Decentralized Safety Control for Multi Agent on Flexible Manufacturing System

1st Jakaisa Riskhalifah Bhuwana
Faculty of Electrical Engineering
Telkom University
Bandung, Indonesia
jakaisaa@student.telkomuniversity.ac.id

2nd Muhammad Zakiyullah Romdlony
Faculty of Electrical Engineering
Telkom University
Bandung, Indonesia
zakiyullah@telkomuniversity.ac.id

3rd Angga Rusdinar
Faculty of Electrical Engineering
Telkom University
Bandung, Indonesia
anggarusdinar@telkomuniversity.ac.id

***Abstract* — This project presents a simulation-based approach for coordinating multiple Autonomous Mobile Robots (AMRs) in a flexible manufacturing system (FMS). In this study, we used the Dijkstra path planning algorithm and CLBF to control AMR so that it runs according to the path plan determined. In this scenario, the AMR will be given a mission, namely to pick up goods from the station and deliver the goods to another station safely and without crashing into other AMRs. In this scenario, there will also be several obstacles that were not previously registered, as well as narrow passages. Then a good communication and coordination system between AMRs will also be designed so that AMRs will not experience deadlocks and live locks. In its application, the research in this paper uses ROS. This experiment will use 2 AMRs with mechanum wheels, encoder sensors, and LiDAR sensors.**

Keywords: FMS; Multi AMR; CLBF; ROS

I. INTRODUCTION

In this rapidly developing era, market needs are also changing dramatically, causing conventional manufacturing systems not able to find a balance point between demand and production[1][2]. Which affects manufacturing profit margins. How to speed up the manufacturing cycle and increase the utilization rate of production resources is the key to the development of the intelligent manufacturing industry[3]. Flexible manufacturing systems offer features that can be a solution to these problems because FMS provides flexibility features that can change according to situations quickly[4]. The manufacturing paradigm has expanded towards flexibility to face various production demands[5]. Because the system will be made more flexible, the system will also be more complex than a conventional manufacturing system[6]. Autonomous mobile robots can increase material handling flexibility and enable distributed control processing in flexible manufacturing systems[7].

AMR is a mobile robot that can move independently or autonomously indoor or outdoor[8][9]. Autonomous mobile robot (AMR) systems, are an important aspect of low-level to middle-level manufacturing, including in flexible manufacturing systems, and the service industry, where they deliver goods such as mail, laundry, and food to hospitals[10]. Flexible production networks based on autonomous mobile robots (AMRs) have the potential to increase industrial flexibility and productivity[11]. AMRs are renowned for their ability to work continuously, safely, and efficiently, delivering various types of loads without human intervention, so multi-AMR systems have become an important part of the automation of manufacturing facilities and warehousing systems [12] [13].

AMR may experience deadlock and live lock behaviours, which are when they become trapped in local equilibria or periodic orbits inside a multi-robot system [14]. Deadlock occurs when two or more AMR s are trapped in a process that is not running because the processes are waiting for each other, which causes the system to stop working [15][16][17]. Meanwhile, the live lock is when AMR continue to change positions or change their actions to avoid collisions or conflicts but remain stuck in movement patterns that will not reach their destination[18]. One important aspect of deadlock resolution is implementing a communication system between AMR so that the AMR can coordinate[19].

The localization, navigation, and control systems, as well as the communication between AMRs, will all be created in the ROS environment during the various stages of AMR design included in this project. ROS enabling easier data analysis[20]. Adaptive Monte Carlo Localization (AMCL) will be used since LiDAR sensors and encoders are being used for localization in this AMR. In the framework of Robot Operating System (ROS), Adaptive Monte Carlo Localization (AMCL) accurately locates mobile robot in unknown territories, with the accuracy depending on the number of particles and the surrounding environment [21][22]. For navigation in this research we will use the Dijkstra method, by choosing the unvisited vertex with the smallest distance and updating its neighbours with shorter distances, Dijkstra's method determines the shortest path[23][24]. And for the AMR control system it will use the Control Lyapunov Barrier Function (CLBF), The CLBF approach substantially enhances the stability and safety of mobile robot systems by avoiding obstacles and obtaining near-perfect alignment with the origin[25].

II. LITERATURE OVERVIEW

In this chapter we will discuss Path plan dan control avoiding, and Communication protocol.

A. Path Plan and Control Avoidance

Path planning is the process by which a robot looks for the best path from one site to another while avoiding items or obstacles along the way[26][27]. Path planning is crucial for AMR because the robot must be able to adapt to changes in the environment and choose the optimal route to achieve its target. Path planning and obstacle avoidance are the

foundations of autonomous mobile robotics, allowing robots to achieve their destination while avoiding obstacles and traveling along optimum paths based on distance, time, or energy[28]. In this sub-chapter we will discuss path plans, using the Dijkstra method. And obstacle avoidance uses the Lyapunov barrier function control method.

a. Control Lyapunov Barrier Function

Control Lyapunov Barrier Function is a collision-free controller that can be created by combining control Lyapunov functions with control barrier functions, therefore avoiding unwanted equilibria at the safe set's border[29]. Control Lya-punov Barrier Function (CLBF) is a control equation that combines and inte-grates CLF and CBF, allowing the utility of the two control functions to be achieved with a simpler control equation[30].

CLF will control AMR from any location to the destination point. This is be-cause CLF has a positive definite equation, which implies it has a negative value for the root value, causing the state value to decrease with time. CBF will manage AMR to avoid an unsafe state and ensure safety. Following the parameter in ref-erence [30], the CLBF pseudocode is constructed in algorithm 1 with AMR target coordinate at (0, 0) and the radius of the obstacle set to 0,6 m, and adding 0,2 m for margin safety and the result is 0,8 m.

CLBF
define x_1 and x_2 as AMR x-axis coordinates, and y-axis coordinates
define x_{1D} and x_{2D} coordinates of the obstacle's center point
define r as distance between AMR and obstacle's center point
while x_1 != 0 and x_2 != 0 **do**
    $L_fV = 0$
    $L_gV = [\,2x_1 + x_2 \quad 2x_2 + x_1]$
    $L_fB = 0$
    if r < 0.8 **do**
        $L_gB = [\,2x_1 - 2x_{1D} \quad 2x_2 - 2x_{2D}]$
    else do
        $L_gB = 0$
    end if
    $L_fW = L_fV + \lambda * L_fB$
    $L_gW = L_gV + \lambda * L_gB$
    b = transpose(L_gW)
    a = L_fW
    $\mathbf{u} = -\frac{\mathbf{a}+\sqrt{\mathbf{a}^2+\gamma\,\|\mathbf{b}\|^4}}{\mathbf{b}^T\mathbf{b}}\mathbf{b}$
end while

Algorithm 1. CLBF pseudocode

CLBF is used for controlling the AMR. Dijkstra's output is simply the path plan, which contains coordinates that will be followed by CLBF. When an obsta-cle on the AMR's path is not recognized on the map server, the CLBF control system will automatically avoid it while still guiding the AMR to its destination. Once the LiDAR no longer detects an unregistered obstacle, the AMR will resume to its original path.

AMR is equipped with LiDAR, which can identify obstacles that are not regis-tered on the map server. LiDAR can determine the distance and angle of an un-registered obstacle to the AMR at any angle. The angle and distance of the obsta-cle relative to the AMR will be calculated using trigonometry to determine its position relative to the AMR. Next, the obstacle position data that is still related to the AMR is combined with the current AMR location to generate global obsta-cle position data. The global obstacle position's x and y axes will be used as input for CLBF.

a. Dijkstra Path Plan

Path planning using Dijkstra on AMR will plot the map provided to AMR by generating a very tiny grid that represents the work environment surrounding AMR. This grid will eventually be utilized as a node by Dijkstra and assigned a weight depending on the accessibility of the environment, which will be determined using the distance necessary to travel through the grid and obstacles on the map. This method will select the path that AMR will travel by calculating the cost of the grid that will be traversed, such that the path with the lowest cost is picked, as well as the shortest path in terms of distance, making the path the most effective[27]. The Dijkstra method in route planning finds an optimum path in an area by utilizing the grid-map structure to construct an accurate approximation of the best path[28].

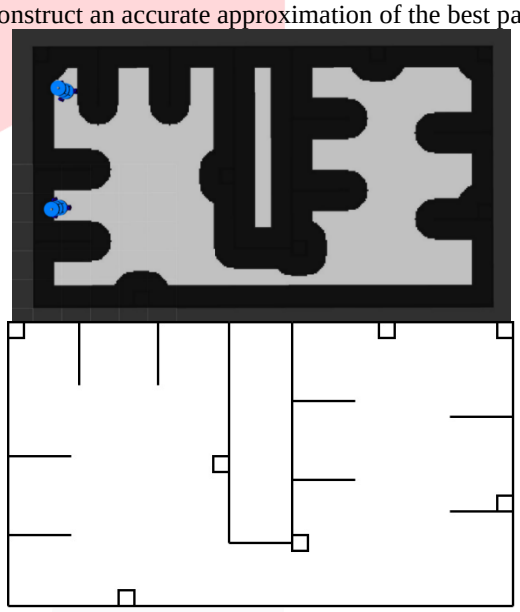


Figure 1. Costmap

The costmap picture in Figure 3 will serve as a map for the Djikstra method. If you look at the server map in figure 3, the partitions and walls are not that thick, but this is critical because if the safe boundary with the wall or partition is too close, and Dijkstra takes the closest path without estimating the area of the AMR itself, because in this research the Dijkstra's method only considers a point, the AMR will crash because the specified path is too close to a wall or partition. To avoid collisions, this safe limit must be bigger than the AMR radius.

B. Comunication Prtotocol

In this research, deadlocks and live locks can be resolved by creating a communication protocol for AMR, so that AMR can coordinate with other AMRs so that deadlocks and live locks caused by disputes over resources will not occur. This communication protocol is meant if there is a conflict between AMRs. When AMRs compete for an area, or in the instance of this research, they pass through a resource or corridor and stations that is only accessible by one AMR. This communication protocol based on priority[29][30] and firstly get in to the resource[31]. The pseudocode below explains the AMR communication protocol.

Conflict Resolution

```
define s as distance between AMRs
define c as distance between path plan
define j as resource being occupied
define m_1 and m_... as priority of each robot
if s < 3 do
    if c < 1 do
        if j = 1 do
            command the robot to wait
        else if m_1 < m_x do
            command the robot to yield
        end if
    end if
end if
```

Algorithm 2. Conflict resolution algorithm

First, AMR will determine whether other AMRs enter the private zone. The private zone is defined as a three-meter radius around the AMR. This private zone serves as a signal for accessing the communication protocol. If one AMR detects that another AMR has entered the private zone, each AMR will check to see if their trajectories will intersect. whether a collision occurs, AMR will figure out whether the resource, or in this example the corridor and station, is it occupied or whether another AMR is nearby. If the resource is full, AMR will enter the "wait" protocol, which will stop AMR in its tracks and wait until there is a "clear" order from AMR who was using the resource.

Next, AMR will check the priority for each AMR. If the AMR has a lower priority, it will enter "gave in" protocol. This protocol puts an artificial barrier on the resource such that AMR can move around it but can't get past through, allowing the AMR with a higher priority to enter without being blocked by the AMR currently in front of the resource. Because the resource entrance is narrow, this is important to prevent deadlocks. If the AMR with high priority has been finished and is at a safe distance, then the AMR will transmit a "clear" signal, removing the artificial barrier at the resource and the AMR with low priority will continue its mission normally.

Next, AMR will check If the AMR with the highest priority, or the AMR that enters the resource that both AMRs seek first, has done using its resource and is at a safe distance, it will send a "clear" message to the other AMR, indicating that the resource is accessible, empty, and safe to use. Other AMRs will subscribe to other AMR coordinates or /amcl_pose topics to obtain the coordinate location for every AMR. The coordinates of the other AMRs will be examined so that they can determine which resources are being utilized by each AMR, as well as so that an AMR can detect when another AMR entered its private zone. Apart from subscribing to coordinate topics, AMR will also publish and subscribe to priority or /prior topics so that AMR can know the priorities of other AMRs so that they can determine whether the AMR has low priority or high priority. Fellow AMRs will then subscribe to the route plan or /Globalplanner/plan topics. This topic covers the information of the path plans for each robot. Information about other AMR travel plans is required so that each AMR knows the destination and route that the other AMR is taking. The information contained in the path plan is, of course just the coordinates that the AMR will go through, so it must be processed so that other AMRs can determine which resource the AMR will visit and which will be used.

III. METHOD

This section discusses the design of the FMS application system with multi-AMRs in the FMS system.

A. Desigin System

This study will create an FMS system that will use two AMRs to be handed some missions, which include the pickup goods in station and deliver it into its destination point. Then, this AMR will be equipped with LiDAR for localization and navigation in order to detect obstacles that are not registered on the map server. The map server will be sent to the AMR, allowing it to navigate without having to examine the region. Priority negotiations require an inter-AMR communication mechanism to avoid deadlocks or live-locks. In addition, the Control Lyapunov Barrier Function (CLBF) approach is used to avoid barriers that have not yet been recorded on the map server while moving towards the objective. The purpose of this research is to create a multi-AMR with a decentralized control system that can connect with one another and deliver items in an FMS environment while avoiding dead locks and live locks, as well as obstacles that are not registered on the server map.

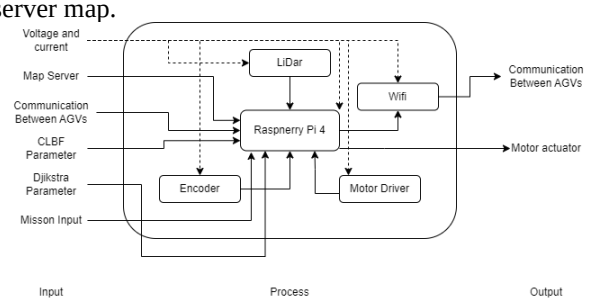


Figure 2. Main Idea

The author provides a solution idea design, as seen in the image above. To get started, the AMR will be loaded with CLBF parameters and a map on the map server. When the power is switched on, the mission is randomly assigned to the AMR, which then travels to the pick-up location using the CLBF and Dijkstra algorithms to determine the shortest route. LiDAR and the AMR's encoder are linked to a microprocessor, which processes localization and navigation using maps from the map server. LiDAR will also scan the surroundings for obstacles that are not recorded on the map server. When LiDAR identifies an obstacle that is not registered on the server map, it uses CLBF to avoid it and continue on to the destination. To minimize deadlocks and livelocks, the communication input to the AMR is considered, as well as whether another AMR or this AMR comes first. After reaching its destination, the AMR will be assigned a new task and will continue to repeat itself until the instruction to halt is issued.

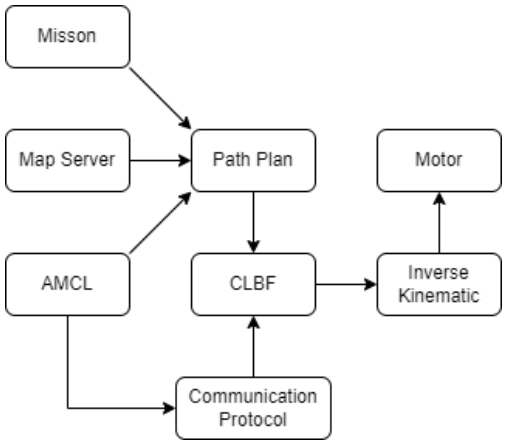


Figure 3. Concept

Here's an overview of how the AMR will move and interact with the environment. The first mission will be the input for the Dijkstra parameters. Then, using server maps and localization, Dijkstra will decide the overall path, such as which corridors and terminals to take. Next, Dijkstra's output is in the form of coordinates, which are used as input for CLBF. CLBF will then follow the path plan that has been given; however, if the path plan provided by Dijkstra runs across an obstacle that is not on the server map, CLBF will avoid it and continue along the planned course. Of course, the LiDAR sensor will identify any dangerous conditions that are not on the server map. The CLBF control system outputs the speed of each axis. So inverse kinematics is required; this stage converts data from the speed of each axis to the speed of each motor on the AMR. The location provided by AMCL will then be processed so that AMR can coordinate effectively and avoid live-locks and deadlocks. The procedure will be repeated till the user terminates it.

B. ROS Desigin

ROS facilitates the development of ideas on robotics subjects by allowing researchers to run simulations [32]. The ROS system for this research will be separated into numerous sections. The first portion will go over localization and LiDAR. The second section covers path planning, move_base, and CLBF. Here's an overview diagram.

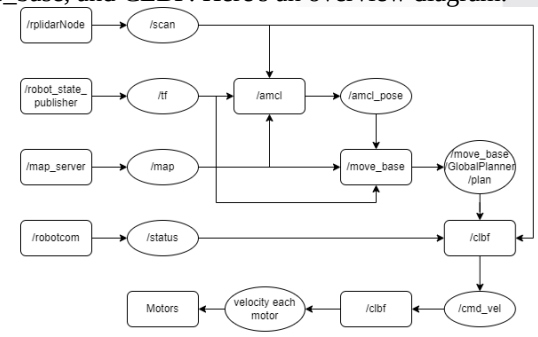


Figure 4. ROS Concept

In the Ros diagram above there are several processes. the square-shaped ones are nodes and the oval-shaped ones are topics. The first process is at the /rplidarnode node. This node contains coding which will take and process the rplidar into scan data and is called topic /scan, topic /scan contains information about each distance and angle from the lidar to objects around it. Then there is the /robot_state_publisher node, this node contains coding about the transfer function of the amr itself, and issues a /tf topic whose contents are the transfer function. After that there is the /map_server node, this node contains coding information about the map that will be used, and a published topic with the name /map which contains information about the map that will be used. Then there is the /robotcom node, this node contains the communication protocol coding between AMR, and the publish topic /status, which contains the contents of the communication protocol.

Then there is the amcl node, this node contains coding for AMR localization, this node requires information from the topics /map, /scan, and /tf, the output from this node is information about the AMR coordinates on the map that has been given which is called the topic /amcl_pose. The next node /move_base, is a node that contains the code to produce a path plan using the Dijkstra method, this node requires information from the topics /amcl_pose, /tf, and /map, which will publish the topic /move_base/GlobalPlanner/plan which contains the coordinates of the resulting path plans. Then the /clbf node is the node that will process information from the topics /move_base/GlobalPlanner/plan, /status, /scan, and /amcl_pose, into speed on the x-axis and on the y-axis AMR, with the topic /cmd_vel. /cmd_vel will enter inverse kinematics which is outside ROS, which will produce speed information for each motor.

IV. RESULT

The simulation in this research uses Gazebo and Rviz as data visualization tools. In the process, before the direct experiment, a simulation will be tried first.

A. CLBF Simulation

The first experiment is designed to demonstrate the safety aspect of the Control Lyapunov Barrier Function (CLBF) as a control method for Autonomous Mobile Robots (AMRs). The experiment is conducted in Figure 5 where the robot navigates from its initial coordinate (-4, -7) or in station 1, to its target coordinate (0, 0) or station 5 in the presence of circle unsafe state centered at (-1, -4), detailed information about the specific coordinates is illustrated in Figure 6. The author attempts to simulate CLBF with results and parameters as follows, the equations used in this experiment are adapted from [30].

$$V(x) = x_1{}^2 + x_1x_2 + x_2{}^2$$
$$\mathrm{LgV} = [2x_1 + x_2 \quad x_1 + 2x_2] \qquad (1)$$

Where represent the coordinates of the robot along the x-axis and y-axis, respectively. The unsafe region is defined as a circular area with a radius of meters around the point (-$x_{1\mathcal{D}}$, -$x_{2\mathcal{D}}$), and $\mathcal{D}$ is a subset of $\mathcal{X}$.

$$\mathcal{D} = \{x \in \mathcal{X} \mid (x_1 - x_{1\mathcal{D}})^2 + (x_2 - x_{2\mathcal{D}})^2 < r^2\} \qquad (2)$$

Where r is the radius of the obstacle, $x_{1\mathcal{D}}$ and $x_{2\mathcal{D}}$ is coordinates of the obstacle's center point. for the application procedure see Algorithm 1 and the equation model in Equation (36) in [30]. for more clarity there is a similar procedure in [30] in sub section 6.2.Following are the simulation results.

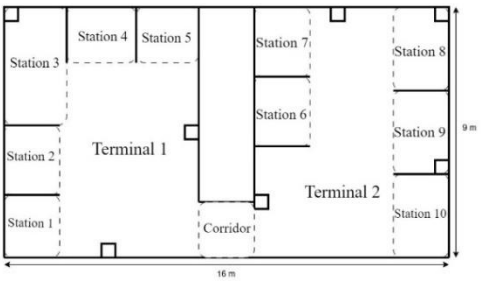


Figure 5. Map of the Environment

Following Figure 6 is the first experiment simulation results demonstrating CLBF can guarantee the safety of the AMR and keep it on its way to its destination.

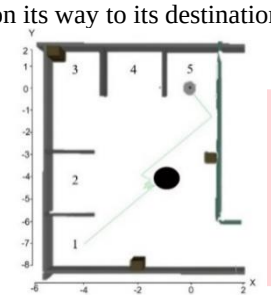


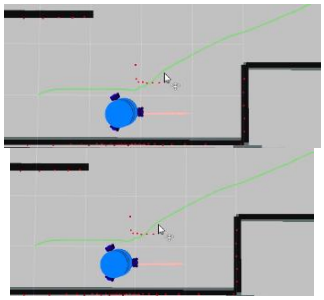
Figure 6. CLBF simulation

The simulation in Figure 6 shows that AMR departs from station 1 to station 5. But it encounters an unregistered obstacle, represented as a black circle in the middle of the workspace. In addition, the AMR is represented as a grey circle with a dot at its center. The edges of the workspace are equipped with registered obstacles, which are visualized by square shapes near the walls. The trajectory of the AMR can be clearly seen from the green line, which shows that it successfully avoids the unregistered obstacle and reaches station 5 without any problem.

B. Application of CLBF and Dijkstra Path Plan

Below, in this research simulates a combination of CLBF and Dijkstra path plans so that the robot can reach its goal. This simulation, determines the AMR's destination, that is the predetermined stations, and the AMR's destination to which station is determined randomly. Below, it's simulate a combination of CLBF and Dijkstra path plans that the robot can reach its goal. In this simulation, the author determines the AMR's destination, namely predetermined stations, and the AMR's destination to which station is determined randomly.

It can be seen below AMR encountered an obstacle that is not listed on the server map. There are several red dots visible in front of the AMR, this is laser scan data which detects obstacles even though they are not registered on the server map. And here are AMR's movements when trying to avoid this obstacle.



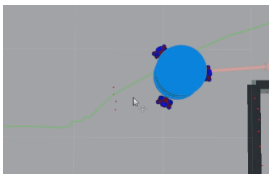


Figure 7. AMR interaction against obstacles that are not registered on the server map

It can be seen from above that AMR can avoid these obstacles safely and AMR continues to its destination station.

C. Simulation Result

The simulation used in this research uses 2 AMRs which are given missions in the form of picking up goods and delivering them randomly, which will be given a time of 3, 5 and 8 minutes. With the second AMR having the highest priority. This will be carried out on a smooth and flat area with a width and length of 12m x 5m, which has several features, namely a corridor with a width of 2m, then there are 5 stations which will be separated by partitions. Like the following picture.

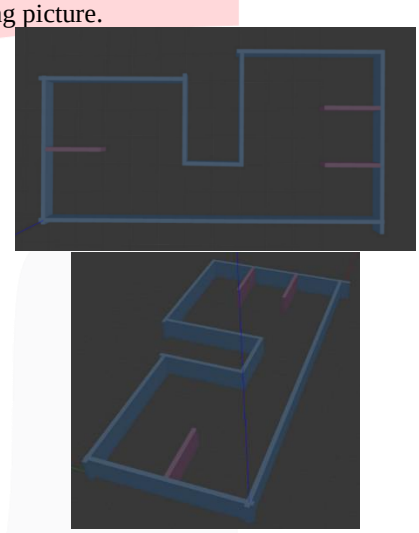


Figure 9. map image of 12m x 6m terrain

The following is data on the success rates of two AMRs in completing their missions.

TABLE 1. Simulation Result with two AMRs in 12x6 workspace

	3 minutes		5 minutes		8 minutes	
	AMR1	AMR2	AMR1	AMR2	AMR1	AMR2
First Experiment	0 Mission	1 Mission	1 Mission	2 Missions	2 Missions	4 Missions
	1 Missions		3 Missions		6 Missions	
Second Experiment	1 Mission	1 Mission	1 Mission	2 Missions	2 Missions	3 Missions
	2 Missions		3 Missions		5 Mission	
Third Experiment	1 Missions	2 Mission	1 Mission	3 Missions	3 Missions	4 Missions
	2 Missions		3 Missions		7 Missions	
Forth Experiment	1 Mission	1 Mission	2 Missions	2 Missions	3 Missions	4 Missions
	2 Missions		4 Missions		7 Missions	
Average	0,7 Mission	1,2 Missions	1,2 Mission	2,2 Missions	2,5 Missions	3,7 Missions
	1,75 Missions		3,25 Missions		6,25 Missions	

And here is the success rate data from one AMR in the same work space

TABLE 2. Simulation Result with one AMR in 12x6 workspace

	3 minutes	5 minutes	8 minutes
First Experiment	9 missions	16 missions	24 missions
Second Experiment	7 missions	12 missions	19 missions
Third Experiment	9 missions	14 missions	23 missions
Average Missions	8.3 missions	14 missions	22 missions

Next, there was an experiment with a different work space, namely with an area of 16m x 9m, equipped with a corridor with a width of 2m and also having 10 stations as an additional feature. The following is a picture of the second work space.

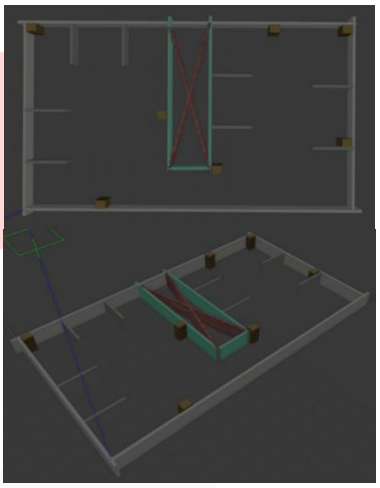


Figure 10. map image of 16m x 9m terrain

The following is data on the success rates of two AMRs in completing their missions.

	3 minutes		5 minutes		8 minutes	
	AMR1	AMR2	AMR1	AMR2	AMR1	AMR2
First Experiment	1 Mission	1 Mission	1 Mission	2 Missions	2 Missions	3 Missions
	2 Mission		3 Missions		5 Missions	
Second Experiment	0 Mission	1 Mission	1 Mission	1 Mission	2 Mission	2 Missions
	1 Mission		2 Mission		4 Mission	
Third Experiment	1 Mission	1 Mission	2 Missions	1 Mission	3 Missions	2 Missions
	2 Missions		3 Missions		5 Missions	
Forth Experiment	0 Mission	2 Mission	1 Mission	3 Missions	1 Mission	4 Missions
	2 Missions		4 Missions		5 Missions	
Average	0,5 Mission	1,2 Mission	1 Mission	1,7 Missions	1,7 Missions	2,7 Missions
	1,75 Missions		3 Missions		4,75 Missions	

And here is the success mission rate data from one AMR in the same work space

	3 minutes	5 minutes	8 minutes
First	2 missions	3 missions	4 missions
Second	0 mission	1 mission	3 missions
Third	1 mission	2 missions	3 missions
Forth	1 mission	2 missions	3 missions
Average	1 mission	2 missions	3,25 missions

Based on the data above, AMR2 on average completes the most missions compared to AMR1. This is because AMR2 has a higher priority than AMR1, so AMR1 gives in more and prioritizes AMR2 first. It can be seen from the data above that adding twice as many robots does not mean increasing the double number of missions completed. There are several factors that influence the number of missions completed, there are random missions, because the missions given in this experiment are given randomly, so each trial has a different and random path length, thus affecting the AMR travel time.

The next factor is communication between AMRs. One AMR working in one space does not need to interfere to another AMR because there are no other AMRs, so the AMR does not experience interference and its path will be faster. When more than one AMR is working in the work space, a communication system is needed to avoid deadlocks, live locks and collisions with other AMRs. The communication

system in this research will control AMR queuing so that it is more orderly and avoids deadlocks, live locks, and collisions with other AMRs.

We can observe the difference in data based on the work space area, particularly on a 12 x 6 work space, the difference between the usage of 2 AMR and 1 AMR is at the 3rd minute 0.25, at the 5th minute 0.75, at the 8th minute 1.5, and at the 16 x 9 at the 3rd minute 0.75, 5th minute 1, 8th minute 1.75.

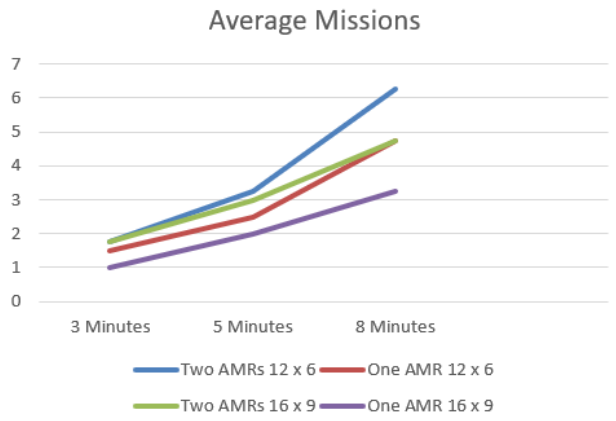


Figure 11. Average mission graph

It can be seen from the graph and the difference above that the difference between the usage of 2 AMR and 1 AMR in the 16 x 9 work space is greater than the difference between the usage of 2 AMR and 1 AMR in the 12 x 6 work space. This occurs because the wider area makes 2 AMRs less likely to conflict with other AMRs and with many stations it also reduces the possibility of AMR fighting over resources so that it is considered more efficient compared to 12 x 6 filed. Because conflict between AMRs will reduce the efficiency of the AMR, if more and more AMRs are added but the work space area does not increase, then conflicts between AMRs will occur more frequently so that the FMS system becomes increasingly inefficient. These two experiments show that the same number of AMRs but with different filed areas can influence the number of missions and the difference in missions from the addition of AMRs.

In the following experiment, we'll add obstacles that are not registered on the map server to the 16x9 work space. Four additional obstacles have been added, which are located in the terminal area of the work space and not in the resource area. This obstacle can be put anywhere as long as it does not cover the resource area or its safety margin, allowing the Dijkstra to locate a safe entrance while the CLBF avoids the obstacle.

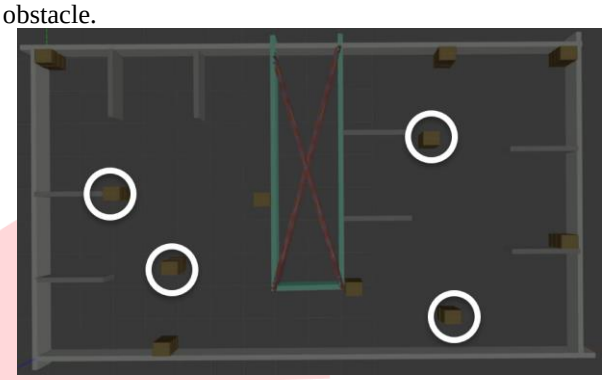


Figure 12. map image of 16m x 9m terrain with unregistered obstacle

Figure 12. map image of 16m x 9m terrain with unregistered obstacle

In the image above, the white circle is an obstacle that is not registered on the server map. When AMR runs and encounters the obstacle while the coordinates provided by the Dijkstra algorithm are within the obstacle, CLBF will prevent AMR from crashing into the obstacle. And with the help of LiDAR, AMR will register the obstacle in the map server, so that AMR will call the Dijkstra route again which has considered the obstacle and a new route will appear where AMR will avoid the obstacle safely. And here is the data from the experiment above.

TABLE 5. Simulation Result with one AMRs in 16x9 workspace with unregistered obstacle

	3 minutes		5 minutes		8 minutes	
	AMR1	AMR2	AMR1	AMR2	AMR1	AMR2
First Experiment	2 Missions	4 Missions	4 Missions	6 Missions	6 Missions	9 Missions
	6 Missions		10 Missions		15 Missions	
Second Experiment	5 Missions	4 Missions	9 Missions	6 Missions	16 Missions	12 Missions
	9 Missions		15 Missions		28 Mission	
Third Experiment	2 Missions	3 Missions	4 Missions	6 Missions	6 Missions	10 Missions
	5 Missions		10 Missions		16 Missions	
Average Missions	3 Missions	3,6 Missions	5,6 Missions	6 Missions	9,3 Missions	10,3 Missions
	6,6 Missions		11,6 Missions		19,6 Missions	

It can be seen from the data above that the comparison of the presence and absence of obstacles that are not listed in the work space greatly influences the success rate of the mission carried out by AMR. The following is a comparison table of 2 AMRs carried out in a 16 x 9m area without any obstacles that are not registered on the server map with work space that have obstacles that are not registered on the server map.

TABLE 6. result difference between works space with and without obstacle

	3 minutes	5 minutes	8 minutes
Average without unregistered obstacle	9 Missions	15,3 Missions	24,3 Missions
Average with unregistered obstacle	6,6 Missions	11,6 Missions	19,6 Mission
Reduction percentage	26,6%	24,1%	19,3%

The lower success missions rate for AMR in carrying out its mission is due to extra obstacles that are not registered on the server map. This is what makes AMR less flexible than before, forcing it to go a greater distance to escape the obstacle at hand. Aside from that, AMR requires time to identify a new path, which will delay AMR's arrival at the target station. If you look at the table above, you can see that the percentage decrease in mission success rates achieved by AMR is reducing with time. This is because when AMR is first run, it encounters multiple new obstacles and registers these obstacles. As time goes by, all obstacles that were not previously registered will be registered on the server map, so Dijkstra will create a path with more complete considerations.

D. *Implementation Result*

In this test, the AMR will be given a random mission, and the AMR will operate for 3, 5, and 8 minutes. From the results of this experiment, will be served how many missions have been accomplished by the multi-AMR. From the experiment, we can also see the distance travelled by the AMR. The test will be carried out using one AMR and placed in a closed room with an area of 4m x 2 m, which contains several stopping stations. The base for implementing this experiment is tiles. The following is a picture of the implementation area.

Figure 13. Map Implementation

In the picture, this is a map that is used as a mapserver which will later be used for localization. The following is an overview of the stations that have been determined. The following are the results of an experimental implementation of one AMR with an area of 4m x 12m.

TABLE 7. Implementation result

	3 minutes	5 minutes	8 minutes
First Experiment	3 missions	5 missions	9 missions
Second Experiment	2 missions	4 missions	6 missions
Third Experiment	2 missions	4 missions	7 missions
Average Missions	2,3 missions	4,3 missions	7,3 missions

In this table above are the data from the implementation results which can be seen even though the distance traveled is closer than the simulation but the mission completed is much lower. This is because the AMR speed in the simulation is faster, and there are other factors such as tire slip and the error percentage during localization which also greatly influences the AMR travel time.

IV. CONCLUSIONS

This research succeeded in designing a multi-AMR with a decentralized control system that can communicate with each other so that it can deliver goods safely in the FMS environment. In this research, we apply CLBF to control and avoid obstacle AMR, and then Dijkstra finds an optimal route plan. The communication protocol system used in this research is also based on priority and who uses it first. This research is tested using a scenario of delivering goods from one station to another, given two work spaces with dimensions of 16m x 9m and 12m x 6m then, AMR will be given a mission to pick up and deliver goods, in this research the mission will be given randomly in the simulation Rviz and Gazebo. With the average results on the 16m x 9m work space at the 3rd minute it is 9 missions, at the 5th minute it is 15.3 missions, at the 8th minute it is 24.3 missions. Then in the 12m x 6m work space, at the 3rd minute it is 5.3 missions, at the 5th minute it is 9 missions, at the 8th minute it is 14.6 missions. This research was additionally tested by introducing multiple unregistered obstacles on the server map with a work space of 16m x 9m and using Rviz and Gazebo simulations. According to the average test results provided by multiple obstacle that are not registered on the server map, the third minute is 6.6 missions, the fifth minute is 11.6 missions, and the eighth minute is 19.6 missions. This research also tested the experiment by implementing it on AMR with mechaunm wheels, in a 4m x 2m workspace. With the average results, namely in the 3rd minute it is 2.3 missions, in the 5th minute it is 4.3 missions, and in the 8th minute it is 7.3 missions.

[1] C. Brecher, “Lecture Notes in Production Engineering Advances in Production Technology,” 2015. [Online]. Available: http://www.springer.com/series/10642

[2] Y. Yin, K. E. Stecke, and D. Li, “The evolution of production systems from Industry 2.0 through Industry 4.0,” *Int J Prod Res*, vol. 56, no. 1–2, pp. 848–861, Jan. 2018, doi: 10.1080/00207543.2017.1403664.

[3] Q. Liu, N. Wang, J. Li, T. Ma, F. Li, and Z. Gao, “Research on Flexible Job Shop Scheduling Optimization Based on Segmented AGV,” *CMES - Computer Modeling in Engineering and Sciences*, vol. 134, no. 3, pp. 2073–2091, 2023, doi: 10.32604/cmes.2022.021433.

[4] P. Udhayakumar and S. Kumanan, “Task scheduling of AGV in FMS using non-traditional optimization techniques,” *International Journal of Simulation Modelling*, vol. 9, no. 1, pp. 28–39, Mar. 2010, doi: 10.2507/IJSIMM09(1)3.139.

[5] M. Azangoo, A. Taherkordi, J. O. Blech, and V. Vyatkin, “Digital Twin-Assisted Controlling of AGVs in Flexible Manufacturing Environments,” in *IEEE International Symposium on Industrial Electronics*, Institute of Electrical and Electronics Engineers Inc., Jun. 2021. doi: 10.1109/ISIE45552.2021.9576361.

[6] A. Kusiak, “Flexible manufacturing systems a structural approach,” *Int J Prod Res*, vol. 23, no. 6, pp. 1057–1073, 1985, doi: 10.1080/00207548508904765.

[7] I. J. Milberg and D.-I. P. Lutz, “Integration of Autonomous Mobile Robots into the Industrial Production Environment by,” 1987.

[8] C. Lin, P. Hsieh, S. H. Wang, and H. H. Chiang, “Automatic Data Acquisition of Indoor Environment Using the Autonomous Mobile Robot,” Taiwan: 2021 IEEE International Conference on Consumer Electronics-Taiwan (ICCE-TW), Sep. 2021, pp. 1–2.

[9] C. Morris and V. Chauhan, “Design of a Low-Cost Autonomous Mobile Robot for Outdoor Applications,” *ASME International Mechanical Engineering Congress and Exposition*, vol. 86670, 2022.

[10] T. Ganesharajah, N. G. Hall, and C. Sriskandarajah, “Design and operational issues in AGV-served manufacturing systems.”

[11] G. Fragapane, D. Ivanov, M. Peron, F. Sgarbossa, and J. O. Strandhagen, “Increasing flexibility and productivity in Industry 4.0 production networks with autonomous mobile robots and smart intralogistics,” *Ann Oper Res*, vol. 308, no. 1–2, pp. 125–143, Jan. 2022, doi: 10.1007/s10479-020-03526-7.

[12] I. Draganjac, T. Petrović, D. Miklić, Z. Kovačić, and J. Oršulić, “Highly-scalable traffic management of autonomous industrial transportation systems,” *Robot Comput Integr Manuf*, vol. 63, Jun. 2020, doi: 10.1016/j.rcim.2019.101915.

[13] M. Čech *et al.*, “Autonomous mobile robot technology for supplying assembly lines in the automotive industry,” *Acta Logistica*, vol. 7, no. 2, pp. 103–109, Jun. 2020, doi: 10.22306/al.v7i2.164.

[14] B. Weng, H. Chen, and W. Zhang, “On the Convergence of Multi-robot Constrained Navigation: A Parametric Control Lyapunov Function Approach,” *2022 International Conference on Robotics and Automation (ICRA)*, pp. 4972–4978, 2022.

[15] N. Du and H. Hu, “A Robust Prevention Method for Automated Manufacturing Systems with Unreliable Resources Using Petri Nets,” *IEEE Access*, vol. 6, pp. 78598–78608, 2018, doi: 10.1109/ACCESS.2018.2885116.

[16] N. Du and H. Hu, “Robust Deadlock Detection and Control of Automated Manufacturing Systems with Multiple Unreliable Resources Using Petri Nets,” *IEEE Transactions on Automation Science and Engineering*, vol. 18, no. 4, pp. 1790–1802, Oct. 2021, doi: 10.1109/TASE.2020.3019684.

[17] F. Čapkovič, “Dealing with Deadlocks in Industrial Multi Agent Systems,” *Future Internet*, vol. 15, no. 3, Mar. 2023, doi: 10.3390/fi15030107.

[18] J. C. Mogul and K. K. Ramakrishnan, “Eliminating receive livelock in an interrupt-driven kernel,” *ACM Transactions on Computer Systems*, vol. 15, no. 3, pp. 217–252, 1997.

[19] G. G. Stetsyura, “Fast decentralized algorithms for resolving conflicts and deadlocks in resource allocation in data processing and control systems,” *Automation and Remote Control*, vol. 71, no. 4, pp. 708–717, 2010, doi: 10.1134/S0005117910040119.

[20] A. Araújo, D. Portugal, M. S. Couceiro, and R. P. Rocha, “Integrating Arduino-Based Educational Mobile Robots in ROS,” *Journal of Intelligent and Robotic Systems: Theory and Applications*, vol. 77, no. 2, pp. 281–298, Feb. 2015, doi: 10.1007/s10846-013-0007-4.

[21] I. Wasisto, N. Istiqomah, I. K. N. Trisnawan, and A. N. Jati, “Implementation of mobile sensor navigation system based on adaptive Monte Carlo localization,” in *2019 International Conference on Computer, Control, Informatics and its Applications (IC3INA)*, IEEE, 2019, pp. 187–192.

[22] M.-A. Chung and C.-W. Lin, “An Improved Localization of Mobile Robotic System Based on AMCL Algorithm,” *IEEE Sens J*, vol. 22, no. 1, pp. 900–908, 2022, doi: 10.1109/JSEN.2021.3126605.

[23] A. , Chandak, R. , Bodhale, and R. Burad, “Optimal shortest path using HAS, A star and Dijkstra algorithm,” *Imperial journal of interdisciplinary research*, vol. 2, 2016.

[24] R. Srivastava and P. Singh, “A Novel Optimized Travel Planner,” *Journal of Informatics Electrical and Electronics Engineering (JIEEE)*, vol. 3, no. 1, pp. 1–17, Apr. 2022, doi: 10.54060/JIEEE/003.01.007.

[25] G. H. Widiarta, M. Z. Romdlony, M. R. Rosa, and B. R. Trilaksono, “Control Lyapunov — Barrier Function Implementation for Mobile Robot Model with Hardware in the Loop,” in *2021 International Conference on Mechatronics, Robotics and Systems Engineering (MoRSE)*, Bali, 2021, pp. 1–6.

[26] M. Z. Romdlony and B. Jayawardhana, “Stabilization with guaranteed safety using Control Lyapunov-Barrier Function,” in *Automatica*, Elsevier Ltd, Apr. 2016, pp. 39–47. doi: 10.1016/j.automatica.2015.12.011.

[27] Y. Yu and M. Wang, “The path planning algorithm research based on cost field for autonomous vehicles,” in *Proceedings of the 2012 4th International Conference on Intelligent Human-Machine Systems and Cybernetics, IHMSC 2012*, 2012, pp. 38–41. doi: 10.1109/IHMSC.2012.105.

[28] A. Ammar, H. Bennaceur, I. Châari, A. Koubâa, and M. Alajlan, “Relaxed Dijkstra and A* with linear complexity for robot path planning problems in large-scale grid environments,” *Soft comput*, vol. 20,

no. 10, pp. 4149–4171, Oct. 2016, doi: 10.1007/s00500-015-1750-1.

[29] J. R. Gonz~ilez De Mendivil ~', F. Farifia, C. F. Alastruey, and R. Garitagoitia, "A safe distributed deadlock resolution algorithm," 1998.

[30] W. Lu, Y. Yang, L. Wang, W. Xing, and X. Che, "A novel priority-based deadlock detection and resolution algorithm in mobile agent systems," in *Proceedings - DMS 2016: 22nd International Conference on Distributed Multimedia Systems*, Knowledge Systems Institute Graduate School, 2016, pp. 61–68. doi: 10.18293/DMS2016-014.

[31] D. B. Lomet, "Subsystems of processes with deadlock avoidance," *IEEE Transactions on Software Engineering*, no. 3, pp. 297–304, 1980.

[32] Estefo, Pablo, J. Simmonds, R. Robbes, and J. Fabry, "The robot operating system: Package reuse and community dynamics," *Journal of Systems and Software*, vol. 151, pp. 226–242, 2019.