Design of Systems and Optimizations for Autonomous Agents using passive RFID Localization Techniques

Recycling Collaborative Robots

A Thesis
Presented to
The Academic Faculty

by

Parsa Piroozi Esfahani

In Partial Fulfillment
of the Requirements of the Degree
Bachelor of Science in Computer Science in the
College of Computing

Georgia Institute of technology
May 2020

Computer Science College of Computing— Research Option Thesis

HPArch-research (High Performance Computer Architecture) Research Lab
SARAV— Systems and Architectures for Robotics and Autonomous Vehicles Research Group

Table of Contents

Abstract

This paper aims to describe the work done towards designing and implementing systems and optimizations for a set of autonomous robots that intend to collaborate towards accomplishing the specific common goal of transporting recycled objects. At first the paper dives into the aspects of autonomous behavior and describes what exactly constitutes autonomous behavior and then proceeds to explain the specifics of the research work in our lab at Georgia Tech and also mentions the importance and reasons behind performing such research. The paper then goes into an extensive literature review of autonomous collaborative topics and puts emphasis on RFID localization techniques. And finally describes the results and discusses the outcomes of the project. Having the research abruptly paused due to the COVID-19 pandemic in Spring of 2020, prevented us from getting to implement the collaborative medium for the robots and putting into a software service box for shipment, however we were able to discover many new findings in the fields of autonomous behavior development and implement a successful and consistent RFID reader-tag duo for our robots to be next used in implementing a collaborative medium for the robots.

Special thanks and gratitude towards professors, advisors, UROP representatives and instructors, and graduate students who helped me and our research group in conducting this great research.

Introduction

The need for autonomous behavior has been increasing at a very high rate in the past decade. Autonomous cars, drones, and even surgical arms are being found more and more involved in our everyday life with applications in medicine, industry, and even entertainment (Sukhan Lee). In the field of autonomous behavior development, any piece of machinery that can operate autonomously can be referred to as an autonomous agent.

The power of autonomous behavior is determined by the strength of the technological tools and computation machines behind their implementation and development. The more capable the computers behind the development of autonomous agents' behavior, the more advanced their behavior will be. As an example we can refer to the infamous comparison between AlphaZero and Stockfish, two of the most powerful chess engines currently being used. Despite Stockfish's longer history and higher popularity, AlphaZero has proven to be a "superior" agent due to the much higher performance offered by the computers and compute power behind AlphaZero developed by Google Inc (Pete). However, this gain in performance comes with a cost in power and resources which explains why AlphaZero is not widely used or as available as the Stockfish engine. On the other hand, better and more complicated behavior can also be achieved, to some extent, on the same computational models and devices through optimizations.

The research behind the development and optimization of autonomous capability, as done in our lab, is based heavily on implementation and design. This concludes that our lab first spends the majority of our time designing, coding, and testing our ideas, then proceeds to gather data from testing and experimenting with our own design, as opposed to running already tested

models. Our approach also establishes the fact that our most eminent source of information in our research is extracted from the past work (previous semesters) done, in our lab, on this same exact research project. In our lab, we want to design and implement systems and optimizations for autonomous agents.

Our research lab, HPArch-research, which strives to enable high-performance and energy efficient computing from microarchitectures to compilers, divides its SARAV (Systems and Architectures for Robotics and Autonomous Vehicles) research group into three teams of Connected Cars, Collaborative Robots, and Drones. As a part of the Collaborative Robots team, we focus on designing and implementing an autonomous system for a set of robots to work together, *collaboratively*, towards performing a defined common goal. The common goal/objective that our collaborative bots are designed to perform is *recycling* of objects. We strive for our final design system to have a group of robots that sort objects by transporting them to their designated areas based on their identity, recognized by Computer Vision. Despite having a defined goal that our system would be designed for (Recycling), the course of the action is never defined for our collaborative robots, which means that, for example, the active bots in a run of the system have no knowledge of the placement of objects as they are always placed in the field randomly. The emphasis of this autonomous system is on *collaborative-work*. However, each individual robot still has its own stages of autonomously accomplishing one milestone of the goal (sorting a single object). These stages consist of Navigation, Detection, Recognition, and Approach & Taking-control. Each of these stages utilizes many different technologies such as Robotic Perception, Computer vision, OpenCV, etc.

To aid with the implementation of the detection and recognition stages of the collaborative system where robots can openly communicate with one another, we have chosen to put our focus on using a protocol that allows robots to passively locate each other and other objects nearby. This system is called RFID Localization. RFID which stands for Radio-frequency identification uses electromagnetic waves in a field to automatically detect and scan tags associated with other objects. Compared to similar protocols such as NFC, RFID localization allows us to detect objects at short-to-medium ranges using radio frequency waves while keeping power consumption at much lower rates. Compared to NFC, RFID allows for longer read/scan ranges, but might be more costly than barcode scanners.

In our research, we want to design a highly optimized computation model for our autonomous collaborative system in order to have the participating/active bots perform their common and collaborative goal efficiently. This will aid us in resolving some of the most crucial questions that we're aiming to answer in our research. In our research we seek to investigate the answers to questions such as, how should we efficiently maximize the throughput of *each* robot, in isolation and in our Collaborative System? What is the best method of communication between the robots? and where is the best place for the bulk of the computations and processings? Locally on the robots, or remotely on a separate server/machine (communicated to the robot)?

The robots used in our research-lab, as our autonomous agents, are called iRobots. An extensive reference to *iRobot Roomba 600 Open Interface specification* documentation is provided in the Works Cited section. We conduct our research by exploring the possibilities of using a network system of Raspberry Pi(s) along with an RFID Localization protocol to run,

control, and process our implemented autonomous and collaborative behavior system of our robots. Raspberry Pi(s) allow us to easily manage library dependencies and implement portable code that can be executed on any other Linux based machine across the research community and industry, while the RFID technology allows us to implement a passive detection and recognition algorithm to successfully localize and nearby objects and other robots. Seven of the other technologies and platforms used in our research are RPLidar, Minoru 3D Webcam Camera, Raspberry Pi Camera, MeArm Pi Robot Arm, Python, as our most used programming language, and ROS (Robot Operating System) as a flexible framework for writing robot software and a source for tools, libraries, and packages for the other technologies to utilize (references to specific APIs, repositories, and libraries are included later in the paper when mentioned specifically and are also present in the Works Cited section).

Our research explores the possibilities of utilizing autonomous agents in the industrial field of sorting and recycling. However, the one of the most important and compelling reasons as to why this area/field of *autonomous collaborative control* research is necessary and significant, lies behind the fact that this research can be extrapolated into development work for other fields such as Transportation and Communication, Rescue Organizations & Disaster Recovery, and etc. Our research which is focused on designing systems, architectures, and optimizations for robotics and autonomous agents, can bring numerous benefits and thus help millions of people around the world by bringing about developmental work that can foster great innovations such as Network of Autonomous Vehicles, Fully Autonomous Production/Assembly Line, Rescue Robots, and etc. in those fields.

Literature Review

The behavior of Autonomous agents is defined as the operation of non-human controlled agents based on a perceived notion of their surroundings through different means such as computer vision and color recognition without any human intervention (Nayeripour, Majid, et al., 2010). Design of Autonomous systems and architectures for robotics and Autonomous Vehicles has been one of the most trending and heated fields and areas of research in recent technological developments. From designing a Network of Autonomous Vehicles to innovative autonomous surgical arms the field of medicine, design and control of autonomous agents has been seeking to make the lives of people around the world easier and more efficient.

Even though, design of Autonomous systems, architectures, and optimizations can be done on interfaces developed on different communication platforms or transmission protocols, in our lab, SARAV (Systems and Architectures for Robotics and Autonomous Vehicles) research teams have decided to implement our design on an SSH interfaced (Secure Shell) network of Raspberry Pi(s). Interfaced Raspberry Pi networks are indeed one of the most prominently used platforms for communication, development, and control of bots in this area of autonomous behavior development and research (Yan, Zhi, et al., 2013) and are widely seen in the literature for this field. Raspberry Pi(s) are complete computers which can be programmed to perform functions of a computer without human interaction (no interfacing I/O systems). The Operating System installed on Raspberry Pi(s) is usually a version of the many available and published Linux distributions. A Linux OS (Operating System) allows for a more streamlined download, install, and utilization of software packages distributed for interacting and controlling robots. Our Collaborative Robots team has chosen to use the HypriotOS, as it is a GUI-less (Graphical User

Interface) version of the strong Debian Linux distribution. This decision will thus allow for faster and more optimized processings and communications due to utilizing a lighter OS, while still providing all the feature sets of having a strong Linux Operating System. One of the most significantly used robot interaction interface packages provided and accessible in Linux OS distributions is the ROS, Robot Operating System. ROS can be installed on Ubuntu or Debian variants of Linux and provides packages for interacting with different aspects of a robot. In our lab, we're currently utilizing ROS to control the Robot Arm and the dual webcam camera on our iRobots. We have been observing a great increase in the presence and utilization of ROS in Robotic Coordination and Control literature and research fields in the recent years (Yan, Zhi, et al., 2013) since its first stable release in 2007. This can possibly be due to the *thread-safe* aspect of ROS that makes it a rational decision for robotics developers to implement thread-consuming robotic software and control programs. This thread-safe feature is indeed our lab team's initial reason for exploring the utilization of ROS for controlling and enabling coordination and interaction between robots in their autonomous and collaborative effort in performing their common objective/goal.

Our Collaborative Robots lab team is responsible for designing and implementing an autonomous and collaborative system of robots for recognizing and sorting recyclable materials by putting them in specified locations based on their identity. A collaborative system such as the one being designed in our research problem is known as a Multi-Robot System, MRS. MRSs have significantly grown recently in both importance and size (Yan, Zhi, et al., 2013). Our system is also classified as a mobile MRS due to having a controlling and processing unit away and housed outside of the participating/active robot itself. An MRS contains more than one robot

agent and is highly influenced by collaboration. MRS systems have a much better "spatial distribution" compared to a single-robot system and introduce robustness into the system due to having more than one data collecting unit in the active field (Yan, Zhi, et al., 2013). The robustness aids in having more reliability and flexibility in accomplishing the objective of our system. An MRS also allows for more diverse scalability opportunities due to having more units in the system to improve upon. For example, in our system, to increase the efficiency of our system we can improve the performance on any of our robots and trigger a large efficiency improvement in the system as a whole. On the other hand, one of the downsides of an MRS system is the resulting need for a more sophisticated and perhaps inefficient protocol for communication to and between the active robots in the system. As mentioned, our lab, and also many MRS oriented labs, uses the Secure Shell (SSH) protocol as the needed communication platform.

Autonomous agents run on software that take actions "without user intervention". The lack of human/user intervention introduces the idea of an "adaptive" robot control system. The principle behind an adaptive system emphasizes designing a system that improves in performance the more number of times it attempts/finishes executing its objectives. In other words, the system gathers data and learns from its own courses and actions. However, implementing an adaptive robotic control system is a very demanding and heavy approach at developing and implementing autonomous behavior (Lieberman, Henry, 1999), as it has to process massive supplies of data from the working robots and thus requires strong computers that can deliver high compute resources and can supply high electric power. The technique mainly used for developing an adaptive robotic system is Machine Learning. Machine Learning

algorithms strengthen themselves and enhance their own performance the more they are run, by gathering and processing data on their performance, and thus leading to more optimized algorithms and much more accurate performances. In our lab we're currently using Machine learning in recognizing the identity of the detected objects in the course that are going to be sorted. The use of Machine Learning in our robots' recognition algorithms of the systems will be one of the most processing heavy aspects of our design and implementation, because our MRS would need to deal with multiple problems/data collecting units (robots) at a time whereas most single-robot systems would have to deal with only a single processing problem at a given time (Maes, Pattie, 1995) during their course of the execution.

In order to be able to detect the positions of different robots and secondary objects in an MRS, there is always a need for a localization technique that can localize both the objects and the participating/active robots in the medium/field. One of the used techniques and protocols for solving localization problems is through the use of RFID readers and tags. RFID allows for localizations of other robots through the use of a few static RFID tags in the field and allows for the localization of objects by having tags on the objects themselves. RFID offers a passive detection and recognition schema where RFID readers are able to read/scan tags that do not consume any power by themselves. The power is supplied and used only by the reader device. The RFID reader sends out electromagnetic waves and attempts to read waves that are reflected off of the tags. The reader then proceeds to process the reflected wave and output a distinctive value associated only with the scanned tag. RFID readers and tags can be combined with different algorithms to enable localization of fields of objects and robots. One of the most widely used algorithms here is the greedy algorithm. According to Diallo in their research and paper

regarding Wireless Indoor Localization Using Passive RFID Tags, the greedy algorithm allows readers that are mounted on top of robots to cover the most amount of area in search for passive RFID tags. As explained RFID tags are passive and do not emit any sort of waves or signals by themselves, therefore this means that readers need to actively look and search for tags in a field. This approach has opened the door for many different path finding and search algorithms to enter the field. One of these other algorithms is the A* (pronounced A-star) search and path finding algorithm (Huang). However, one of the complications that is carried with the implementation of such algorithms is the presence of conflicting and blocking objects in the straight wave and signal travel between an RFID reader and tag. As one of the limitations of RFID, waves need to be directed and travel between a tag and the reader with little to no blockage. However, when using A* algorithm with RFIDs, which strives to optimize the amount of distance covered during search while delivering the maximum results, the autonomous agents might have trouble finding and detecting all the objective tags in a field (Buffi). Such complications and limitations convinces researchers to use more greedy and perhaps less optimized algorithms that depend on direct undisturbed communication.

With any open interface comes security and privacy concerns. RFID communication raises questions such as, what if another nearby RFID reader scans tags that are confidential and should be kept private, or how can the identity of readers and tags be ensured? Even though it is possible for unauthorized RFID readers to read nearby confidential tags (maybe such as the ones in credit cards), however according to Mohite these concerns can be reduced through the use of techniques such as Renewable Identity approach, where each scan of a tag produced results that can not be reused or deciphered by any unauthorized RFID reader.

Even though Autonomous systems are designed to carry the course of their objective execution in almost all situations without expecting any user input/aid, most Autonomous agents are "programmed in parallel with the user programs" (Lieberman, Henry, 1999) which will allow for user interactions to be involved at a time when it's necessary during the course of the robots' actions in the system. Humans might still need to interact with autonomous robots and their mediums in different actions such as bringing a single robot or the entire system's execution to a full halt, triggering an endless search, and etc. That is because, even the most sophisticated autonomous systems in the industry or research need to have and present the capability of accepting different types of user input, asynchronously, at any time during the system's execution, in case of a need. In our Collaborative Robots System, we have designed and implemented multiple programs, with the help of the safe multi-thread consuming aspect of ROS, that can take actions (inputs) from the user at any time during the system's course of execution and then adjust the active robots' course of actions accordingly.

Methodology

Methods

Every step in our research towards designing and implementing an autonomous collaborative system of robots starts with a problem statement. We start by defining our problem and writing, documenting, and recording our set of goals and constraints in the defined problem statement. We then proceed to prioritize each goal in order to allow for more rational decision making in the case of an encountered problem later during implementation. Prioritizing our goals allows us to, for example, pick the best option between many that all achieve the entire or a portion of our goals. For example, during our research we are striving to find out what number of

robots will yield the highest efficiency in the system with respect to the most amount of work being done. On the other hand we also need to figure out what sort of computation protocol (for example, on a Raspberry Pi on the robot vs communicated computation from a laptop) will result in the most efficient system. Different computation protocols along with the use of different libraries and/or algorithms can yield many different systems that all vary in effectiveness and efficiency. By prioritizing our system we can then decide between different algorithms, where each satisfies a different goal, if not all.

To put into perspective, we set getting concrete results from RFID reads as one of our top prioritized goals. We started by using a public API and Python wrapper repository with documentation called the '*python-mercuryapi'*, referenced in Works Cited section and is owned by Petr Gotthard (GitHub username: *'gotthardp'*), that is specifically implemented towards the use with our version of RFID reader (M6E Nano model, however, given the full model details in the materials section). We aimed to get both continuous and discrete and periodic reads from our RFID reader. These steps were not yet implemented on the robot and were being done locally on an off-load Pi. We find through these experiments that enabling periodic reads at specific time intervals yields the most consistent results compared to continuous reads enabled by the '*start_reading(callback)*' method, which results in inconsistent reads after some time, due to performing too much activity and too many reads that therefore cause overheatings.

After our preliminary steps are done, we then move on to our implementation stage that starts off by designing an algorithm and then writing a pseudocode for a given task that our robot needs to perform. Pseudocode is written in pure english and contains little to no code specific constructs so that it can be implemented in any different coding language. After designing the

algorithm and writing the pseudocode code we then go on to implement our solutions in our chosen programming language, which is Python. We implement our solution's code in separate functions using the public *PyCreate2 iRobot API*, referenced in Works Cited section and is owned by Brandon Pomeroy (GitHub username: *'pomeroyb'*), so we can result in a modular code that can be reused, tweaked, and overloaded with different combinations of parameters if necessary and runs directly on our iRobots. After adding our solutions to the code base, using GitHub, as our client for *git*, we branch our code base to a separate branch, commit, and push our changes to our remote code base repository, that is privately owned by our HPArch research-lab's GitHub Enterprise under Georgia Tech. By using version control we make sure that we keep all the different versions of our code that we have ever created remotely so we would be able to access them if need be later in the timeline of our research. By this protocol it will also be much elaborate to find issues and bugs in our code later during the implementation phase.

After implementing our solution we enter the testing and experimenting phase where we run our code for accomplishing the problem statements and goals defined during earlier stages. By running our code we can debug any problems that might be present in the code, syntactically or logically. After having fully tested our design and implementation we start to run experiments and observe the performance of our solution. With this approach we can iteratively add to our solution and debug as we progress through our problem statements designs.

So our methods that utilizes our setup of Raspberry Pi, iRobot, RFID reader, and RFID passive tags follow a structured flow. This flow that is described in the following diagram describes the action of a single robot on a designated 5 x 5 grid field (2.5 meters on each side) in

our lab. The actions of a robot starts by robot movement that is followed by a continuous search for objects. Afterwards once an object is detected the OpenCV algorithm is supplied information by the Minoru 3D camera on the robot for object recognition. The computation of the recognition is done on the Raspberry Pi housed on the robot. Then the bot moves to a drop of location, drops the object at the designated location based on the identity of the object and continues the search for objects. These steps reflect the states of a state machine where each robot after being done with relocating an object starts back at the start state. On a side note, all the actions of getting a hold and control of an object is done with the use of a MeArm robot arm.

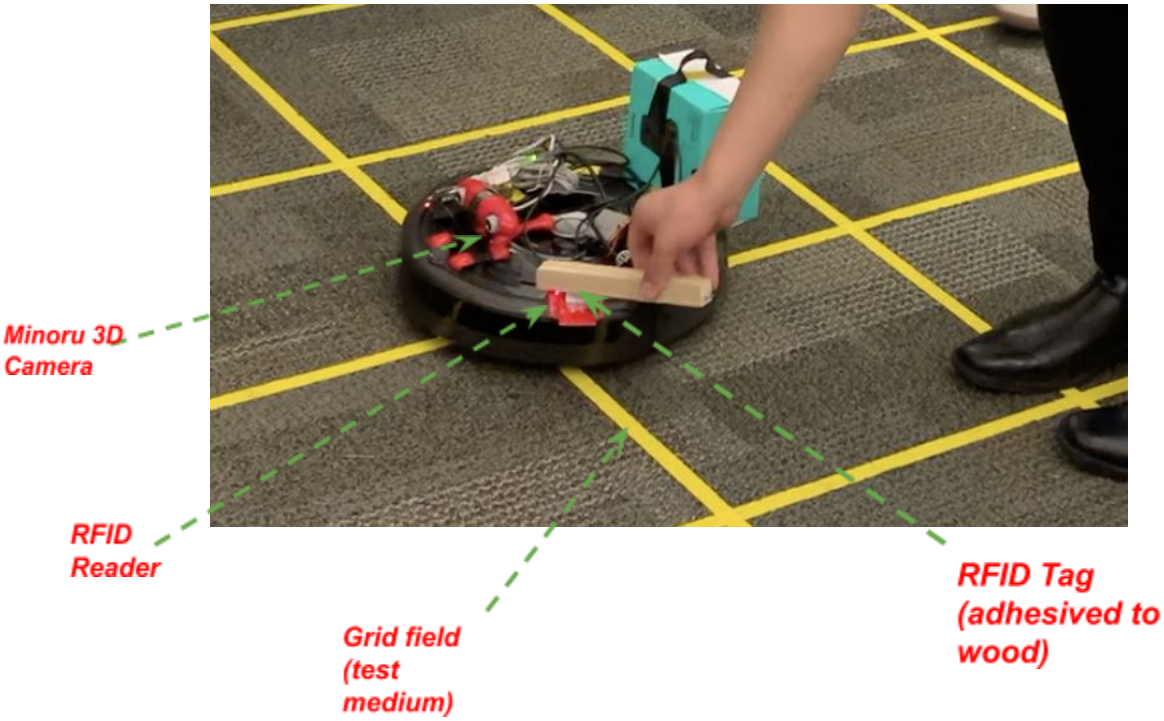

Figure. iRobot in the grid field, RFID module being tested, labels provided.

While the robot continues to follow and adhere to its normal routine function related to motion and path following, the RFID reader is continuously attempting to read RFID tags in a

periodic manner. With each recognition of a tag the bot is signaled to act accordingly. The scanned tag can also offer valuable information such as the type of the nearby object and whether or not it's a recycle object or a fellow robot.

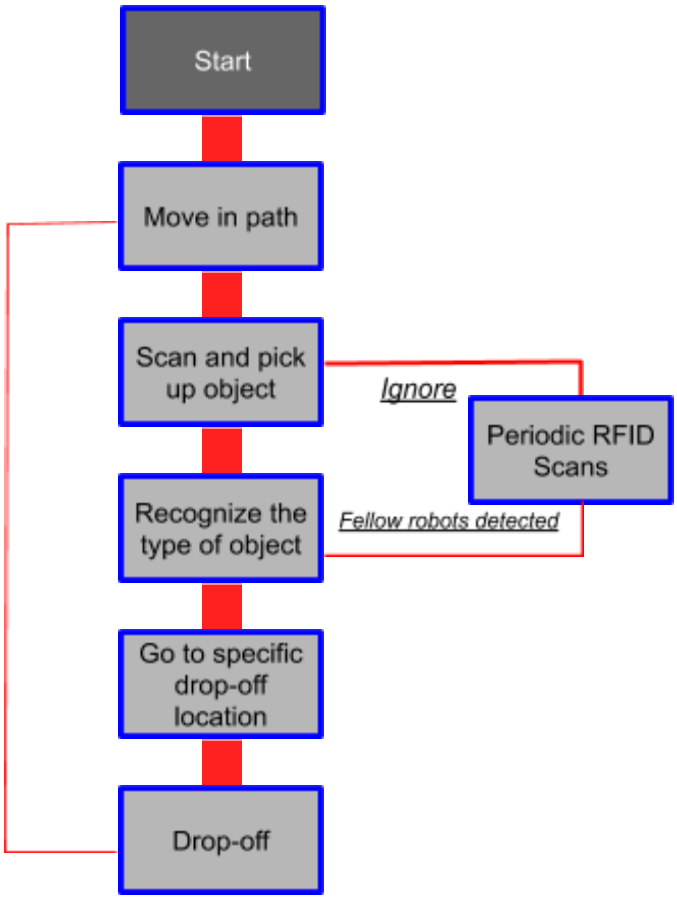

Figure. Downward State flow Diagram of robot actions.

Materials

In our research we utilize a wide variety of materials, most of which are related to computer hardware and software engineering/development. In creating our evaluation platforms, our most prominent tools and materials included Roomba iRobots, Raspberry Pi 3s, Lidar sensor,

Raspberry Pi Cameras, Minoru 3D Cameras, Minoru 3D Camera, and RFID module components and Tags.

Our RFID module consisted of *SparkFun Simultaneous UHF RFID Tag Reader* version M6E Nano (SEN-14066), *SparkFun Simultaneous UHF RFID Tag - Adhesive* (WRL-14151), and *SparkFun Serial Basic Breakout* model CH34OG (DEV-14050), all from SparkFun Electronics. In the set of our secondary tools some important materials to mention are more virtual and they consist of the SSH protocol to log into the iRobot's Raspberry Pis microcontrollers, git as Version Control, and Python as our programming language of choice.

Discussion and Results

We have exposed our iRobots to a new environment of interacting with not just user input but also RFID read results. We enabled our robot to move in a square path, by being programmed to make a 90 degree square turn only when they scanned with a particular RFID tag value. This allows our bots to follow a specific routine for following different paths with different characteristics. On the other hand, the way that the correct tag number/value that the robot should react to is introduced to the robot is by introducing the correct tag value as the first tag that gets scanned by the robot. This means that the first tag that gets scanned by the RFID mounted on the robot is going to be stored inside of the bot's memory and accessed every time another tag is scanned. The bot will react only and only when it scans the first tag again by checking the value of the tag that it reads. This allows for different approaches to be implemented and shows how the bot and the RFID can make a clear distinction between the tags that it reads. Therefore in later cases, for example, by implementing this approach the bot can act accordingly based on the identity associated with the tag of an object that it scans.

Throughout these experiments we were able to identify 3 key points with regards to implementing an RFID localization system in a collaborative medium.

1. **65% RFID reader Duty-Cycle**: Every time our RFID reader got hot and the temperature inside of it increased a certain threshold, the reads that were coming from the reader and processed into the Pi were very inconsistent. This means that values were either coming in late or certain tags were not being read despite being very close to the reader. This was due to the fact that the reader was doing a lot of unnecessary work and sending a lot of waves that were not being reflected. By decreasing the duty-cycle of the reader we were able to decrease the amount of work being done by the reader and thus make the readings much more consistent, since the reader did not heat-up anymore as much.
2. **JC Type 5V Battery Powered:** During our first experimental runs of implementing the RFID we were getting very rare and inconsistent results with even extreme close ranges. We figured out that this was due to insufficient power supply from the Raspberry Pi. The Pi was not able to supply enough power to the reader to perform the necessary reads therefore we soldered on a JC Type power connecter and powered our reader with a 5V battery which enhanced our results dramatically.
3. **Antenna:** Despite improving our results the range of reads that we were able to get from our RFID reader did not exceed more than a few inches. Therefore, we proposed getting a corresponding Antenna for our RFID reader to improve read/scan ranges.

As you can see RFID localization can provide a great medium and protocol for implementing a collaborative robotic medium. RFIDs enable robots to scan and detect other fellow robots in nearby ranges and communicate accordingly.

According to literature using RFID protocols for performing localization techniques can be made streamlined through the use of everyday devices such as mobile phones, if they are able to emit electromagnetic waves for RFID communication (Buffi). However, every implementation of RFID protocols requires the developers to adhere to two primary tasks which are Tracking and Validation. While adhering to these tasks and implementing the protocol to successfully perform the requirements of each stage of these tasks, RFID can provide *scalability* and *reliability* when done properly for localization purposes (Jian). The architecture of RFID readers along with geometry principles, and localization and search algorithms can allow devices/robots to localize and gain knowledge of multiple RFID tagged objects in close proximity (Liu). As described in our research and just similar to the choices made in our experiments, localization through RFID implementation can be done through Continuous-scanning or by performing periodic-scans while also having defined duty cycles though using Time-cost analysis. Despite the fact that some RFID reader architectures can allow for continuous reads with no low cycles, implementing and finding a time and cost efficient duty cycle and scan/read time can greatly enhance the accuracy and consistency of the RFID reads. According to Liu, the following is a formula derived for finding an effective scan time for the

$$T = \frac{(2r + W)\left\lceil \frac{L-r}{2r} \right\rceil}{s_x} + \frac{\left\lceil \frac{L-r}{2r} \right\rceil 2r}{s_y}$$

RFID to be actively looking for tag reads; where T is the continuous scan/read time period, r is the minimum radius of the read region of the RFID at low power, W and L are the width and length of the 'warehouse' architecture of the RFID reader chip and S_x S_y are the speed of the

movement of the RFID reader chip with respect to x and y axes respectively, if moving at all. Roughly speaking, an RFID chip can read the same tag 20 times per second which imposes much extra work-load on the reader chip. Therefore this read rate can be lowered by defining a periodic read/scan instead of a continuous one along with implementing a power duty-cycle. Even though all of these techniques are approached to improve the performance and efficiency of the RFID chip, however, the performance of a localization algorithm and its run-time complexities are mostly affected by the performance of the search and localization algorithms choices implemented on the autonomous agents and how they perform in different cases.

As described in the abstract of the paper even though implementing the collaborative protocol for the robots was one of the goals of the project, we unfortunately were not able to get to any development or experiment to this part because of the abrupt stop to our research due to the COVID-19 pandemic. Despite not being able to fully devote time and development effort to this part we were able to extensively study this aspect of MRS systems and present our findings in the literature review section of the paper. As a part of the future works for this research, it is intended for the future research students to pick up this research and use our RFID localization findings and utilize them in implementing the collaborative medium as the very nexts step.

Conclusion

Our research problem started with a simple question of whether or not robots can communicate and talk to each other and use various detection techniques to navigate and perform goals together. Despite not being able to move forward completely and finish the research due to the pandemic situation in Spring of year 2020, we ended our research with being able to use

Computer vision techniques and speech recognition commands for controlling our set of robots while implementing different localization techniques and algorithms using RFIDs to allow robots to know of each others location towards the goal of collaboration. This research was a great opportunity for discovering new techniques in the field of robotics and will definitely be continued by future students with the immediate goal of enabling collaboration within a set of robots in achieving a certain task and/or goal.

At last as mentioned before, these efforts in finding efficient communication, collaboration, and localization techniques using RFIDs are all towards the common goal of decreasing the need for human resources, increasing productivity, and facilitating the efficient and precise accomplishment of different tasks of important industries such as medicine/medical.

Works Cited

A. Back, D. Rus, "Towards task-directed coordinated manipulation", *Proceedings of the IEEE/RSJ 1993 International Conference on Intelligent Robots and Systems*, 1993.

Buffi, Alice, and Paolo Nepa. “An RFID-Based Technique for Train Localization with Passive Tags.” 2017 IEEE International Conference on RFID (RFID), 2017, doi:10.1109/rfid.2017.7945602.

Diallo, Alseny, et al. “Wireless Indoor Localization Using Passive RFID Tags.” Procedia Computer Science, vol. 155, 2019, pp. 210–217., doi:10.1016/j.procs.2019.08.031.

Glorennec, Pierre-Yves. “Coordination between Autonomous Robots.” International Journal of Approximate Reasoning, vol. 17, no. 4, 1997, pp. 433–446., doi:10.1016/s0888-613x(97)00004-2.

G. Oriolo, G. Ulivi, M. Vendittelli, "On-line map building and navigation for autonomous mobile robots", Proc. 1995 IEEE Int. Conf. Robotics Automation, pp. 2900-2906, 1995.

Gotthardp. “Gotthardp/Python-Mercuryapi.” GitHub, 7 Feb. 2020, github.com/gotthardp/python-mercuryapi.

Huang, Baohu, et al. “RFID Fingerprint-Based Localization Based on Different Resampling Algorithms.” Journal of Computer Applications, vol. 33, no. 2, 2013, pp. 595–599., doi:10.3724/sp.j.1087.2013.00595.

IRobot Roomba 600 Open Interface Spec. cdn-shop.adafruit.com/datasheets/create_2_Open_Interface_Spec.pdf.

Jian, Mian, and Yasutake Takahashi. “2A2-L05 Investigation on Mobile Robot Self-Localization Based on RFID System Using Less RFID Readers.” The Proceedings of JSME Annual Conference on Robotics and Mechatronics (Robomec), vol. 2015, no. 0, 2015, doi:10.1299/jsmermd.2015._2a2-l05_1.

Lieberman, Henry. “Intelligent Interface Agents.” Proceedings of the 4th International Conference on Intelligent User Interfaces - IUI 99, 1999, doi:10.1145/291080.291083.

Liu, Jinkai, et al. “RILS: RFID Indoor Localization System Using Mobile Readers.” International Journal of Distributed Sensor Networks, vol. 14, no. 4, 2018, p. 155014771877128., doi:10.1177/1550147718771288.

Maes, Pattie. “Modeling Adaptive Autonomous Agents.” Artificial Life, 1995, doi:10.7551/mitpress/1427.003.0009.

Mohite, Sangita & Kulkarni, Gurudatt & Sutar, Ramesh & Polytechnic, D & Kolhapur, & Mandal, Mitra & Polytechnic, & Pune,. (2013). "RFID Security Issues". International Journal of Engineering Research & Technology. Volume 2. 746-748.

Nayeripour, Majid, et al. “Frequency Deviation Control by Coordination Control of FC and Double-Layer Capacitor in an Autonomous Hybrid Renewable Energy Power Generation System.” Renewable Energy, vol. 36, no. 6, 2011, pp. 1741–1746., doi:10.1016/j.renene.2010.12.012.

Pete. “AlphaZero Crushes Stockfish In New 1,000-Game Match.” Chess.com, Chess.com, 17 Apr. 2019.

Pomeroyb. “Pomeroyb/Create2Control.” GitHub, github.com/pomeroyb/Create2Control#create2api.

R. Simmons, J.L. Fernandez, R. Goodwin, S. Koenig, J. O'Sullivan, "Lessons learned from Xavier", *Robotics & Automation Magazine IEEE*, vol. 7, no. 2, pp. 33-39, 2000.

Sukhan Lee, Hun-sue Lee, Dong-wook Shin, "Cognitive Robotic Engine for HRI", *Intelligent Robots and Systems 2006 IEEE/RSJ International Conference on*, pp. 2601-2607, 2006.

Yan, Zhi, et al. “A Survey and Analysis of Multi-Robot Coordination.” International Journal of Advanced Robotic Systems, vol. 10, no. 12, 2013, p. 399., doi:10.5772/57313.