MULTI ROBOT COVERAGE PATH PLANING AND NAVIGATION USING DARP (Divide Areas Algorithm for Optimal Multi-Robot Coverage Path Planning) AND A*ALGORITHM

Abstract- **This project presents a novel method of warehouse automation by using a group of e-puck robots as a proof of concept for efficient path planning and goods transportation. The primary objective is to facilitate streamlined, effective and automated transport of goods within the warehouse environment. DARP (Divide Areas Algorithm for Optimal Multi-Robot Coverage Path Planning) algorithm, is designed to efficiently achieve this task. Warehouse operations benefit greatly from the algorithm's innate capacity to deliver a non-backtracking solution with lowest coverage path length, as it ensures resource and time efficiency. A key aspect of our method is that it is client-server based and centralized. With this architecture, the robot fleet may be efficiently coordinated and managed from a central control unit. To determine the distance between each robot and the target site, we integrate the A* algorithm. We can find the best robot (based on its unique ID) for every work by finding the shortest path, which guarantees the quickest response and delivery times. Our technology greatly reduces setup complexity and time because it doesn't require any pre-mapping of the warehouse or preparatory step. The DARP algorithm and A* pathfinding are integrated, allowing our system to dynamically adjust to work demands and changes in the warehouse environment. This proof of concept illustrates the viability and efficiency of our methodology in a warehouse environment. We offer a scalable and intelligent solution by combining the area optimization of DARP with the path efficiency of A*, creating the foundation for further developments in automated warehouse logistics. This research lays the groundwork for more difficult jobs like automated pick-and-place operations in addition to demonstrating the possibility of robotic automation in warehouse management.**

Keywords: DARP ((Divide Areas Algorithm for Optimal Multi-Robot Coverage Path Planning), A* algorithm, e-puck, Centralized, lowest coverage

1. INTRODUCTION

Since the 1970s, autonomous robots have been instrumental in various high-impact applications, from deep-sea and space exploration to integration in almost all aircraft. Today, in the era of multi-robot systems, the challenges of robotics, previously solved for single robots, are being reevaluated to optimally incorporate multi-robot dynamics.

The advent of robotics in industrial applications, particularly in warehouse management, has revolutionized the efficiency and effectiveness of operations in such environments. An encouraging answer to the growing need for accurate and quick handling of commodities is the use of autonomous robotic systems. With the employment of a group of e-puck robots, this project seeks to advance this developing field by showcasing a creative method of warehouse automation.

Coverage path planning (CPP) is a fundamental robotics problem that is especially relevant to this project. CPP is vital in many robotic applications, including vacuum cleaning,

autonomous underwater vehicles[4], unmanned aerial vehicles, demining operations, automated harvesters, planetary exploration, and search and rescue missions. It involves determining an optimal path that encompasses all points of a given area, avoiding certain sub-areas like obstacles or no-fly zones.

The primary objective of this project is to showcase a proof of concept for efficient path planning and area coverage in a warehouse setting using e-puck robots. This study aims to optimize robot navigation and task allocation for goods transportation, without simulating pick-and-place tasks, by utilizing the Divide Areas Algorithm for Optimal Multi-Robot Coverage Path Planning (DARP)[2] and A* for pathfinding algorithm[5].

The scope of this project is encapsulated in a series of structured sections: Initially, in the first section, we delve into the related works on multi-robot coverage path planning (mCPP), setting the stage for our approach [1]. The next section is dedicated to the development of a kinematic model for the e-puck robot, which is fundamental to understanding and optimizing its movements within a warehouse environment. Next, we focus on creating a controller for the DARP (Divide Areas based on Robot's initial Positions) algorithm, which is instrumental in generating an optimized map that includes obstacles, enhancing navigation efficiency. This optimized map is then integrated into a centralized server, as outlined, ensuring coordinated and effective robot deployment. And then we discuss the implementation of the A* algorithm controller, leveraging the optimized map to calculate the most efficient paths for the e-puck robots. The project culminates in the final Section with a discussion on future work and a conclusion that encapsulates the achievements of the project and its potential implications for the advancement of robotic warehouse automation.

2. RELATED WORKS

Extensive research has been conducted on the use of multi-robot systems in warehouse automation. This includes

studies on robotic coordination, path planning, and task allocation to enhance efficiency and reduce operational costs. Advanced approaches for multi-robot coordination in logistic scenarios, as discussed in the provided literature, offer valuable insights into the complexities and solutions in this field.

When working with Unmanned Ground Vehicles (UGVs), particularly in warehouse environments, We have selected Navigation as a property, which encompasses a broad spectrum of behaviors. This includes collective exploration, and collective localization is crucial, enabling each UGV to accurately determine its position within the complex maze layout.

Furthermore, we have incorporated Task allocation property, which plays a pivotal role in the effective functioning of Unmanned Ground Vehicles (UGVs). This involves developing behaviors such as consensus among the robots for task distribution, ensuring that each robot is assigned roles most suited to its capabilities and current position. And division of labor, which is essential for maximizing efficiency in our project.

The e-puck robot, known for its simplicity and versatility, has been a popular subject in robotics research. Its applications span across various domains, including swarm robotics, autonomous navigation, and sensor-based tasks [5]. These studies provide critical insights into how e-puck robots can be adapted for specific tasks in warehouse-like environments, contributing to practical problem-solving in robotics.
.
The DARP algorithm stands out as a significant contribution to multi-robot coverage path planning, particularly relevant to warehouse applications. Its ability to optimally divide an area among multiple robots aligns well with the logistics of warehouse automation. Additionally, the A* algorithm, known for its efficient pathfinding capabilities, has been widely utilized in various contexts, demonstrating its effectiveness in navigating complex environments.

The integration of the DARP algorithm and A* for pathfinding is a novel approach in robotic path planning. This combination has the potential to address critical challenges in warehouse automation, such as optimizing area coverage and enhancing the speed and accuracy of goods transportation[6]. This integration is particularly relevant to your project, as it seeks to leverage these algorithms for effective coordination and navigation of e-puck robots in a warehouse setting.

II. Mathematical Model

1. Kinematic model

Differential-wheeled robots, a prominent type in current research, are exemplified by the e-puck robot (Figure:1). This robot features two independently controlled drive wheels, oriented parallel and opposite to each other, with a rotating idler wheel at the rear for balance. Forward movement involves applying equal torque to both wheels, while different torques result in left or right turns. A notable advantage is the ability to execute zero-radius turns by applying opposing torques. This robot is "non-holonomic," requiring orientation before moving in a specific direction.

The e-puck robot is a small, circular, two-wheeled robot widely used in education and research due to its versatile and open-source design. The e-puck is equipped with various sensors and actuators, making it highly adaptable for different tasks and environments. Its compact size and agility make it ideal for navigating through warehouse environments, where space can be limited, and maneuverability is crucial. The e-puck's diverse sensor suite enables it to effectively navigate[10] and interact with its environment, an essential feature for path planning and obstacle avoidance in a warehouse setting.

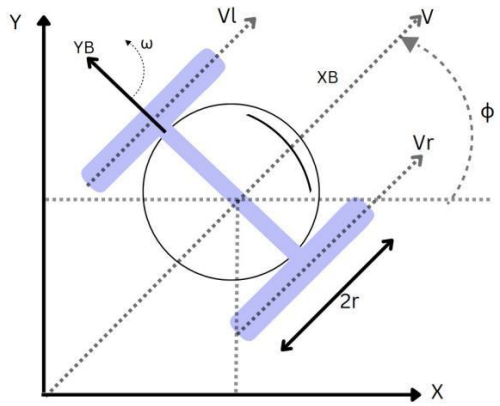

Figure1: Kinematic model of a differential drive e-puck robot

By leveraging the e-puck robot in this project, we aim to develop and demonstrate an efficient, scalable solution for warehouse automation that can potentially be applied to larger, more complex robotic systems in the future.

Information on the robot's position at all times (during the sample time) is required to implement the control algorithm that maintains the e-puck's route. We will use the special signal provided by the e-puck wheels, which is the signal of activation of the step motors, to measure the traveled distance as the robot lacks a physical encoder. One way to use this signal is as a virtual encoder. The e-puck's motors are capable of 1000 steps per revolution (360°), which is equivalent to the virtual encoder's resolution.

The kinematic equations governing differential-wheeled robots:

$$\dot{x}_{B} = v_i \sin(\varphi_i) \quad (1)$$

$$\dot{z}_{i} = v_i \cos(\varphi_i) \quad (2)$$

$$\dot{\varphi}_{I} = \omega_i \quad (3)$$

Where,

$\dot{x}_{B}$, $\dot{z}_{I}$ - absolute velocity
$\emptyset_{I}$ - angular velocity
V_i - linear velocity,

ω_I - rotational velocity,
X_i, $Z_{i,}$ $\emptyset_I$ - absolute position and orientation

In the specific case of e-puck, starting off at an initial position, P=[x y θ]T the actual position of the robot P′=P+[△x △y △θ]T can be determined knowing the change in the virtual values encoders of the wheels left $\triangle s_l$ and right $\triangle s_r$ and the distance between wheels b, by using the equation (1) ,(2) and (3).

$$P' = \begin{bmatrix} x \\ y \\ \theta \end{bmatrix} + \begin{bmatrix} \frac{\triangle s_r + \triangle s_l}{2} & \cos(\theta + \frac{\triangle s_r - \triangle s_l}{2b} \\ \frac{\triangle s_r + \triangle s_l}{2} & \sin(\theta + \frac{\triangle s_r - \triangle s_l}{2b} \\ & \frac{\triangle s_r + \triangle s_l}{b} \end{bmatrix}$$

2. Divide Area Algorithm based on Position (DARP)

In this section we describe DARP (Divide areas based on robots' initial position) a optimized technique that divides the given terrain into no_r robot exclusive regions[10]. Initially we divide the environment with the obstacles as shown in Fig (2).

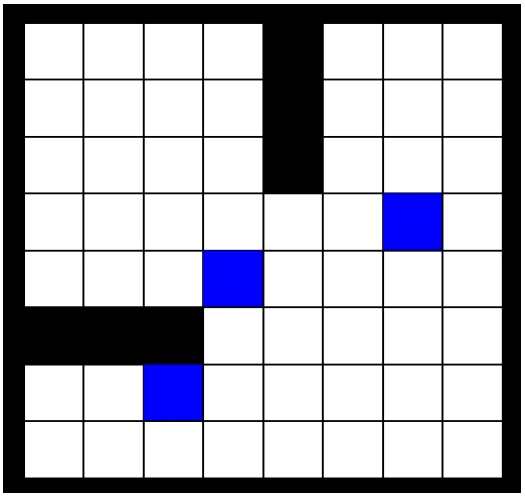

Figure 2: Initial Grid environment with the Robots and Obstacle

For easy understanding , it is assumed that the terrain which is to be explored is constrained with in a rectangular area bounded by (x,y) coordinates and discretized into finite set of equal cells.

$$U = x, y : x \in [1, rows], y \in [1, columns] \quad (4)$$

where rows, cols denote the number of rows and columns that remain after the terrain to be covered has been discretized [10]. It appears that the formula n = rows × cols gives the total number of cells in the terrain. Additionally, it is believed that U is not obstructed in any known a-priori places. The collection of unidentified challenges is shown as:

$$B = \{(x, y) \epsilon \text{ u: } (x, y) \text{ is occupied}\} \quad (5)$$

And the overall set is reduced : $\mathcal{L} = u \backslash B$, and the number of cells to be covered is reduced to $l = n - n_o$

Two cells (x_i,y_i) and (x_j,y_j) is considered adjacent if :
$\|x_i - x_j\| + \|y_i - y_j\| \leq 1$

As valid robot path of length m is considered every sequence of cells:
$X = ((x_1, y_1),\ldots\ldots(x_m,y_m))$
where the following constraints are hold – $(x_i, y_i) \in L$, $\forall i \in \{1,...,m\}$ – every two sequential cells, i.e. (xi, yi) and (xi+1, yi+1), are adjacent (Definition 1), $\forall i \in \{1,...,m-1\}$. Moreover, a closed path of length m is a path, as defined previously, where the additional condition is maintained.
Moreover, a closed path of length m is a path, as defined in previously, where the additional condition is hold – (x1, y1) and (x_m, y_m) are adjacent The robot positions are defined as:

$$\chi i(t) = (x_i, y_i) \in L, \forall i \in \{1,...,nr\} \quad (5)$$

where t denotes the specific time-stamp of the coverage path and n_r denotes the number of operational robots. The (given) initial position of the ith robot inside L is represented as $\chi_i(t0)$. Having the above formulation in mind, the mCPP problem2 can be transformed to calculate the robots' paths $X* i \forall i \in \{1,...,nr\}$ so as, minimize:

$$X \max i \in \{1,...,nr\} |Xi| \text{ subject to } X1 \cup X2 \cup \cdots \cup X_{nr} \supseteq L \quad (6)$$

where |Xi| denotes the length of the path Xi.

Definition 3 A selection $\{L_1, L_2, \ldots, L_{n_r}\}$ composes an optimal solution for the mCPP, iff

1. $L_i \cap L_j = \emptyset, \forall i, j \in 1, \ldots, n_r, i \neq j$
2. $L_1 \cup L_2 \cup \cdots \cup L_{n_r} = \mathcal{L}$
3. $|L_1| \approx |L_2| \cdots \approx |L_{n_r}|$
4. L_i is connected $\forall i \in 1, \ldots, n_r$
5. $\chi_i(t_0) \in L_i$

Figure 3: Constraints for optimal solution

If all the conditions from Figure 3, are satisfied we can tell that the solution is optimal.

3. A* Algorithm and usage

A fundamental tool in computer science, especially for pathfinding and graph traversal, the A* (A-star) algorithm is essential to our study since it allows for effective navigation in a warehouse setting. A* is distinguished by its capacity to determine the shortest path between a given beginning point and a target location. It was initially developed as an intelligent search algorithm. It finds the least expensive path by quickly navigating a graph. This algorithm balances the exploration of uncharted territory (heuristic function)[4] with the evaluation of the previously traveled path (cost function), combining in a

novel way the elements of Dijkstra's algorithm and the Best-First-Search technique.

The expression for this function is the sum of two functions:

1) The cost from the starting node to the current node is essentially represented by the path cost function, g (n).
2) A heuristic approximation of the desired node's distance from the present node. It has the value h(n).

From the starting point to the goal point, this evaluation function, f(n) = h(n) + g(n), keeps the two functions in balance. Beginning with the start node, A* keeps track of the nodes to be visited in order of priority (or "open list"), together with their associated costs (value of f(n). The nodes that have been visited are also included in another list known as the "closed list"[10]. The back pointer to the visited node is also included in this list. The back pointer indicates the node that was the source of the visited node.

A* is employed for dynamic and effective navigation, particularly in situations where the robots are not in their starting positions, while DARP effectively partitions the warehouse area for exploration[4]. It efficiently navigates the robots across the grid of the warehouse by calculating the best routes from their current places to their intended destinations. The robot fleet's operational efficiency is greatly increased by the algorithm's implementation, which guarantees the shortest possible journey time and distance.

IV. Theoretical Analysis

In this section we try to validate how the primary behaviors of our Multirobot system: Collective Exploration and Collision avoidance is achieved and justify it.

Initially the DARP algorithm adopts the following cell to robot assignment scheme. For every robot an evaluation matrix E_i is maintained, this matrix expresses the reachability between the cells and Robot position .

$$A_{x,y} = \text{argmin } E_{i|x,y,} \forall(x, y) \in \mathcal{L} \quad (7)$$

During every iteration the A matrix is calculated as mentioned and the robots's region L_i matrix can be calculated as:

$$L_i = \{(x, y) \in \mathcal{L}: A(x, y) = i\}, \forall \in \{1, \ldots\ldots, n_r\} \quad (8)$$

The number of the assigned cells per robot can be defined as the cardinally set L_i

$$K_i = | L_i |, \forall_I \in \{1, \ldots\ldots, n_r\} \quad (9)$$

In practical terms, the first condition states that a single cell can only be allocated to a single robot; the second condition states that each cell is assigned to a robot's operation plan; and the fifth condition assumes that the initial robot positions are always assigned to the relevant robot area. The DARP method, to put it briefly, is an iterative procedure that alters the robots' judgments Ei in a coordinated way.

Initially the evaluation matrix E_i only contains the distance information:

$$E_{i|x,y} = d(X_i(t_o),[x,y] \hat{} \tau), \forall_i \in \{1, \ldots\ldots, n_r\} \quad (10)$$

Where (d) denotes the chose Euclidean distance function and thus the initial matrix A is a classical Voronoi Diagram[10]. The Main idea of the DARP algorithm is it can be corrected by a term m_i which is as follows :

$$E_i = m_i E_i \quad (11)$$

Where m_i is the Scalar correction factor for the *i*th robot
The third condition is equivalent to the minimization of

$$J = \frac{1}{2} \sum_{r=1}^{nr} (ki - f)^2 \quad (12)$$

A standard Gradient Descent method for updating m

$$mi = mi - \eta \frac{\partial J}{\partial mi}, \eta > 0, \forall i \in \{1,.....,nr\} \quad (10)$$

can be employed to minimize the cost function, but there are two issues, first $\partial j / \partial m_i$ cannot be calculated algebraically as the analytical form that relates j and m_i is not available and secondly there is no guarantee that the cost function j has only one (global) minimum.

An approach known as cyclic Coordinate Descent (CD) is used to get around the issues [10]. Coordinate descent algorithms work by iteratively executing approximation minimization along coordinate directions or coordinate hyperplanes to solve optimization problems. For every coordinate, the global cost function is minimized cyclically while the remaining coordinates are fixed at their most recent updated values. Since each of these subproblems is a scalar minimization issue, it is usually easier to solve than the original problem.

The update rule of m_i can be computed directly for each objective function individually as :

$$m_i = m_i - \eta \frac{\partial J_i}{\partial mi} \quad (11)$$

$$= m_i - \eta (k_i - f) \frac{\partial k_j}{\partial mi}$$

In summary, it can be determined that although the global cost function f may not always be convex, as it depends on the configuration of the robots and obstacles, it will always have multiple local minima[10]. However, the contribution of each robot, Ji, is convex in relation to the controllable parameter mi. In cases where this property is true, cyclic Coordinate Descent methodologies can reach a global optimal solution set, denoted by $m*$ i.e.:

$$J(m^*) \leq J(m), \forall m \in \text{dom}(J) \quad (12)$$

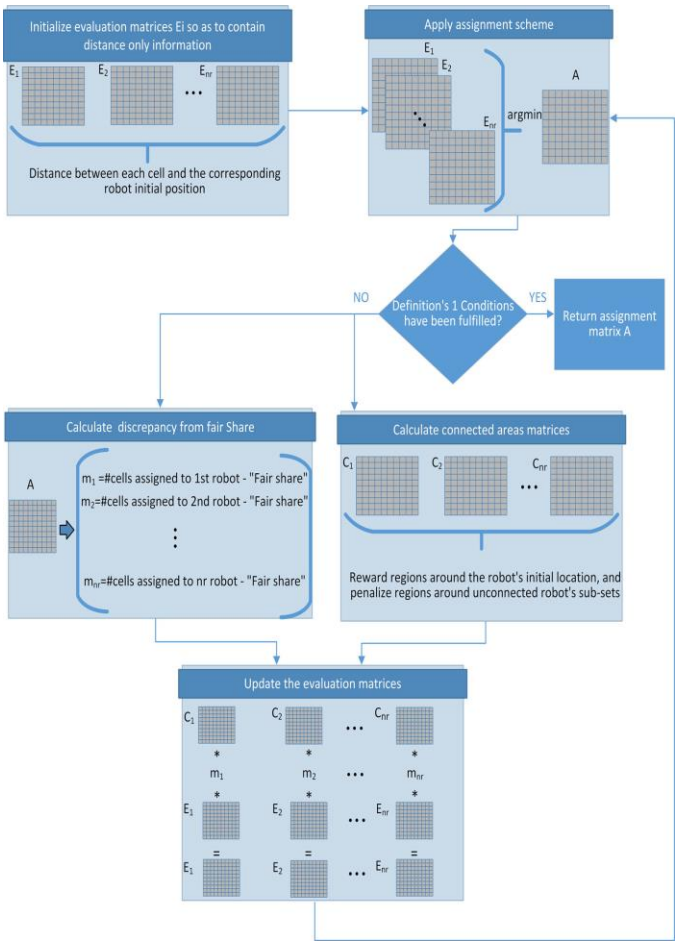

Figure 3: DARP algorithm flowchart - divide areas based on robot's initial positions [10].

In this project the A* algorithm is designed in such a way that the Robot positions are also assumed as obstacles, so the algorithm takes in count the initial robot positions [10] (The n other robots) and computes the shortest distance to the target. If the user specified point (Target) Matches with any of the Robots initial position, then the Target is not accepted. This way the second behavior Robot collision is satisfied .

V. Validation in Simulations

4.1. Simulations and experiments

The simulation setup for our centralized multi-robot system (MRS)[4] involves an intricate process where robots interact and coordinate through a central server. This setup allows for efficient data exchange and communication among robots, pivotal in conducting the experiments. The system's scalability is a key feature, allowing adjustments in grid size and robot count as needed.

Figure 4: Simulation of Initial position of E-puck robots in workspace (Simulated using WEBOTS)

Phase I: Exploration using the DARP Algorithm:

In the initial phase, the Division and Repartition (DARP) algorithm plays a crucial role in distributing the exploration area among the robots. For this experiment, we focus on a scenario with three robots [4], although the setup can accommodate different numbers. The DARP algorithm ensures equitable distribution of the grid cells to each robot.

Each robot, equipped with distance sensors, navigates its assigned area. Upon detecting walls, the robots update the grid data on the server, methodically mapping the environment. This iterative process continues until all cells in the grid reflect accurate environmental data. This meticulous mapping forms the foundation for subsequent exploration and goal navigation tasks.

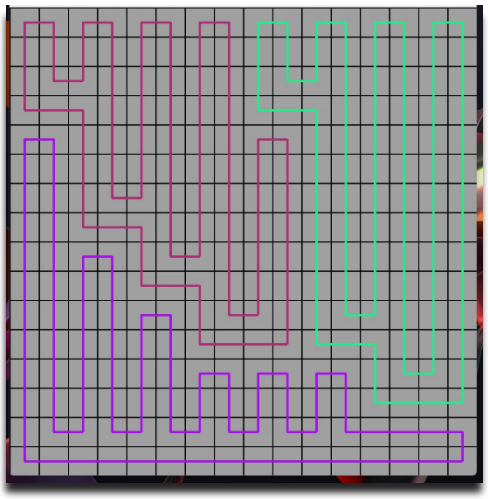

Figure 5: Exploring the map using DARP without walls.

4.2 Warehouse Exploration and Object Detection

Following the initial mapping, the DARP algorithm generates updated paths for more nuanced exploration within the

warehouse environment. These paths are based on the grid data acquired earlier. As robots traverse these paths, they detect and identify objects, updating this information to the server in real-time.

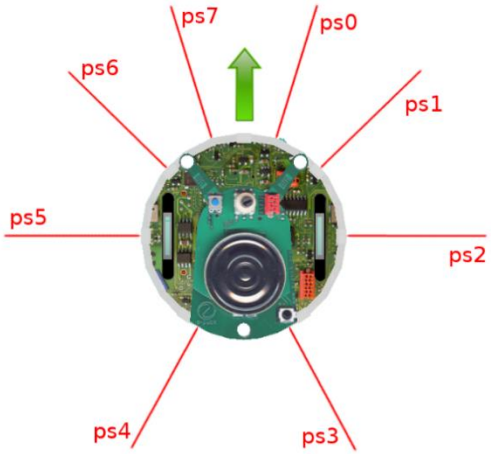

Figure 6: Distance sensor location of a simulated E-puck robot

4.3 Warehouse Navigation using the A* Algorithm

For efficient navigation within the warehouse, the A* algorithm is employed. While the DARP[10] generated paths are suitable for exploration within designated areas, the A* algorithm provides a more computationally efficient solution for navigation, especially when robots are not in their initial positions. This algorithm calculates optimized paths for robots, ensuring effective navigation to target locations.
User Interface and Interaction

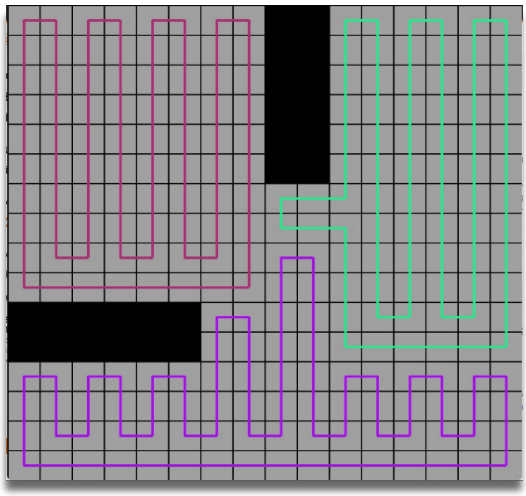

Figure 7: E-puck robot movement after installing walls.

A user interface (UI) served from the central server facilitates human interaction with the system. Users can select desired target locations within the grid via this UI. The system then generates A* paths for all robots, selecting the robot closest to the target for task execution. This robot then follows the calculated trajectory to reach the specified location.

4.4 Server Details and Workflow Integration

The central server, as the linchpin of this Multi-Robot System (MRS), plays a critical role in orchestrating the operations of the robot fleet. It manages data exchange, path computation, and user interaction, ensuring that the robots work in unison, thus optimizing the overall system efficiency and effectiveness. This server-based architecture allows for the centralization of complex computational tasks, such as the real-time processing of environmental data and the generation of navigational paths[10]. The server's capacity to handle these tasks offloads the computational burden from individual robots, enabling them to focus on execution of their assigned tasks with greater agility and precision.

Moreover, this simulation workflow, which leverages the central server's capabilities, demonstrates a robust approach to MRS. It incorporates advanced algorithms like DARP for area division and the A* algorithm for efficient pathfinding, showcasing a sophisticated strategy for exploration and navigation within dynamic environments[4]. This approach is underscored in academic discussions, particularly in the lectures on "Multi-Robot Systems: Challenges and Perspectives" and "Exploration and Mapping with Group of Robots," which highlight the importance of adaptability, coordination, and efficient data handling in modern robotic systems. The central server's role in facilitating these functions exemplifies how to enhance the operational efficiency of multi-robot systems, particularly in complex and changing environments like those encountered in warehouse automation.

S. No	Method and Route	Description
1	GET /update	This route is used to receive and update the collaborative physical mappings (CPMs) with the details (name, radius, speed) of a robot, sent as query parameters.
2	GET /nav	This endpoint responds with details of the robot currently assigned to navigate to a target location
3	GET /retrieve	This route is utilized to retrieve and send back the current state of all CPMs stored in the serve
4	GET /getposition	This route provides the current positions of all robots, including their coordinates and orientation.

5	GET /updateposition	This endpoint updates the positions of the robots based on the query parameters received, reflecting their current locations in the simulation.
6	GET /target	This route is used for target allocation where the server calculates and assigns the robot with the shortest path to a specified target location, received as query parameters.
7	GET /	Serves the main control HTML page from the server's root directory.
8	GET /maze	This endpoint responds with the current state of the maze, including wall positions and other relevant details.

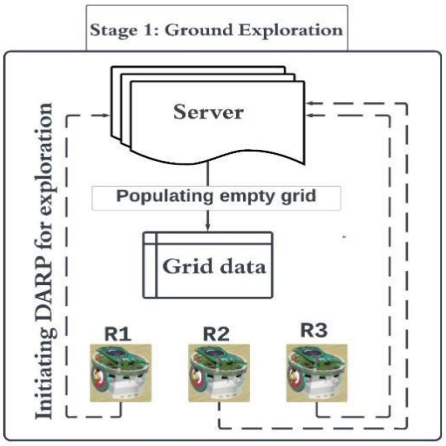

Figure 8: Workflow network during exploration.

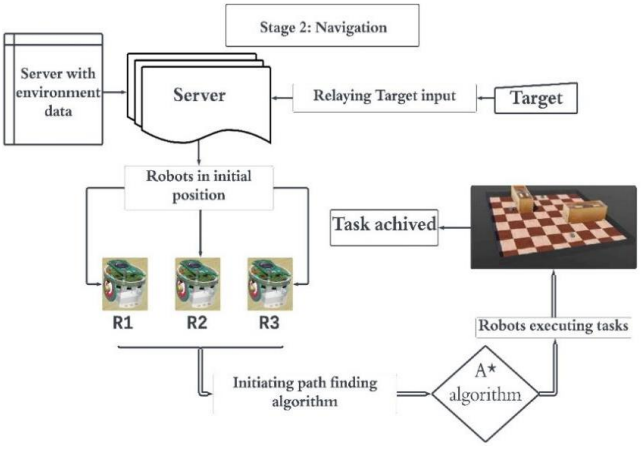

Figure 9: Workflow network of E-puck during Navigation.

VI. FUTURE WORK

Several avenues of exploration are left open for future work in this field, and intend to few improvements to our model. Integrating our project with Simultaneous Localization and Mapping (SLAM) using LIDAR technology, enhancing the practicality and effectiveness. The E-puck robot, currently devoid of a grabbing mechanism, In future will be equipped with a grabber to facilitate pick-and-place tasks, which will be controlled through computer vision. Decentralizing this model could revolutionize opportunities in transportation and logistics, paving the way for innovative applications. Additionally, the establishment of micro-servers is envisaged to manage a higher volume of input requests, allowing for the parallel distribution of tasks among multiple robots, thereby optimizing efficiency and productivity.

VII. CONCLUSION

Our proposed method strategically coordinates a team of multiple robots to thoroughly explore a designated area. At the core of our method is the Decomposition and Allocation based on Relaxation Point (DARP) methodology, an innovative search algorithm. DARP effectively assigns the most suitable cells to each robot, employing a cyclic coordinate descent strategy. This strategy considers both the initial positions of the robots and the layout of any obstacles.

The result of the DARP algorithm is a series of distinct operating zones, each assigned to a different mobile robot. These zones are then relayed to the individual robots' planners. Here, using the A*algorithm, the precise path that each robot will follow to cover its assigned area is determined. Our comprehensive navigation system ensures that the entire operational area is covered efficiently, avoiding revisiting areas already covered, and starting from the robots' exact initial locations. To our knowledge, this approach is unique in literature, combining all these elements simultaneously.

REFERENCES

[1] A. Farinelli, N. Boscolo, and E. Zanotto, "Advanced approaches for multi-robot coordination in logistic scenarios".

[2] G. A. Vargas, O. G. Rubiano, R. A. Castillo, O. F. Avilés, and M. F. Mauledoux, "Simulation of e-puck path planning in Webots," International Journal of Applied Engineering Research, ISSN 0973-4562, Volume 11 Number 19 (2016) pp. 9772-9775.

[3] M. S. Sumbal, "Environment Detection and Path Planning Using the E-puck Robot," Muhammad Saleem Sumbal.

[4] A. Ch. Kapoutsis, S. A. Chatzichristofis, and E. B. Kosmatopoulos, "Department of Electrical and Computer Engineering, Democritus University of Thrace, Xanthi, Greece; Information Technologies Institute CERTH, Thessaloniki, Greece," A. Ch. Kapoutsis et al..

[5] L. Marín, M. Valles, A. Valera, and P. Albertos, "Implementation of a bug algorithm in the e-puck from a hybrid control viewpoint ".

[6] Olivier Idir and Alessandro Renzaglia, "Multi-Robot Weighted Coverage Path Planning: A Solution based on the DARP Algorithm".

[7] M. S. Sumbal, "Environment Detection and Path Planning Using the E-puck Robot," e-ISSN: 2395 -0056, Volume: 03 Issue: 01, Jan-2016, www.irjet.net, p-ISSN: 2395-0072.

[8] Yufan Huang, "A Multi-robot Coverage Path Planning Algorithm Based on Improved DARP Algorithm," April 2023, arXiv:2304.09741v1 [cs.RO].

[9] O. Idir and A. Renzaglia, "Multi-Robot Weighted Coverage Path Planning: A Solution based on the DARP Algorithm".

[10] A. Ch. Kapoutsis, S. A. Chatzichristofis, and E. B. Kosmatopoulos, "DARP: Divide Areas Algorithm for Optimal Multi-Robot Coverage Path Planning".

Contributions:

Ezhilan Veluchami: Webot simulation, DARP algorithm setup in python and Corresponding report documentation

Mohanraj babu: Path finding (using A*algorithm), Constructing the Mathematical model For the Robot and Corresponding report documentation.

Shiva sam Kumar Govindan: Theoretical Analysis, Computing Mathematical model for DARP and Corresponding report documentation.