Distributed Autonomous Robotic Systems for Co-operative control

Daniel Robin K.
Department of ECE
Vidyavardhaka College of Engineering
Mysuru, Karnataka, India

Darshan Kumar V.
Department of ECE
Vidyavardhaka College of Engineering
Mysuru, Karnataka, India

Harshini Gowda C.
Department of ECE
Vidyavardhaka College of Engineering
Mysuru, Karnataka, India

Kaveramma M. G.
Department of ECE
Vidyavardhaka College of Engineering
Mysuru, Karnataka, India

D. J. Ravi
Department of ECE
Vidyavardhaka College of Engineering
Mysuru, Karnataka, India

Rohith K.
Department of ECE
Vidyavardhaka College of Engineering
Mysuru, Karnataka, India

***Abstract*— This paper presents a substantial design of a multi-robot system. The proposed system is more robust than individual robots. Till date, several technologies have been developed for multi-robot systems (MRS) focusing on path planning, exploration and mapping techniques, task allocation, coalition and team formation. This set-up employs three robots similar in hardware design and is interfaced with a set of open source software OpenCV and is programmed using Python. Positioning equations, guidance feedback laws were derived to make the system work in complete co-operation. Topic of interests largely incorporate image processing (IP), swarm control algorithm and correction feedback.**

Keywords*— *Multi-Robot System (MRS), Swarm robotics, Graph theory, Image Processing, OpenCV, Python, MQTT Protocol, Internet of Things

I. INTRODUCTION

The design of a robotic system can dramatically impact its work efficiency and cost. A stiff design limits the operating spatial distribution and increases the purpose of self-dependency on the system overall. Incorporating the added feature of handling the allocated task with a multi-agent system increases the overall robustness. The proposed setup, however gives an open approach on using all available open-source software and implementing using available limited hardware resources.

The literature survey referred by us presented a formation control algorithm. In a given dynamic environment, co-ordination needs to be adapted to different constraints at hand (consider the example, where there is drastic loss in performance due to an unreliable communication network influenced by external parameters). To address this issue, we contribute our rather conventional approach for coordination among the robots. Such an approach allows a robotic swarm to exploit environmental knowledge and adapt to various challenges encountered, enhancing its threshold performance. This result can be achieved by readily adapting the configurable task assignment and distributed world representations, based on the current state of the existing environment.

The underlying aspects of establishing a communication network, communication bandwidth, hardware analysis, software requirements must be shown to be scalable and possibly adaptive. However, some changes in the scenario might turn into a necessity to change the co-ordination strategy. The primary inspirational example for our research was the controlled swarm strategy observed in animal colonies. This biological model has highly capable efficiency in the environment with controlled scenarios.

II. APPROACH

To address the above-mentioned application domains, we generalize the approach into two main sets. The set-I being, the complete implementation of the positioning system for the constituent robots which is dubbed as the co-ordinate system and is carried out by utilizing OpenCV which is an open source library which uses inbuilt database functions to aid us in the domain of Image Processing and return the results as co-ordinates. The set-II being the implementation of the swarm algorithm and designing it based on the pre-defined assumptions and the forayed ground rules.

While there are sometimes applications such as tracking a lost target in surveillance applications, search and rescue scenarios including service robots operating in indoor environments.

It also portrays a challenge of handling the changes in environment requires a complete change in the strategy. Any

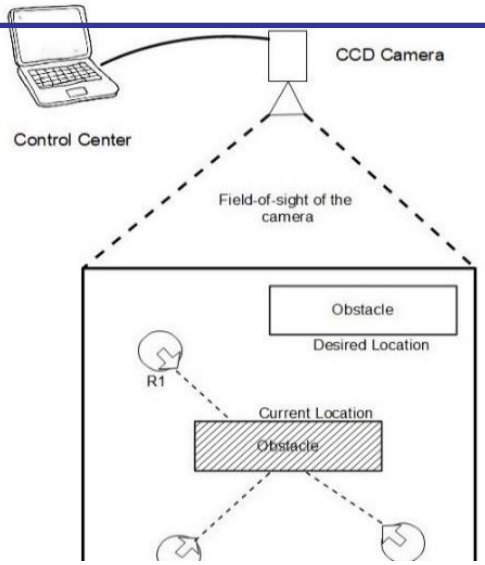

Fig. 1. The Proposed Set-up

known possible change is addressed as contextual knowledge where these robots have the independence of determining the underlying changes as contexts. Our proposed set-up is as shown in Figure 1.

In summarizing the proposed approach in total:

- Combining multiple independent robotic agents to give-out the distributed working model scenario.
- Use the computational outputs from the image-processing functions to obtain positioning Co-ordinates.

III. ARCHITECTURE OF THE SWARM SYSTEM

There are certain fundamental requirements to develop a distributed autonomous robotic system for co-operative control. This includes a robotic module for locomotion, camera module for image processing, wireless microcontroller and the arena setup necessary to execute all the control algorithms, neighbor-to-neighbor communication.

The robots are centimeter scale agents and are designed as line-followers. Although we never exploit it's "line-following" features. These bots combined with the external camera system form the basic architecture. Each robot requires to be programmed independently and is so small and robust that it can fit inside a cube of 17 cm dimension.

A. The Robotic Platform and its features

Two DC geared motors and caster wheels as support with a top speed of 15 cm/sec. Based on ATmega16A microcontroller comes with 7.2V 600mA NiMH battery. It is provided with a support for the servo mounted sensor pod which can make 180-degree scan for the map making. It is coupled with a 2x16 alphanumeric LCD Display. Two DC geared motors and caster wheels as support with a top speed of 15 cm/sec. Motors are controlled by L293D motor driver. This robot has an onboard socket for multi-robot and robot to PC communication. It also employs two wheel encoders for precise movement of the robot and this feature is widely used in this project. The Spark V platform is shown in Figure 2.

It supports many operational modes Standalone (Autonomous Control), PC as master and Robot as slave, Distributed (Multi Robot Communication). In our system, we employ the second operational mode where the PC is the master and the robotic agents are the slaves. We use Wi-Fi to establish control over the robot using a discrete module which will be explained in detail below. Thus, robot-to-robot communication is established indirectly.

B. Wireless Microcontroller

ESP8266 is a Wi-Fi enabled Microcontroller used to control each robotic agent and employs the Mosquitto open source tool by Eclipse which serves as a message broker for implementation of the Message Queuing Transport Telemetry (MQTT) protocol (client-server protocol) which is designed for machine-to-machine (M2M) communication and runs over TCP/IP. The current version of MQTT is 3.1 and the unique features of this standard of communication gives the following advantages:

- To avoid polling of sensors, allowing data to be sent to interested parties the moment it is ready.
- Lightweight, so that it can be used on very low bandwidth connections.

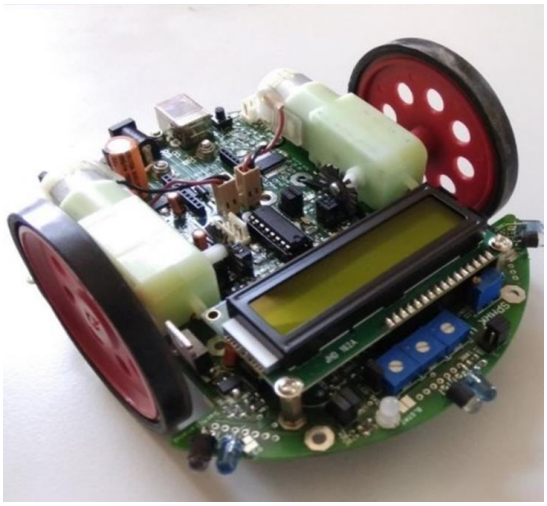

Fig. 2. The Spark V platform

Here, we can "publish" messages in a given topic and the devices "subscribed" to the it receives the same message after publishing provided, the client device and the server device are connected to the same Wi-Fi network. This can be programmed by using the Paho library available for Python which enables easy publishing and subscribing commands in the given Wi-Fi network. The subscribed devices in this case would be the robots themselves and the publishing device would be the laptop or the command center. Camera and Vision System

The exact position of the robot is measured through an external readily available camera system. The robot is tagged and marked with different tags that are unique. The camera acts as a Visual GPS and send the positional coordinates to each robot at a frame rate of 30fps at about 100Ghz. The visual tags add precise identification attributes for the system. The whole setup spanned around 120 cm^2 area. The vertical position of the system can be increased or decreased depending on how large the testing area of the application requires. The camera setup and the testing area are well illustrated in figure 2.

IV. ROBOT KINEMATICS

The Spark V has same maneuverability as of a differential drive robot. The planar 2-D motion is described by the vertical and horizontal movement. It provides a way to exploit the high-level formation of the surrounding environment. We were given the challenge of synchronizing our robots to overcome hardware differential factors such as wheel alignment, wheel direction and initial errors.

We address the set of robots as R={R1+R2+R3}. The Movement information for our framework influences the regular execution of movement of the individual agents.

The swarm control algorithm responds to the overall coordination protocol locally run by the control unit i.e. the laptop module. The neighboring robots evaluate the data sources and external environmental conditions. The Control System redirects the control signals for the neighboring robot agents. The entire multi-robot system will be able to communicate with each other. At this point the whole system can estimate the surrounding.

Noise was ignored. Real robots experience noise in both their actuators, and detection systems. To save computational resources we planned to ignore this. Methods for dealing with noise in robotics applications exist - notably in the field of Probabilistic Robotics - and may need to be considered when building real world models.

The basic ground rules considered for the robot kinematics:

- Making assumptions of the system to achieve simple goals
- Introduction to a larger environment and adapting to it
- Examination of the different behaviors for varying parameters both globally for all robots and by choosing random parameters for each robot

V. SWARM CONTROL ALGORITHM

This forms the integral part of our system and the intelligence programmed is straightforward. We generally set rules and constraints to the movement of each robot in the arena and is expected to behave accordingly.

Considering the arena, it uses a grid-wise topology as shown in Figure 3. Using this type of segmentation technique, the mapping and localization of each robot becomes easier. Each grid in the arena is roughly $17cm^2$ in size and can occupy only one robot at a given instance of time. The entire arena constitutes about 7x7 grids (square topology). Thus, it forms a graph with 49 vertices with each vertex having a maximum of four paths connected with its neighboring vertices.

Regarding the motion controls of the robots, there are four basic controls:

- Move Forward by one grid
- Turn Left by 90 degrees (spot-rotation)
- Turn Right by 90 degrees (spot-rotation)
- Stop

This form of traversal of the robots offers high accessibility and can cover the entire arena within a given time-frame.

Firstly, the vision system captures frames of the arena and performs the necessary processing techniques to obtain the grid locations of all the three robots and the object taken into consideration.

The entire algorithm is split into two different operations:

- Path Tracing
- The Drill

Under Path Tracing operation, each robot is given a specific grid location to occupy based on the orientation of the object. This allows the robots to manipulate the object appropriately.

A. Path Tracing

Based on the initial locations of the robots, we divide the swarm into two different teams consisting of two in one team and one in another. The first team is designated with the job of traversing the object vertically i.e. closer to one of the edges of the arena. The second team which is the third robot is given the responsibility to traverse the object side-ways i.e. horizontally into one of the corners of the arena.

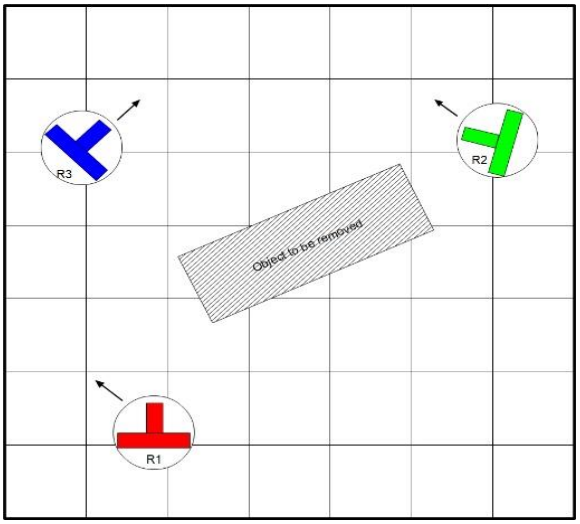

Fig. 3. Grid-based Topology

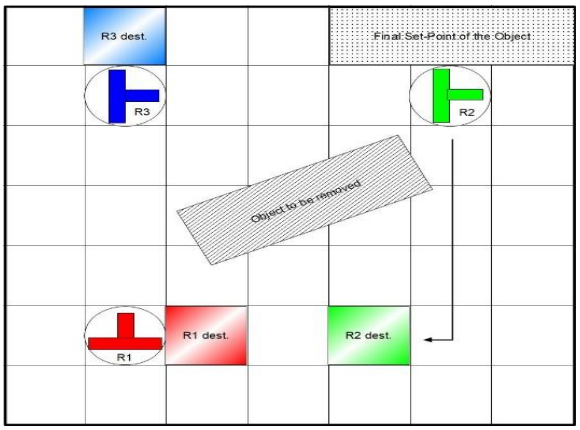

Fig. 4. Path Tracing

Maze routing techniques were initially (and still is) employed in the field of VLSI and PCB design. Different auto-router software utilizes this technique to efficiently route copper tracks across the board. We decided to use one of the

prominent algorithms in this field called the Soukup's Algorithm. Although we do not base the entire maze routing technique based on this algorithm, but the foundation for our path tracing technique is derived from this and we have made considerable amount of modifications to suit our design.

Before each robot is allocated with a specific path, certain pre-processing is required to check whether there are any obstacles along the way. The obstacles here for a specific robot would be the other two robots and the object itself. We store these grid locations in a separate list in the program.

Our algorithm follows three main operations:

- Length-first
- Breadth-first
- Over-take (special case)

The breadth-wise (vertically) algorithm finds a path to the destination grid by traversing breath-wise first and then length-wise(horizontally).

Given the condition that the source grid and the destination grid is not located in the same row or column (if they are in the same row or column the traversal becomes easier and this algorithm is not required), we traverse the robot in length-first operation by default. The algorithm is programmed to check for a restricted grid in every single iteration. If the program comes across a restricted grid, it immediately switches into breadth-first operation. And if another restricted grid is encountered again, it switches back to length-first mode. Therefore, these two operations are consecutively switched until the destination grid is reached. The whole list of generated vertices is then stored in a variable and the command for actuation is given by the Command Center.

One special case which could be possibly encountered would be the presence of an obstacle in the same row/column of that of the source and restricted grids. In this scenario, we designed the algorithm to "Over-take" the obstacle and continue its journey further on using the length-breadth operation. Path Tracing is shown in Figure 4.

B. The Drill

Once all the robots occupy their respective destinations, they must "co-operate" and manipulate the object to take it to the desired position as shown in figure 5. This is carried out by the two teams as mentioned before.

The team comprising of two robots move the object forward to one of the edges of the arena and the other team with one robot move the object towards the corner along the edges. The Drill operation is as shown in Figure 5.

This operation is simultaneous and works only when the object's alignment is same as that of the destination location's alignment. This whole algorithm is written in Python.

VI. IMAGE PROCESSING USING OPENCV

As stated earlier, the vision system is responsible for locating the robots and the object in real-time for our application. This acts like a pseudo-GPS model. The function must return the value of the grid locations of the robots and the object whenever called by the main program.

We decided to use distinct shapes as markers for our design. The shapes used were: triangle, square, rectangle, pentagon and hexagon. The entire algorithm for this process has three operations:

- Shape determination using contour detection
- Computing the center of the contour
- Estimation of grid location using a reference shape

Fig. 5. Shape detection using OpenCV

A. Shape Determination

We determine the shape of the contour by using contour approximation. contour approximation is an algorithm for reducing the number of points in a curve with a reduced set of points - thus the term approximation. This algorithm is commonly known as the Ramer-Douglas-Peucker algorithm, or simply, the split-and-merge algorithm. Contour approximation is predicated on the assumption that a curve can be approximated by a series of short line segments. This leads to a resulting approximated curve that consists of a subset of points that were defined by the original curve. Contour approximation is already implemented in OpenCV via the `cv2.approxPolyDP` method. To perform contour approximation, we first compute the perimeter of the contour, followed by constructing the actual contour approximation.

It's important to understand that a contour consists of a list of vertices. We can check the number of entries in this list to determine the shape of an object.

For example, if the approximated contour has three vertices, then it must be a triangle. If a contour has four vertices, then it must be either a square or a rectangle. To determine which, we compute the aspect ratio of the shape, which is simply the width of the contour bounding box divided by the height. If the aspect ratio is ~1.0, then we are examining a square (since all sides have approximately equal length). Otherwise, the shape is a rectangle. If a contour has five vertices, we can label it as a pentagon. The detected shapes on the arena is shown in Figure 5.

B. Computing the center

This part becomes fairly straight-forward once we find the contours and their vertices. All we do is calculate the center of the bounding box we use to depict the picture on the screen.

C. Estimation of Grid Location

We have used the square as the reference shape in this procedure. We place the square marker in the corner-most part of the arena and remain stationery all through-out the

task. Before calculating the relative distances (in pixels), we manually calculate the length of each grid in pixels by placing the centers of any two shapes exactly one grid apart (in our case it's 17 cm).

Thus, we establish the relationship between the distance in pixels and the actual distance on the arena. Using this information, we can calculate the relative distance and roughly obtain the grid location of the subject in consideration.

VII. APPLICATION SCENARIOS

Some of the most prominent applications of multi-robot cooperation include:

A. Factory floor emergency recall response

We can readily implement the existing multi-robot system that is currently used in this demonstration with considerable modifications for the application of warehouse management systems considering the improvements in parameters such as the amount of weight, each robot can handle and allowable maximum freedom of movement.

B. Outdoor industrial operations

Currently, available robotic agents drive employed production with automation. This basically includes tasks related to movement of equipment along a recurrent programmed route to open spaces. Therefore, multiple agents can aid to efficient and accurate results.

C. Humanitarian assistance and disaster relief (HA/DR)

Considering the scope of information-gathering activities required for planning, it provides the responder tools to sense and act in dangerous environments. For example, during an earthquake several of these small robotic agents can creep into every crevice and check for possible survivors.

D. Space exploration

The exploration of space presents numerous challenges, including an unpredictable environment and significant limitations on the mass and volume of equipment used to study that environment. Since one set of modules can be configured to perform many tasks, these robots can solve challenges while occupy little space and weight as compared to multiple devices. These bots can also be packaged in a convenient way to meet the volume constraints of spacecraft. Once on site, modules can be used to build structures, navigate across terrain, perform scientific studies, et cetera.

VIII. RESULTS OBTAINED

The formation control swarm algorithm is successfully tested and validated in the proposed methodology. The ability of the proposed experimental setup is demonstrated and met the expected results. The Swarm Algorithm is efficient, and the argument is supported by the co-ordinates provided by the outputs of Image Processing. The final set-up is shown in Figure 7. Although, there were a few technical glitches with robots themselves. The robots sometimes failed to turn exactly 90 degrees left or right, which had some minor effects during "The Drill". This issue could be solved by replacing higher resolution position wheel encoders for the robots.

IX. CONCLUSION AND FUTURE SCOPE

This Multi-Robot System is designed in such a way that it can be readily accessed and manipulated to design the necessary experiment to demonstrate the capabilities of the system. The Robot's modular design allows users to develop any custom modifications to be incorporated into the system. This robot's capabilities are demonstrated with proof-of-concept experiments on object manipulation, shortest path tracing, image processing and collective-collaborative transportation.

In the distant future, this proof-of-concept experiment can be increased in scale and the decision making computational capabilities can be decentralized and it wouldn't need any central control module to do the necessary computations.

Towards the end we will consolidate a wireless communication strategy to autonomously recharge its batteries. In further collaborative transportation experiments, we will include enough computational processing hardware within the robot to exclude any external processing requirement from the control system, therefore giving each robot the independence to both communicate and to compute the necessary. Necessary future work can be assimilated into the following control strategies:

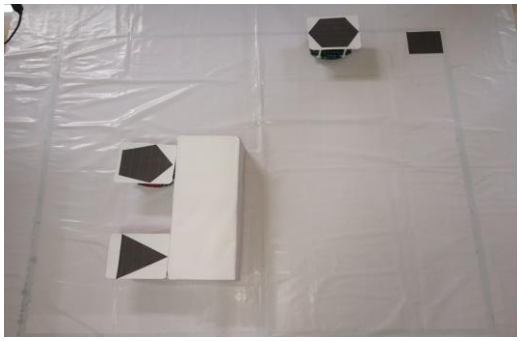

Fig. 6. Enhancement Module

- Enhancements to the hardware in terms of actuation and sophisticated sensors
- Enhancements in the simulation model to suit varying applications
- Partial communicating autonomy for self-maintenance
- Using Machine learning for autonomous path-tracing for improved efficiency

Significant amount of further enhancements are necessary for improvement in system functionality and potential improvements in compromising between just allowing the robot to observe the obstacle from a distance through the vision-system. More precise encoder motors give improved accuracy.

It is observed that the speed for simulation increases as more obstacles are added to the environment. The decrease in speed may be enhanced by using a more robust hardware design. The demonstration presented in this paper were setup so that an independent robot is guided by a central control system and the commands flow in and out the control framework. Further contradictory add-ons might enable the same set of frameworks to improve the efficiency by

allowing the necessary computational hardware to be implemented on each robot individually.

REFERENCES

[1] Francesco Riccio, "Multi-Robot Search for a Moving Target: Integrating World Modeling, Task Assignment and Context", 2016.
[2] Dylan Fyler, "Distributed Object Manipulation Using Mobile Multi-Agent System", 2015.
[3] Sean Wilson, Pheeno, "A Versatile Swarm Robotic Research and Education Platform", IEEE Robotics and Automation Letters, 2015.
[4] Peter Sauer, "Using Internet of Things Technology to Create a Ready Platform Independent Robotics Framework", 2016.
[5] Abhijit Makhal, Manish Raj, Karan Singh, P.Chakraborty and G.C.Nandi, "Path Planning through Maze Routing for a Mobile Robot with Nonholonomic Constraints", 2012.
[6] Wanrong Huang, Yanzhen Wang and Xiaodong Yi, "A Deep Reinforcement Learning Approach to Preserve Connectivity for Multi-robot Systems", 2017
[7] J. Soukup, “Fast Maze Router” 1978
[8] Ying Zou, Mao Shan, Mingyang Guan, Changyun Wen, Kwang-Yong Lim, "A Trajectory Reconstruction Approach for Leader-Following of Multi-Robot System", 2017
[9] Hasan Ercan, Pinar Boyraz, "Design of a Modular Mobile Multi Robot System: ULGEN (Universal-Generative Robot) ", 2016
[10] Victor Hernandez, Erik Schaffernicht and Achim J. Lilienthal, "Bayesian Gas Source Localization and Exploration with a Multi-Robot System Using Partial Differential Equation Based Modeling", 2017.
[11] Made Widhi Surya Atman, Julian Hay, Junya Yamauchi, Takeshi Hatanaka1 and Masayuki Fujita, "Two Variations of Passivity-Short-Based Semi-autonomous Robotic Swarms" -Society of Instrument and Control Engineers International Symposium on Control Systems(SICE ISCS) Tokyo, Japan, March 9-11, 2018.
[12] Toshiyuki Yasuda , Kazuhiro Ohkura, "Collective Behavior Acquisition of Real Robotic Swarms using Deep Reinforcement Learning", Second IEEE International Conference on Robotic Computing, 2018.