Course Project
WiLiE: Wireless LiDAR Explorer, a Cost-Effective Solution to Routine Network Metric Collection

Ashton Palacios, Christopher Kitras

April 21, 2022

Introduction

IT departments across the world spend countless hours debugging random WiFi issues. Many of these issues seem to only randomly manifest themselves in certain parts of a room. Our project focuses on creating a system that can accurately create WiFi signal strength heat maps. These heat maps show the highs and lows in signal strength throughout a room or building. There are many off the shelf products that create these heat maps, but maybe do not do it in the most accurate way. Or, if they do it in an accurate way they are cost prohibitive. Many companies and people do not have access to these tools. The Wireless LiDar Explorer (WiLiE) aims to create a system that is easy to use and generally uses off the shelf hardware. The current iteration of WiLiE is a remote controlled robot that a user can drive around accurately collecting WiFi metrics. These metrics can then be compiled down into heat maps that the user may use to make informed network design decisions. WiLiE is open source and can be found on GitHub[6].

Related Work

A quick internet search reveals that there are many off the shelf solutions to create heat maps. Two of the most popular are Netspot[4] and Ekahau[5]. Ekahau is the suite that BYU uses for designing, debugging, and improving their networks. There are many others like these tools. All of these tools claim to give accurate network statistics. These tools usually require a user to walk around with a computer in their hands. Then the user has to add spots of where they think they are in the building on a map. Most of the tools show only collecting one or two data points per room. Our solution included building a robot that can accurately build a map from Lidar sensor readings[1]. Included in this map, is accurately placing tens of received signal strength data points in the same room. These commercial solutions are cost prohibitive to many people. IoT devices are many times placed at the fringe of a network. There has been research done to have these IoT devices still be able to communicate despite having a bad connection[3]. Our direction is different in that we want to place these sensors in good spots or create a network that has good signal in most places. Our project is similar to the work done by the Technical University of Sofia[2]. They created a complex machine that accurately moves a software defined radio around a room collecting received signal strength data points. They were then able to create a 3D heat map of the room. This heat map allows the researchers to accurately detect high interference zones within the room. This approach, while being accurate, does not scale well. It is hard to move from room to room. Our approach differs in that to move room to room all we have to do is drive the robot.

Methodology

For this system we decided to build most things from the ground up. By creating a new robot on a familiar platform to most robot designers with commodity hardware, we invite the possibility of this project to easily be open sourced and improved. In this iteration we have left out certain aspects like a single LAN for the unit via ethernet and self-driving. Both of these factors could easily be considered for a future version.

Equipment

One main goal for this project was to make a device that would collect the necessary network metrics with commodity hardware. We accomplished this objective by using several common yet effective hardware platforms to control each of the system components. Raspberry Pi 4 Bs were used for the brains of the system (i.e. motor movement, metrics, data transmission, etc). A Teensy 4.1 was used as the main brains to collect odometry information from the motor encoders. An RPLidar A1M8 module was used to collect environmental data to create a map. A TL-SG105 was used with the intention to centralize all network communication locally. This unit was ultimately not used in the final design due to a lack of time for proper integration, but has been left in place for future iterations. Finally, a PlayStation 5 (PS5) controller was used for its bluetooth functionality to wirelessly control the movements of WiLiE.

System Components

At its core, WiLiE is divided into several system components: wheel odometry, motor control, visualization, and data collection. To have all of these components communicate effectively we used the Robot Operating System v2 (ROS2) platform which contains a publisher/subscriber framework to facilitate communication between components.

The wheel odometry component reads the odometry values from the motor encoders which was published on a specific topic used for localization with respect to a reference point. The motor control component receives values from the PS5 controller over a

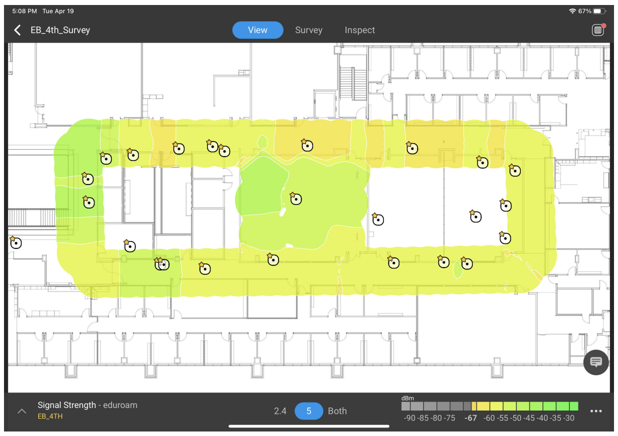


Figure 1: Ekahau heat map for the fourth floor of the Engineering Building

Bluetooth connection. These values are then passed into a node which controls the speed and direction of the motor spin. The visualization component receives values from the A1M8 and publishes it to a ROS2 node and using `rviz2` the readings are converted into a map that shows the area surrounding WiLiE and his movements. Finally, the data collection component makes use of the `/proc/network/wireless` file which reports RSSI values on any NICs (or wireless devices) attached to the Raspberry Pi.

Data Collection

Data was collected through several steps. The data collection system component was responsible for most of the gathering and then a brief processing portion to render the data readable was employed. To begin the reading process, we powered on all of the involved ROS2 nodes and then started driving WiLiE around. The metrics were matched with location by the localization node subscribing to the metrics (or `/WILIE2`) topic. We opted for ROS2 to coordinate the data collection because of our familiarity with the project but do recognize that other tools like ZeroMQ can handle the project as well.

Implementation/Experimentation

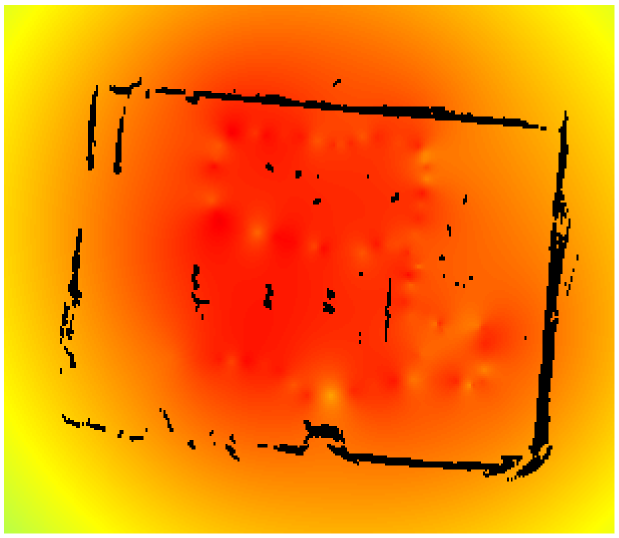
Our implementation took longer than expected, but yielded much experience in creating a functional robotic system. Despite these set backs, we feel we were able to adequately prove the utility of a system like WiLiE in many budget IT departments. Our suite of tests aims to prove the functionality of such a platform and also how it compares to a professional platform.

Implementation

Some of the hurdles we had to overcome were learning mechanical design, PCB design, hardware selection, motor encoder operation, control theory, ROS2 framework, power management, and taking networking measurements on a Raspberry Pi. The mechanical design included doing CAD work in Fusion 360 to create the chassis and connecting pieces. We needed custom PCBs for the power management and wheel encoder boards which were designed in EAGLE. We learned the theory on how quadrature encoders work and how to interface with them. On top of that we learned how to calculate wheel odometry based on the differential drive model, which our robot approximates. Control theory was used to create smooth driving and some what consistent drive speed. The power management board was designed to use low cost buck converters to regulate 12 volts down into multiple 5V buses for the various electronics. We learned the ROS2 framework, as stated in the methodology section, to communicate between the various computers involved with the WiLiE platform. There were many pitfalls that we encountered with all of these aspects. Nevertheless, all of these complex moving parts worked together to create a robot that can collect various WiFi metrics while mapping and localizing its location.

Experimentation

There were two parts to our experimentation process: gathering ground truth data from the Ekahau Sidekick and running WiLiE to gather the experimental data. While we were expecting to run the Ekahau and WiLiE site surveys in the same location, we came


(a) RSSI measured from external antenna

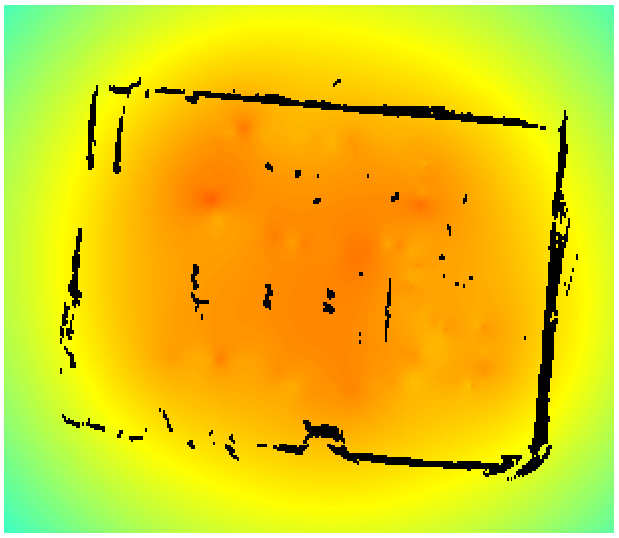

(b) RSSI measured from onboard antenna

Figure 2: RSSI heat maps generated by WiLiE

across several issues which prevented us from doing so. Regardless, we believe that when comparing the data collected by the Ekahau as opposed to WiLiE, the differences in data quality and methods are apparent enough and appropriate for the scope of this course project. A more rigorous testing in the same location by both platforms is required for any definitive conclusions can be reached.

As seen in Figure 1, we surveyed the halls and some labs on the fourth floor of the Engineering Building. We were constrained to this floor due to CAD model availibility, which has proved to be an unfortunate piece of overhead to using such a system. While the platform allows for measuring points, lines, or in an AR assisted manner, the results in Figure 1 were collected using the first two. It is apparent that care is taken in measuring RSSI by the plaform and that mathematical abstractions are employed to provide different boundaries between discovered APs. It is also apparent that the platform has the possibility of picking up duplicate APs (i.e. when two symbols are significantly close to one another or overlapping).

The experience of using WiLiE differed greatly from using the Ekahau. In this preliminary design, WiLiE reads the RSSI values from APs that it is already connected to as opposed to being in monitor mode like the Ekahau does. This is an obviously drawback of design but was decided upon due to time constraints but is a recommended step for any future investigations. We originally brought WiLiE and a separate router to the fourth floor of the Engineering Building to account for these constraints. However, we came to the quick realization that the router was faulty and we could not get WiLiE to connect reliably to it or an open networks. Defeated, we brought the router back to our lab and conducted the experiment there. This could potentially be mitigated by routing all information between components with a switch as noted in the previous section.

After turning on all of the nodes, we were able to drive WiLiE around the lab and collect RSSI data. By using a combination of matching the metric points up with locations points, fixing the scaling, and creating a heatmap, we were able to get more detailed renderings as visible in Figure 2 where (a) shows readings from an external NIC plugged into the Pi, (b) shows the onboard antenna, and (c) provides a map of the environment with a path of the robot. Most of this was manually computed and compiled, however, with more time, these individual processes can be pipelined in an automation script and made seamless.

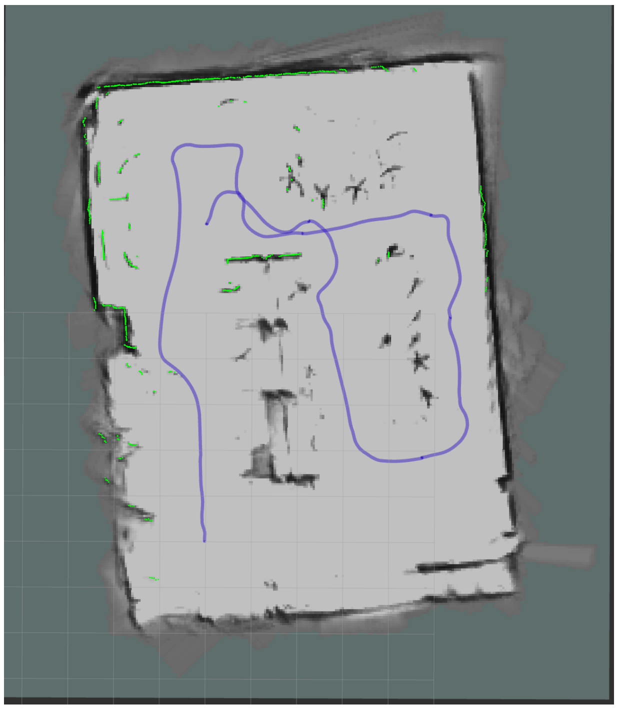


Figure 3: Visualization of room from WiLiE LiDAR and its path.

Conclusion

Our WiLiE platform was able to collect and make WiFi heat maps. There were quite a few hiccups along the way. Designing and making the all electronics fit within out form factor was a challenge. Having the EE shop mill our PCBs lead to many of the PCBs being re-milled. Getting the ROS2 nodes to play nice with each other was a pain. On the flip side, we are able to control and know where the robot is in space. We are able to collect WiFi metrics to within about 50cm accuracy. We are able to see that there are micro-pockets where the WiFi signal is weaker than the surrounding areas. We learned that there could be real research done in this area.

Currently available commercial software and hardware to perform the same task are prohibitively expensive to many people. We recognize that a robot driving around collecting WiFi metrics is not the perfect solution to this problem, but it is a start to making these tools more accessible to the common lay person. In the future we would like to redefine the WiLiE project. We would like to find the niche area that WiLie provides meaningful insight. Some of those areas may include robotic localization maps, more informed dynamic WiFi networks, open source WiFi measuring software and hardware, and channel secrecy capacity measuring tools. The heatmaps and other metrics that WiLiE can currently provide are insightful, but there is still work to be done.

References

[1] Wolfgang Hess and et al. "Real-time loop closure in 2D LIDAR SLAM". In: *2016 IEEE International Conference on Robotics and Automation (ICRA)* (2016). DOI: `10.1109/icra.2016.7487258`.

[2] Antoni Ivanov and et al. "3D interference mapping for indoor IOT scenarios". In: *2020 43rd International Conference on Telecommunications and Signal Processing (TSP)* (2020). DOI: `10.1109/tsp49548.2020.9163556`.

[3] Philip Lundrigan and et al. "On-off Noise Power Communication". In: *The 25th Annual International Conference on Mobile Computing and Networking* (2019). DOI: `10.1145/3300061.3345436`.

[4] NetSpot. *Wi-Fi Site Surveys, Analysis, Troubleshooting.* `https://www.netspotapp.com/`. Jan. 2022.

[5] *Wi-Fi Heatmap software - visualize coverage and capacity.* `https://www.ekahau.com/solutions/wi-fi-heatmaps/`. Dec. 2020.

[6] *WiLiE*. 2022. URL: `https://github.com/NET-BYU/WiLiE.git`.