support to the motors and also to ensure that the motors are aligned towards the mecanum wheels.
- Mounting the mecanum wheels on the TT motor aligning them diagonally to each other to enable an omnidirectional movement. Mounting the mecanum wheels to the TT motor in a diagonal manner to each other to allow an omnidirectional motive
- Connecting the TT motors to the L298N H-bridge motor drivers: the front wheels to one and the back wheels to the other. Connection of TT motors to the L298N H-bridge motor drivers: the front wheels to one and the rear wheels to the other, to facilitate the provision of varying speed and easy directional movement of the TT motor
- Mounting the Arduino Uno on the chassis. Attaching Arduino Uno R3 to the chassis to prevent open wires, solder joints and short circuiting.
- Connecting the signal pins of the H-bridge to the Arduino Uno R3. Attaching the signal pins of the H-bridge to the Arduino Uno R3 so that the Arduino can control the activity of the motors (speed, start, stop and direction)
- Connecting the servo motors to power and the Arduino Uno. Connecting the servo motors to the Arduino Uno to enable precise control of the ultrasonic sensor placed on the servo motors and a stable operation of the prototype carrier
- Connecting the HC-SR04 ultrasonic sensor to power and the Arduino Uno. Attaching the HC-SR04 ultrasonic sensor to the Arduino Uno to allow the prototype carrier to sense the distance to obstacles and make a required decision to evade them.
- Mounting the two servo motors on the front of the chassis. Connecting the two servo motors to the front of the chassis so that the prototype can read its surroundings by sweeping across them.
- Mounting the ultrasonic sensors on the servo motor. Installation of the ultrasonic sensors to the servo motor that allows the sensors to travel between (0 degrees -180 degrees).
- Mounting the battery pack on the rear of the chassis. Installation of the battery pack in the back of the chassis to avoid short circuits.
- Mounting the two buck converters on the chassis. Attaching the two buck converters to the chassis to supply a voltage (12v-5v) stepped down to the Arduino Uno.

3.5. The programming used for the work

The code is the brain of the process, it makes decision; comparing it to human beings its like detecting something and been able to process the logic behind the part being detected. In the case of this research once the prototype detects an obstacle, the aim is to avoid it and move towards a free direction. The code helps us know when to avoiding an obstacle and how much distance it is supposed to attain to avoid the obstacle. The codes are always stored on a micro controller which is a hardware. The codes are written using C language which are then transferred to the micro controller. Fuzzy logic was useful in this work in terms of intelligent decision-making in the presence of uncertainty, which allows the prototype carrier to avoid obstacles and control motion in an efficient way. But the full potential was not achieved due to sensor limitations and mechanical latencies. The next improvement should concern the refinement of the fuzzy rule base and the implementation of higher-resolution sensors (LiDAR) to improve route following and dynamic obstacle tracking. There is an interface called the Arduino integrated development environment (IDE) acting as a compiler that converts the codes into machine language. In this study, the code was written on C language which is a high-level language. The IDE serves as the interpreter that interprets the machine codes into binaries so that it can be stored on the micro controller's memory. So, there is a chip called an AT mega, which does the interpretation so that's the process of code transfer.

3.6. Major parts of the codes

- Variables
- Macros
- Functions
- Algorithms

The functions are forward-left, forward-right, backward-left, backward-right, forward, backward, strafe leftwards, strafe rightwards, rotate-left, and rotate-right. These are the motions the prototype is meant to take. Whenever the automated carrier prototype is in motion and it sees an obstacle it can easily call the function and avoid that obstacle.

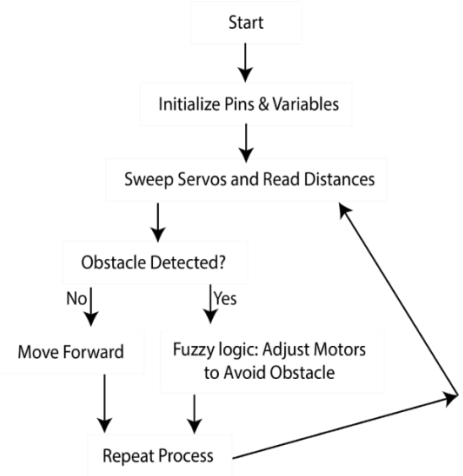

Figure 9. Flowchart of the programming aspect of this work

Figure (9) shows the procedure conducted to achieve the programming aspect of the work. Gain insight into the programming aspect, and visualisation is aided by Figure (9), which shows steps that the automated carrier takes from receiving a command to the end, where loads are dropped off the carrier. Moreover, the flowchart relates to results as it maps out the behaviour of the automated system. By following the logic illustrated in Figure (9), the results, such as the efficiency of the process or the time taken by the carrier in responding to the input and converting it to outputs, are aided by Figure 9. Moreover, before obtaining the result, the earlier task of the flowchart in Figure 9 is to assist in visualising the methodology, detailing the approach to implement the logic of the program.

4. Results and discussion

The suggested model of automated carrier simulates a smart manufacturing process where semi-knockdown vehicles are conveyed to the assembly line. In developing the proposed model, the parameters to model the layout of the assembly plant and operation of a semi knockdown assembly process were introduced. These parameters include the distance between stations (0.303 x 0.5 m^2) and designed speed (255 – 75 bits). The model is a prototype of an automated carrier which is smart and operates in limited or cluttered industrial locations. This prototype uses a Reactive Control Model (also referred to as a Behavior-Based or Stimulus-Response Model). Motion on the sideways was done by mecanum wheels, angular rotation of the ultrasonic sensors was done by a servo motor, stepping down the voltage output with buck converter, and wheels connection with L298n motor driver.

Furthermore, the following information are used in the design: TT motor gear ratio 1:220 and 60mm diameter mecanum wheels. To calculate a precise estimated speed, the following are inputs:

Motor gear ratio: 1:220,

Wheel diameter: 60 mm = 0.06m

Radius = 30 mm = 0.03 m

Wheel circumference = $\pi \times$ diameter = 3.1416×0.06 m ≈ 0.1885 m

Typical TT motor (1:220) no-load RPM at 12V: ~100 RPM

Under load (realistic for robot chassis): ~60-80 RPM

(Using 70 RPM as an average speed value under load; accounting for friction, weight, and slight voltage drop.)

Speed Calculation:

Linear Speed (m/s) = (RPM × Wheel Circumference)/60 = $(70 \times 0.1885)/60 \approx 0.2199$ m/s

Convert to km/h: 0.2199 m/s × 3.6 = ~0.792 km/h

Obstacle avoidance decreases mean speed however; the load carrier will not be operating at maximum speed at all times since the obstacle avoidance is being done by it. It will:

Slow down near obstacles

Stop, turn or strafe left or right (because of mecanum wheels)

Accelerate/decelerate frequently

So average speed of operation in the navigation usually is between 40-60% of max speed.

$0.2199 \text{ m/s} \times 0.5 \approx 0.11$ m/s

Convert to km/h:

0.11 m/s × 3.6= ~0.396 km/h

The loading vehicle was travelling at 0.396km/h on average. The load carrier has its time limits of carrying out some activities. The prototype is proved to prevent obstacle when the sensor transmits waves. Whereas, it is an infinite loop model, hence, it continues running till it is switched off using the switch.

The prototype being tested is shown in Figures (10a) to (10d)

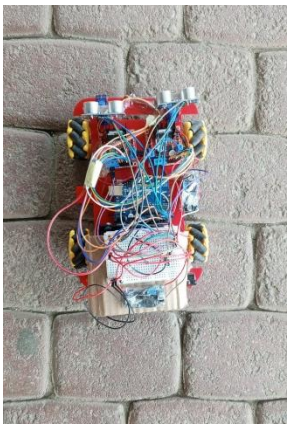

Figure 10a. Testing of the prototype on a rough surface for no obstacle condition

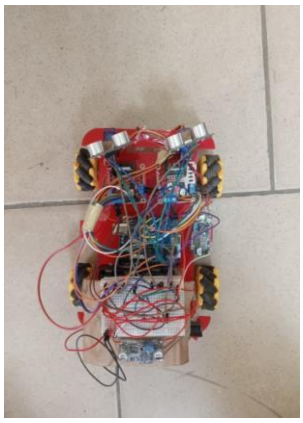

Figure 10b. Testing of the prototype on a smooth surface for no obstacle condition

Figure 10c. Testing of the prototype with 1 obstacle placed at the front on a rough surface

Figure 10d. Testing of the prototype with the Second obstacle on a rough surface

Figures (10a) and (10b) show images during operation. Figure (10a) was a test on the rough surface primarily to yield an understanding of the mechanical durability of the automated carrier. The rough ground and debris give insights into the ability of the mecanum wheels to withstand usage without sudden failure. Figure (10b) shows the test on the smooth surface, which allows the carrier to be tested from a baseline performance perspective. The dynamics braking efficiency, handling attributes, sensor performance and acceleration of the automated carrier could be measured continuously and used as a standard with which the performance of the automated carrier on the rough road or combined smooth and rough road could be judged. Figures (10c) and (10d) show tests during obstacle avoidance. In Figure (10c), the distance of the obstacle to the sensing part of the carrier is shown and the obstacle is smaller in size than in Figure (10d). A shorter distance between the obstacle and the carrier ensures targeted testing, promoting efficiency and controlled automated carrier testing. However, in the case of Figure (10c) where the distance between the obstacle and carrier is longer, it enables testing to assess performance for a sustainable situation, possibly revealing a drift in the sensor on the carrier.

Testing Process

Case 1: When under load and on a rough surface with no obstacle (This test was varied in 20 steps of bits from (255 – 155) bits)

Figures (11) to (16) illustrate the trend of the correlation between time (s) and speed (m/s). The slower the speed, the slower it takes to cover the different distances. The following factors are also taken into consideration that contributed to this other than the relationship between speed, time and distance which includes; the uneven floor surface, voltage drop during reduction in health of the battery and slippage. The horizontal axis presents the average time following the five times of observing the speed of the prototype and recording the behavior to ensure the average time was applied and the related it to the distance in which the average speed in each aspect was realized. The correlation of time, speed and distance is justification of the conduct of the gradient. The distance covered by the motor is slower in lower motor speed to cover the same distance because speed is less = distance ÷ time. This means that the slower the speed the more time will be required to maintain the same distance

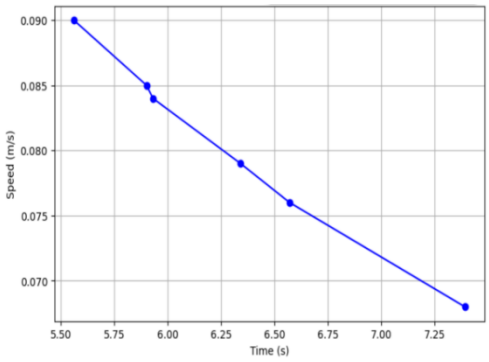

Figure 11. Speed (m/s) against time (s) considering no obstacle and a rough surface; Distance = 0.5m
Cross sectional Area = 0.303m x 0.5m

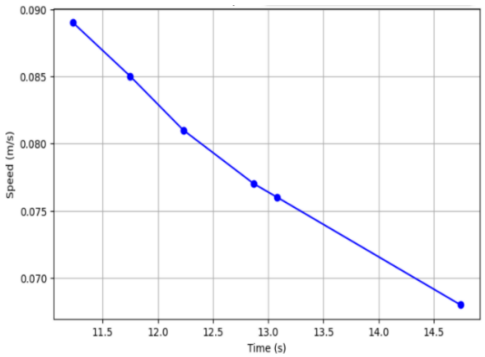

Figure 12. Speed (m/s) against time (s) considering no obstacle and a rough surface; Distance = 1.0m
Cross sectional Area = 0.303m x 1.0m

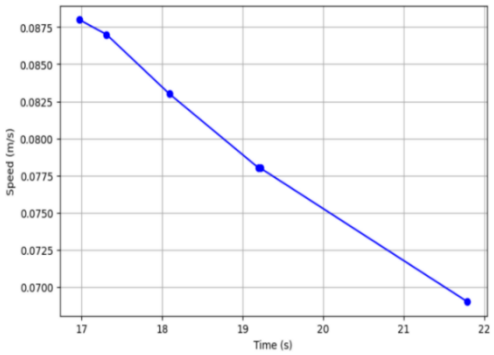

Figure 13. Speed (m/s) against time (s) considering no obstacle and a rough surface; Distance = 1.5m

Cross sectional Area = 0.303m x 1.5m

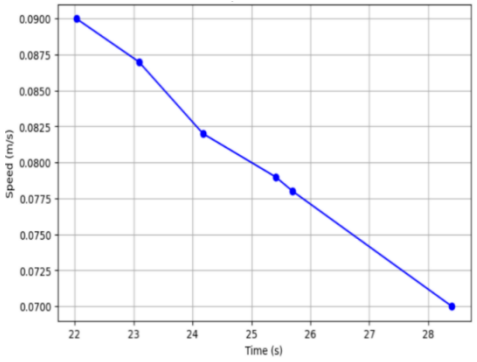

Figure 14. Speed (m/s) against time (s) considering no obstacle and a rough surface; Distance = 2.0m
Cross sectional Area = 0.303m x 2.0m

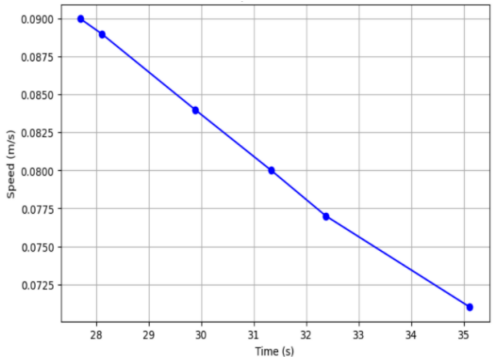

Figure 15. Speed (m/s) against time (s) considering no obstacle and a rough surface; Distance = 2.5m
Cross sectional Area = 0.303m x 2.5m

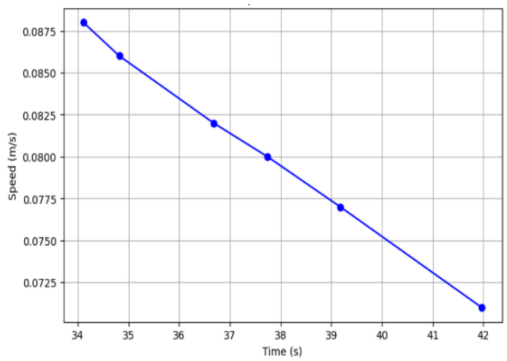

Figure 16. Speed (m/s) against time (s) considering no obstacle and a rough surface; Distance = 3.0m
Cross sectional Area = 0.303m x 3.0m

Case 2: When under load and on a smooth surface (This test was varied in 20 steps of bits from (255 – 75) bits)

Figures (17) to (22) illustrate the correlation between speed(m/s) and time(s). The slow the speed, the longer is the time to cover the end of each distance. The horizontal axis plot represents the average time of observing the speed of the automated carrier prototype five times and recording the behavior such that the average time was used referring it to the distance at which the average speed of each stage was acquired. The correlation that time is related to speed and distance is the reason behind the behavior of the gradient. At a slower motor speed the distance covered by the motor will take a longer time to cover the same distance as the speed is less = distance ÷ time. Hence, with low speed, time must no longer be at the same distance as with low speed. At some points on the graph, it was seen that the speed remained constant at some varying time a fact that could be attributed to irregular floor surface, voltage drop that took place across the batteries or variation in the friction of the wheels like wheel slip. The data provided by Khan et al. [29] on "no obstacle" which indicates a forward movement when no obstacle is detected confirms the result obtained

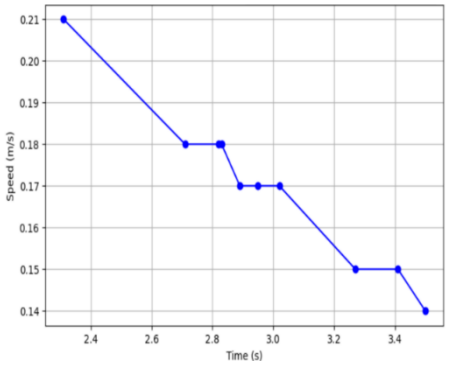

Figure 17. Speed (m/s) against time (s) considering no obstacle and a smooth surface; Distance = 0.5m
Cross sectional Area = 1.55m x 0.5m

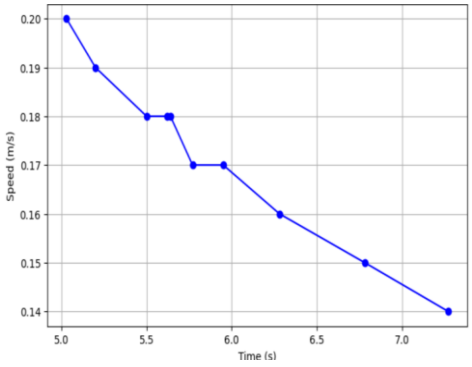

Figure 18. Speed (m/s) against time (s) considering no obstacle and a smooth surface; Distance = 1.0m
Cross sectional Area = 1.55m x 1.0m

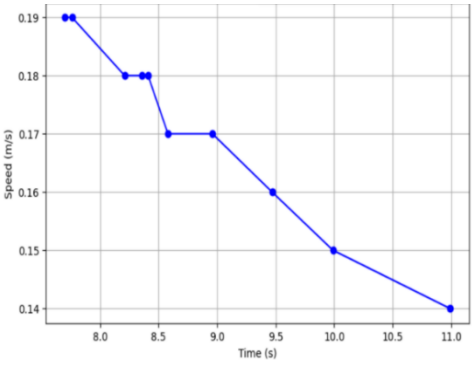

Figure 19. Speed (m/s) against time (s) considering no obstacle and a smooth surface; Distance = 1.5m
Cross sectional Area = 1.55m x 1.5m

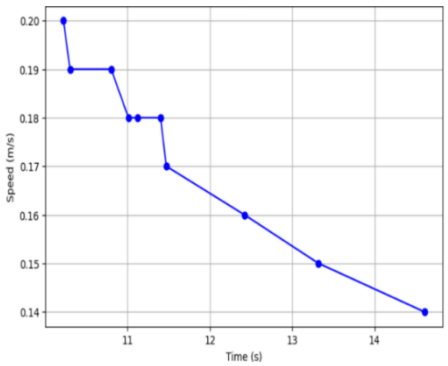

Figure 20. Speed (m/s) against time (s) considering no obstacle and a smooth surface; Distance = 2.0m Cross sectional Area = 1.55m x 2.0m

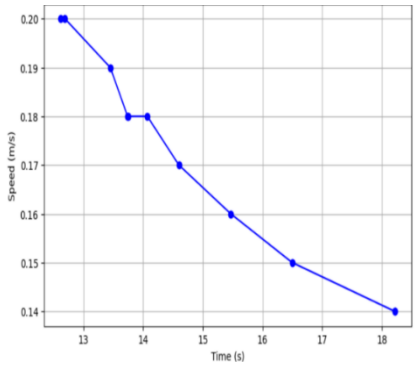

Figure 21. Speed (m/s) against time (s) considering no obstacle and a smooth surface; Distance = 2.5m Cross sectional Area = 1.55m x 2.5m

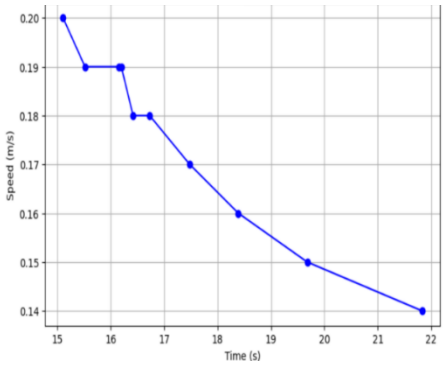

Figure 22. Speed (m/s) against time (s) considering no obstacle and a smooth surface; Distance = 3.0m Cross sectional Area = 1.55m x 3.0m

Case 3: When under load, a rough surface, 1 and 2 obstacles (This test was varied in 20 steps of bits from (255 – 75) bits)

Figures (23) to (26) demonstrate the reason for the variations in speed at different distance for each of the surfaces due to the prototype not moving in a specified direction. The automated carrier prototype does not have barriers but it is not specified in such a way that it can prevent it along a pre-determined route so it will move in a random manner without hitting an obstacle on the way. This is the reason for the huge gap in the speed for the 0.5m and 1.0m in each of the surfaces unlike for the no obstacle part of the result that move in a straight direction. The data provided by Khan et al. [29] on "obstacle avoidance" which indicates an all-direction movement when an obstacle is detected confirms the result obtained.

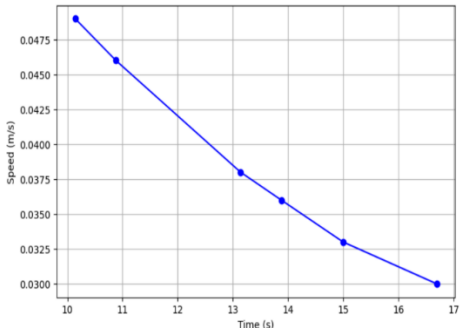

Figure 23. Speed (m/s) against time (s) considering load, a rough surface and 1 obstacle; Distance = 0.5m, Cross sectional Area = 1.55m x 0.5m, Size of obstacle = 0.17m x 0.104m x 0.055m

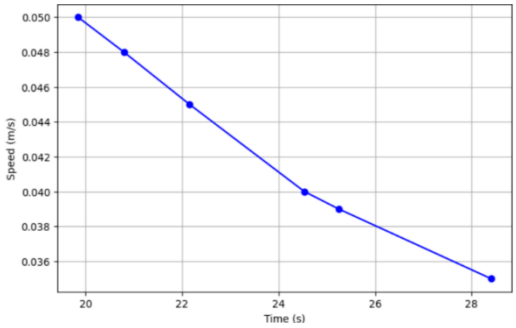

Figure 24. Speed (m/s) against time (s) considering load, a rough surface and 2 obstacles; Distance = 1.0m, Cross sectional Area = 1.55m x 1.0m, Size of obstacle 1 = 0.17m x 0.104m x 0.055m; Size of obstacle 2 = 0.275m x 0.215m x 0.105m

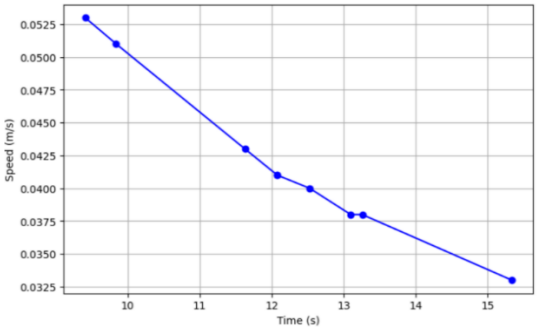

Figure 25. Speed (m/s) against time (s) considering load, a smooth surface and 1 obstacle; Distance = 0.5m, Cross sectional Area = 1.55m x 0.5m, Size of obstacle = 0.17m x 0.104m x 0.055m

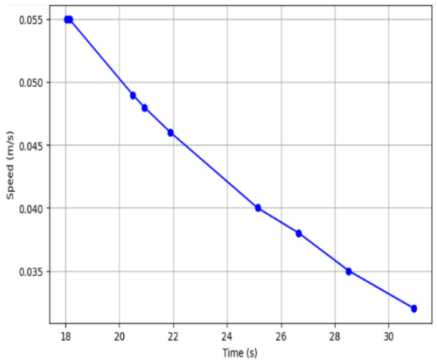

Figure 26. Speed (m/s) against time (s) considering load, a smooth surface and 2 obstacles; Distance = 1.0m, Cross sectional Area = 1.55m x 1.0m, Size of obstacle 1 = 0.17m x 0.104m x 0.055m, Size of obstacle 2 = 0.275m x 0.215m x 0.105m

Case 4: For various distances, correlation between rough (Y) and smooth (X) surfaces with no obstacles

Case 4 contains the results of Figures (27), to (38). These parameters are compared for "during operation" and "during obstacle avoidance" situations. The aim of comparing the outcomes involves understanding the differences, which provides a basis to make more informed decisions on the performance optimisation of the various components of the automated carrier. In all the Figures (27) to (38), correlation experiments were first conducted. This entails collecting data on pairs of parameters, such as the speed of the automated carrier for specific distances, when rough and smooth surfaces were considered. Different scenarios were created when obstacles were present or in their absence. For example, Figure (27) considers the speed between situations where the carrier travels on a smooth surface (X) and a rough surface (Y). No obstacles are placed in front of the carrier in both instances, but the distance was restricted to 0.5 m. To obtain the correlation coefficient, statistical analysis was conducted with software such as Microsoft Excel, which requires pairs of data, processes it using the analysis tool kit and returns a result between 0 and 1. This index is called Pearson's correlation coefficient. Next, regression analysis is conducted based on the same data. In this instance, plots of the paired data sets are conducted as scattered plots. Then the regression analysis function is used to produce the regression equation. In sum, Figure (27) implies that, the more the X (speed) the more the Y (speed) with a very strong relationship that is almost linear. Figure (27) shows the relationship between the correlated speeds and the equation defining them is y = 1.6889x – 0.2389. Figure (28) implies that X and Y are on an increasing trend but it is not a very tight moderate linear association. Figure (28) shows the relationship between the correlated speeds and the equation defining them y = 1.9453x – 0.287

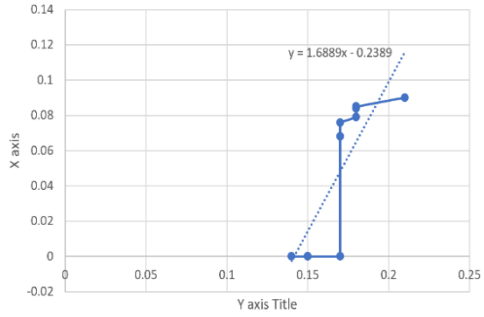

Figure 27. X (speed) against Y (speed) for 0.5. For distance 0.5m, correlation between rough (Y) and smooth (X) surfaces with no obstacles, r = 0.8

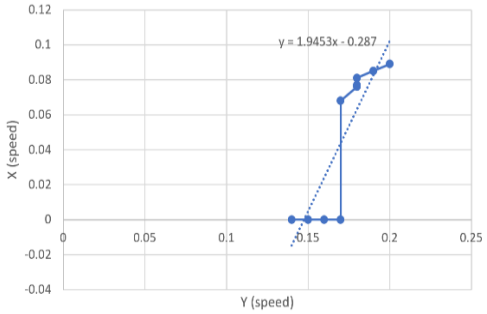

Figure 28. X (speed) against Y (speed) for 1.0. For distance 1.0m, correlation between rough (Y) and smooth (X) surfaces with no obstacles

Figure (29) means at this condition, the relationship between X (speed) and Y (speed) is almost non-existent (barely linear).The above graph shows the relationship between the correlated speeds and the equation defining them y = 2.1715x – 0.323. Figure (30) implies that, the more the X (speed) the more the Y (speed) with a very strong relationship that is almost linear. Figure (30) shows the relationship between the correlated speeds and the equation defining them y = 1.9772x – 0.2954. Figure (31) implies that, the more the X (speed) the more the S (speed) with a strong relationship that is nearly linear. Figure (31) shows the relationship between the correlated speeds and the equation defining them y = 0.5694x – 0.059. Figure (32) implies that, the more the X (speed) the more the S (speed) with a very strong relationship that is almost linear. Figure (32) shows the relationship between the correlated speeds and the equation defining them y = 0.8088x – 0.0993.

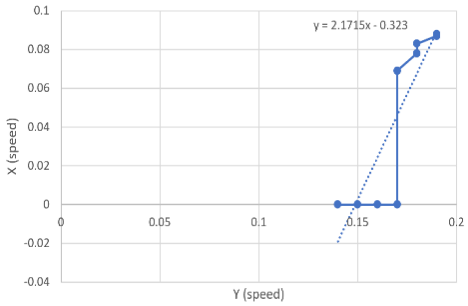

Figure 29. X (speed) against Y (speed) for 1.5. For distance 1.5m, correlation between rough (Y) and smooth (X) surfaces with no obstacles

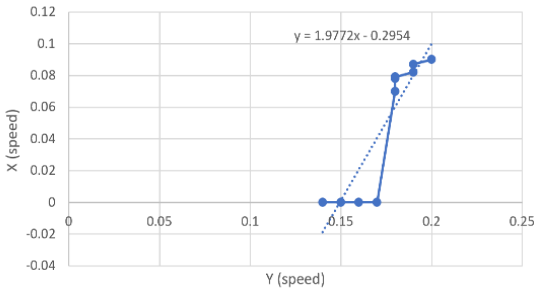

Figure 30. X (speed) against Y (speed) for 2.0. For distance 2.0m, correlation between rough (Y) and smooth (X) surfaces with no obstacles

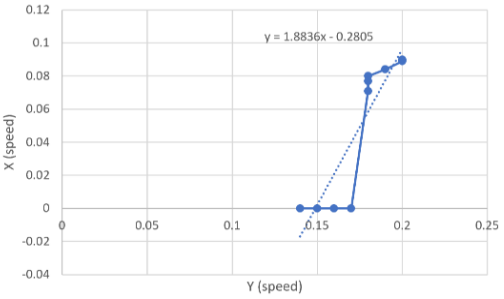

Figure 31. X (speed) against Y (speed) for 0.5. For distance 0.5m, correlation between smooth (S) and smooth (X) surfaces for no obstacle and an obstacle avoidance

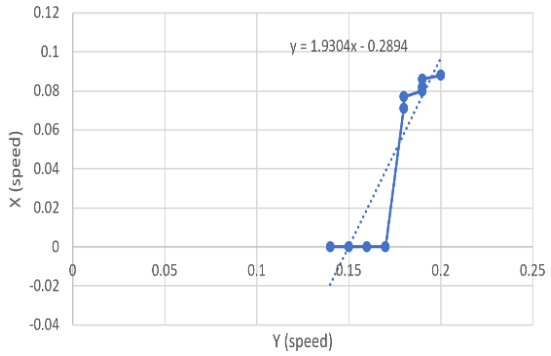

Figure 32. For distance 1.0m, correlation between smooth (S) and smooth (X) surfaces for no obstacle and an obstacle avoidance

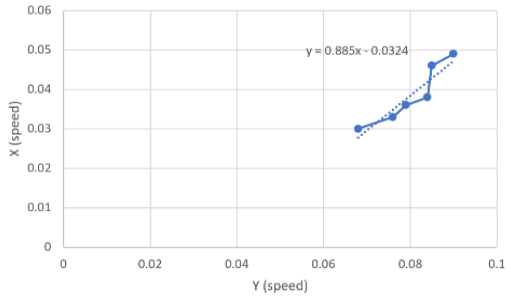

Figure 33. X (speed) against Y (speed) for 0.5. For distance 0.5m, correlation between rough (Y) and rough (Z) surfaces for no obstacle and an obstacle avoidance

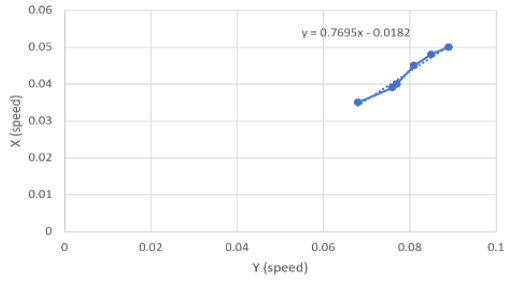

Figure 34. For distance 1.0m, correlation between rough (Y) and rough (Z) surfaces for an obstacle avoidance

For Figure (33), the value, r cannot be 2.1, the data is not very precise for rough surfaces especially in obstacle avoidance. Figure (33) shows the relationship between the correlated speeds and the equation defining them y = 0.885x – 0.0324. For Figure (34), the value, r cannot be 3.7, the data is not very precise for rough surfaces especially in obstacle avoidance. Figure (34) shows the relationship between the correlated speeds and the equation defining them y = 0.7695x – 0.0182

Description	Correlation (r)
No obstacle, smooth and rough surfaces; 0.5m	0.8
No obstacle, smooth and rough surfaces; 1.0m	0.5
No obstacle, smooth and rough surfaces; 1.5m	0.1
No obstacle, smooth and rough surfaces; 2.0m	0.89
No obstacle, smooth and rough surfaces; 2.5m	0.89
No obstacle, smooth and rough surfaces; 3.0m	0.9
Obstacle avoidance, smooth and rough surfaces; 0.5m	0.84
Obstacle avoidance, smooth and rough surfaces; 1.0m	0.88
Obstacle avoidance and no obstacle, smooth and smooth surfaces; 0.5m	0.78
Obstacle avoidance and no obstacle, smooth and smooth surfaces; 1.0m	0.89

In summary, the approximate and smooth surfaces timing are so remote to each other which is because of each of the following:

(1) Friction and rolling resistance: Wheels move with little resistance to smooth surfaces whereby less energy is lost in the movement of the wheels, providing greater speeds and reduced time to accomplish the movement tasks. The surface irregularities cause a rolling resistance when the wheels were experimented on rough surfaces. It can also be observed on the results, the loss of energy occurs as it vibrates and in exchange of heat in a way of forcing more torque by the motor in order to sustain the motion at a lower speed and over a long period of time.

(2) Traction and slippage: Wheels can also be rotating in high speed whereby carrier is moving at low speed providing low effective speed and unpredictable time. The rough surfaces provide more grip to the wheels resulting in less slip and transferring more of the torque to motion.

(3) Power Consumption and Consumption of Motor Load: Motors get hotter on rough surfaces as the motor will pull the current of more resistance and thus battery is exhausted more quickly leading to the motor running faster.

Novelty Statements

-The novelties of this article are described as follows:

Testing a prototype automobile automated carrier during operation and under obstacle avoidance conditions on smooth and rough surfaces at different distances (0.5m, 1m, 15m, 2m, 2.5m and 3m). This is a shift from manual handling of semi–knockdown automobiles that is labour-intensive, inefficient, and unsafe to an automatic, efficient and safe system. This idea is unique in that it considers distances and load, which guides in understanding the optimal operational ranges of sensors driving the carriers.

-Establishing the correlation between smooth and rough surfaces under loads at different distances with and without obstacles. This is new for automobile carriers in that the dynamic physical interfaces coupled with the complex vehicle – road surface system have been omitted in the literature to date.

5. Conclusions

Today, despite the level of technology available to engineering workshops, very little of these have been used to benefit the workforce in automobile workshops. This is at the expense of potential accidents, excessive human effort utilization and the risk of low productivity. The argument in this work is that load-carriers should be developed and used to load and unload semi-knockdown vehicles as they are received from the suppliers. In this work, the design, development and testing of a load carrier for semi-knockdown vehicles of (1540 - 2030)kg vehicle is proposed. An (L * B * H)m load carrier of (J Kg) was developed using a prototype scale which could be up-scaled in future design modifications. It was designed and built in the mechanical engineering computational laboratory at the University of Lagos, Nigeria.

The automated carrier prototype design performance assessment was carried out under varying conditions of surfaces, illustrating the relationship that exists between the surface features (smooth and rough surfaces) and the mobility behaviour of the system in line

graphs. Based on the graphical analysis, it can be concluded that the smoother surfaces offered better traction and more stable movement, taking less time to reach their destination, and the rough and irregular surfaces offered more levels of slippage and lower efficiency in motion, taking more time to reach their stop point. The findings also showed that the prototype carrier had a poor capability of sustaining a consistent motion at low levels of voltage supply, especially on rough surfaces. This implies that inadequate power delivery restricts torque production at the mecanum wheels, thus contributing to high slippage and a reduction in general propulsion efficiency. On the other hand, the increase and constant levels of voltages led to better repeatability of performance across surfaces.

In general, the outcomes of the experiment confirm the significance of surface conditions and power availability in automated carrier systems. This work can be useful in highlighting the weakness of the present prototype, navigation control and the necessity of the traction control. These observations provide an excellent foundation for the improvement of design and optimisation of the system in the future.

Sensors and control limitations of the work

This paper has demonstrated the use of a prototype automated carrier in providing real-time intelligent navigation data with accurate regulation tasks [36]. Nonetheless, certain limitations were encountered that could be tackled in future studies to make the idea of intelligent navigation and control discussed in this work more robust. These limitations are presented in the following points:

1. Operational constraint of ultrasonic sensor and detection accuracy: The ultrasonic sensor has a small field of coverage beyond which the prototype carrier fails to notice obstacles, thereby leading to the detection of false obstacles by deceitful echoes of the floor. The replacement of the ultrasonic sensor with a LiDAR sensor placed at the centre of the prototype carrier, with higher capacity, could resolve this problem. Furthermore, activating more than one ultrasonic sensor at a particular time, and an inadequate delay between successive activation cycles can cause cross-talk of the sensor, leading to inaccurate distance measurement. The LiDAR sensor could equally resolve this challenge.
2. Dynamic obstacle tracking constraints: During the obstacle avoidance test, the automated carrier prototype continued to evade obstacles but failed to follow a specific route. It moves sporadically, evading obstacles. Moreover, the system is not effective in dynamic environments, as it hardly follows speeding obstacles. This limitation could also be tackled by the LiDAR sensor of higher capacity than the ultrasonic sensor used in this work.
3. Response of servo motors and mechanical delays: The servo motors used for sensor rotation in the prototype operated at low speed, thereby experiencing a delay in obstacle detection. To resolve this, a LiDAR sensor could be introduced.
4. The present investigation has been limited to successfully navigating load carriers only but further works will be directed to the weight applied on the load carrier.

Working on semi knockdown vehicles, the design and utilization of load carriers has implications for ergonomics, efficiency, safety and cost. From the ergonomic perspective, using load carriers reduces the risk of musculo-skeletal disorders, lowers the operational cost of the workshop by reducing injury associated expenses and re-orientate both the workers and system owners towards human-centered activities. In this situation, the design proposed in this work attempts to reduce awkward postures of the car body operators. These postures may include uncontrolled twisting and bending of the body therefore, the present design aids in optimizing movement parts through secure loading and unloading of the semi-knockdown vehicles. Based on the proposal prototype, the load carrier enhances efficiency in the workshop since it reduces movement. It also optimizes space utilization, coupled with the stabilization of loads. As the efficiency is improved, weak security and consequent release of the components is made while the center of gravity of the load remains well managed to afford a stable vehicle dynamic. Moreover, engineering design cost is central to the management of new designs in automobile centers. With the current design, it is possible to optimize such cost which includes material selection, expert analysis and software design cost.

6.References

[1] J. Christian and G. Jochen, "How to load your auto carrier: a hybrid packing approach for the auto-carrier loading problem," Eur. J. Oper. Res., vol. 315, pp. 1167-1181, 2024. https://doi.org/10.1016/j.ejor.2024.01.001

[2] A. Saikumar, P. Pranay, N. Sarvigari, N. Sai Kumar, and G. Vikram, "Design and fabrication of compact load carrier," Int. J. Res. Publ. Rev., vol. 6, pp. 8310-8318, 2025.

[3] K. P. Devesh, S. Tushar, R. Suraj, and V. K. Ojha, "Ergonomic design of multipurpose load carriers: a comprehensive review," Int. J. All Res. Educ. Sci. Methods., vol. 11, pp. 2639-2644, 2023.

[4] K. Karthik and N. Rakesh, "Autonomous load carrier," UCPH Res. Portal, p. 3, 2020.

[5] C. Valentin, C. David, V. H. Edwin, D. Stijn, and V. Thierry, "Automation in cargo loading/unloading processes: do unmanned loading technologies bring benefits when both purchase and operational cost are considered?," J. Shipp. Trd., vol. 8, p. 20, 2023. https://doi.org/10.1186/s41072-023-00146-9

[6] O. Esan and J. Ajilore, "Design, fabrication and evaluation of an autonomous guided vehicle with a unit load carrier," Int. J. Sci. Technol. Res., vol. 10, Article 13140, 2020.

[7] K. S. Reethya, "Design and development of a plant watering robot using Arduino microcontroller," Int. J. Sci. Res. Eng. Manag., vol. 10, Article 55041, 2024.

[8] A. Muazu, S. Abdulkadir, and A. Abdulrazaq, "Design and construction of a low-cost autonomous firefighting robot using Bluetooth module and Arduino," Global J. Res. Eng. Comput. Sci., vol. 10, Article 5281, 2024.

[9] I. N. Mukthar, A. B. Muhammad, and A. Shehu, "Simulation of obstacle avoidance robots," Middle East Res. J. Eng. Technol., vol. 3, pp. 66-74, 2023. https://doi.org/10.36348/merjet.2023.v03i05.003

[10] K. G. Danladi, A. B. Muhammad, A. Hussein, and A. S. Nuhu, "Drawbacks of employing self-driving automobiles," IPHO-J. Adv. Res. Sci. Eng., vol. 2, Mar. 2024.

[11] A. B. Muhammad, "The several uses for obstacle-avoidance robots," Global J. Res. Eng. Comput. Sci., vol. 3, pp. 11-17, 2023.

[12] Z. Anye, L. Yongyang, T. Einat, and A. Shubham, "Car-following behavior of human-driven vehicles in mixed-flow traffic: a driving simulator study," IEEE Trans. Intell. Veh., vol. 8, 2023.
https://doi.org/10.1109/TIV.2023.3257962
[13] N. Konda, V. K. Naveen, and Y. S. Padma, "Real-time traffic signs and obstacle detection in self-driving car," Int. J. Innov. Technol. Explor. Eng., vol. 10, Article 35940, 2020.
[14] A. F. Faysal, R. Asef, I. Amirul, and A. Ajmy, "Arduino-controlled multi-function robot with Bluetooth and NRF24L01+ communication," IJRCS, vol. 4, pp. 1353-1381, 2024.
https://doi.org/10.31763/ijrcs.v4i3.1517
[15] A. B. Muhammad and I. B. Mukthar, "Elements needed to implement the obstacle-avoidance robots," GJRECS, vol. 3, pp. 18-27, 2023.
[16] U. Faikul, M. Fuad, S. Iswanto, and M. Alfian, "Obstacle avoidance based on stereo vision navigation system for omni-directional system," JRC, vol. 4, Apr. 2023.
https://doi.org/10.18196/jrc.v4i2.17977
[17] F. Morariu, M. Timotei, and R. Sever-Gabriel, "Mobile robot vision navigation strategy based on Pixy2 camera," Eur. J. Oper. Res., vol. 10, Article 1016, 2023.
https://doi.org/10.1016/j.matpr.2023.04.327
[18] T. Nuntachai and C. Phichitphon, "Design and construction of electric wheelchair with mecanum wheels," JRC, vol. 4, pp. 71-82, Jan. 2023.
https://doi.org/10.18196/jrc.v4i1.17095
[19] U. N. Akhtar, S. Saifullah, and A. N. Raheel, "Design and implementation of force sensation and feedback systems for telepresence robotic arms," JRC, vol. 3, Sept. 2022.
https://doi.org/10.18196/jrc.v3i5.15959
[20] S. Kolapo, O. O. Moses, O. A. Solomon, M. A. Ogunlade, C. M. Anthony, A. Oladimeji, and O. D. Samuel, "Development of an Arduino-based obstacle avoidance robotic system for an unmanned vehicle," ARPN J. Eng. Appl. Sci., vol. 13, 2018.
[21] N. Chidvilasini, M. P. Giri, L. S. Lalith, and S. S. Vivek, "Arduino-based ultrasonic distance measurement and analysis system," IJERT, vol. 14, Aug. 2025.
[22] A. Wadea, M. G. Atef, M. Rafeek, and A. Abdulaziz, "Evaluating instructional format and performance metrics in manual assembly task," J. Work., vol. 10, Article 1177, 2025.
[23] L. Ruimin and L. Yanjiao, "Geometry accuracy of carriers in automobile assembly workshop and its check methods," J. Mech. Sci. Technol., vol. 38, pp. 6841-6846, 2024.
https://doi.org/10.1007/s12206-024-1135-4
[24] T. Amirreza, "A unified MILP for choosing between line stocking, manual kitting, and hybrid human-robot kitting in car assembly," vol. 10, Article 13140, 2025.
[25] W. Manuel, S. Daniel, K. Alexander, M. Philipp, P. Corina, and Z. Helmut, "The integration of smart systems in the context of industrial logistics in manufacturing enterprises," Procedia Comput. Sci., vol. 200, pp. 727-737, 2022.
https://doi.org/10.1016/j.procs.2022.01.271
[26] S. Prathamesh, S. P. Chaitanya, S. S. Ganesh, and S. P. Srishti, "Transforming MSME assembly operations: smart manual assembly table for improved productivity," IJBAS, vol. 10, pp. 40-51, 2025.
https://doi.org/10.14419/am35a906
[27] J. Ish, "Actual and contractual carriers," vol. 10, p. 12, 2025.
[28] E. O. Lois, "Prevalence of work-related musculoskeletal disorders among male manual material handling in markets across Port Harcourt metropolis," TRESJ, vol. 17, 2025.
[29] B. Khan, N. Sehrish, N. H. Kazi, N. Saad, and N. Sarwat, "A prototype of obstacle avoidance for autonomous vehicle," Iraqi J. Sci., vol. 63, pp. 2203-2210, 2022.
https://doi.org/10.24996/ijs.2022.63.5.33
[30] E. Gernot, V. Rien, L. Stefan, H. Hunter, S. Raffaele, P. Rodolfo, P. Thomas, and S. Karl, "Advances in cable yarding: a review of recent developments in carriers for mobile skyline cable yarding," Curr. For. Rep., vol. 11, Article 14, 2025.
https://doi.org/10.1007/s40725-025-00246-8
[31] R. Andrea, D. Pierpaolo, and B. Francesco, "Prototyping of automated guided vehicles for teaching practical mechatronics," Educ. Sci., vol. 15, Article 3390, 2025.
https://doi.org/10.3390/educsci15030294
[32] P. Thibault, A. Nabil, B. Valeria, and C. Diego, "Optimization of human-aware logistics and manufacturing systems: a survey on the human-aware models," Eur. J. Oper. Res., vol. 13, Article 100137, 2024.
https://doi.org/10.1016/j.ejtl.2024.100137
[33] P. Thibault, A. Nabil, B. Valeria, and C. Diego, "Optimization of human-aware logistics and manufacturing systems: a comprehensive review of modeling approaches and applications," Eur. J. Oper. Res., vol. 13, Article 100136, 2024.
https://doi.org/10.1016/j.ejtl.2024.100136
[34] A. L. Simon and N. M. Munashe, "Innovative approaches to enhancing logistics for adapting to the evolving demands of manufacturing companies in East Africa through improved lean strategies," WJARR, vol. 23, Article 30574, 2024.
https://doi.org/10.30574/wjarr.2024.23.3.2840
[35] S. Kazemi and S. Gentes, "Development of robot system as a tool-carrier machine for the decontamination process of nuclear power plants," SaND, vol. 16, pp. 52-54, 2025, 5th IFSA Winter Conf. Automation, Robotics & Commun. Industry 4.0/5.0 (ARCI'2025), Granada, Spain, Feb. 19-21, 2025.
[36] T. Thierry and V. Sri Sudha, "Augmented reality for quality inspection, assembly and remote assistance in manufacturing," Eur. J. Oper. Res., vol. 232, pp. 533-543, 2024.
https://doi.org/10.1016/j.procs.2024.01.053

APPENDIX

- **Forward motion only without obstacle**

```
// === Pin Definitions ===
// front right wheel
const int enA_F = 5;
const int in1_F = 4;
const int in2_F = 2;

// front left wheel
const int enB_F = 3;
const int in3_F = 13;
```

```
const int in4_F = 12;

// back right wheel
const int enA_B = 9;
const int in1_B = 11;
const int in2_B = 10;

// back left wheel
const int enB_B = 6;
const int in3_B = 8;
const int in4_B = 7;

// === Setup ===
void setup()
{
 // set motor pins as outputs
 pinMode(in1_F, OUTPUT);
 pinMode(in2_F, OUTPUT);
 pinMode(in3_F, OUTPUT);
 pinMode(in4_F, OUTPUT);
 pinMode(in1_B, OUTPUT);
 pinMode(in2_B, OUTPUT);
 pinMode(in3_B, OUTPUT);
 pinMode(in4_B, OUTPUT);

 pinMode(enA_F, OUTPUT);
 pinMode(enB_F, OUTPUT);
 pinMode(enA_B, OUTPUT);
 pinMode(enB_B, OUTPUT);
}

// === Loop ===
void loop()
{
 // Move forward
 forward(215); // speed = 200 (0–255)
 // delay(10000);
}

// === Functions ===
void forward(int speedVal)
{
 // //front right
 digitalWrite(in1_F, LOW);
 digitalWrite(in2_F, HIGH);

 // front left
 digitalWrite(in3_F, LOW);
 digitalWrite(in4_F, HIGH);

 // back right
 digitalWrite(in1_B, LOW);
 digitalWrite(in2_B, HIGH);

 // back left
 digitalWrite(in3_B, LOW);
 digitalWrite(in4_B, HIGH);

 // set speed
 analogWrite(enA_F, speedVal);
 analogWrite(enB_F, speedVal);
 analogWrite(enA_B, speedVal);
 analogWrite(enB_B, speedVal);
}
```

- **Forward motion and obstacle avoidance**

```
/* mecanum_obstacle_controller_safe.ino
  GNU89-style Arduino sketch
  Two ultrasonic sensors on complementary servos (default: left=83°, right=72°).
  Fuzzy logic + sweep search to pick manoeuvre. Mecanum motor controls provided.
  Safety: maintain MIN_SAFE_DIST (15 cm) at all times.
*/

#include <Arduino.h>
#include <Servo.h>

/* ====== TUNABLE CONSTANTS ====== */
const int SERVO_LEFT_DEFAULT  = 83;  /* default forward-facing for left sensor */
const int SERVO_RIGHT_DEFAULT =72;  /* default forward-facing for right sensor */

const int SWEEP_DELTAS[]     = {0, 15, 30, 45}; /* degrees to add to defaults when sweeping */
const int SWEEP_STEPS        = sizeof(SWEEP_DELTAS) / sizeof(SWEEP_DELTAS[0]);

const int SWEEP_SETTLE_MS    = 160; /* wait after moving servo before sampling */
const int CROSS_TALK_GAP_MS  = 60;  /* time between sensor triggers to reduce cross-talk */
const int SENSOR_SAMPLES     = 3;   /* average samples per reading */
const int SENSOR_SAMPLE_GAP  = 20;  /* ms between samples */
const long SENSOR_TIMEOUT_US = 30000L;/* pulseIn timeout (microseconds) */

const int MIN_SAFE_DIST      = 15;  /* cm - robot must never go closer than this */
const int SAFE_RESUME_DIST   = MIN_SAFE_DIST + 3; /* hysteresis before resuming forward */

const int CLOSE_THRESH       = 20;  /* cm - triggers obstacle handling */
const int FAR_THRESH         = 60;  /* cm */
const int PREFERENCE_MARGIN  = 10;  /* cm advantage to prefer a side */
const int MEDIUM_THRESH      = (CLOSE_THRESH + FAR_THRESH) / 2; /* 40cm */

const int NORMAL_SPEED       = 255; /* PWM 0..255 */
const float CAUTIOUS_FACTOR  = 0.5f; /* reduce speed when medium is detected */

const int TURN_STEP_MS       = 200; /* small rotate step while re-checking */
const int ANGULAR_STEP_MS    = 350; /* small angular-forward step */
const int BACKUP_MS          = 450; /* small reverse in ms */
const int UTURN_MS           = 1600; /* conservative 180 deg rotate time (tune) */

const int MAX_ESCAPE_ATTEMPTS = 3;  /* try a few things before U-turn */

/* ====== PINS (as provided) ====== */
/* motors */
#define FORWARD 1
#define BACKWARD 2
#define RELEASE 3

/* front left motor */
const int enA_F = 5;
const int in1_F = 4;
const int in2_F = 2;

/* front right motor*/
const int enB_F = 3;
const int in3_F = 13;
const int in4_F = 12;
```

```
/* back left motor*/
const int enA_B = 11;
const int in1_B = 9;
const int in2_B = 10;

/* back_right motor*/
const int enB_B = 6;
const int in3_B = 8;
const int in4_B = 7;

/* servos & sensors */
const int SERVO1_PIN = A0;   /* left servo */
const int SERVO2_PIN = A1;   /* right servo */

const int TRIG_LEFT  = A3;
const int ECHO_LEFT  = A2;
const int TRIG_RIGHT = A5;
const int ECHO_RIGHT = A4;

/* ====== GLOBALS ====== */
Servo servoLeft;
Servo servoRight;

int motorSpeed = NORMAL_SPEED;

/* ====== PROTOTYPES ====== */
int  readDistanceSingle(int trigPin, int echoPin);
int  readDistanceFiltered(int trigPin, int echoPin);
void moveServosToDefault(void);

/* fuzzy prototypes */
float mem_close(int d);
float mem_medium(int d);
float mem_far(int d);
int   fuzzy_decide(int leftDist, int rightDist);

/* motor prototypes (we use motorSpeed) */
void front_right(int dir);
void front_left(int dir);
void back_right(int dir);
void back_left(int dir);

void stopMotors(void);
void forward(void);
void backward(void);
void left(void);
void right(void);
void forwardLeft(void);
void forwardRight(void);
void backwardLeft(void);
void backwardRight(void);
void rotateLeft(void);
void rotateRight(void);
void setSpeed(int pwm); /* set global speed - future: ramping */

/* higher-level behaviours */
int  scanForBestCandidate(int *bestSide, int *bestDelta, int *bestClearance);
int  executeManoeuvreChoose(int side, int delta, int clearance);
void performUTurn(void);
void smallRotateStep(int side); /* side: 0=left,1=right */

/* helpers */
int  clampServoAngle(int a);
int  oppositeSide(int s);

/* ====== SETUP ====== */
void setup()
{
  Serial.begin(9600);
  /* motor direction pins */
  pinMode(in3_F, OUTPUT);
  pinMode(in4_F, OUTPUT);
  pinMode(in1_F, OUTPUT);
  pinMode(in2_F, OUTPUT);
  pinMode(in3_B, OUTPUT);
  pinMode(in4_B, OUTPUT);
  pinMode(in1_B, OUTPUT);
  pinMode(in2_B, OUTPUT);

  pinMode(enA_F, OUTPUT);
  pinMode(enB_F, OUTPUT);
  pinMode(enA_B, OUTPUT);
  pinMode(enB_B, OUTPUT);

  /* ultrasonic pins */
  pinMode(TRIG_LEFT, OUTPUT);
  pinMode(ECHO_LEFT, INPUT);
  pinMode(TRIG_RIGHT, OUTPUT);
  pinMode(ECHO_RIGHT, INPUT);

  /* attach servos */
  servoLeft.attach(SERVO1_PIN);
  servoRight.attach(SERVO2_PIN);

   /* default positions (complementary: same direction but different base
angles) */
  servoLeft.write(SERVO_LEFT_DEFAULT);
  servoRight.write(SERVO_RIGHT_DEFAULT);

  delay(300);

  stopMotors();

      Serial.println("Mecanum fuzzy obstacle avoidance - ready
(MIN_SAFE_DIST enforced)");
}

/* ====== MAIN LOOP ====== */
void loop()
{
  int leftDist  =readDistanceFiltered(TRIG_LEFT, ECHO_LEFT);
  delay(CROSS_TALK_GAP_MS);
  int rightDist = readDistanceFiltered(TRIG_RIGHT, ECHO_RIGHT);

  /* Immediate safety: if inside the safety bubble, stop and back up */
  if (leftDist< MIN_SAFE_DIST || rightDist< MIN_SAFE_DIST)
  {
    Serial.println("EMERGENCY: too close to obstacle - backing up");
    stopMotors();
    backward();
    delay(BACKUP_MS);
    stopMotors();
    /* after backing up, return to loop to re-evaluate sensors */
    return;
  }

  /* continuous default sensing + cautious approach at medium */
  if (leftDist> CLOSE_THRESH &&rightDist> CLOSE_THRESH)
  {
    /* if medium anywhere, approach cautiously */
          if (leftDist<= MEDIUM_THRESH || rightDist<=
MEDIUM_THRESH)
    {
      setSpeed((int)(NORMAL_SPEED * CAUTIOUS_FACTOR));
    }
    else
    {
      setSpeed(NORMAL_SPEED);
    }
```

```
    /* continue forward and keep sensing (loop will re-sample) */
    forward();
    return;
  }

  /* Obstacle detected (within CLOSE_THRESH): stop and begin sweep search */
  stopMotors();
  Serial.println("Obstacle detected - searching for path");

  int bestSide = -1;      /* 0 = left, 1 = right */
  int bestDelta = 0;
  int bestClear = 0;

  int found = scanForBestCandidate(&bestSide, &bestDelta, &bestClear);

  if (!found)
  {
    /* no candidate found - attempt a small backup and re-scan a few times */
    Serial.println("No candidate found - backing up and retrying");
    backward();
    delay(BACKUP_MS);
    stopMotors();

    static int escapeAttempts = 0;
    escapeAttempts++;
    if (escapeAttempts>= MAX_ESCAPE_ATTEMPTS)
    {
      Serial.println("Escape attempts exhausted - performing U-turn");
      performUTurn();
      escapeAttempts = 0;
    }
    return; /* loop will re-evaluate sensors after backup/UTURN */
  }

  /* Prefer side with significantly greater clearance */
  int otherSide = oppositeSide(bestSide);
  int otherClear = 0;
  if (otherSide == 0)
    otherClear = readDistanceFiltered(TRIG_LEFT, ECHO_LEFT);
  else
    otherClear = readDistanceFiltered(TRIG_RIGHT, ECHO_RIGHT);

  /* If difference less than margin, pick the actual farther side */
  if (otherClear>bestClear + PREFERENCE_MARGIN)
  {
    bestSide = otherSide;
    bestClear = otherClear;
    Serial.println("Opposite side preferred due to clearance advantage");
  }

  /* Choose and execute manoeuvre */
  int ok = executeManoeuvreChoose(bestSide, bestDelta, bestClear);
  if (!ok)
  {
    Serial.println("Manoeuvre failed, will attempt alternate strategies next loop");
  }
  else
  {
    Serial.println("Manoeuvre executed - resuming forward");
    moveServosToDefault();
    setSpeed(NORMAL_SPEED);
    forward();
  }

  /* next iteration will re-sample sensors */
}

/* ====== SENSOR & SERVO HELPERS ====== */

/* trigger and read one ultrasonic pulse; returns distance in cm or 999 on timeout */
int readDistanceSingle(int trigPin, int echoPin)
{
  unsigned long duration;
  int distance = 999;

  digitalWrite(trigPin, LOW);
  delayMicroseconds(2);
  digitalWrite(trigPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin, LOW);

  duration = pulseIn(echoPin, HIGH, SENSOR_TIMEOUT_US);
  if (duration == 0)
    return 999;

  distance = (int)((duration * 0.034) / 2.0);
  return distance;
}

/* average multiple sensor samples, ignoring timeouts if possible */
int readDistanceFiltered(int trigPin, int echoPin)
{
  int sum = 0;
  int count = 0;
  int i;
  for (i = 0; i< SENSOR_SAMPLES; i++)
  {
    int d = readDistanceSingle(trigPin, echoPin);
    if (d > 0 && d < 900)
    {
      sum += d;
      count++;
    }
    delay(SENSOR_SAMPLE_GAP);
  }
  if (count == 0)
    return 999;
  return sum / count;
}

void moveServosToDefault(void)
{
  servoLeft.write(SERVO_LEFT_DEFAULT);
  servoRight.write(SERVO_RIGHT_DEFAULT);
  delay(SWEEP_SETTLE_MS);
}

/* ====== SCANNING & CANDIDATE SELECTION ======
  Scans deltas (same sign applied to both servos so sensors look same direction),
  takes sequential readings (left then right) to avoid cross-talk,
  and returns best candidate found by clearance.
  bestSide: 0 left, 1 right; bestDelta is delta index; bestClearance is measured cm.
*/
int scanForBestCandidate(int *bestSide, int *bestDelta, int *bestClearance)
{
  int i;
  int bestC = -1;
  int bestS = -1;
  int bestD = 0;

  for (i = 0; i< SWEEP_STEPS; i++)
  {
    int delta = SWEEP_DELTAS[i];

        int leftAngle =clampServoAngle(SERVO_LEFT_DEFAULT + delta);
```

```
      int rightAngle = clampServoAngle(SERVO_RIGHT_DEFAULT + delta);

      servoLeft.write(leftAngle);
      servoRight.write(rightAngle);
      delay(SWEEP_SETTLE_MS);

      int leftClear = readDistanceFiltered(TRIG_LEFT, ECHO_LEFT);
      delay(CROSS_TALK_GAP_MS);
            int rightClear = readDistanceFiltered(TRIG_RIGHT, ECHO_RIGHT);

      if (leftClear>bestC)
      {
        bestC = leftClear;
        bestS = 0; /* left */
        bestD = delta;
      }
      if (rightClear>bestC)
      {
        bestC = rightClear;
        bestS = 1; /* right */
        bestD = delta;
      }

      if (bestC>= FAR_THRESH)
        break;
    }

    /* opposite-direction sweep if needed */
    if (bestC< FAR_THRESH)
    {
      for (i = 0; i< SWEEP_STEPS; i++)
      {
        int delta = -SWEEP_DELTAS[i];

         int leftAngle =clampServoAngle(SERVO_LEFT_DEFAULT + delta);
         int rightAngle = clampServoAngle(SERVO_RIGHT_DEFAULT + delta);

        servoLeft.write(leftAngle);
        servoRight.write(rightAngle);
        delay(SWEEP_SETTLE_MS);

        int leftClear = readDistanceFiltered(TRIG_LEFT, ECHO_LEFT);
        delay(CROSS_TALK_GAP_MS);
              int rightClear = readDistanceFiltered(TRIG_RIGHT, ECHO_RIGHT);

        if (leftClear>bestC)
        {
          bestC = leftClear;
          bestS = 0;
          bestD = delta;
        }
        if (rightClear>bestC)
        {
          bestC = rightClear;
          bestS = 1;
          bestD = delta;
        }

        if (bestC>= FAR_THRESH)
          break;
      }
    }

    if (bestC< 0)
      return 0;

    *bestSide = bestS;
    *bestDelta = bestD;
    *bestClearance = bestC;

    Serial.print("Best candidate: side=");
    Serial.print(bestS == 0 ? "LEFT" : "RIGHT");
    Serial.print(" delta=");
    Serial.print(bestD);
    Serial.print(" clearance=");
    Serial.println(bestC);

    return 1;
}

/* ====== MANOEUVRE SELECTION & EXECUTION ======
   Maps candidate into manoeuvre types and executes with continuous sampling.
*/
int executeManoeuvreChoose(int side, int delta, int clearance)
{
   /* if very large clearance -> angular-forward (safe to drive diagonally) */
   if (clearance >= FAR_THRESH)
   {
     if (side == 0) /* left */
       forwardLeft();
     else
       forwardRight();

     unsigned long start = millis();
     while (millis() - start < ANGULAR_STEP_MS)
     {
       int l = readDistanceFiltered(TRIG_LEFT, ECHO_LEFT);
       delay(CROSS_TALK_GAP_MS);
       int r = readDistanceFiltered(TRIG_RIGHT, ECHO_RIGHT);
       if (l < MIN_SAFE_DIST || r < MIN_SAFE_DIST)
       {
         stopMotors();
           Serial.println("Emergency stop during angular-forward: within MIN_SAFE_DIST");
         return 0;
       }
     }
     stopMotors();
     return 1;
   }

   /* if moderate clearance -> rotate in place toward side (sharp) */
   if (clearance >= CLOSE_THRESH + 6)
   {
     unsigned long start = millis();
     unsigned long timeout = 3 * TURN_STEP_MS;
     while (millis() - start < timeout)
     {
       if (side == 0)
         rotateLeft();
       else
         rotateRight();

       delay(TURN_STEP_MS);
       stopMotors();

       int l = readDistanceFiltered(TRIG_LEFT, ECHO_LEFT);
       delay(CROSS_TALK_GAP_MS);
       int r = readDistanceFiltered(TRIG_RIGHT, ECHO_RIGHT);

       /* require safe resume distance before continuing forward */
       if (l > SAFE_RESUME_DIST && r > SAFE_RESUME_DIST)
       {
         return 1;
       }
```

```
        if (l < MIN_SAFE_DIST || r < MIN_SAFE_DIST)
        {
          stopMotors();
                Serial.println("Emergency stop during rotate: within MIN_SAFE_DIST");
          return 0;
        }
      }

      /* rotation timed out; fail */
      return 0;
    }

    /* no safe manoeuvre -> back up then U-turn */
    if (clearance > 999)
    {
      /* fallback */
      return 0;
    }

    backward();
    delay(BACKUP_MS);
    stopMotors();
    performUTurn();
    return 1;
}

/* U-turn: rotate ~180 degrees conservatively */
void performUTurn(void)
{
    Serial.println("Performing U-turn");
    rotateRight();
    delay(UTURN_MS);
    stopMotors();
    moveServosToDefault();
}

/* small rotate step (left/right) helper */
void smallRotateStep(int side)
{
    if (side == 0) rotateLeft();
    else rotateRight();
    delay(TURN_STEP_MS);
    stopMotors();
}

/* ====== FUZZY LOGIC (membership & decision) ====== */

float mem_close(int d)
{
    if (d <= 0) return 1.0f;
    if (d >= CLOSE_THRESH) return 0.0f;
    return (float)(CLOSE_THRESH - d) / (float)CLOSE_THRESH;
}

float mem_medium(int d)
{
    if (d <= CLOSE_THRESH || d >= FAR_THRESH) return 0.0f;
    float mid = (CLOSE_THRESH + FAR_THRESH) / 2.0f;
    if (d <= mid)
      return (float)(d - CLOSE_THRESH) / (mid - CLOSE_THRESH);
    else
      return (float)(FAR_THRESH - d) / (FAR_THRESH - mid);
}

float mem_far(int d)
{
    if (d <= CLOSE_THRESH) return 0.0f;
    if (d >= FAR_THRESH) return 1.0f;
      return (float)(d - CLOSE_THRESH) / (float)(FAR_THRESH - CLOSE_THRESH);
}

int fuzzy_decide(int leftDist, int rightDist)
{
    float Lc = mem_close(leftDist);
    float Lm = mem_medium(leftDist);
    float Lf = mem_far(leftDist);

    float Rc = mem_close(rightDist);
    float Rm = mem_medium(rightDist);
    float Rf = mem_far(rightDist);

    float forward_strength = 0.0f;
    float turn_left_strength = 0.0f;
    float turn_right_strength = 0.0f;
    float back_away_strength = 0.0f;

    /* Forward: both far */
    forward_strength = minimum(Lf, Rf);

    /* Medium contributes via OR */
    if (Lm> 0.0f || Rm > 0.0f)
    {
      float med_strength = maximum(Lm, Rm) * 0.6f;
      forward_strength = maximum(forward_strength, med_strength);
    }

    /* turning */
    turn_right_strength = minimum(Lc, Rf);
    turn_left_strength = minimum(Rc, Lf);

    /* one side close & other medium increases turn */
    turn_right_strength = maximum(turn_right_strength, minimum(Lc, Rm) * 0.8f);
    turn_left_strength  =maximum(turn_left_strength,  minimum(Rc, Lm) * 0.8f);

    back_away_strength = minimum(Lc, Rc);

    /* choose max */
    float best = forward_strength;
    int action = 0; /* forward */
    if (turn_left_strength> best) { best = turn_left_strength; action = 1; }
    if (turn_right_strength> best) { best = turn_right_strength; action = 2; }
    if (back_away_strength> best) { best = back_away_strength; action = 3; }

    if (best < 0.05f) action = 0;
    return action;
}

/* ====== MOTOR CONTROL FUNCTIONS (use motorSpeed) ====== */

void setSpeed(int pwm)
{
    motorSpeed = pwm;
    if (motorSpeed> 255) motorSpeed = 255;
    if (motorSpeed< 0) motorSpeed = 0;
}

void front_right(int dir)
{
    switch (dir)
    {
    case FORWARD:
      analogWrite(enA_F, motorSpeed);
      digitalWrite(in1_F, LOW);
      digitalWrite(in2_F, HIGH);
      break;
    case BACKWARD:
```

```
      analogWrite(enA_F, motorSpeed);
      digitalWrite(in1_F, HIGH);
      digitalWrite(in2_F, LOW);
      break;
    default:
      analogWrite(enA_F, 0);
      digitalWrite(in1_F, LOW);
      digitalWrite(in2_F, LOW);
      break;
    }
}

void front_left(int dir)
{
    switch (dir)
    {
    case FORWARD:
      analogWrite(enB_F, motorSpeed);
      digitalWrite(in3_F, LOW);
      digitalWrite(in4_F, HIGH);
      break;
    case BACKWARD:
      analogWrite(enB_F, motorSpeed);
      digitalWrite(in3_F, HIGH);
      digitalWrite(in4_F, LOW);
      break;
    default:
      analogWrite(enB_F, 0);
      digitalWrite(in3_F, LOW);
      digitalWrite(in4_F, LOW);
      break;
    }
}

void back_right(int dir)
{
    switch (dir)
    {
    case FORWARD:
      analogWrite(enA_B, motorSpeed);
      digitalWrite(in1_B, LOW);
      digitalWrite(in2_B, HIGH);
      break;
    case BACKWARD:
      analogWrite(enA_B, motorSpeed);
      digitalWrite(in1_B, HIGH);
      digitalWrite(in2_B, LOW);
      break;
    default:
      analogWrite(enA_B, 0);
      digitalWrite(in1_B, LOW);
      digitalWrite(in2_B, LOW);
      break;
    }
}

void back_left(int dir)
{
    switch (dir)
    {
    case FORWARD:
      analogWrite(enB_B, motorSpeed);
      digitalWrite(in3_B, LOW);
      digitalWrite(in4_B, HIGH);
      break;
    case BACKWARD:
      analogWrite(enB_B, motorSpeed);
      digitalWrite(in3_B, HIGH);
      digitalWrite(in4_B, LOW);
      break;
    default:
      analogWrite(enB_B, 0);
      digitalWrite(in3_B, LOW);
      digitalWrite(in4_B, LOW);
      break;
    }
}

/* convenient motion functions using mecanum drive primitives */
void stopMotors(void)
{
    front_left(RELEASE);
    front_right(RELEASE);
    back_left(RELEASE);
    back_right(RELEASE);
}

void forward(void)
{
    front_left(FORWARD);
    front_right(FORWARD);
    back_left(FORWARD);
    back_right(FORWARD);
}

void backward(void)
{
    front_left(BACKWARD);
    front_right(BACKWARD);
    back_left(BACKWARD);
    back_right(BACKWARD);
}

void left(void)
{
    /* strafe left (mecanumbehaviour) */
    front_left(BACKWARD);
    front_right(FORWARD);
    back_left(FORWARD);
    back_right(BACKWARD);
}

void right(void)
{
    /* strafe right */
    front_left(FORWARD);
    front_right(BACKWARD);
    back_left(BACKWARD);
    back_right(FORWARD);
}

void forwardLeft(void)
{
    front_left(RELEASE);
    front_right(FORWARD);
    back_left(FORWARD);
    back_right(RELEASE);
}

void forwardRight(void)
{
    front_left(FORWARD);
    front_right(RELEASE);
    back_left(RELEASE);
    back_right(FORWARD);
}

void backwardLeft(void)
{
    front_left(BACKWARD);
    front_right(RELEASE);
    back_left(RELEASE);
```

```
  back_right(BACKWARD);
}

void backwardRight(void)
{
  front_left(RELEASE);
  front_right(FORWARD);
  back_left(FORWARD);
  back_right(RELEASE);
}

void rotateLeft(void)
{
  front_left(BACKWARD);
  front_right(FORWARD);
  back_left(BACKWARD);
  back_right(FORWARD);
}

void rotateRight(void)
{
  front_left(FORWARD);
  front_right(BACKWARD);
  back_left(FORWARD);
  back_right(BACKWARD);
}

/* ====== small helpers ====== */
int clampServoAngle(int a)
{
  if (a < 0) return 0;
  if (a > 180) return 180;
  return a;
}

int oppositeSide(int s)
{
  return (s == 0) ?1 : 0;
}

/* min / max helpers for floats */
float minimum(float a, float b) { return (a < b) ?a : b; }
float maximum(float a, float b) { return (a > b) ?a : b; }
```