Indoor Wireless Beacon Tracking Using ESP32

Gautam Singh, G.V.V. Sharma

Indian Institute of Technology Hyderabad, India

Email: cs21btech11018@iith.ac.in, gadepall@ee.iith.ac.in

***Abstract*—This document is a report which demonstrates the use of machine learning in beacon tracking using an unmanned ground vehicle (UGV) and a WiFi-enabled microcontroller such as the ESP32.**

1 Introduction

Positioning, localization and navigation (PLAN) technology has become an essential capability for consumer electronics and autonomous vehicles alike. This fast growing market has seen the entry of big tech giants such as Amazon, Apple and Google. Use cases of PLAN include safety, medical applications and assistance for the elderly and the specially abled. Reliable PLAN technology can reduce road accidents, congestion, energy and time consumption. However, many existing PLAN technologies cater to industrial use cases and are expensive, complex and require hardware that might be scarcely available to consumers.

To overcome the above challenges, we propose a simple, effective and robust algorithm for indoor beacon tracking using a single WiFi or BLE beacon and a low-cost easily available microcontroller such as the ESP32 that is suitable for consumer electronics such as robot vacuum cleaners. The main contributions of this paper are as follows.

1) A simple yet robust algorithm for indoor beacon tracking using only RSSI measurements from a single target beacon.
2) A practical implementation of the algorithm on an ESP32 microcontroller mounted on a UGV chassis using open-source frameworks such as PlatformIO.

The remainder of this paper is organized as follows. In Section 2, we present a brief survey of related work in the field of indoor beacon tracking. In Section 3, we formulate the optimization problem corresponding to this task and describe our approach. In Section 4, we present the implementation details of the algorithm on the ESP32 microcontroller. In Section 5, we present the experimental results and performance evaluation of our proposed approach. Finally, in Section 6, we conclude the paper and discuss possible extensions of our work.

2 Related Work

A comprehensive survey of PLAN technology presented in [1] categorizes PLAN technologies based on the primary sensors used in navigation and illustrates the trade-offs between accuracy, cost and complexity. Some of these technologies are briefly described below.

The authors of [2] present a fuzzy controller system which consists of three ultrasonic sensors and one camera to detect obstacles in the robot's vicinity. The navigation logic is implemented on BLE tag modules using the angle of arrival (AOA) method and is suitable for unmanned nursing in hospitals, especially during pandemic outbreaks. The authors of [3] propose iDROP, a 3D localization scheme for drones in indoor GPS-denied environments that is robust against noise and multipath fading and provides location estimation with high accuracy. The authors of [4] present a UGV implemented on the robot operating system (ROS) that uses light detection and ranging (LiDAR) for accurate navigation even in dynamic conditions and is suitable for various indoor robotic use cases such as in warehouses or construction sites. The authors of [5] present a scalable navigation system using multiple ESP32 microcontrollers acting as BLE beacons. A neural network models the position of the robot based on the signal strength received from the various beacons and the Levenberg-Marquardt method is used to accurately estimate the position of the robot. The authors of [6] present a strategy for landing a UAV using a radio beacon that exploits a special structure occurring when approaching the target beacon from above to reduce the flight time required for landing near the beacon.

Methods based on artificial intelligence (AI) have gained popularity in recent literature. The authors of [7] propose an easy to implement method for multi-object tracking (MOT) using an encoder-decoder mechanism. Their proposed method achieves comparable performance with state-of-the-art frameworks on the challenging MOT dataset. The authors of [8] leverage deep reinforcement learning (RL) to train an agent for tracking dynamically moving objects using unmanned surface vehicles (USV) in marine environments. The trained RL agent performs comparably to the analytically derived trajectory in the steady state. The authors of [9] explore multiple RL algorithms using LiDAR inputs on TurtleBot3, showing that the twin delayed deep deterministic policy gradient (TD3) is the most efficient and robust across varied environments.

Many of the above methods are either too complex or too expensive to be implemented on cheap hardware such as an ESP32 microcontroller. Furthermore, a lot of supporting infrastructure needs to be set up to enable autonomous navigation. Our work aims to address these challenges by providing a lightweight and cost-effective solution for indoor beacon

tracking using a single wireless beacon and easily available wireless-capable microcontrollers such as the ESP32. It builds upon the algorithm proposed in [10].

3 Proposed Navigation System

In this section, we formulate the beacon tracking problem as an optimization problem. Then, we describe our algorithm for solving this optimization problem using gradient ascent and compare it to a similar algorithm proposed in [10].

3.1 Problem Statement

To estimate (radial) distance to beacon, we use its signal strength which is usually given by the Received Signal Strength Indicator (RSSI). The RSSI (in dBm) at radial distance of r metres is defined as

$$R(r) = R(1) - 10\log_{10}(r), \tag{1}$$

where $R(1)$ is the RSSI at a distance of 1 metre from the beacon. Using this definition, the beacon tracking problem can be formulated as the following optimization problem.

$$\max_r R(r) \text{ s.t. } r > 0. \tag{2}$$

The derivative and second derivative of $R(r)$ is

$$R'(r) = -\frac{10}{\ln 10}\frac{1}{r}, \tag{3}$$

$$R''(r) = \frac{10}{\ln 10}\frac{1}{r^2} > 0. \tag{4}$$

Notice that for $r > 0$, $R'(r) < 0$, thus $R(r)$ is a decreasing function of r. This implies that the maximum RSSI is at $r = 0$, as expected. Since $R(r)$ is a convex function of r, we can use gradient ascent [11] to recursively find the point where the RSSI is maximum, which would correspond to the location of the beacon. Using (3), the gradient ascent update equation for the radial distance r is given by

$$r_{n+1} = r_n + \alpha R'(r_n) = r_n - \frac{10\alpha}{r_n \ln 10} \tag{5}$$

where r_i is the radial distance at the i-th step and α is the step size and r_0 is the initial radial distance of the UGV. Since $r_n > 0$, we can see that $r_{n+1} < r_n$ for all n. In other words, the radial distance to the beacon is decreasing with each step. This means that the UGV will converge towards the beacon. However, due to the explosion of the gradient in (3) at $r = 0$, the update steps become larger as r_n decreases, which could lead to overshooting the beacon. We now compare two algorithms that overcome this limitation by using a recursive approach that keeps the step size constant.

3.2 Beacon Tracking Algorithm

The proposed beacon tracking algorithm broadly be divided into two stages at each step. The first stage is the *probing stage*, where the UGV measures the RSSI at various points in its vicinity. The second stage is the *decision stage*, where the UGV decides its direction of movement based on the RSSI measurements.

The algorithm proposed in [10] is described in Algorithm 1. The probing phase consists of taking RSSI measurements in place but at various orientations. The decision phase consists of moving in the direction where the signal is the strongest. A major drawback of this approach is that there may not be appreciable difference in the measured RSSI values at the various orientations in the same place, especially if the beacon is far away. This could lead to the UGV making arbitrary moves instead of converging towards the beacon.

Algorithm 1 Beacon Tracking Algorithm of [10].

Input: RSSI threshold T, number of steps N

while GetRSSI() $< T$ **do**
  $r_F \leftarrow$ GetRSSI().
  Turn left by 90 degrees in place.
  $r_L \leftarrow$ GetRSSI().
  Turn right (from initial orientation) by 90 degrees in place.
  $r_R \leftarrow$ GetRSSI().
  if $r_F \geq r_L$ and $r_F \geq r_R$ **then**
    Turn to the initial direction in place.
  else if $r_L \geq r_F$ and $r_L \geq r_R$ **then**
    Turn to the left direction in place.
  else$r_R \geq r_F$ and $r_R \geq r_L$
    Turn to the right direction in place.
  Move forward in the chosen direction by one step.

To overcome this challenge, we require the UGV to travel a significant distance in the probing stage to obtain some differences in the measured RSSI values. Thus, in the probing stage of our proposed algorithm, the UGV takes N steps in a straight line in its current orientation, measuring the RSSI at each step. In the decision stage, the UGV then moves to the position where the maximum RSSI was measured. If this is at the beginning or end of the probing trajectory, the UGV moves backward or forward respectively. Otherwise, it turns perpendicular to its probing trajectory at that point (either left or right). Our proposed algorithm is described in Algorithm 2. The main shortcoming of this particular algorithm is the amount of movement required in the probing stage for making one simple move in the decision stage.

Algorithm 2 Proposed Beacon Tracking Algorithm

Input: RSSI threshold T, number of steps N

while GetRSSI() $< T$ **do**
  for i = 0 to N **do**
    $r_i \leftarrow$ GetRSSI().
    Move forward by one step.
  $j \leftarrow \arg\max_i r_i$.
  Move to the position at step i.
  if $i = N$ **then**
    Move one step forward.
  else if $i = 0$ **then**
    Move one step backward.
  else
    Turn left.

4 Implementation

The code for the beacon tracking algorithm is implemented in C++ using the PlatformIO framework. They are openly available on GitHub.

4.1 Assets

1) UGV chassis with DC motors
2) ESP32 microcontroller with Type-B USB cable
3) L293D Motor Driver IC
4) Breadboard and Jumper Wires
5) Android phone
6) (Optional) USB 2.0/3.0 Hub

4.2 Procedure

1) Make the connections as per the wiring diagram in Figure 1.
2) Connect the ESP32 board to your Android Phone.
3) Generate the firmware by entering the following commands.

```
$ cd codes
$ pio run
```

4) Go to ArduinoDroid and select

```
Actions → Upload → Upload Precompiled
```

and choose the firmware file at

```
codes/.pio/build/firmware.hex
```

5) Now put the phone at a reasonable distance from the UGV with no obstacles in the way and then turn on the hotspot. The UGV should travel towards the phone and stop near it.

5 Results

The UGV eventually converges close to the beacon (here, the hotspot). However, if there are a lot of nearby obstacles, the UGV may not converge close to the location of the beacon. It may either get physically blocked by the beacon or the signal interference may be too high.

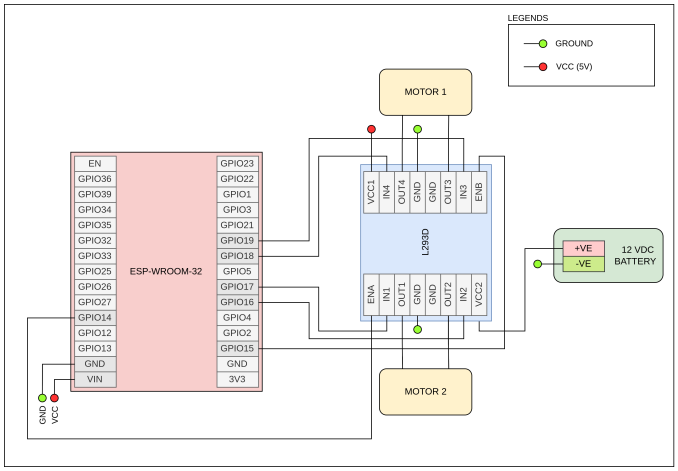

Fig. 1: Wiring Diagram for Beacon Tracking.

6 Conclusion and Future Work

In this paper, we presented a cost-effective and robust beacon tracking algorithm that uses RSSI measurements to approach the beacon at each step. Our work uses easily available hardware and software tools to implement the algorithm on an ESP32 microcontroller mounted on a UGV chassis. A further extension to this work could be to use additional sensors such as a gyroscope or an accelerometer to improve the accuracy of the UGV's position and prevent deviations from the intended path of the UGV.

References

[1] N. El-Sheimy and Y. Li, "Indoor navigation: State of the art and future trends," *Satellite Navigation*, vol. 2, no. 1, p. 7, May 2021.

[2] K. L. Narayanan, R. S. Krishnan, L. H. Son, N. T. Tung, E. G. Julie, Y. H. Robinson, R. Kumar, and V. C. Gerogiannis, "Fuzzy Guided Autonomous Nursing Robot through Wireless Beacon Network," *Multimedia Tools and Applications*, vol. 81, no. 3, pp. 3297–3325, Jan. 2022.

[3] A. Famili, A. Stavrou, H. Wang, and J.-M. Park, "iDROP: Robust Localization for Indoor Navigation of Drones With Optimized Beacon Placement," *IEEE Internet of Things Journal*, vol. 10, no. 16, pp. 14 226–14 238, Aug. 2023.

[4] M. H T, M. L, and M. Sajjad, "ROS Powered Autonomous Mobile Robot for Indoor Applications," in *2024 International Conference on Intelligent Algorithms for Computational Intelligence Systems (IACIS)*, Aug. 2024, pp. 1–7.

[5] D. A. Yukhimets, I. E. Naidanova, and N. V. Andrukhovsky, "Local Navigation System for a Robot Based on a Network of BLE Beacons," in *2024 International Russian Automation Conference (RusAutoCon)*, Sep. 2024, pp. 611–616.

[6] N. Stefas, H. Bayram, and V. Isler, "UAV Landing at an Unknown Location Marked by a Radio Beacon," in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Nov. 2019, pp. 4409–4414.

[7] Y. Liu, E. Wang, S. Xu, Z. Wang, M. Liu, and W. Shu, "Simple Online Unmanned Aerial Vehicle Tracking with Transformer," in *2021 IEEE 20th International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom)*, Oct. 2021, pp. 1235–1239.

[8] I. Masmitja, M. Martin, K. Katija, S. Gomariz, and J. Navarro, "A reinforcement learning path planning approach for range-only underwater target localization with autonomous vehicles," in *2022 IEEE 18th International Conference on Automation Science and Engineering (CASE)*, Aug. 2022, pp. 675–682.

[9] A. K. Kashyap and K. Konathalapalli, "Autonomous navigation of ROS2 based Turtlebot3 in static and dynamic environments using intelligent approach," *International Journal of Information Technology*, Mar. 2025.
[10] S. O. Dubey, "Navigation and Communication for UGV/UAV," Master's thesis, Indian Institute of Technology, Hyderabad, Jun. 2022.
[11] S. P. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge university press, 2004.