Control and Path Planning of Mobile Swarm Robots Using Blockchain Technology with Particle Swarm Optimization

Temidayo Samuel Adeoti and Yekeen Olajide Olasoji

***Abstract* — Technological advancement has made robots become rampant in industrial automation and globalization, as it's fast and efficient in delivering tasks in industries with no supervision. This work shows the use of blockchain technology (smart contract) in controlling a swarm of robots combined with particle swarm optimization for solving the navigation path. The technique in this research modeled a new fitness function, that uses the optimal path generation and barrier avoidance for the mobile robot's movement within the swarm, while the blockchain smart contract was integrated to control the robot's speed. Simulated results validate path optimization and speed control with PSO and BT.**

***Keywords* — Autonomous Mobile Robot, Blockchain Technology (BT), Particle Swarm Optimization (PSO), Robotchain, Smart Contract (SC), Swarm Robotics.**

I. Introduction

Robots are becoming rampant in industrial automation, especially in a major task that is collectively carried out by humans in factories and other day-to-day activities. This involves several tasks such as fixing bolts to devices, movement of devices, selection of products, and rendering of services. In the early integration of robots in industry globalization, these tasks carried out by robots are being supervised by a human in order for the robotics to deliver optimally on their actions. This demonstrated the part of Human-Robot Interaction (HRI) in the early stage of industrial globalization, where the major work carried out by robots needs to be supervised and control manually [1].

The modern industrial and automation application cannot be carried out by a single robot, but by a collaborative work of the robot. Application using collaborative works of robots has now been in their dominant phase for industrial expansion and automation. The new era of robot collective labor is mostly for group tasks that cannot be completed by a single robot. It entails workplace monitoring in some factories in order to avoid collisions through the use of vision-based reactive planning [2] or external sensors that utilize the robot's distance from humans to halt it [3]. Other implementations employ pressure sensors to stop or slow down the robot when there is a space violation [4]. Robotic swarming is a multi-robot field in which a large number of robots are coordinated and controlled in a dispersed and decentralized manner to complete a set of tasks. It is based on the employment of local rules and basic robots in a swarm to improve communication and productivity in relation to the job complexity.

A hard task done by social insects with a unique sense of cooperation and collaboration based on a large number of basic robots that can complete difficult tasks more effectively than a single robot, giving strength and flexibility to the team, inspired the study of swarm robotics. Many applications and existing robotic integration methods might be disrupted by swarm robotics. The robot swarm is strong and adaptable because each robot in the swarm can perform independent operations and can be programmed to do a range of jobs without requiring physical change [5]. The information transferred between the nodes determines the success of a swarm of robots. One of their most significant issues is the use of local vs global information and control methods in robotic networks [6]. Because each robot in the swarm has limited detecting and computational capability, it is impossible for it to have a full perspective of the whole swarm system. A swarm of robots, unlike a single robot that fails during a single operation, will not fail during the execution of the activity.

Blockchain technology was introduced by Satoshi Nakamoto to implement the cryptocurrency known as Bitcoin in 2008. Since their inception, both have grown in terms of worldwide adoption, overall value, levels, and popularity [7]. This paved the way for researchers to study them. The impending value of blockchain is not only holding cryptocurrencies for financial purposes but also allowing the combination of a large number of different networks on the same defined platform in a secure and decentralized way to achieve uniformity in results [8]. The creation of Ethereum was proposed in 2013 as the second stage in the advancement of blockchain technology [9] It introduced new features in blockchain technology, such as smart contracts (this is a set of rules that are applied in real time on the blockchain, when certain contract submission conditions are met.) and proof of authority, giving it the dominant power to integrate more services and have more value across many disciplines and areas of academic research. In the field of robotics, where integration with blockchain is still in its infancy, there are few approaches that show how the two technologies can be used together to overcome the challenges. The introduction of blockchain technologies in robotic systems, especially in a large number of robots, can solve many of the problems

Submitted on July 21, 2022.
Published on August 16, 2022.
T. S. Adeoti, Department of Electrical and Electronics Engineering, Federal University of Technology Akure, Nigeria.
(corresponding e-mail: temidayoadeoti@yahoo.com)
Y. O. Olasoji, Department of Electrical and Electronics Engineering, Federal University of Technology Akure, Nigeria.
(e-mail: yoolasoji@futa.edu.ng)

facing systems [10] The problem to be solved is not limited to the following: The first issue it can address is security: while many systems suffer from data integrity and dependability difficulties, blockchain can allow secure peer-to-peer connection over an untrustworthy network. Another benefit of this integration is the ability to make distributed decisions in the swarm system, as blockchain can ensure that all participants in a decentralized network share the same views of the connected system.

Blockchain technology's applications and widespread adoption in the sector are currently getting extensive attention. For information security, we can use blockchains in Industrial Control Systems (ICS). Blockchain's core technologies, including as distributed ledgers, asymmetric cryptography, consensus algorithms, and smart contracts, can be applied in domains other than cryptocurrencies. The technology, with its inherent distinguishing characteristics of decentralization, security, immutability, transparency, and high availability, can increase robotic swarm collaboration and decision-making. The proper use of blockchain technologies to robotic systems could overcome a number of issues that these systems currently face. [11] Shows how to create a blockchain and keep robotic events on it. This approach allows for the creation of smart contracts that employ data acquired in the wild by various robots and initiate actions based on the contracts recorded and validated on the blockchain network. This can increase industrial output and save time spent on tasks such as replacing tools for a robot that used the blockchain to convey its need for new tools to continue working.

As a continuation of past work, we suggest the establishment of a blockchain for robotic event registration using Tezos's technology [12], It makes use of the enhanced security provided by Tezos' explicit verification system. This ongoing work will support smart contracts that execute AI code on the blockchain and have been verified to be valid (to do exactly what their specification defines). They also want to adapt the blockchain to allow for several more transactions per second than the present standard permits, allowing the system to accommodate a large number of interacting robots. One such example may be found in an Amazon warehouse, where an army of robots works together to run the facility.

The benefits of merging blockchain technology with robotics, particularly swarm robotics and robotic hardware, are presented in a study undertaken by [13]. Both the simplicity of scalability and the durability against failure are advantages of robotic swarms. These benefits stem from the fact that these swarms' members are dispersed. We can also see how the industrial sector is increasing and allowing enterprises to reach increased productivity. One of the major issues in robotic swarms is that robots employ local information, that implies how they only have knowledge about themselves and/or other robots in their immediate vicinity. If blockchain is incorporated into these systems, however, it may be able to send global information to the robots, allowing all of the robots in the system to collaborate. It could provide the robots global knowledge, allowing all robots in the system to be aware of all shared data, which might be useful in a number of situations. Because global information enables the entire system to instantly adapt behavior in response to particular robot requests, blockchain integration can help the system change behavior more quickly. This might be accomplished using a controller robot that uses blockchain data to evaluate the status of the system and subsequently commits the changes to the blockchain. Higher productivity and autonomous robot's swarms could result from these enhancements and the system's global information. Robotic swarm, anti-collision, and obstacle avoidance are critical for task implementation.

In this work, the robotics system will be combined with blockchain technology to achieve autonomous robot swarms for workspace monitoring, as well as particle swarm optimization to solve the mobile robot navigation path. The method used in this study optimizes the path generated by intelligent mobile robots within a workplace from their origin to their destination without colliding. To achieve the desired outcome, a novel fitness function was developed that satisfies obstacle avoidance and optimal path generation for the swarm's robots, while a blockchain smart contract was incorporated to manage the robots' speed during navigation.

II. Materials And Methods

A. Proposed Methodology Overview

Using blockchain technology, the approach described in this work of directing robots to communicate worldwide as opposed to local accessibility of robots with each other in a swarm and avoid collisions while delivering tasks. Even though the robots will be equipped with sensors to detect the presence of other members in their region in order to avoid collisions, it is critical to force the robots to stop or change direction in the shortest path possible before a collision occurs, ensuring that no damage is done to the other robot or the material picked or carried. A PSO method would be used to obtain the robot pathways in order to achieve the desired result.

B. Mobile Robot Navigation Architecture

By establishing a fitness function that converts the problem into a minimization problem, PSO would be used to navigate swarm mobile robots. Two conditions will be considered in order to achieve our desired results and robot movement efficiency first, the robot must develop trajectories by escaping traps and avoiding obstacles; and second, the robot must get to its destination by covering a small distance in the shortest time feasible. This method will employ the defined system architecture to develop optimal travel paths for an autonomous mobile robot.

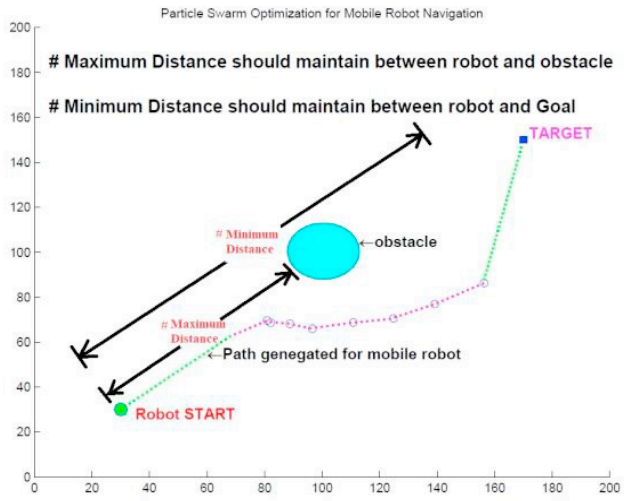

Fig. 1. Robot navigation framework [14].

Initially, without utilizing any intelligence techniques, the robots move in diverse directions directly towards the goal. When a robot's sensor detects another robot on its route, the PSO algorithm is immediately activated to find a suitable path to avoid collision and achieve the target in the shortest time possible. Fig. 5 shows how the suggested PSO algorithm allows the robot to entirely avoid the barrier and follow the desired path.

C. *Mathematical Modelling of PSO*

Individuals are referred to as particles in PSO, while the population is referred to as a swarm. So, in PSO, a swarm is defined as a set R = Q1, Q2, Q3 to Qn. The Q1, Q2, Q3 to Qn are the swarm's particles numbers symbolized with 'n'. The predefined particles are thought to travel inside the search space. Using an appropriate position shift termed velocity, the particles' new locations may be updated while they are going. Using the sent information of particles inside the swarm, the latest speed of every particle is determined. It may be done using memory, in which each particle saves the best spot it finds during its search. The best position is denoted by Ypbest and refers to the ideal location as decided by each particle. As a consequence, the swarm's 'n' particles have a total of 'n' ideal location values. The swarm's particles are exchanging information and will ultimately converge on a single global optimal location, which will be visited by all particles. The global optimal location may be determined by determining the fitness function of an individual particle in the swarm. The global best position, which is indicated by the symbol Ygbest, might be regarded as the particle with the best fitness. The decision of Ygbest indicates when a PSO iteration is complete. This approach will be performed till the robot arrives at its target or until the maximum number of iterations is achieved. Following the discovery of each Ypbest and swarm Ygbest, both the velocity and location of each particle would be updated using (1) and (2).

$$v_l(j+1) = v_l(j) + C_1 * \text{rand } 1 * \left(Y_{\text{pbest}} - x_l\right) + C_2 * \text{rand } 2 * \left(Y_{\text{gbest}} - x_i\right) \quad (1)$$

$$x_l(j+1) = x_l + v_l(j+1) \quad (2)$$

The iteration counter is represented by j; rands represent random variables while cognitive and social factors are given C1 and C2.

III. System Operation and Structure

Fig. 2 depicts the architecture for the suggested technique. It is made up of the RobotChain ledger, which includes smart contracts that run code to generate a robot state. A controlling unit sends data from the robot's condition, such as location, velocity, and energy, as well as imagery from a webcam that inspects the place where the packages must be picked, to the RobotChain. Both have access to the same computing unit in this situation, but this has no bearing on the processing or the blockchain because they are processed individually. We just record the robotic events (logs) and a tuple consisting of a Hash and an ID that reflects the picture of the packages on the blockchain because it can quickly grow in size. A database is used to store the image. The Hash of the photos adds protection to the database, which can be easily corrupted if not produced properly, because the information saved in the blockchain cannot be changed. A trusted external Oracle uses the information on the blockchain to process the photos and determine number of packages present. This data is then sent to a smart contract, which decides whether the robot should return to its home location, signaling that there are no more parcels to pick, or slow down or speed up, indicating that there are few or many packages to pick.

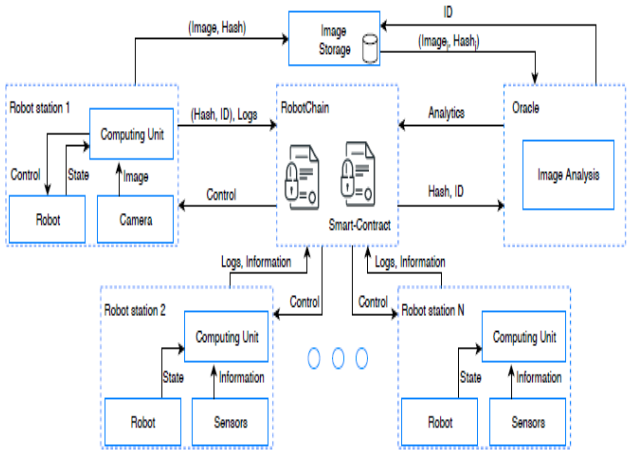

Fig. 2. System Operational Block Diagram

A. *RobotChain Module*

RobotChain would be utilized as a decentralized database to safely and quickly retain robotic activities and other data. RobotChain is a collaborative blockchain developed for industrial applications that can manage a high volume of transaction activities per second and uses off-chain mechanisms to deal with blockchains' exponential expansion. [12]. Oracles would be used to process images, with smart contracts handling data storage and logic for controlling robots. The robots were modeled and experiments were conducted using MATLAB. This universal technique may be used to control any robot in the swarm as long as the robot allows external instructions to change its speed and location, and halt it. Smart contracts autonomously conduct response

action when the requirements for them are satisfied, therefore this technology is particularly useful since it does not require involvement from a central authority to operate the robots. One of the techniques' unique features is how easy it is to maintain and adapt to new robots and assignments. To achieve this goal, and address the additional needs, simply establish a new smart contract.

B. Fitness Function Development

By establishing a fitness function that converts the problem into a minimization problem, PSO would be used to navigate swarm mobile robots. Two conditions will be considered in order to achieve our desired results and robot movement efficiency first, the robot must develop trajectories by escaping traps and avoiding obstacles; second, the robot must get to its destination by covering a small distance in the shortest time feasible. This method will employ the defined system architecture to develop optimal travel paths for an autonomous mobile robot. During the robot's navigation to the objective. If there are no barriers in the robot's environment, it can travel toward its target. As a result, no adaptive mechanism is required to move the robot inside its working area. However, generating paths for an autonomous mobile robot that detects impediments in its environment is a challenging task. When the robot detects impediments in its work region, the current research study proposes a PSO structured system for achieving optimum route trajectories. The fitness function must be modeled to match the following conditions when determining the fitness value of individual particle in the swarm:

As a primary priority, the particle's fitness should keep it as far away from the nearest hindrance as possible; in other phrases, the fitness function is straightforwardly relative to the particle's location from the nearest obstacle. As a result of this situation, a repulsive force is formed between both the particle and the hindrance.

$$F_1 \propto \left(1/dist_{P_iCOb}\right) \text{ for } 1 \le i \le n \quad (3)$$

where $dist_{P_iCOb}$ as used above indicates the distance between ith particle and the nearest hindrance.

Secondary priority condition: The particle's fitness should keep it as close to the robot's goal as possible; in other phrases, the fitness function is dependent on the distance between the particle and the target. In order to advance the robot towards its destination, an attractive action is established between the particle and the target as a result of this requirement.

$$F_1 \propto \left(dist_{p_iT}\right) \text{ for } 1 \le i \le n \quad (4)$$

where $dist_{p_iT}$ as used above indicates the distance between ith particle and the target position.

Tertiary priority condition: The fitness value of the particle should keep it as close to the robot's mission as feasible; in other phrases, the fitness value is proportionate to the particle's position from the target. As a result of this desire, an enticing action is created between the particle and the objective, allowing the robot to go closer to its goal.

$$F_1 = W_1 * \text{dis } t_{P_1T} + W_2 * \left(1/dist_{P_iCOb}\right) \quad (5)$$

The constants/controlling parameters W1 and W2 are specified as the proportionality constants/controlling parameters that may be adjusted depending on the particle, target, and nearest obstruction locations within the space.

C. Robot Path Generation

The robot would follow the shortest path to the target in order to meet the PSO criterion. We must consider the obstacle position when monitoring the shortest path for each robot. To avoid a collision, the robot must maintain the greatest possible distance from the object. Once the mobile robot has detected obstacles (Sob) within its sensing range, it may detect the nearest obstacle based on the strength of reflected radiation from the detected obstacles. The distance between the robot and the nearest obstacles can be calculated using Equation (6).

$$D_{ro(i)=}\sqrt{(R_X - Ob_X)^2 + (R_y - Ob_y)^2} \quad (6)$$

$$\text{for } 1 \le i \le S_{ob}$$

Where; i= Iteration number;
$D_{ro(i)=}$Function for Robot and obstacle distance in the *i*th iteration;
Rx= Robot current position in x-co-ordinate;
Ry= Robot current position in y-co-ordinate;
Obx(i)= Obstacle point in x-co-ordinate;
Oby(i)= Obstacle point in y-co-ordinate.

If the robot follows the shortest path, then distance between robot and goal should be minimum This can be determined using Equation (7).

$$D_{rg(i)=}\sqrt{(R_X - G_X)^2 + (R_y - G_y)^2} \quad (7)$$

Where; i= Iteration number;
$D_{rg(i)=}$ Function for Robot and Goal distance in the *i*th iteration;
Rx= Robot current position in x-co-ordinate;
Ry= Robot current position in y-co-ordinate;
Gx(i)= Goal point x-co-ordinate;
Gy(i)= Goal point y-co-ordinate.

The obstacle with the smallest distRob can be chosen as the closest obstacle using the determined 'Sob' number of distance values. When the robot discovers a nearby obstacle around its sensing region, it would spawn a haphazard population/swarm in its vicinity. One fitness function F is required to determine the fitness of individual particle in the swarm for subsequent robot motions. The length between each particle and the robot's destination and nearest barrier may be calculated if the particle, target, and nearest barrier locations are represented as (pxi; pyi), (goalx; goaly), and (Nobx;Noby).

$$\text{dist}_{p_i}T = \sqrt{\left(p_{x_i} - \text{ goal } x\right)^2 + \left(p_{y_i} - \text{ goaly }\right)^2} \quad (8)$$

$$\text{dist}_{p_i}NOb = \sqrt{\left(p_{x_i} - NOb_x\right)^2 + \left(p_{y_i} - NOb_y\right)^2} \quad (9)$$

To prevent a collision and reach the objective safely, individual particles in the swarm would compute the length between the robot and the target and barrier. The particle's fitness function would be determined using the Fitness function for each particle.

$$F_i = D_{rg} + D_{ro} \quad (10)$$

Ygbest is a particle that is distant from the closest barrier and close to the target location with the lowest fitness value. The Ygbest would be determined for multiple cycles until the robot is free of sensed impediments or has arrived at its target. Velocities of individual particles in the swarm are utilized to calculate their best position and the swarm's global agreed best position, but they have no effect on the robot's velocity. When the robot identifies the swarm's global best position, it will begin moving towards the Ygbest. Iterations will continue until the robot is no longer in range of the detected obstacles or until the maximum number of cycles is reached.

D. Mobile Robot Navigation Algorithm using PSO

Start

Sequence 1*: Set the starting point and target locations for the robot.*

Sequence 2*: The robot will continue to move at its initial speed until it detects any impediments or its intended spot.*

Sequence 3*: When the robot detects an impediment, use PSO.*

Sequence 4*: Set the random population's locations and velocities.*

Sequence 5*: Calculate swarm Ygbest along each particle's Ypbest.*

Sequence 6*: Using Eqs (1) and (2), get new locations and velocities for each particle.*

Sequence 7*: Sequence 4, 5, and 6 should be repeated until the robot gets clear of the detected impediments.*

Sequence 8*: Sequence 2 should be repeated until the robot reaches its goal.*

Stop

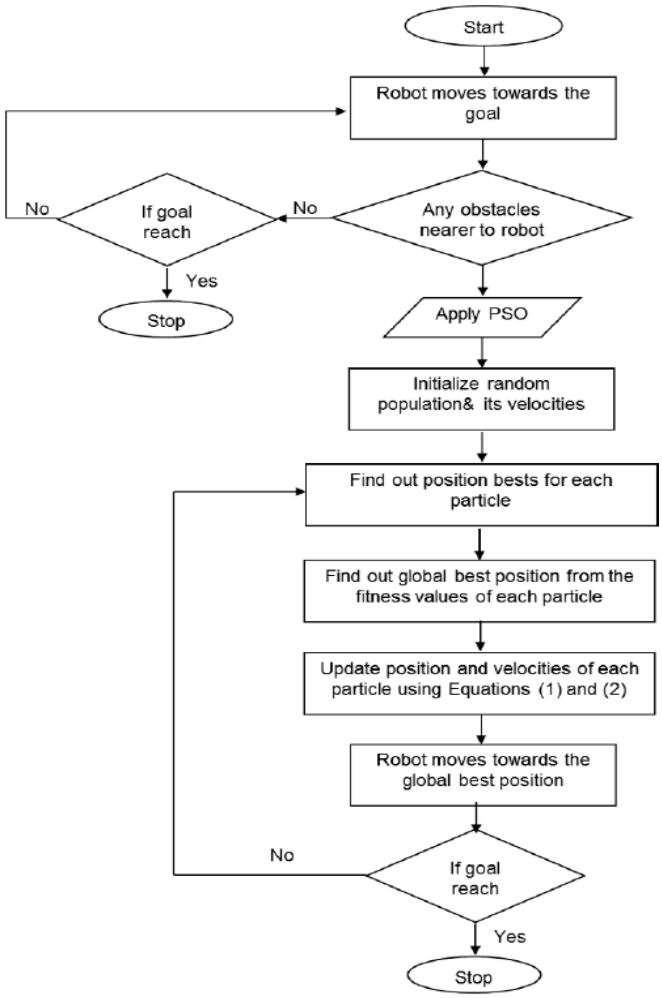

Fig. 3. Proposed Particle Swarm Optimization Flowchart.

IV. Result And Discussion

The results and analysis of this work are divided into three sections. The first section shows the analysis of the workspace, considering the robots moving to their set goals(target) without applying the PSO algorithm scheme upon sensing other robots on its path to get it shortest path to the target. The second section shows the performance of particle swarm optimization techniques in determining the shortest distance to avoid any collision or stoppage of a robot upon sensing another robot within its path at a particular range. The third section shows how the speed and movement of the robots are control through the secure data stored in the smart contract based on number of target available to pick. The time taking for robots to reach target and their respective path lengths per experiment results were evaluated and discussed. In a virtual environment, robot trajectory planning and control are developed. MATLAB 2019 is used to run the simulation. The workspace environment is a size of 20 × 20 units in the x and y axis, the unit is read as centimeters. The simulated working environment as shown in Fig. 4, it has a starting point and goal point for the robots.

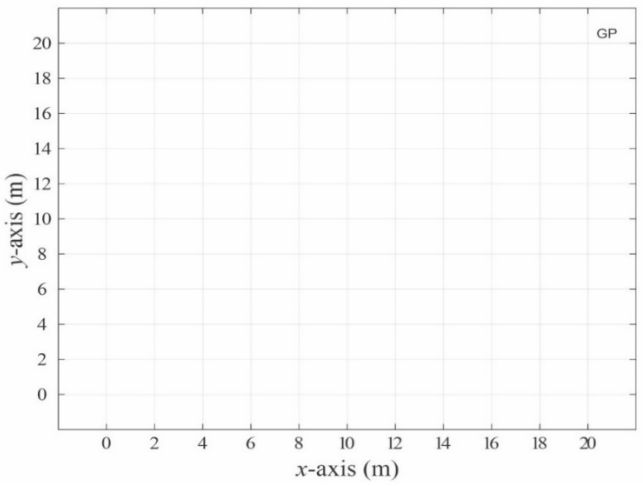

Fig. 5. Simulated Environment for the robots.

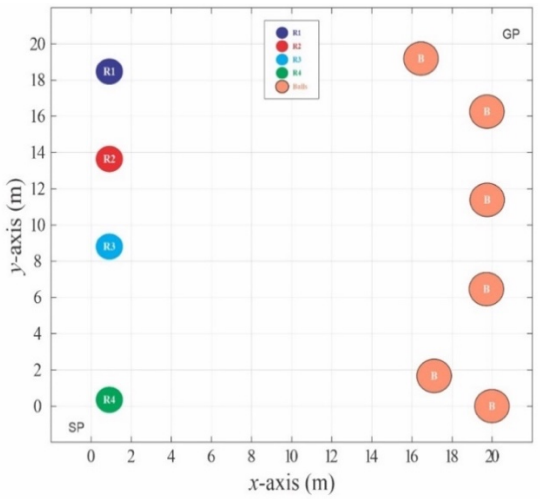

Fig. 6. Simulation showing robots and targets' position.

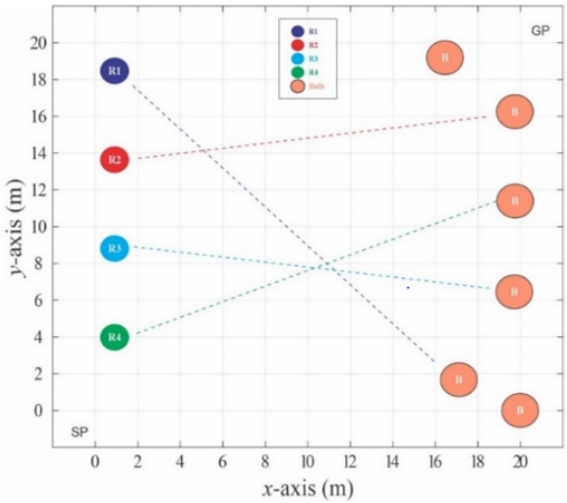

Fig. 7. Simulation showing robots and targets within workspace.

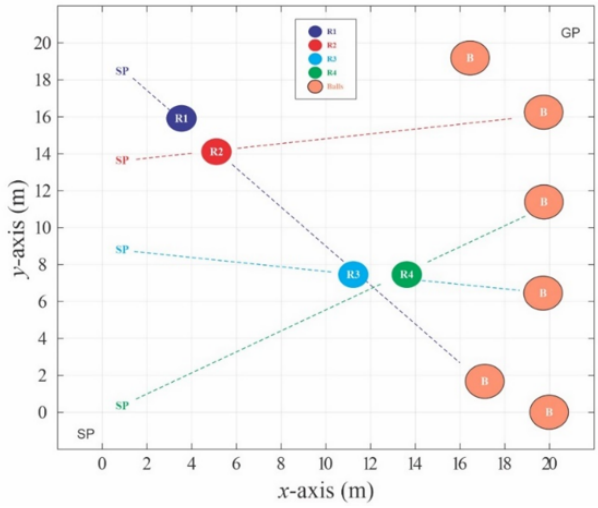

Fig. 8. Simulation showing navigation without adaptive techniques within workspace.

The robots can move in any direction within the simulated workspace depending on the target and ensure collision-free with another robot within its path range randomly in the environment. The results were obtained from MATLAB 2019 under various computations and iterations. The robot's initial positions within the workspace are shown in Fig 6.

The starting points of the robots and goal (balls to be picked up) point for the robot are shown with SP and GP respectively. The brown circle represents the target to be reached by each robot and their respective points are noted within the workspace. Because the robot is initially distanced away from the determined target, it uses its intellect and sensors to go directly towards the goal. In the simulation run, robots are represented by a ball shape with a separate route and color code. In terms of power, the robot is self-contained. It is omnidirectional; its movement speed may be limited, but it is movable. The robots are given a predetermined starting position and the desired goal location. The trials were carried out under various settings, with identical beginning velocities assigned to each robot at the start of the program.

TABLE I: THE INITIAL PARAMETER OF THE SIMULATION WITHIN THE 20×20 GRID

Robots	Robot Position (x,y)	Target Position (x,y)	Initial Velocity (m/s)
Robot 1	(1,18.5)	(17,18)	5
Robot 2	(1,13.8)	(19.8,16)	5
Robot 3	(1,9)	(19.6,6.4)	5
Robot 4	(1,4)	(19.4,11.4)	5

Table I shows the initial position of the robots along their respective velocities, the position of targets (Ball) is also inputted. These initial values are specified to set up the simulation parameter.

Once the simulation is run robot first checks for available targets based on updated parameters of other neighboring robots. In moving from the initial point to the target point as stated in the table above the robot would set it target value. Following that, a path is established for the robot to navigate and follow within the working environment after establishing target values and defining other variables. If there is an obstacle (another robot in this case) at each of its sensing regions, a conditional statement set is used to check if there is an obstacle. If there is an obstacle, an action is taken to wait for it in the neighborhood of X direction (axis) and Y direction (axis).

The next stage is to determine the distance between each point the robot passes through while navigating from the initial to the goal point.

As shown in Fig. 7, the simulation was set up with 4 robots and represented as small filled disks (Rob. 1 as blue, Rob.2 as red, Rob. 3 as light blue, and Rob. 4 as green). The goal is to guide each mobile robot (Rob. 1 to Rob. 4) to its respective objectives (Targ.1 to Targ.4) without colliding with the other robots. While navigating from the initial location to the target point, each robot employs the induced reference trajectory.

TABLE II: PATH DISTANCE WITH THE TIME TAKEN FOR ROBOTS WITHOUT ADAPTIVE TECHNIQUES

Robots	Path Length Covered by Robots (m)	Robots Duration from start point to Target (mins)
Robot 1	17.72	5.22
Robot 2	14.45	4.36
Robot 3	13.00	4.15
Robot 4	15.52	4.97

As tabulated in Table II shows the total distance of each robot to navigate and reach its desired target on simulation. It also shows how long it takes each robot to get to its desired target without colliding with any other robots. Robot 3, has the best minimum path and time to reach the target despite having to wait for Robot 4 during the simulation. The mobile robot's speed was kept constant at 5 m/s to traverse from its SP to GP using self-navigation based on information gathered through its sensors. The sensing method is achieved by the mobile robot through virtual sensors attached around the omnidirectional mobile robot with sensing range (SR) of 1.5m. Once the robot senses any obstacle within it sensing range, it triggered it mode to calculate the distance between it and the obstacle (in this case another robot). This computing process help robot to know which one has the minimum distance from each other and stop for the other to transverse on the path. As shown in Fig. 11, each robot (Rob. 1 as blue, Rob.2 as red, Rob. 3 as light blue and Rob. 4 as green) uses equation 11 to calculate it distance to obstacles sensed 1.5 m along it predefine path then share globally with other robots within it sensing range to compare their estimated distances. The simulation result as display in Fig. 12, shows Rob1 has a maximum distance calculated compared to Rob2. Rob 1 waited for the other Rob2 to pass through the path before traversing to target. This effected it time taken to reach target point. Also, Rob3 has a maximum distance calculated compared to Rob4. Rob 3 waited for Rob4 to pass through the path before traversing to target. This effected it time taken to reach target point as shown in table above. This could be addresses and solved applying adaptive techniques such as particle swarm optimization to create another shortest path to avoid collision.

A. Robot Navigation with Adaptive Techniques (PSO)

The proposed method would avoid collisions by optimizing the trajectory generated by the mobile robot from its initial point to its target position inside the work environment. We modelled new fitness method to satisfy the hindrance avoidance and optimal path movement criteria without needing to halt for each other in order to avoid collision. The robot goes towards the particle with the best fitness value to avoid obstacles and approach the objective with the shortest path possible, based on each particle in the swarm estimated fitness values. The practicality of the presented methodology is validated using simulation results.

TABLE III: CONTROL PARAMETERS USED FOR THE PSO SIMULATION

Parameters Name	Parameter Values
No of swarm Particles	50
Particles velocity range	0-3
Personal Coefficient C1	1
Global Coefficient C2	1
Rand 1and Rand 2	1
Controlling Parameter W1	0-1
Controlling Parameter W2	120-800

The particle population is set to 50 in Table III during the simulation. They are initialized at random by establishing their locations and velocities in the vicinity of the robot along its directional sensing range (say 1.5 meters for calculating purposes), with particle velocities ranging from 0 to 3. Each particle's velocity is determined by the parameters C1 and C2 as well as random1 and random2. Normally, the random variables random1 and random2 vary between 0 and 1, and these values influence velocity of the particles but not travel distance of robot. For convenience, these parameters are set to a unique value of '1'. Fitness parameters W1 and W2 would be changed based on distance traveled by the mobile robot inside its work space. It means that while changing the parameters W1 and W2, C1 and C2 are meanderingly affected to determine Ygbest. The simulation experiment was run using C1= 1 and C2= 1 for ease of consideration. rand1 = rand2 = C1 = C2= 1 PSO parameters.

Fig. 9 shows a simulation with four robots interacting with different static targets. Little filled disks are used to illustrate the robots (Rob. 1 as blue, Rob.2 as red, Rob. 3 as light blue and Rob. 4 as green). The goal is to move each mobile robot (Rob. 1 to Rob. 4) as close as possible to its respective targets (Targ.1 to Targ.4) while avoiding collisions with other robots. While navigating from the initial location to the target point, each robot employs the generated reference trajectory. When it detects an obstruction in its course, it uses PSO techniques to discover an optimal path to avoid the barrier and successfully reach its destination, as seen below.

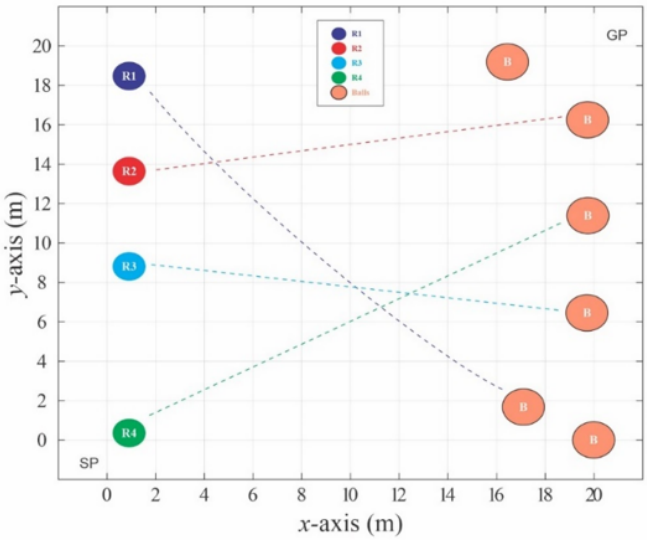

Fig. 9. Robots and targets initial position during simulation.

Because one of the goals of this research is to create a collision-free path with target searching as a secondary goal, the barrier avoidance parameter W2 was given greater weight than the target focus parameter W1 to achieve the best outcome for the report. W1=0.5 and W2=750 were used in the simulation, as well as Particle swarm optimization parameters C1=1 and C2=1, and rand1&rand 2=1.

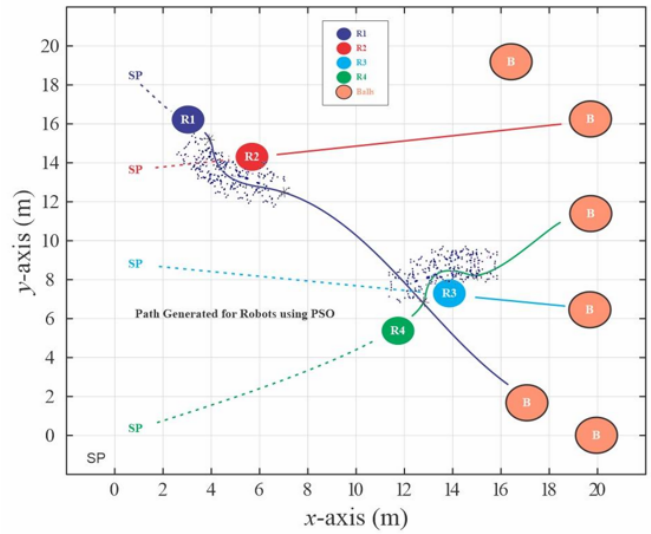

Fig. 10. Simulation showing generated PSO techniques with particles along path

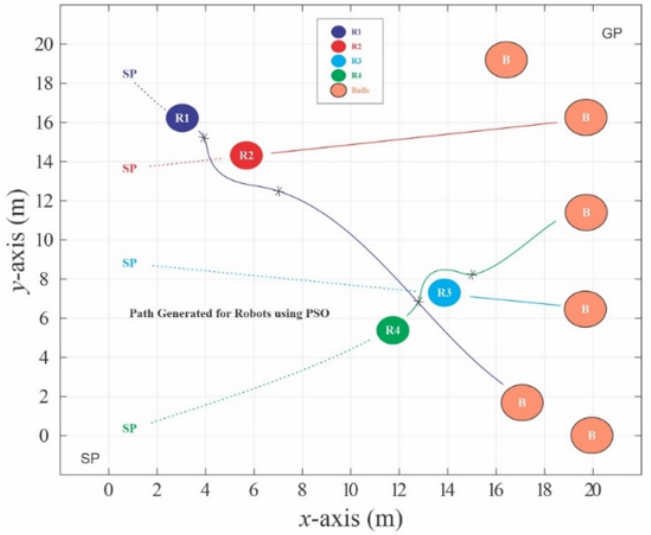

Fig. 11. Robots navigation with PSO techniques within workspace.

Fig. 15 depicts the simulation outcome. Remember from methodology Equation of the approach that the PSO particle with the lowest fitness value within the swarm is referred to as Ygbest. W1 being the first controlling parameter represents the particle's proximity to robot's destination target, while the other controlling value parameter W2 tells how distance the particle is from the nearest obstacle. So, as demonstrated in fitness equation, Ygbest can be attained by minimizing the fitness function. W1 with a high value implies the particle is close to the robot's target, whereas W1 with a low value signifies the particle is far away. Similarly, a high W2 value indicates that the particle is keeping a greater displacement from the nearest barrier, while a low W2 value shows that the particle is relatively close to barrier. As a result, the fitness function's governing parameters must be adjusted to low W1 and high W2 values, as specified in this simulation experiment.

TABLE IV: SIMULATION RESULT SHOWING TRAJECTORY LENGTH AND TIME TAKEN BY ROBOTS USING PSO

Robots	Distance covered by robots (in meters)	Time taken for robots to reach target from start point (in mins)
Robot 1	13.12	5.09
Robot 2	11.04	4.00
Robot 3	9.11	3.75
Robot 4	12.12	4.56

The total distance traveled by each robot to navigate and reach its desired goal is shown in Table IV. In comparison to the result obtained in table 4 without PSO, the path covered by each robot was reduced using particle swarm optimization. It also illustrates how each robot's time to reach its desired destination without colliding with other robots has decreased. During the simulation, Robot 3 has the best minimum path and time to reach the destination, followed by Robot 2. The mobile robot's speed was adjusted to 5 m/s to allow it to travel from its SP to GP utilizing self-navigation based on data collected by its sensors. Each robot has a 1.5 m sensing range (SR). When the robot detects an obstacle within its sensing range, it activates its distance calculation mode, which calculates the distance between the robot and the obstacle (in this case another robot). This computational procedure assists robots in determining which robot is closest to the other and applying PSO to generate a new optimum path away from the other robot in order to reach the destination. As shown in Fig. 10, each robot (Rob. 1 represents blue, Rob. 2 represents red, Rob. 3 represents light blue, and Rob. 4 represents green) uses equation 6 to calculate the distance to obstacles sensed 1.5 m along its predefined path, which it then shares globally with other robots to compare with their estimated obstacle distances for minimum distance determination.

Fig. 11 displays the simulation result, which demonstrates Rob1 has determined the maximum distance when compared to Rob2. As a result, Rob 1 used particle swarm optimization to create an optimum path around Rob2 as it traveled to its destination. In comparison to the previous experiment, the time taken to reach the goal position was lowered. Similarly, when compared to Rob3, Rob4 has a determined maximum distance. As a result, Rob 4 also established an optimum path to travel around Rob3 as it traversed to its objective utilizing particle swarm optimization.

In comparison to the previous experiment, the time taken to reach the goal position was lowered. This illustrates the use of adaptive approaches such as particle swarm optimization to generate a new shortest path to avoid collisions and optimize the robot's course, which has a substantial impact on the time taken to reach the target, as seen in Table V.

Table V compares the robot path and time with and without the use of the PSO model approach. When PSO adaptive strategies were introduced, the robot's path length and time to reach the destination were lowered compared to when they were not. The simulation results demonstrate the model techniques' capabilities, as well as how successfully the robot generates routes using the proposed algorithm, avoids obstacles, and arrives at its desired position inside the workspace surroundings. In each iteration, the robot's trajectories are selected by picking the global best site. The particle with the least fitness is considered to be the global best point among the swarm. As a result, the robot travels toward the global best position, which is repeated multiple times until the robot reaches its target position.

TABLE V: COMPARISON OF THE PSO TECHNIQUE WITH AND WITHOUT PSO TECHNIQUES ON THE ROBOTS

PSO Parameter Criteria	Robots Name	Path Length Without PSO (m)	Robot to Target Time mins	Path Length With PSO (m)	Robot to Target Time mins
No of particle=50 Particle Velocity=3 Personal Coefficient C1=1 Global Coefficient C2=1 Rand1&Rand2=1 Parameter W1=0.5; W2=750	Robot 1	17.72	5.22	13.12	5.09
	Robot 2	14.45	4.36	11.04	4.00
	Robot 3	13.00	4.15	9.11	3.75
	Robot 4	15.52	4.97	12.12	4.56

B. Robot Velocity Control

To combine robotics and blockchain, we followed the proposed way in methodology. This was utilized to manage the robot velocity in a simpler method between the initial point and their respective targets; its integration with the simulation demonstrates a revolutionary approach to controlling robots that can be tailored to various conditions and scenarios. To avoid failures, the smart contract utilized for the blockchain might be tweaked to ensure that other parameters of the robot, such as temperature and other data, do not exceed a threshold. The simulated result was safe since no one can change the control logic without the smart-contract's approval. Furthermore, the blockchain serves as a ledger that securely saves data across the swarm's robots. The velocities of each robot during the simulation are shown in the table below.

Table VI shows the robot's velocities under various settings, with velocities measured in meters per second (robot speed) and retrieved at various times (based on when conditions are met). The robot is at rest when its velocity is equal to zero. The simulation began with all robots moving at the same speed, with each robot's speed being regulated by the smart contract's defined criteria (each time obstacle like other robot is detected, PSO condition is activated to create an optimal path to navigate through obstacle and deactivated). The values displayed for each robot were within the simulation frame; however, when Robot 1 detected another robot in its route, its velocity was automatically decreased to 3m/s. This velocity was maintained until PSO was engaged to navigate the obstacle; in order to navigate and develop an optimum path, the robot changed its velocity to 2.5 m/s and then reverted to normal 5 m/s after navigating the obstacle.

TABLE VI: VELOCITY, IN M/S FOR THE RESPECTIVE ROBOTS

Velocity Condition Criteria	Robot 1 Velocity m/s	Robot 2 Velocity m/s	Robot 3 Velocity m/s	Robot 4 Velocity m/s
No obstacle detected	5	5	5	5
Obstacle detected	3	3	3	3
PSO Activated	2.5	-	-	2.5
PSO Deactivated	5	-	-	5

The same is true for every other robot. The table demonstrated how proposed technology is sufficient of managing the modelled robots' velocities and tracking speeds in real-time. The fact that the approach modifies the velocities based on the conditions met is one quality that can be noticed in the table. The approach constantly insisting on the robot moving at a specific speed.

V. CONCLUSION

This paper demonstrates how path formed by a mobile intelligent robot from its initial source location to its destination position within its work environment is optimized using a particle swarm optimization approach. To acquire fitness values for individual particle in the swarm and allow each robot to travel along the particle with the best fitness value, a novel fitness function modelled to meets the hindrance avoidance and optimal path movement requirements was utilized. During iteration, the robot's paths are selected by picking the global best site. The particle with the lowest fitness is considered the swarm's global best location. The robot moves closer to the global best position, which is repeated until the robot achieves its goal position. Simulation results were used to verify the feasibility of the suggested technique. Finally, utilizing RobotChain as a decentralized ledger, simulated results indicated an innovative way for connecting blockchain with robots. Robotic events may be registered in the simulated architecture, and smart contracts might be used to control robots. This approach restrict modification to earlier states deposited into the blockchain. Ensures smart-contract output is always free of human modifications, suggesting that no one can change the logic of a smart contract after it has been published on the blockchain. The simulation findings demonstrate that to link blockchain technology with robotics system is possible. The combination has benefits beyond merely providing possible ways to safe and transfer data; in the context of this study, the data might also be used to control robot characteristics such as velocity.

REFERENCES

[1] Thomas B. Human robot interaction. *Status and Challenges*. 2016: p. 525-532.

[2] Gautier D, Guido M, Michel D, Ambroise C, Daniel S. Reactive Planning on a collabrotive robot for industrial application. *12th International conference on information in control, automation and*

robotics; 2015. p. 450-457.
[3] Vicentini F, Pedrocchi N, Giussani M, Tosatti L. Dynamic Safety in Collaborative Robot Workspace through a Network of Devices Fufilling Functional Safety Requirements. *41st International Symposium on Robotics*; 2014: ISR/Robotik. p. 1-7.
[4] Sarkar S, Ghosh G, Mohanta A, Ghosh A, Mitra S. Arduino Based Foot Pressure Sensitive Smart Safety System for Industrial Robots. *2017 second international conference on Electrical, computer and communication technologies(ICECCT);* 2017. p. 1-6.
[5] Nishida Y, Kaneko K, Sharma S, Sakurai K. Suppressing chain size of blockchain-based information sharing for swarm robotic systems. *Sixth International Symposium on Computing and Networking Workshops (CANDARW*); 2018; Takayama, Japan: IEEE; 2018. p. 524-528.
[6] Khaldi B, Cherif F. An Overview of Swarm robotics: Swarm Intelligence applied to Multi-robotics. *International Journal of Compter Applications;* 2015. p. 3-10.
[7] Raja. *Blockchain infographic: Growth, use cases & facts.* [Online].; 2018 [cited 2021 02 15. Available from: https://www.dotcominfoway.com/blog/growth-and-facts-of-blockchain.
[8] Blockchain. *Blockchain charts - the most trusted source for data on the bitcoin.* [Online].; 2018 [cited 2021 3 24. Available from: https://www.blockchain.com/en/charts.
[9] Wood G. Ethereum: A secure decentralised generalised transaction ledger. *Eyherium Project Yellow Paper*. 2014: 1-32.
[10] Lopes V, Alexandre L. An Overview of Blockchain Integration with Robotics and Artificial Intelligence. *Ledger Journal.* 2019; 4: 1-6.
[11] Degardin B. Blockchain for Robotic Event ecognition Covilha: Universidade da Beira Interior; 2018.
[12] Fernandes M, Alexandre LA. Robotchain: Using Tezos Technology for Robot Event Management. *First Symposium on Blockchain and Robotics. MIT Media Lab*. 2018: p. 32-41.
[13] Castell´o F E. *arXiv (2016).* [Online].; 2021. Available from: http://arxiv.org/abs/1608.00695.
[14] Harshal S, Prases K M, Shubhasri K. A Robust Path Planning For Mobile Robot Using Smart Particle. *International Conference on Robotics and Smart Manufacturing;* Procedia Computer Science. 2018: 290–297.

Adeoti Temidayo Samuel. Is a lifetime learner and longtime focus on blockchain technology, smart contract, payment system governance and fintech.

He has a keen interest and advancement in blockchain technology and its applications beyond the financial scope to solve human-oriented problems.

Mr. Samuel's love for the decentralization of networks to achieve a unique autonomous system triggered this research paper on controlling a swarm of robots using blockchain technology.