School of Electronic Engineering

Design of a discreet building access management system using wireless indoor location tracking and fuzzy logic

Project Portfolio

Name: Aisling Lee
ID: 20216371

August 2022

MEng in Electronic and Computer Engineering

Supervised by Dr. Derek Molloy

Declaration

I hereby declare that, except where otherwise indicated, this document is entirely my own work and has not been submitted in whole or in part to any other university.

Signed:  Date: 22/02/2020

Design of a discreet building access management system using wireless indoor location tracking and fuzzy logic

Aisling Lee, Derek Molloy
aisling.lee32@mail.dcu.ie, derek.molloy@dcu.ie

***Abstract*— This paper looks at the present challenges in implementing accurate indoor location tracking within the use case of a discreet building access management system. This report covers the investigation into existing works and technologies, design and development of the physical and software architectures required to achieve this in Python, the testing covered, Analysis of results obtained and suggestion for future improvements of similar designs.**

Keywords— Indoor location tracking, UWB, Fuzzy Logic, Python, distributed systems, edge computing

1. Introduction

The internet of things has allowed for a new era of invention and integration of technology in our daily routines. We saw the advent of "smart devices" from watches, TVs, house hold utilities etc become common place in consumers lives. As the world becomes more connected it is also expected that actions become more seamless as well as providing useful data to be leveraged for decision making.
Indoor location access management has long been a concern for many agencies operating out of buildings which require access restriction or controls. Outdoor location tracking has been prevalent for centuries with its resolution improving greatly in the last few decades however, it is only recently that indoor wireless location tracking has become viable as a result of wireless technologies like Wi-Fi, Bluetooth and Ultra-Wide band (UWB).

1.1. Problem statement

Using a portable wireless technology integrated with a data analysis system to produce a discreet access management system. This problem can be broken down into 3 pillar components:

1. *Portable wireless technologies:* Analysis, selection and integration of appropriate wireless technology that can be carried by users without impeding on their routines
2. *Data analysis for decision making:* Careful analysis as to the data provided to ensure an appropriate and timely decision is made as to granting or denying access.
3. *Access provision based on a decision:* Once the decision has been made, facilitating the actuation of door access control mechanisms

2. Background

2.1. Wireless devices

The research for and selection of a wireless device type was based on technical criteria devised. The scope of technologies reviewed was determined by prevalence of (likelihood of users to already own device using that technology) and commercial availability for off the shelf. The most prominent off the shelf devices were Tile products (Bluetooth), Apple Airtags (UWB) and Mobile phones (Wi-Fi).
Other technologies ruled out of scope but worth noting include the likes of Zig-Bee which boasts 65k nodes per network and 128-bit AES, and LoRa which has impressive low battery rates and long range where there is also ongoing research into indoor positioning systems being undertaken. They were ruled out due to not yet having a device with a small footprint (pocket sized) and as readily adopted and widely used in the consumer market.

2.2. Security of Wireless devices

Consideration was given to the security profiles of each of the 3 technologies.

- Bluetooth GATT is the interface protocol for direct connection between central and peripheral devices, while this does provide a private connect further encryption of packets on the network is advised due to data potential being able to be captured via MIM (man in the middle) or sniffing attacks [1]. All current
- UWB implements a scrambled time stamp with BPSK [2] for encryption to ensure the data is secure but retained when being transmitted.
- Wi-Fi networks are capable of implementing WPA2-PSK (AES) which is at present the highest level of encryption commercially available (WPA3 is not yet fully developed) and is highly dependable

2.3. Selection Matrix

There were several criteria used to select which technology from the 3 in scope primarily they were accuracy, power efficiency, scalability, real-time achieve-ability and reliability [3]. Comparing the metrics of the devices combined with the findings as per section 2.2 showed the following results as seen in Table 1 - Wireless Protocol

Decision Matrix. In Table 1 - Wireless Protocol Decision Matrix the ranking of scalability is based on the number of devices that can be on a network before expansion and with respect to the ease of expansion should inclusion of more devices be required on a network. With regards to this UWB stands out as most scalable as it supports significantly more devices with repeatably showing capacity for over 6000 [4] before expansion of network access points is required. Decwave DWM1001 specifically has a capacity of 15 tags at 10Hz and up to 9000 at 0.01667Hz advertising rates [5]. Expansion of these networks is relatively easy by comparison to its counterparts whereas Bluetooth has shown a maximum of 14 devices [6] and Wi-Fi can support up to 255 [7] but industry recommendation is 30-40. The reliability of Wi-Fi and Bluetooth can vary due to high volume usage of the 2.4GHz and higher risk (than UWB) of interruptions/packet loss they operate in although noted Wi-Fi is expanding out into 5GHz and 60GHz presently.

Technology	Accuracy	Tx Power	Scalability	Realtime	Reliability	Security
BLE	3m	1mW	Low	Yes	Medium	Low
UWB	0.3m	0.07mW	High	Yes	High	High
Wi-Fi	10m	1W	Medium	Yes	Medium	High

Table 1 - Wireless Protocol Decision Matrix [3]

Based off of the matrix and reviewing previous case studies [8] [9] UWB emerged as the best wireless technology to proceed with.

2.4. *Indoor tracking location methods*

Methods to discern a devices location as part of an indoor positioning system/location tracker are numerous and can leverage analysis of phase, signal strength and time analysis of packets. Based off of the literature survey [3] devising location based on signal strength and packet time are the most popular and most suitable to implement within this solution. RSSI (relative signal strength index) however is not as reliable as time-based analysis thus the solution will implement packet time analysis. As part of the Decawave devices and software supplied TWR (two-way ranging) is already implemented. TWR times the packets between 2 devices, in this solution 1 of known fixed location and calculates any delay and assigns an appropriate distance value

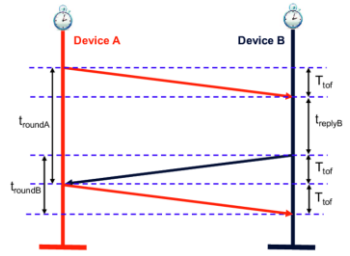

Figure 1 - Two way ranging diagram [10]

3. TECHNICAL DESCRIPTION

3.1. *Architecture*

Due to the intended application of this system, it was decided a distributed system across multiple nodes would be most suitable as well as affording the implementation of edge computing. In Figure 2 - Full E2E UWB Technical Architecture the final design to be implemented is below. The notation in this diagram correlates with Figure 10 - System Swimlane Diagram. Within this section each component and its services will be outlined.

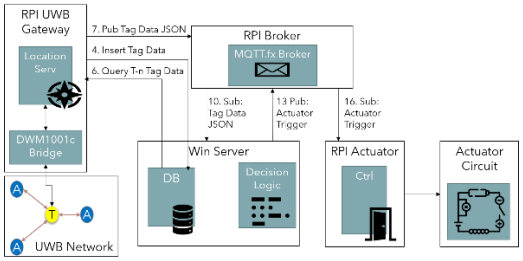

Figure 2 - Full E2E UWB Technical Architecture

3.1. *Test Environment*

The test environment consisted of 3 Decawave DWM1001c UWB devices being placed in line of sight (LoS) of the doorways and keeping the tags at roughly the same height so as to mitigate for tag positioning on the Z axis having an impact.

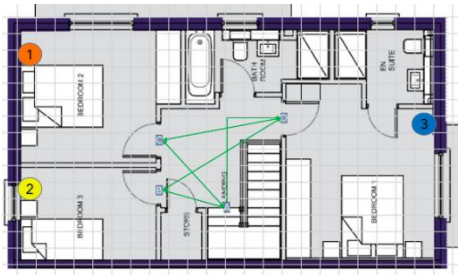

Figure 3 - Test Environment Floorplan

3.2. *Hardware*

3.2.1. *Location tracking*

The UWB anchor components were physically Installed in the configuration as outlined in 3.1. The bridge device was connected to the gateway as shown in Figure 4 - Decawave DWM1001 Gateway

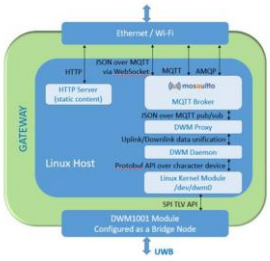

Figure 4 - Decawave DWM1001 Gateway [5]

3.2.2. *Broker*

The system communicates between all nodes via an MQTT Broker. The GUI version used was an MQTT.fx and MQTT Paho libraries installed on all nodes to enable connection, publishing and subscribing.

3.2.3. *Actuator*

The final node in the chain is the actuator node. This is a standard raspberry PI connected in this instance up to 4 LEDs which are used to show connectivity to the broker and a physical representation of potential actuation of a door (e.g.,

lock/unlock). In Figure 5 - RPI Actuator node Circuit diagram we see this, note it also included a CPU fan for the node to assist with cooling.

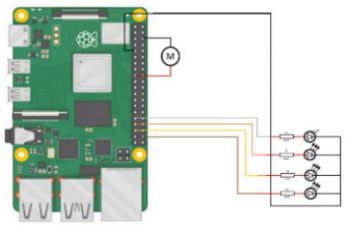

Figure 5 - RPI Actuator node Circuit diagram

3.3. Software

3.3.1. Database

The database was created using SQLite for python to store relevant data. A tag location table with columns: tag id, x co-ordinate, y co-ordinate and time stamp were created. On a movement event the data pertaining to that tag is inserted into the database for future use. In this system there is only 1 tag being tracked however recording a tag ID would allow future integration with a permissions repository such as active directory and then can be included in the decision logic. The time store is stored as an integer as seconds since the epoch as, SQLite does not have a datetime variable and doing so facilitates easier calculations later.

3.3.2. Tag Calculations

3.3.2.1. Speed

The speed of the tag is calculated using Equation 1 - Speed formula where S is the speed calculated, T is the time taken to traverse between the 2 points (x1,y1) and (xn,yn) where, x1 and y1 are the xy co-ordinates and present time and xn and yn are the historical co-ordinates at nth prior interval. In this application it was found the 5th previous interval recording in the series was old enough to capture the movement of the user and not so old as to introduce errors where the user may have been stationary or not moving toward any door way.
The D is the distance between the 2 points. The distance is derived using Equation 2 - Distance formula for 2 co-ordinates. Taking the same 2 points as above.

$$S = \frac{D}{T}$$

Equation 1 - Speed formula

$$D = \sqrt{(x1 - xn)^2 + (y1 - yn)^2}$$

Equation 2 - Distance formula for 2 co-ordinates

The Python function written *for calculating* this is as below in Code 1 - Speed calculation function. Prior to calling this function a data base query is ran to acquire the required nth interval time and location values.

```
Def speed_calc(x1_coord, y1_coord, time1, xn_coord, yn_coord, timen):
    # calc average speed = distance/time
    distance_covered = ((x1_coord - xn_coord) ** 2 + (y1_coord - yn_coord) ** 2)
** 0.5
    time_taken = time1 - timen
    tag_speed = int(distance_covered / time_taken)
    return tag_speed
```

Code 1 – Speed calculation function

3.3.2.2. Distance

The tag distance was devised between the current co-ordinates (x1,y1) and the xy co-ordinates of the centre point of the door frames and again using Equation 2 - Distance formula for 2 co-ordinates as explained in 3.3.2.1. The distance between the device and all of the doors is calculated within the function to be then passed to the Fuzzy logic control system.

```
def distance_calc(x1_coord, y1_coord, xn_coord, yn_coord):
    ## Door frame centre co-ords
    # office = (420 450)
    x_office = 420
    y_office = 450
    # Box rm = (420 590)
    x_boxrm = 420
    y_boxrm = 590
    # Bed rm = (770 400)
    x_bedrm = 770
    y_bedrm = 400

    # distance between 2 pts = sqrt[(x1-x2)^2 + (y1-y2)^2]
    distance_office = int(((x1_coord - x_office) ** 2 + (y1_coord - y_office) **
2) ** 0.5)
    distance_boxrm = int(((x1_coord - x_boxrm) ** 2 + (y1_coord - y_boxrm) ** 2)
** 0.5)
    distance_bedrm = int(((x1_coord - x_bedrm) ** 2 + (y1_coord - y_bedrm) ** 2)
** 0.5)

    print("office ", distance_office, " boxrm ", distance_boxrm, " bedrm ",
distance_bedrm)
    return distance_office, distance_boxrm, distance_bedrm
```

Code 2 - Distance Calculation function

3.4. Decision logic

3.4.1. Fuzzy logic

The decision to select Fuzzy logic over other machine learning models or AI decision systems such as Bayesian logic or neural networks was based on there being no need for training data sets but rather defining the logical rules and the ranges of the categories under which the antecedents (inputs) would fall under. The system is based on rules which govern over defined ranges which can be analysed in a qualative or quantitive nature.

3.4.2. Fuzzy Logic control system

The fuzzy logic is implemented by creating sets in which the antecedents may fall and a set similarly for the consequent. These require a name, a range (a minimum and maximum value) and the resolution (increment steps within a given range). For this solution 3 sets were defined, speed and distance as the antecedents and intent as the consequent The values derived for these sets were taken from maximum width in the test environment and initial data capture to find the maximal speed achievable. The set values defined can be seen in Code 3 - Fuzzy Set Values and Figure 6 - Fuzzy Logic sets shows the graphing of these sets

```
speed = ctrl.Antecedent(np.arange(0, 70, 1), 'speed')  #
range from 0 to 70 in 1 unit increments
distance = ctrl.Antecedent(np.arange(0, 800, 10),
'distance')  # range from 0 to 800 in 10 unit increments
intent = ctrl.Consequent(np.arange(0, 10, 1), 'intent')
# range from 0 to 10 in 1 unit increments
```

Code 3 - Fuzzy Set Values

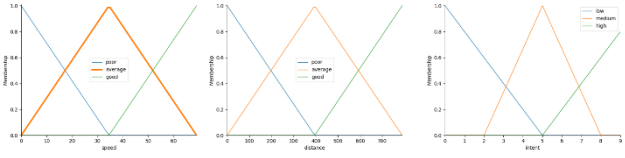

Figure 6 - Fuzzy Logic sets

3.4.2.1. Intent membership fucntions

The output will belong to one of the subsets of the consequent function which must be defined by the system designer. These are provided here in Code 4 - Intent Consequent Membership Function. A category name, range and apex value for each of the 3 sub sets is defined and attributed to the intent object.

```
intent['low'] = fuzz.trimf(intent.universe, [0, 0, 5])
# triangle pts abc negative
intent['medium'] = fuzz.trimf(intent.universe, [2, 5, 8])
# triangle pts abc inconclusive
intent['high'] = fuzz.trimf(intent.universe, [5, 10, 10])
# triangle pts abc affirmative
```

Code 4 - Intent Consequent Membership Function

3.4.2.2. Rules

The rules system required within Fuzzy logic contains the antecedent(s) set(s) with a qualifier for the data that will be input and what sub set of the consequent it should fall into. In Code 5 - Fuzzy Rule set we see the rules which were implemented in the final system. However, there was testing done with up to 7 rules but was refined down to 3.

```
# Define rules under which to consider
rule1 = ctrl.Rule(speed['poor'] | distance['poor'], intent['low'])
rule2 = ctrl.Rule(distance['average'] & speed['average'], intent['medium'])
rule3 = ctrl.Rule(distance['good'] & speed['good'], intent['high'])
```

Code 5 - Fuzzy Rule set

3.4.2.3. Calculating Intent

The consequents is calculated by identifying where the received values fall within the subsets (see Figure 7 - Triangular Set Calculation), and giving then a weighting based on the components outlined in 3.4.2 and combing them to then be used in identifying where the value lies on the consequent function and its subsets.

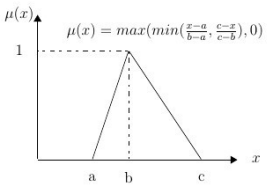

Figure 7 - Triangular Set Calculation [11]

The representation of this in the decisions logic code can be seen in Code 6 - Computing intent value function. Once the data generated from the preceding system parts is passed into the function it performs the set analysis of the data to return a value which in this system is used as a probability of intent to use a doorway.

```
def fuzzylogic_funct(tag_speed, distance, door):
    # Function to calculate the intent value upon which decisions will be made
    # Pass inputs to the Control System using Antecedent labels with Pythonic API
    intention.input['speed'] = tag_speed
    intention.input['distance'] = distance
    intention.compute()
    probability_intention = intention.output['intent']
    print(" Door - ", door, " intention - ", probability_intention)
    intent.view(sim=intention)
    return probability_intention
```

Code 6 - Computing intent value function

3.4.2.4. Analysing data and Decision making

Once the intent consequent value had been generated for each door way this was then checked against the threshold value (~80% likely hood of intent to use door) and then compared against the other 2 doorways in the system to see if it is the emerging preference. If the value passes these logical tests the decision logic program will publish a message to the broker to trigger actuation of the appropriate doorway. This can be seen in Code 7 - Fuzzy logic Actuation trigger function

```
def actuator_function(office_prob, boxrm_prob, bedrm_prob):
    # Function to trigger actuation based off of data based decision
    # If intent value exceeds threshold and is higher than that of the
    # Other 2 rooms publish to corresponding topic a 1 for actuation to
    # occur and a second later publish 0 to trigger deactivation

    if (office_prob >= threshold) & (office_prob > boxrm_prob) & (office_prob > bedrm_prob):
        client.publish("actuator_topic/act1", payload="1", qos=1)
        time.sleep(1)
        client.publish("actuator_topic/act1", payload="0", qos=1)

    if (boxrm_prob >= threshold) & (boxrm_prob > office_prob) & (boxrm_prob > bedrm_prob):
        client.publish("actuator_topic/act2", payload="1", qos=1)
        time.sleep(1)
        client.publish("actuator_topic/act2", payload="0", qos=1)

    if (bedrm_prob >= threshold) & (bedrm_prob > office_prob) & (bedrm_prob > boxrm_prob):
        client.publish("actuator_topic/act3", payload="1", qos=1)
        time.sleep(1)
        client.publish("actuator_topic/act3", payload="0", qos=1)
```

Code 7 - Fuzzy logic Actuation trigger function

3.5. Full E2E architecture

Due to the issues experienced with collecting and transmitting tag data, the simulator was designed (Figure 8 - RTLS Simulator) to substitute the data generation that would have taken place using the UWB tag and anchors. The simulator connects into the system replacing the Decawave network and software on the gateway RPI. The simulator operates based on similar principals of posting tag data (co-ordinates and time stamps) so that the rest of the solution would be valid for.

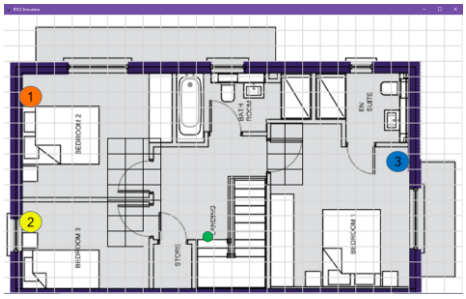

Figure 8 - RTLS Simulator

The final design as a result, for this investigation can be seen in Figure 9 - Full E2E Simulator Technical Architecture. The design is that of a distributed system to facilitate edge computing and better distribution of services from a processing perspective.

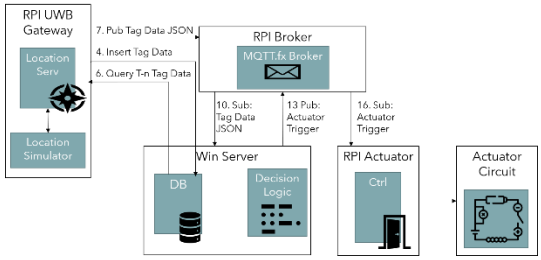

Figure 9 - Full E2E Simulator Technical Architecture

The full end to end (E2E) process is also depicted in Figure 10 - System Swimlane Diagram where you can see the systems functionality and operation in its entirety from a tag moving event to actuation should an affirmative decision be made again; the simulator replaces the tag devices and Decawave software used to determine xy co-ordinates but does not alter the flow of the system.

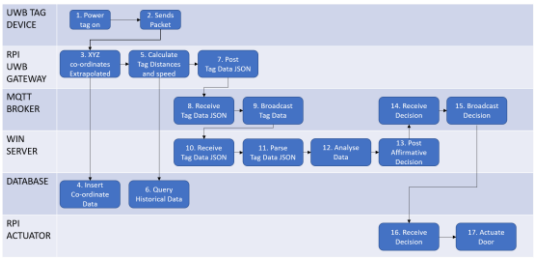

Figure 10 - System Swimlane Diagram

4. RESULTS

4.1. Isolated Testing

4.1.1. Decawave network

Initially the network showed promise once established. Confirmation of operability of the network was done using Bluetooth connection however it was noted there was a significant delay between movement and the tag data updating in the application. Small performance improvement was achieved when going to the floor below and tracking through the ground floor. The filtering qualities of the internal building materials improved the response.

Unfortunately, there was significant compatibility issues with the bridge node and moving onto testing of the UWB component was not achievable. As a result, a simulator was built which would allow generation of new data in real time as well as injecting data gathered from the Bluetooth testing.

4.1.2. Simulator

The simulator was developed over an iterative process first having an accurate representation of a tag device moving across a given space. It was decided a visual representation of this which a user-controlled object on the screen would be the closest representation and allow a more seamless integration into the overall system design. Testing of this included verifying that the object could be controlled and tracked within the bounds of a given area/canvas with a visual representation. It also required establishing new measurements of the door frame locations within the canvas image as the scale is not a 1:1 of the physical test environment. As a result, the distances units are not measures in meters or centimetres but rather pixels.

4.1.3. Actuator

The actuator node was the simplest of items to test as the complexity was quite low. All it required was validating the wiring and LED circuitry to the PI and ensuring the correct libraries were installed to interact with the GPIO pins

4.1.4. Decision logic

When testing the logic and libraries used first step was to verify configurations and packages required were present this first step was crucial as there were a number of packages needed. Once that was done running a sample solution to verify the library's worked locally was able to produce same inputs as those in the examples. From there the modifications were made to produce the code presented in section 3.4.2. It was during the initial testing that it was identified that the distance values needed to be inverted as large values were treated with positive outcomes.

Rule	Speed	Distance	Intent	Decision
1	5	20	1.965	Negative
2	30	400	4.628	Inconclusive
3	70	750	7.667	Affirmative

Table 2 - Fuzzy logic Rule Testing

Injecting the above sample data set in Table 2 - Fuzzy logic Rule Testing into the code produces the following results. Figure 11 - Fuzzy logic isolated rules testing outcomes shows graphs for each test case. For rule 1 an expected negative outcome as the speed is low and inverted distance is small (both considered poor) and the resultant intent value generated falls int the low likely hood sector. Rule 2 testing uses middle values for distance and speed and shows an inconclusive result. Lastly, we see rule 3 testing produces an affirmative result albeit slightly lower than anticipated.

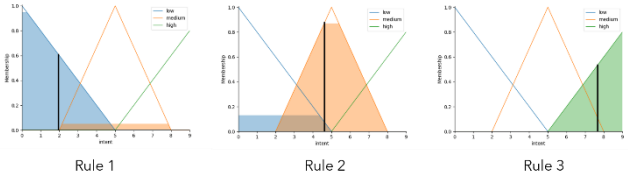

Figure 11 - Fuzzy logic isolated rules testing outcomes

It was noted that with values injected where a predicted value would produce a value the system would be slightly under e.g., rule 2 would expect to return a 5 (50%) chance but produces a 4.6 rule 3 test would have expected a higher degree of certainty since it is at max speed and only 50 pixels away from the max distance point. The data injected as per testing rule 3 was done with max speed and distance and produced a similar result to 2 decimal places. This is worth noting that in this solution under valuing can ensure eventual actuation is not done so unless the system is more certain it would require further investigation and refinement as there will be a degree of error on cases that fall just under the threshold while being legitimate cases a door should actuate for.

4.2. Integrated technical and functional testing

Once isolated testing had been complete and validated the operability of each component integrating the functionality of each was the next task.

4.2.1.1. Component – Broker

First test case was to verify each individual component was able to publish and subscribe to the broker to ensure message handling between components would be successful. This was completed by monitoring the broker via the GUI and having each component publish to it as well as manually publishing messages to the topics each was subscribed to.
It was noted early on here that when posting to the broker from the simulator based on key press events that rapid pressing/holding a key would destabilase the connection. This affect was present throughout the rest of testing.

4.2.1.2. Gateway – Broker – Actuator

Once broker comms was stablished the next was verifying that based on an event occurring in the simulator triggering an LED output in the actuator. This started out as based on a

key press event to turn all on to then using the distance values and speed to turn on and off individual LEDs

4.2.1.3. Gateway – Broker – Algorithm

Once confirmed that data generated from one node could be received and used to actuate the circuitry on the RPI actuator node the next step was integrating the simulators generated data with the decision logic. Again, we saw performance issues with throttling and this then required the decision logic to be refactored and as the maximum values emitted were not inline (2 units lower on average) with those achieved in the isolated testing. After being able to generate results by way of intent values the decision triggers were then incorporated to post to the broker, results were again observer through the GUI to validate.

4.2.1.4. Gateway – Broker – Algorithm – actuator

After confirming the functionality of various nodes individually and in controlled integration the last step was to fully integrate. Demonstration video EE580_AislingLee_E2E_FullProjectDemo.mp4 [12] shows the full integration. It was at this last stage the persistent issue of broker performance became more prominent and destabilised the solution, this of course was due to the simulator and decision logic both posting data and all 3 nodes subscribing to the broker. As a result a decisions to move to time rather than event based publishing alleviated this problem however introduce a delay into the system.

5. Analysis

5.1. Accuracy of Results

The decisions made by the system were accurate and aligned with the intent of the user when based off of the decision logic accompanied with further refining logic. It was noted some edge cases where incorrect decisions were made but was ~1%. While the decisions the system made and actuation were valid the timing was not acceptable for a real time system. In larger areas the tolerance/time to make a decision may not have shared the same issue system throughput capacity was the primary weakness.

5.2. Dependability and Suitability of Solution

Initially the designed logic was to intended to trigger a door actuation if it was the emerging preference (of the 3 doors analysed) and above a set threshold for intent value (8.0 or 80%). While in isolated testing the tuning implemented on the fuzzy control system reliably provided the above result the performance limitations of the broker message handling system meant that while posting on an event basis, we were unable to move at higher speeds due to the higher publishing rate throttling and destabilizing the broker. This produced a behaviour where the broker would rapidly disconnect and reconnect to the offending node and at times causing it to crash. As a result, it was elected to publish on a time interval of once every 500 milliseconds. It was identified any more frequent that that triggered the above failures. This posed issues as (see Table 3 - Common area traversal data) From one point to another only 3 to 6 data capturing events could be taken in any given route. As a result, the initial tuning implemented which sought an intent value of 8 (80%) from a range of 1 to 10 or higher required was lowered to a 5 as most data returned was between 1.6 and 6. The value of 5 being selected as it represented an 83%. Under the conditions the system controlled this was sufficient for correct decisions to be made albeit the timing between decision and actuation was not adequate.

Route Taken	Average Time	Shortest Distance
A - B	1.66s	170cm
A - C	1.97s	240cm
A - D	2.71s	323cm
B - D	3.02s	312cm
C - D	2.51s	351cm

Table 3 - Common area traversal data

5.3. Suggested modification

Due to the aforementioned nature of the architecture and rate limiting that was required the delay between the user's actions indicating intent to enter a room and the triggering of access actuation meant that often the user would have had a 3+ second wait in which they would have already passed through the threshold of the door way. Increasing the broker pub/sub rate would help with this.

Adding another layer of complexity to the decision logic such as weighting towards a doorway if the user is already in the room as it was note when in one room there was a small margin of error that the door for the one next to be temporarily triggered this would ensure a greater degree of robustness against incorrect door selection as in common areas the correct doors were actuated in line with the user's intent.

Regarding the consequent categories, for this implementation it could be refined to only having 2 a yes or no versus low medium and high however further expansion on intent and pattern analysis could be used in the likes of an unsupervised system to correlate the inconclusive results to further predict if a person may be intending to use a door but account for use cases where they may have stopped/slowed due to a number of reasons.

The membership functions could be modified to be quantitive rather than qualitive which was used in this investigation. The rules implemented could be further expanded on. Initially the design was to include more rules but during testing it was seen that the values never reached the 80% (on the scale of 1 to 10) and initially reducing the 7 rules to 3 did slightly improve the magnitude of intent result values however it is now known system throughput had the greater impact and thus revisiting rule design to account for more scenarios.

Regarding the UWB device used leveraging opensource material would provide greater wealth of materials to access especially with regards to trouble shooting.

6. Conclusion

In reflection there were 3 main lines of inquiry to pursue within this research Portable wireless technologies, Data analysis for decision making and, Access provision based on a decision. As outlined the latter 2 were achieved and

suggestions made as to refine and improve their implementations. The research and initial testing of the wireless devices was achieved however integration into the wider architecture was not fully realised due to hardware compatibility issues which were unresolvable during the duration of this study.

As mentioned previously the IoT and advances in wireless communications has provided an opportunity to implement new systems previously unattainable. There is a significant gap in the market for indoor location tracking systems. This investigation contributes to this area of research in:

- Integrating location tracking with an AI data analysis algorithm
- Implementing it over a distributed system with edge processing
- Demonstrating the combined use of indoor location tracking and the decision logic in a real-world application with highlighting of further areas of development and integration open to it

In summary, while the final system design was not as performative on a real-time basis due to component limitations the architecture implemented and the decisions which arose were successful and most definitely warrant further reinvestigation for future refinement

7. Appendices

Attached to this portfolio are the following support documentation: A. Literary Review and its oral presentation B. Design plan, C. Risk assessment, D. Project development log, E. Research log, F. Source code, G. Demonstration Video, H. Test data, and I. Images collated from testing and configuration.

All of the afore mentioned appendix materials are located here: https://github.com/ashification/ee580_meng.git

8. References

[1] Bluetooth SIG, “GATT Specification Supplement,” Bluetooth SIG, 2022.

[2] Lite Point, “Ultra Wideband Testing: Practical Introduction to UWB Measurements,” Lite Point, 2020.

[3] A. Lee, “Indoor wireless location tracking for use in a smart building access management system,” 2022.

[4] M. Ridolfi, S. V. d. Velde, H. Steendam and E. D. Poorter, “Analysis of the Scalability of UWB Indoor Localization Solutions for High User Densities,” *Sensors*, vol. 18, no. 6, p. 1875, 2018.

[5] Decawave, “DWM1001 System Overview and performance V2.0,” Decawave, 2018.

[6] E. H. Reich, B. Ghita, M. Wagner and J. Schäfer, “Performance Evaluation of Bluetooth in a Wireless Body Area Network for Practical Applications,” 2020 IEEE 11th Sensor Array and Multichannel Signal Processing Workshop (SAM), 2020.

[7] J. Co, G. Duran and C. Sabate, “Raspberry Pi 2 Platform for Coinoperated WiFi HotSpot Kiosk,” Computer Engineering Department, Eastern Samar State University, 2016.

[8] X. Zhao, Z. Xiao, A. Markham, N. Trigoni and Y. Ren, “Does BTLE measure up against WiFi? A comparison of indoor location performance,” European Wireless 2014, 2014.

[9] M. D. S. M. L. M. M. P. a. A. A. A. Martinelli, “Ultra wide Band Positioning in Sport: How the Relative Height Between the Transmitting and the Receiving Antenna Affects the System Performance,” *International Journal of Wireless Information Networks*, vol. 27, p. 18–29, 2020.

[10] C. Lian Sang, A. Michael, H. Timm, H. Marc and P. Mario, “Numerical and Experimental Evaluation of Error Estimation for Two-Way Ranging Methods,” *Sensors*, vol. 19.

[11] C. Gafa, “Fuzzy Inference System implementation in Python,” Towards Data Science, 2020. [Online]. Available: https://towardsdatascience.com/fuzzy-inference-system-implementation-in-python-8af88d1f0a6e.

[12] A. Lee, “EE580_AislingLee_E2E_FullProjectDemo.mp4,” [Online]. Available: https://github.com/ashification/ee580_meng/blob/master/Appendix%20Materials/EE580_AislingLee_E2E_FullProjectDemo.mp4.