Text

Description automatically generated

SCSE22-0215

**Occupancy Detection Using IR-UWB**

Venkat Subramanian

U1921075D

Supervisor: Associate Professor Luo Jun

# Abstract

Academic libraries have undergone renovations to improve the comfort and user experience of their users. During peak times like midterms and finals, libraries tend to be crowded with students seeking a quiet and productive place to work. Students tend to wander around the library to find a study spot which can be time-consuming. In this project, we present a real-time seat occupancy detection system using impulse radar and frequency-modulated continuous wave.

We used IR-UWB as the radar technology and applied Constant False Alarm Rate (CFAR) and Hidden Markov Model (HMM) for target detection and prediction of seat occupancy state. The system includes a graphical user interface (GUI) application that allows users to select the orientation of the seats and input their coordinates for analysis. The system is able to process the radar signal, perform clutter suppression, and predict the occupancy state of each seat in close to real time. This system can be utilized in various settings such as classrooms, lecture halls, and public transportation to provide real-time seat occupancy information to users.

# Acknowledgement

I would like to express my deepest gratitude to Professor Luo Jun for providing me with guidance and support throughout this project. His quick response to my emails and his calm demeanour was greatly appreciated. His expertise and guidance were invaluable in helping me to complete this project successfully. I would also like to extend my thanks to Zhang Shujie for her advice and suggestions. I am grateful for her patience and willingness to help me collect the sensor even when I was late. Her contributions were vital in the completion of this project. I would like to thank both Professor Luo Jun and Zhang Shujie for their time and dedication to this project. Their support and guidance have been instrumental in its success.

I would also like to extend my gratitude to my friends who took time out of their schedules to come down to school and help me with the data collection. Their support and assistance were greatly appreciated. Lastly, I would like to thank my family for their unwavering support throughout my years in NTU. Their love and encouragement, even during the toughest times, have been invaluable to me. I am truly grateful for their support throughout this project and always.

Table of Contents

[Section 1: Introduction 3](#_Toc125143443)

[Section 1.1: Current Strategies 3](#_Toc125143444)

[Section 1.2: Potential Strategy 5](#_Toc125143445)

[Section 2: Seat Occupancy Detection using IR-UWB 6](#_Toc125143446)

[Section 2.1: Background Information 6](#_Toc125143447)

[Section 2.2 Hardware & Software Implementation 8](#_Toc125143448)

[Section 2.3 Signal Pre-Processing 8](#_Toc125143449)

[Section 2.3.1 Background subtraction 8](#_Toc125143450)

[Section 2.3.2 Adaptive Clutter Suppression 11](#_Toc125143451)

[Section 2.3.3 Envelope 13](#_Toc125143452)

[Section 2.3.4 Distance Compensation 14](#_Toc125143453)

[Section 2.4 Target Detection 15](#_Toc125143454)

[Section 2.4.1 Background Information 15](#_Toc125143455)

[Section 2.4.2 CA-CFAR 15](#_Toc125143456)

[Section 2.4.3 MATLAB Implementation 16](#_Toc125143457)

[Section 2.5 Target Localisation 18](#_Toc125143458)

[Section 2.6 Pose Estimation 18](#_Toc125143459)

[Section 2.6.1 Hidden Markov Model Background 18](#_Toc125143460)

[Section 2.6.2 HMM Algorithm for In-Bed detection 19](#_Toc125143461)

[Section 2.6.3 MATLAB Implementation 19](#_Toc125143462)

[Section 2.7 Real-Time System 25](#_Toc125143463)

[Section 2.8 Application Development 26](#_Toc125143464)

[Section 3: Seat Occupancy Detection Using FMCW 34](#_Toc125143465)

[Section 4: Reference List 35](#_Toc125143466)

# Section 1: Introduction

In recent years, academic libraries have undergone renovations to create more conducive environments for their users [1]. These changes have focused on improving the comfort, accessibility, and overall user experience of the library. As a result, academic libraries have become popular places for students to study and work on assignments, especially during exam periods like midterms and finals when demand for study space is high. During these times, libraries tend to be crowded with students seeking a quiet and productive place to work. Many students have reported difficulties finding a seat in the library, with nearly two-thirds of students at the University of Illinois [2] and over 61.2% of students at UC Berkeley citing this as an issue [3]. A survey at the University of Washington found that 84.2% of respondents preferred to wander around the library to find a study spot [4], which can be time-consuming and decrease the overall user experience. The lack of knowledge about seat availability in libraries may contribute to these difficulties.

## Section 1.1: Current Strategies

Currently, NTU students are required to scan their matriculation card at the gantry to gain access to the Lee Wee Nam Library.



Figure 1: Student entering the library through the gantry

(Source: https://blogs.ntu.edu.sg/ntulibrary/2018/12/05/lee-wee-nam-library-reflections-from-a-librarian/)

By doing so NTU is able to keep track of the number of students that have entered the library. Additionally, there is a web application that provides an estimate of the library's level of crowdedness based on the number of matriculation card scans.

Diagram

Description automatically generated

Figure 2: Live seat availability (Source: https://libapp.ntu.edu.sg/)

While this may provide an indication of the overall level of crowding, students may still need to search for an available seat once inside.

Graphical user interface, application

Description automatically generatedA picture containing text, stone

Description automatically generatedOther universities have implemented the Waitz application, which displays the level of crowdedness at a section level within the library. Waitz is an application launched in 2017 for occupancy monitoring [5]. It utilizes custom Internet of Things (IoT) sensors that are mounted on the plugs in each section of the library (shown in figure 3). These sensors will scan for radio signals picking up connected devices such as laptop, cell phones and wearables. The total signal activity of these connected devices is then measured and analysed to predict number of people present in a specific area [6], which is displayed in the Waitz app (shown in figure 4).

Figure 4: Waitz App (Source: https://waitz.io/index.html)

Figure 3: Waitz sensor being mounted (Source: https://www.facebook.com/mcgill.library/videos/find-a-seat-with-waitz/816791778959406/)

While Waitz helps students understand the level of crowding in different library sections, it does not provide information on individual seat availability. Sensors could be utilized to monitor individual seat availability.

Mechanical sensors are frequently employed to monitor seat occupancy [7]. Mechanical sensors can either be resistive [8], capacitive [9], or inductive [10]. These sensors are installed in each seat and are able to detect whether the seat is occupied or available. However, mechanical sensors have the limitation of being unable to distinguish between people and objects placed on them. To address this limitation, research has examined using a single force-sensitive resistor (FSR) for seat occupancy detection [7]. This approach may be effective at determining the availability of individual seats, but it is expensive to implement due to the requirement of installing sensors in every seat in the library.

An alternative approach to monitoring seat availability is to use sensors that capture a region of space instead of a single seat. Cameras are one example of such a sensor, as they can periodically capture the area of interest and use object detection to identify occupied seats. However, the usage of camera will raise privacy concerns [11] as individuals are filmed and they would have to give consent every time they enter the library which might deter students from using the facility.

As a result, it is important to consider alternative sensors that can monitor a region of space without invading privacy. An example of such a sensor is grid-EYE produced by Panasonic. This sensor uses an array of infrared (IR) sensors to capture temperature readings in a 2-dimensional area [12], making it capable of detecting the presence of people or objects in a specific region. The detection area of this sensor is around 5.3m [13] making it suitable to measure 4 to 5 seats spread across a table. Students at the University College London have created a prototype that uses the grid-EYE sensor for live seat availability tracking [14]. In addition to seat occupancy monitoring, the grid-EYE sensor has also been used for applications such as social distancing [15] and passenger detection [16].

## Section 1.2: Potential Strategy

Inspired by the ubiquity of the grid-EYE sensor and its ability to monitor a region of space without compromising privacy, we began to explore the potential use of Radio Frequency (RF) sensing for seat detection. RF sensing utilizes radio frequency signals such as WiFi,4G and Bluetooth to capture events in the IoT environment [17]. When these RF signals are transmitted, they will be reflected, blocked, and scattered by objects [18]. By receiving these fluctuated signals and using machine learning models, RF sensing can be applied in a variety of applications such as gesture recognition [19], pose estimation [20], and human activity recognition [21]. Like grid-EYE RF sensors captures a region of space without invading the privacy of the users. The contactless [22] and low-cost nature [23] of RF sensing make it suitable for a variety of applications, including seat occupancy detection. Therefore, it is worth considering the use of RF sensors for this purpose.

While the application of RF sensors in seat occupancy has not been extensively studied, it can be broken down into three smaller problems that are commonly researched: human detection, human localization and pose estimation. In this report, we will specifically examine the applicability of two types of RF sensors: Impulse Radio-Ultrawide Band (IR-UWB) and Frequency Modulated Continuous Wave (FMCW). We will delve into the capabilities of these sensors in solving the three subproblems of human detection, human localization, and pose estimation, which are necessary for accurately identifying occupied seats.

# Section 2: Seat Occupancy Detection using IR-UWB

## Section 2.1: Background Information

IR-UWB systems transmit and receive data using pulses that are extremely brief, lasting only a few nanoseconds [24] causing it to occupy a large bandwidth [25]. In each frame, there is only one pulse transmitted [26]. The radar transmits frames at a consistent speed and combines the received signals to create a signal matrix r(t) = [r1(t), · · · rn(t)] where t is the fast-time index and n is the slow time index [17], [27]. IR-UWB has been widely studied in the field of medicine for vital sign monitoring [28-36] and tumour detection [37], as well as in applications that involve human detection [38] and gesture recognition [39-42].

During transmission, the baseband gaussian pulse s(t) is modulated by a cosine carrier [43] as a result the transmitted pulse is

where is the carrier frequency. In an indoor environment such as the library, it can be expected that there will be substantial multipath interference [44]. It was stated in [43] that the indoor environment can be modelled as a channel by the following equation

where is the attenuation and is the time delay due to the propagation of the pth reflected path. The received signal y(t) would be a convolution between the transmitted signal x(t) and the channel h(t)

When convolved with a shifted impulse function yields a shifted version of that function [45]. As a result, the time function is x(t) will be replaced with . However, the received signal is still a modulated version of the baseband. To recover the baseband signal, we could employ IQ down conversion together with a low pass filter. The resulting signal , as mentioned in [43], is shown below

(4)

If the threshold is set to be , then the first term of the equation will be removed from the system leaving only With this resulting equation objects at different distance will reach the receiver at different timing. Using the time delay of the pth signal we can deduce the distance the object is located at a centimetre-level precision [46]. The high precision of IR-UWB can be attributed to its high bandwidth [47]. Additionally, the low power of the transmitted pulses makes it a safe and health-risk-free [48] choice for use in situations involving people making an ideal choice for seat occupancy detection.

However, a single IR-UWB sensor is not sufficient to detect humans in a 2D space, as it is unable to resolve equidistant targets. This is particularly important as multiple students may be seated at the same table. The current literature only discusses the number of people located at a particular distance [49]. Therefore, we are unable to identify the exact coordinates of the target. By using two sensors, it is possible to capture both the x and y coordinates of the target, potentially allowing for the occupancy detection of a specific seat in a 2D space. Additionally, breaking the 2D problem down into a 1D problem simplifies calibration of the algorithm and sensors. In the remaining sections we will first explore how IR-UWB is able to identify seat occupancy in a 1D space (i.e. a row of seats or a column of seats) and apply this understanding to detecting occupancy in a 2D space.

## Section 2.2 Hardware & Software Implementation

This project uses two Novelda X4M05 IR-UWB radar transceivers, which are compact and low-cost. These radars operate at a centre frequency of 7.29 GHz with a bandwidth of 1.5 GHz and a sampling rate of 23.328 GHz [50]. The frame rate for these radars is set at 50 fps, and they have a pair of transmit-receive antennas with a field of view of 65 degrees. To control the transceivers and interface with a laptop, we are using a Raspberry Pi single-board computer [51]. The laptop used in this project has an Intel Core i5-8250 CPU, 16 GB of RAM, and a GeForce MX140 graphics card. This project used Python 3.7 to create an application that displays seat occupancy and interfaces with the Raspberry Pi single-board computer. The application uses MATLAB 2020b to analyse the signal and obtain the state information of a specific seat. To avoid disruption to students studying in the library, the project was evaluated in the Computer Networks and Communication Lab.

## Section 2.3 Signal Pre-Processing

### Section 2.3.1 Background subtraction

The received signal must be converted to the baseband signal through IQ down conversion as explained in section 2.1. Novelda X4M05 IR-UWB radar is able to be configured to perform the down conversion automatically [50]. However, the signal is still corrupted with the multipath interference as seen from the summation sign in equation 4. This interference can be removed through the use of background subtraction methods such as the loopback filter [52]. This method is popular in literature due to its fast-processing time [48]. The equation of the loopback filter [53] is

where α represent the signal to clutter ratio and ranges from 0 to 1 [39].

The clutter signal at each frame depends on the frame before it therefore we will initialise the first frame of the clutter signal to be the same as the first frame of the received signal [54]. The loopback filter will calculate the clutter signal from frame 2 to nth frame The value of alpha, which determines the strength of the filter, impacts both the computation time and the robustness to noise [53]. A low value of alpha will result in faster clutter signal estimation but may make the target signal vulnerable to noise. A high value of alpha will increase the estimation time but will also make the target signal more resistant to noise. The optimal value of alpha depends on the specific situation and desired outcome. For example, values of 0.97, 0.95, 0.85 and 0.8 have been used for vital sign monitoring [31], finger counting [39], in-air writing detection [55] and mobile phone usage while driving [31], respectively. However, some literature has used higher values such as 0.97 for hand gesture recognition [41]. Therefore, it is necessary to experiment with different alpha values to determine the optimal value for seat occupancy detection. The performance of different alpha values was evaluated on two types of signals: stationary and moving targets as the system will only be exposed to these 2 different types of signals. The computation time for clutter estimation is shown in tables 1.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Signal Type |  |  |  |  |
| Stationary Target (626 frames) | 0.003626s | 0.003806s | 0.003940s | 0.004595s |
| Moving Target  (941 frames) | 0.006130s | 0.006475s | 0.006928s | 0.007897s |

Table 1: Comparing the speed of different alpha values

As expected, the low alpha values can estimate the clutter signal the fastest, but we also have to consider the noise. Our goal is to create a system that can display results in real-time without causing false positives, such as incorrectly indicating that a seat is taken. Examples of the received signals and their cleaned counterparts are shown below.

Background pattern

Description automatically generated

Figure 5: Received Signal of Stationary Target

A picture containing chart

Description automatically generated

Figure 6: Stationary Target cleaned signal

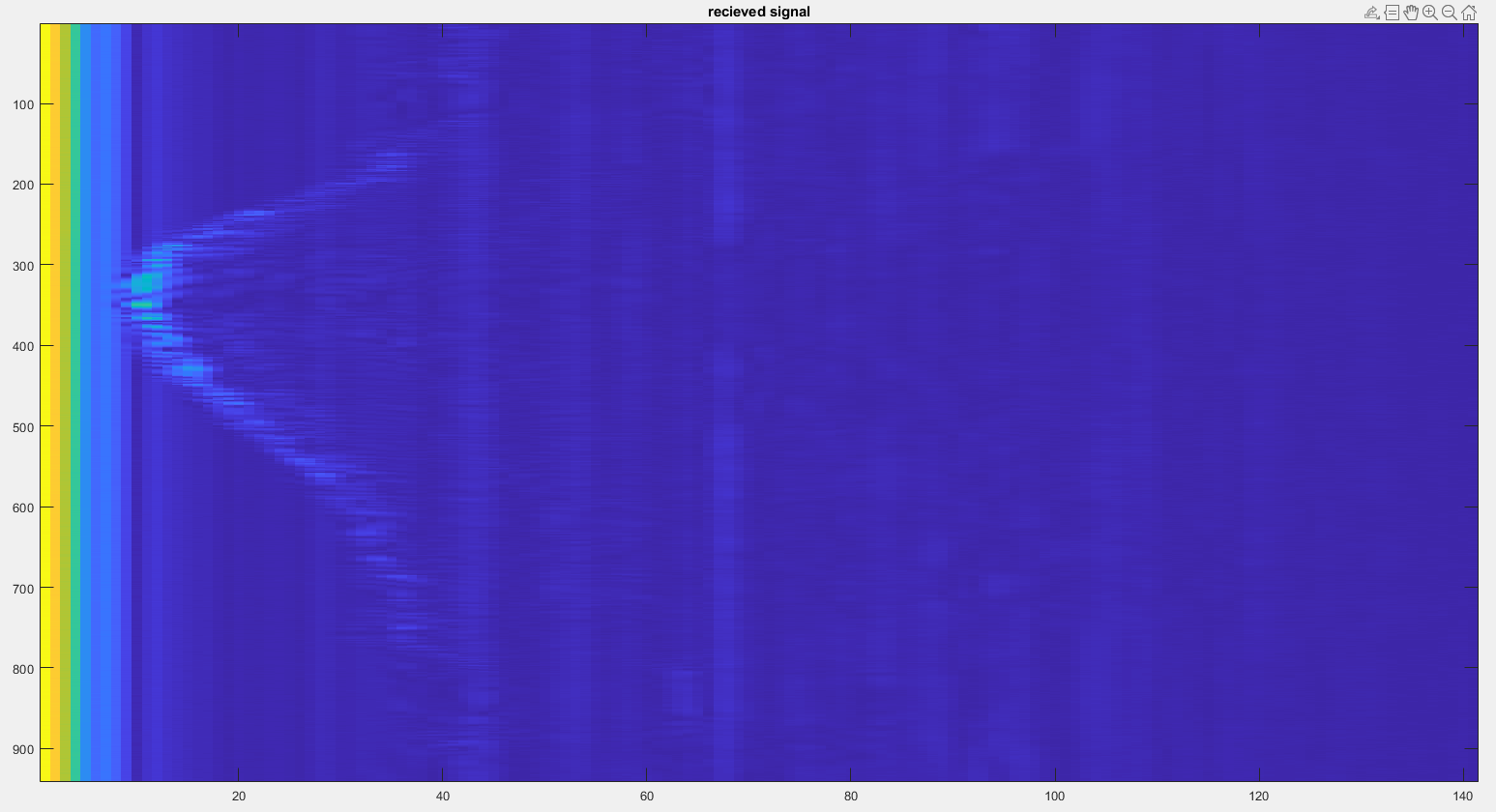


Figure 7: Received Signal of Moving target

Graphical user interface

Description automatically generated

Figure 8: Moving Target Cleaned Signal

Comparing the outputs of the moving target signals, there is little difference observed. However, for the stationary targets, it was found that a higher alpha value resulted in a brighter signal, indicating a higher signal-to-noise ratio (SNR). While lower alpha values are computationally faster, it is important to consider the benefits of a higher SNR for target detection [56]. Therefore, the alpha value used in this project is 0.95, as it offers a balance between computational speed and noise immunity.

### Section 2.3.2 Adaptive Clutter Suppression

To improve the accuracy and efficiency of seat occupancy detection, we need to adaptively change the application ratio (alpha value) applied to the received signal. Higher ratios should be used in areas with recent or expected motion, while lower ratios should be used in areas with no recent or anticipated movement [57]. This helps to maintain a high signal-to-noise ratio for better detection and allows the system to efficiently detect moving objects. The algorithm that was stated in [57] is as follows:

The application ratio at a specific frame and fast time index is given by the minimum between the normalized envelope of the received signal and the target signal at that specific index divided by the normalized envelope. The envelope used here is using the Hilbert transformation and it is normalised so that the application ratio can be that is derived in subsequent frames and fast time index are consistent with the range which is 0 to 1. [57] went on to add a step where they scaled the application ratio, but it requires domain knowledge, so we decided to stick with the default range. The MATLAB code for the above algorithm is shown below

Text

Description automatically generated

Figure 9: MATLAB Code for adaptive clutter suppression

The Hilbert transformation gives us a complex signal and since the application ratio is a positive integer, we will be taking the absolute of the envelope. The normalisation of the envelope was min max normalisation so that it will follow the range of 0 to 1. The envelope of both the signals is stored in a vector so that when the minimum operation is performed it can retrieve the minimum value in each range bin. Since we performed min max normalisation there is a chance that the envelope of the received signal is 0, therefore a small value is added to prevent division by 0. Figure 10 shows the effect of using adaptive clutter suppression as compared to the signal after background filtering on a signal that has 2 targets located at 240cm and 105cm.

Chart

Description automatically generated

Figure 10: Adaptive Clutter suppression vs background subtraction

The red line represents the adaptively suppressed signal while the blue signal represents the signal after loopback filter. While the clutter remains to be unchanged, the strength of the signal where the targets are located is much higher.

### Section 2.3.3 Envelope

After applying adaptive clutter suppression to the received radar signal, the authors took the envelope of the resulting signal for target detection and localization. Envelope detection is often used in radars for non-coherent detection [58], and it was also used in [59] after signal cleaning. Even though we have obtained the baseband signal through IQ down conversion, envelope detection can help to create a stronger signal that aids in the target detection stage. After clutter suppression, the resulting stationary signal may appear discontinuous at certain frames. However, taking the envelope of the clutter-suppressed signal can help to smooth out these discontinuities. The difference in the signal before and after applying envelope is shown in Figure 11.

A computer screen capture

Description automatically generated with low confidence

Figure 11: Comparing the effects of an envelope on clutter-suppressed signal

### Section 2.3.4 Distance Compensation

After the clutter suppression and envelope detection, the signal must be compensated for distance attenuation. This is because targets located closer to the sensor will have a stronger reflected signal, while those further away will have a weaker reflected signal [60]. To address this issue, the envelope of the clutter suppressed signal is multiplied by a vector that increases linearly with distance [60]. The results of this distance compensation are shown below

Chart, histogram

Description automatically generated

Figure 12: Effects of Distance Compensation

## Section 2.4 Target Detection

### Section 2.4.1 Background Information

Once the signal is prepared, we can proceed with identifying human targets. One commonly used method for detecting targets is the Constant False Alarm Rate (CFAR) method [61]. There are several different CFAR algorithms available, such as Cell-Averaging CFAR (CA-CFAR), Order Statistic CFAR (OS-CFAR), Greatest Of CFAR (GO-CFAR), and Smallest Of CFAR (SO-CFAR). The SO-CFAR algorithm was designed to detect closely spaced targets [62], however, it performs poorly in environments with clutter edges. The OS-CFAR algorithm, on the other hand, is designed to be more robust and is able to detect multiple targets that are close together and located in non-homogenous environments [63]. However, both the OS-CFAR and SO-CFAR algorithms require more computational resources, which may delay the processing of the signal and detract from creating a real-time system. Additionally, due to the current COVID-19 pandemic, social distancing measures are enforced, and seats are typically spaced one meter apart, reducing the likelihood of closely spaced targets, further diminishing the need for the use of OS and SO CFAR algorithms. While there are other CFAR algorithms available, they do not have pre-existing functions in MATLAB. As such, we will be using the CA-CFAR algorithm to detect targets, as it exists in MATLAB and does not require a significant amount of time to compute.

### Section 2.4.2 CA-CFAR

CA-CFAR works by computing the power level of a target cell (the cell being analysed for the presence of a target) to the average power level of a reference cell (a group of surrounding cell sells used as a reference for determining the presence of a target). According to [64], the algorithm starts by selecting a reference window, which contains a group of cells surrounding the target cell. The average power level of these reference cells is then calculated. Next a guard window, which is a group of cells surrounding the reference window is selected. The average power level of the guard cells is then subtracted from the average power level of the reference cells resulting in a threshold value. If the power level of the target cell is above this threshold it is determined to be a target. The threshold is calculated based on the probability of false alarm

where α is the threshold. N is the number of training cells and the is the probability of false alarm [65] .

The diagram of the CA-CFAR process is shown below

Diagram

Description automatically generated

Figure 13: CA-CFAR Algorithm depiction (Source: https://www.mathworks.com/help/phased/ug/constant-false-alarm-rate-cfar-detection.html)

### Section 2.4.3 MATLAB Implementation

The MATLAB implementation of the CA-CFAR algorithm is illustrated in Figure 14. Since the signal obtained in Section 2.3 is in baseband form, there is no need to perform square law prior to applying the algorithm. The goal is to identify a target in each frame, so the CFAR algorithm must be applied to each fast time index. Therefore, the algorithm is performed on the transposed signal. The number of training cells and guard cells used in the algorithm are adapted from [66].

Text

Description automatically generated

Figure 14: MATLAB Code for CA-CFAR

However, the method used had a probability of false alarm of 10-4, which did not detect any targets, even when present. To improve detection, we experimented with different PFA values for various signals. PFA of 10-1 provided the most accurate results. The results are presented in Table 2.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Signal Type | Predicted Number of people | | | | Actual Number of People |
| PFA = 10-4 | PFA = 10-3 | PFA = 10-2 | PFA = 10-1 |
| Stationary Target (200cm) | 0 | 1 | 1 | 1 | 1 |
| Stationary Target  (400cm & 525cm) | 0 | 0 | 0 | 1 | 2 |
| Moving Target | 0 | 0 | 0 | 2 | 1 |

Table 2: Comparing Effectiveness of different PFA values

To determine the number of people, present, we first multiplied the detected signal (consisting of 1s and 0s) with the original signal. This eliminated any frames without any detected targets. Next, we utilized the 'findpeaks' function in MATLAB to locate the number of peaks in each frame of the multiplied signal. Each peak corresponds to a unique target [43]. The number of peaks in each frame is added to an array and the predicted number of people is calculated as the median of that array.

Text

Description automatically generated

Figure 15: MATLAB Code for finding the number of people detected

The CA-CFAR algorithm was applied to the envelopes of both the clean and suppressed signals. As shown in Figure 16, the use of an envelope detector significantly improves target detection. This highlights the importance of utilizing an envelope detector, as discussed in section 2.3.3.

A computer screen capture

Description automatically generated with low confidence

Figure 16: Comparing the effects of envelope detector on target detection

## Section 2.5 Target Localisation

Once the targets have been identified in each frame, the next step is to locate these targets. The range resolution can be written as where c is the speed of light and B is the bandwidth of the radar [67]. Thus, the range resolution will be 10cm as the radar has a bandwidth of 1.5GHz. However, after the signal pre-processing, the range resolution might decrease. Based on experiments, we have determined that each range bin roughly represents 5cm

## Section 2.6 Pose Estimation

Once targets have been detected, the next step is to determine if they are seated or standing. If they are seated, we can also identify the specific seat that is occupied by the target. The use of IR-UWB for pose estimation is not a well-researched area. However, some scientists have used Hidden Markov Models (HMM) to estimate if a person is in bed [68] or even the direction in which the subject is facing the sensor [69].

### Section 2.6.1 Hidden Markov Model Background

HMM is a powerful statistical tool that can be used to model the sequence of observable events that are influenced by internal factors that cannot be directly observed [70]. To use HMM in a problem, one must define a set of possible states, a set of observations, and the probabilities of transitioning between states (transition probability) and emitting observations at each state (emission probability) [71]. These probabilities are typically learned from data [68].

### Section 2.6.2 HMM Algorithm for In-Bed detection

The HMM used in [68] has two states S0 & S1, where S0 represents an empty bed and S1represents a target in bed. They have divided the region around the bed into 3 areas: R0 represents the centre area, R1 represents the buffer area and R2 represents the outer area. The centre area is where the bed is located, the buffer area is a 50cm wide circular area that surrounds the bed and any point outside of the buffer area will fall into the outer area. The observation is not a sequence of single states but a sequence consisting of tuple of states Rs and Re. They have broken down the signal into consecutive window of 5 seconds and detect the locations of the targets during this window. The location of the target is then mapped to one of the three areas, R0, R1, R2. Rs is the area that correspond in the first second of the window and Re is the area that correspond to the last second of the window.

### Section 2.6.3 MATLAB Implementation

In order to use the HMM model in MATLAB, we will be using the values 1 and 2 for the states, where 1 represents a seat being available (S0) and 2 represents a seat being taken (S1). For observations, we will be using the values 1, 2 and 3, where 1 represents the central area (R0), 2 represents the buffer area (R1), and 3 represents the outer area (R2). Targets that are located very close to the sensor tend to take up a column of 20cm width as shown below. As a result, we will be taking an adding 10cm to both the left and right side of the seat and label this area as R0.

A screenshot of a computer

Description automatically generated with medium confidence

Figure 17: 2 Stationary Targets at 100 and 240cm

A buffer radius of 5cm was used due to the small size of the chairs. Any locations beyond this buffer radius will be categorized as R3. The HMM model will be evaluated on each individual seat to determine their states. To obtain the observations, we will first divide the signal into windows using a step size of 2 frames, as explained in section 2.7. We will then locate the targets in the first and second frames of the window using the 'findpeaks' function in MATLAB. The indexes returned by this function will indicate the locations where the targets are located. However, if the number of people detected is lesser than the number of seats, it can be difficult to determine the state of each seat. To address this issue, we will append 0s to the index array returned by the 'findpeaks' function to match the number of seats.

Text

Description automatically generated

Figure 18: Code for target detection window

The next step is to iterate through all the seat locations and the detected locations to find the nearest seat for each detected location. Initially, the idea was to assign the strongest peaks, which would indicate people, to the seats. However, it was found that there is a chance that noise could be incorrectly classified as a target. Another idea was to assign the first detected location to the first seat, but this approach does not take into account that the target may be seated at a location that is not nearest to the first seat. Therefore, iterating through all the seat locations to find the nearest one to the target is necessary to ensure accurate mapping of detected locations to seats.

Text

Description automatically generated

Figure 19: Code to map target location to nearest seat

Finally, in order to complete the HMM process, the location of the detected targets must be mapped to one of the 3 areas, R0,R1,R2. To do this, the first and last detected location of the window will be compared with the seats location and the mapping will be stored in a tuple. Each tuple will then be stored in an array, creating the seat observation array, with a dimension of 100, 2\* the number of seats. The first 2 columns will correspond to the observation of the first seat, the next 2 columns will correspond to the observation of the second seat and so on.

Graphical user interface, text, application, email

Description automatically generated

Figure 20: Code to map detection location to observed areas

To obtain the transmission and emission probability matrix, the observation collected for different samples of both moving and stationary targets at various locations was stored in an excel file along with the actual state of the seat. The 'hmmestimate' function in MATLAB was used to estimate the transmission and emission probability matrix. The matrix is then stored in a text file for easier access and use in future analysis.

Graphical user interface, text, application, email

Description automatically generated

Figure 21:Code to train HMM

The state of individual seats can be obtained using the ‘hmmviterbi’ function in MATLAB. Do take note that the first window will give erroneous result as we have initialised the clutter signal to be the same as the received signal.

Text

Description automatically generated

Figure 22: Code to predict state of each chair

The effectiveness of the HMM model is evaluated by calculating the ratio of correct prediction of various signals.

|  |  |  |  |
| --- | --- | --- | --- |
| Signal Type | Predicted States | Actual State | Accuracy |
| Seated Target  (60cm) | 2,2,2,2,2,2,2,2,2,2 | 2 | 100% |
| Seated Target  (200cm) | 2,2,2,2,2,2,2,2,2,2 | 2 | 100% |
| Seated Target  (200cm & 85cm) and a free seat at 400cm | 2,2,2,2,2,2,2,2,2,2  2,2,2,2,2,2,2,2,2,2  1,1,1,1,1,1,1,1,1,1 | 2  2  1 | 100% |
| Moving Target  (Seat located at 190cm) | 2,2,2,…,2,2,2,2,2 | 1,1,1,1,…1,1,2,2,2 | 27% |
| Moving Target  (Seat located at 50cm) | 1,1,1… 2,2,2,2 | 1,1,1,…2,2,2,…1,1,1 | 27% |

Table 3: Effectiveness of HMM model on different signals

The HMM model was found to be effective in detecting stationary targets but not in detecting moving targets. Upon investigation, it was discovered that the use of an envelope detector added a trailing edge to the movement, causing the HMM model to detect the seat as being taken prior to the actual movement. For example in figure … the targets only gets seated at the seat located at 100cm at frame 800 but the envelope detector makes it as if the person has been seating there all this while.

A screenshot of a computer

Description automatically generated with low confidence

Figure 23: Moving Target Signal

When we consider the first 800 frames only, we do not see the trailing edge located at 100cm.

A picture containing graphical user interface

Description automatically generated

Figure 24: Moving Target but only the first 800 frames

Therefore, a real-time system that only considers a subset of the frames at each instance can resolve the aforementioned issue. The implementation of the real-time system will be discussed in Section 2.8.2

## Section 2.7 Real-Time System

. Since the MATLAB code was able to analyse the signal within 0.22 seconds there was no need to implement multi-threading. A Python script was created to incorporate two functions. The first function uses SSH to remotely access an RPI and execute a file that captures 20 frames of data, which are then stored in a shared folder. The second function runs a MATLAB code that accesses the excel file in the shared folder and calculates seat availability. Since the second function is only able to be executed once the excel file is available, after a delay of 2 seconds the second function is executed. The code and its explanation will be given in Section 2.8

## Section 2.8 Application Development

In this section, we will discuss the implementation of an application to visualize seat occupancy using IR-UWB technology. This application, which was developed using the PyQt Python package for basic GUI applications, connects to a MATLAB engine for signal processing. The ultimate goal of this project was to create a real-time system that displays seat occupancy information to students.

### Section 2.8.1 User Interface

In this subsection, the user interface of the application will be discussed in detail

The application displays a waiting image to users while the MATLAB engine is initializing

Graphical user interface, text, application

Description automatically generated

Figure 25:Starting Screen of the application

With the MATLAB engine initialized, users can then choose the sensor type they wish to use for seat occupancy detection. We wanted to explore the applicability of FMCW radars in seat occupancy detection, but due to time constraints, we were unable to pursue this aspect in this project. However, a rough ideation illustrating how FMCW radars could be utilized for seat occupancy detection will be presented in Section 2.9.

Graphical user interface, text, application

Description automatically generated

Figure 26: Home page of the application

Additionally, users will be prompted to select the orientation of the seats, whether they are placed vertically, horizontally, or in a 2D space. By clicking on the home page button, users will be able to navigate back to the initial screen of the application (Figure 29).

A picture containing waterfall chart

Description automatically generated

Figure 27: Schema Selection Page

For all 3 options, the application requires the coordinates of the seats to be entered. Therefore, before the signal analysis can begin, the users must first enter the number of seats. Due to the limited scale of this experiment, the application only accepts a value ranging from 1 to 10 seats.

Graphical user interface

Description automatically generated

Figure 28: User Input for number of seat

Graphical user interface, application

Description automatically generatedIf the user enters an input that is not a whole number or falls outside of the accepted range of 1 to 10 seats, an error message will be displayed, instructing the user to enter a valid input.

Graphical user interface, application

Description automatically generated

Figure 29: Example of Wrong Seat Input Figure 30: Error message for invalid number of seats

Once the user have successfully entered a valid number of seats, they will be prompted to enter the coordinates of each seat. The layout of the page will be the same for both vertical and horizontal seating arrangements, but the instructions displayed will be specific to the chosen option.

Graphical user interface, application

Description automatically generated

Figure 31: User input page for entering coordinates of each seat arranged vertically

Graphical user interface, text, application

Description automatically generated

Figure 32: User input page for entering coordinates of each seat arranged horizontally

When the users have selected the 2D option, they will be prompted to enter the coordinates of each seat in both the x and y directions.

Graphical user interface

Description automatically generated

Figure 33: User input page for entering coordinates of each seat arranged in a 2d space

As seen from the above figures, the number of input fields required for seat coordinates will correspond to the number of seats entered in Figure 31. Users can easily correct their mistake of entering the wrong number of seats by clicking on the back button located at the top right corner of the interface, which will take them back to Figure 31. Furthermore, if users have selected the wrong seat orientation, they can rectify the mistake by closing the window or by pressing the cancel button shown in Figure 31.

To ensure the accuracy of the results, it is important that users enter the correct coordinates for each seat. The coordinates must be within a range of 50cm to 500cm, as this project is only a small-scale experiment and the sensor's range is limited. We collected data using a QlineEdit widget with an integer validator, which limits user input to only digits and reduces the chances of erroneous results.

Graphical user interface, text, application

Description automatically generated

Figure 34: Error message if seats entered are above the range specified

Additionally, the application will also check if the seats are at least 30cm apart from one another, as we mentioned in section 2.7. For 2D space, the difference of 30cm has to be in both x and y axis. If any of the coordinates entered is incorrect or the seats are not sufficiently spaced, an error message will appear prompting the user to enter the coordinates again.

Graphical user interface, application

Description automatically generated

Figure 35: Error message if the seats are not spaced 30 cm apart

Graphical user interface, text, application

Description automatically generated

Figure 36: Error Message for 2D space if x or y coordinates of seats are not evenly spaced

Even though the 2 seats shown in Figure 39 are 30cm apart, the seats are not 30cm apart in both X and Y dimension therefore the error message is shown. This discrepancy is mainly due to multi-threading and we are treating the X and Y axes independently. This will be discussed again in section 2.9.

Once the valid coordinates of the seats are entered, the user will be brought to a page where the coordinates of the seats are highlighted in green. For ease of visualization, the range is broken into 10 bins in both the x and y directions. The seat highlighted is calculated by dividing each user input by 50 and subtracting 1. The resulting float variable is then rounded to the nearest whole number. This number will indicate the index of the seat that is available to monitor. The columns and rows are labelled with the distance away from the sensor in the x and y directions. The back button will bring the users back to the orientation selection page. The schemas for the 3 options are shown below.

Chart

Description automatically generated

Figure 37: Schema for seats arranged vertically

Chart, waterfall chart

Description automatically generated

Figure 38: Schema for seats arranged horizontally

Chart

Description automatically generated

Figure 39: Seats arranged in a 2D space

The back button located at the top left corner of the application allows users to return to the schema selection page depicted in Figure 30. The distance between the seats and the sensor is displayed using arrows, while the legend at the top right corner indicates that a dark green seat is available and a red seat is taken. For 2D seat occupancy detection, a yellow colour is used to indicate ambiguity if a seat is detected as taken in one dimension but not the other.

### Section 2.8.2 Real-Time System Development & Integration

In this sub section we will be discussing how we have developed a real time system and integrate with the App discussed in earlier subsection.

To design an effective Real-Time System, the Nyquist theory must be applied in order to accurately sample the room or area. According to the average walking frequency of 1~2 Hz, the room should be sampled at a rate of 4Hz [72], or every 0.25 seconds or 10 frames. The Hidden Markov Model (HMM) discussed in section 2.6 requires a minimum step size of 2 in order to effectively predict the state of each seat by considering both past and future data. However, with such a short frame duration, the HMM model may not have sufficient data to accurately determine the state of each chair. For example, consider this signal where the seat is located at 50cm.

Chart

Description automatically generated

Figure 40: Stationary Target ,10 frames

And the observations of this signal were all 1s indicating that the target is seated down but the predicted state is 1 implying that the seat is available.

Chart

Description automatically generated

Figure 41: Predicted state for 10 frames of stationary target.

But when the number of frames is increased to 20, the predicted state of the chair is becomes 2

Scatter chart

Description automatically generated with low confidence

Figure 42: Predicted state of the stationary target with 20 frames.

Therefore, to ensure accurate seat occupancy information, using 20 frames instead of 10 is necessary, despite potentially compromising real-time performance.

The radar file in the RPI directory is modified to run a while loop that calls the radar function for 0.55 seconds after which it will break. The results are then saved to a excel filed which will be stored in a shared folder. The shared folder is created using the samba file server. To facilitate a continuous loop the RPIs are connected to the ethernet port using the ethernet cable. As a result, each RPI has a unique IP address which will be used by python to SSH. Since SSH clients are meant to be short lived we will be invoking a shell and be sending the commands to the RPIs using those shell created.

When the start button is pressed is pressed it will display the stop button and SSH into the respective RPIs based on the schema that the users has specified previously. The label will also change to inform the user the sampling procedure has started.

Chart, waterfall chart

Description automatically generated

Figure 43: UI when the user first presses the start button.

Figure 44 UI when the RPI is running the radar thread

After which the application will create a Qthread name Radar\_Main\_Thread where the SSH client created using Paramiko and the channel created will be passed into the thread. The Radar\_Main\_Thread will first change the directory of the RPI to where the modified radar code is found using the shell passed to it.

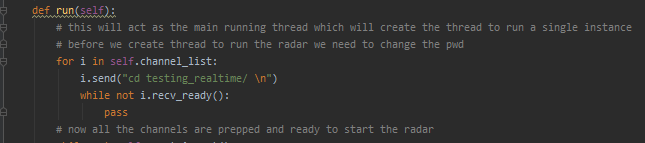


Figure 45: Python Code For Radar Main Thread

After the “cd” command is sent the application monitors the receiver of the shell and once the shell is ready to receive other commands the Radar\_Main\_Thread will create the appropriate number of threads continuously until the stop event is set. The stop event is a threading event created when the app was starting up more specifically in between Figure 25 and Figure 26. The stop event is set when the user presses the stop button.

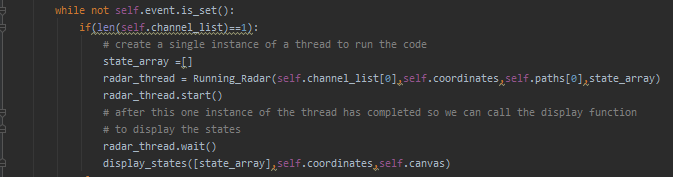


Figure 46: Python code for Running Radar Thread creation in Radar Main Thread

When the Running\_Radar thread starts it will execute the start\_sensor function.

Text

Description automatically generated

Figure 47: Start Sensor Function

The function will send the ./Runme command to run the Radar.cpp file stored in the RPI directory and it waits until the code has completed running. The Radar.cpp will print each frame index while it is running and once it breaks out from the while loop it will print the statement done. Therefore, there is a while loop monitoring the output of the shell

Figure 48 the output of the RPI when running the radar command

After the data from the IR-UWB Radar has been completed it will take some time for the excel file to be reflected in the shared folder on the laptop used for the experiment as a result there will be another while loop to monitor the file size and on average 20 frames of IR-UWB data takes around 52KB or more.

After which, the seat coordinates that were entered by the use is converted into a MATLAB integer array and passed into the IR\_UWB\_function MATLAB function which will compute the state of the chairs individually. After the computation of the state we will remove the file to ensure the next instance of the thread will run on the new set of data. The time taken to complete running the radar code from python takes around 0.8 seconds, the transfer of the file takes 0.0013 seconds and the time taken to run the MATLAB code is 0.17 seconds. Since the bottleneck process is the radar code, we did not implement multi-threading in MATLAB.

From figure 46 It can be observed that the radar\_thread waits until completion before running the display\_states function. The display states function will traverse through the state returned by the MATLAB function and only observes the last known state. If the last known state is 1, it will display the seat as green coloured and if the state is 2 the seat will be displayed in red.

Text

Description automatically generated

Figure 49: Display State function

Due to these various stages, the whole process takes around 1~1.5 seconds which is greatly different from the required sampling rate but as mentioned earlier some of these delays are needed to ensure the accuracy of the system.

### Section 2.8.3 2D System Development & Integration

To find the state in a 2D space, as stated before the Radar\_Main\_Thread will create 2 radar\_threads one of each RPI

Text

Description automatically generated

Figure 50:Radar Main Thread for 2D space

After both threads have started, they will both wait until the threads have completed their individual functions before displaying the states. This will ensure that the excel file received from the RPI measuring the X direction and the RPI measuring the Y direction are close to each other enabling for more accurate results. This is one of the main reasons as to why we have decided to code the application using PyQt5 instead of frameworks such as Tkinter. For Tkinter the GUI freezes when the threads are waiting for one another. If we used a single thread in PyQt the result will be similar therefore there was a need to break it into smaller threads to prevent the GUI from freezing and the user can still interact with it to press the stop button.

For displaying the states, the states in the X direction and the Y direction are added up. If the added state is 2 means that both the x\_state and the y\_state is 1 implying the seat is free. If the added state is 4 implies that both the x\_state and the y\_state is 2 which means the seat is taken. But if either the x\_state or the y\_state is 3 means the seat is taken in one of the axis but not the other which is impossible. Therefore, the seat will be shown in yellow to imply ambiguity so the users have to check for themselves if the seat is taken or not.

Text

Description automatically generated

Figure 51: Displaying state for 2D Space

Since both the threads are waiting for one another the total time taken to display the state is increased to around 2 seconds deviating further away from the ideal real-time system.

Figure 43: Code to run RPI using Python

# Section 4: Reference List

[1] Dilevko J, Gottlieb L. The Evolution of Library and Museum Partnerships: Historical Antecedents, Contemporary Manifestations, and Future Directions: Libraries Unlimited; 2004.

[2] Sanghyuk Lee P-HH, Tzu-Yu Shen. Library Seat Availability Checking System 2011 [Available from: <http://www.eui.illinois.edu/Docs%5CLeeSp11.pdf>.

[3] Z. Dai ML, P. Yang, and Q. Yu. Occupancy Monitoring Application for University Library, Seeat: UC Berkeley; [Available from: <https://www.ischool.berkeley.edu/sites/default/files/sproject_attachments/seeat_final_report.pdf>.

[4] Rachel Chung TH, Monica Ionescu. Libraries Study Spaces Research: University of Washington; 2021 [Available from: <https://www.lib.washington.edu/assessment/projects/UX-session-reports/LibrariesStudySpaces_UX_AssessmentResearch>.

[5] SUBBA S. App implemented to help students know occupancy of library spaces 2021 [Available from: <http://www.campustimes.org/2021/10/03/app-implemented-to-help-students-know-occupancy-of-library-spaces/>.

[6] Wairz. How it Works [Available from: <https://waitz.io/how-it-works.html>.

[7] Sifuentes E, Gonzalez-Landaeta R, Cota-Ruiz J, Reverter F. Seat Occupancy Detection Based on a Low-Power Microcontroller and a Single FSR. Sensors. 2019;19(3):699.

[8] Ralf Oestreicher MH, Harold Lichtinger, Scott Morell, Dan Reich, inventor; Siemens Automotive Corp, assignee. Method and system for determining weight and position of a vehicle seat occupant. U.S.2006.

[9] Boese H, Fuß E, Ehrlich J. A2.2 - Capacitive Sensor Mats for Pressure Detection with High Sensitivity2015. 55-60 p.

[10] Kisic MG, Blaz NV, Babkovic KB, Zivanov LD, Damnjanovic MS. Detection of Seat Occupancy Using a Wireless Inductive Sensor. IEEE Transactions on Magnetics. 2017;53(4):1-4.

[11] Barsocchi P, Calabrò A, Crivello A, Daoudagh S, Furfari F, Girolami M, et al. COVID-19 & privacy: Enhancing of indoor localization architectures towards effective social distancing. Array. 2021;9:100051.

[12] Panasonic. Infrared Array Sensor Grid-EYE [Available from: <https://industrial.panasonic.com/ww/products/pt/grid-eye>.

[13] Panasonic. Grid-EYE Application Note on Social distancing [Available from: <https://mediap.industry.panasonic.eu/assets/custom-upload/Components/Sensors/Industrial%20Sensors/Infrared%20Array%20Sensor%20Grid-EYE/Grid-EYE%20Application%20Note%20on%20Social%20Distancing.pdf>.

[14] Andy Boscor GH, Octav Rusuleanu,Claudia Stanescu,Natcha Sujaritworakun. Study Hunt: Library seat availability live monitoring system 2017 [Available from: <https://community.element14.com/learn/learning-center/stem-academy/stem-projects/b/blog/posts/study-hunt-library-seat-availability-live-monitoring-system>.

[15] Xie C, Daghero F, Chen Y, Castellano M, Gandolfi L, Calimera A, et al., editors. Privacy-preserving Social Distance Monitoring on Microcontrollers with Low-Resolution Infrared Sensors and CNNs. 2022 IEEE International Symposium on Circuits and Systems (ISCAS); 2022 27 May-1 June 2022.

[16] Géczy A, Melgar R, Bonyár A, Harsanyi G. Passenger detection in cars with small form-factor IR sensors (Grid-eye)2020. 1-6 p.

[17] Zheng T, Chen Z, Ding S, Luo J. Enhancing RF Sensing with Deep Learning: A Layered Approach. IEEE Communications Magazine. 2021;59(2):70-6.

[18] Wang X, Wang X, Mao S. RF Sensing in the Internet of Things: A General Deep Learning Framework. IEEE Communications Magazine. 2018;56(9):62-7.

[19] Zhang B, Zhang L, Wu M, Wang Y, editors. Dynamic Gesture Recognition Based on RF Sensor and AE-LSTM Neural Network. 2021 IEEE International Symposium on Circuits and Systems (ISCAS); 2021 22-28 May 2021.

[20] Feng L, Li Z, Liu C, Chen X, Yin X, Fang D. SitR: Sitting Posture Recognition Using RF Signals. IEEE Internet of Things Journal. 2020;PP:1-.

[21] Liu J, Teng G, Hong F. Human Activity Sensing with Wireless Signals: A Survey. Sensors. 2020;20(4):1210.

[22] Ashleibta AM, Taha A, Khan MA, Taylor W, Tahir A, Zoha A, et al. 5G-enabled contactless multi-user presence and activity detection for independent assisted living. Scientific Reports. 2021;11(1):17590.

[23] Nguyen C, Kim S. Theory, Analysis and Design of RF Interferometric Sensors: Springer New York; 2011.

[24] Ghavami M, Michael L, Kohno R. Ultra Wideband Signals and Systems in Communication Engineering: Wiley; 2007.

[25] Zou Z, Tenhunen H, Lande TS. Impulse Radio UWB for the Internet-of-Things : A Study on UHF/UWB Hybrid Solution. Stockholm: KTH Royal Institute of Technology; 2011.

[26] Abbas WB, Che F, Ahmed QZ, Khan FA, Alade T. Device Free Detection in Impulse Radio Ultrawide Bandwidth Systems. Sensors (Basel, Switzerland). 2021;21(9).

[27] Ding S, Chen Z, Zheng T, Luo J, editors. RF-net: A unified meta-learning framework for RF-enabled one-shot human activity recognition. Proceedings of the 18th Conference on Embedded Networked Sensor Systems; 2020.

[28] Cho HS, Park YJ, Lyu HK, editors. Robust heart rate detection method using UWB impulse radar. 2016 International Conference on Information and Communication Technology Convergence (ICTC); 2016 19-21 Oct. 2016.

[29] Khan F, Cho SH. A Detailed Algorithm for Vital Sign Monitoring of a Stationary/Non-Stationary Human through IR-UWB Radar. Sensors. 2017;17(2):290.

[30] Khan N, Khan K, Khan A, Alam I, Ullah F, Khan S, et al. Accommodate Data Loss in Monitoring Vital Signs Through Autoregressive Model. Journal of Medical Imaging and Health Informatics. 2019;9.

[31] Leem SK, Khan F, Cho SH. Vital Sign Monitoring and Mobile Phone Usage Detection Using IR-UWB Radar for Intended Use in Car Crash Prevention. Sensors. 2017;17(6):1240.

[32] Lin JC. Noninvasive microwave measurement of respiration. Proceedings of the IEEE. 1975;63(10):1530-.

[33] Ren L, Wang H, Naishadham K, Kilic O, Fathy AE. Phase-Based Methods for Heart Rate Detection Using UWB Impulse Doppler Radar. IEEE Transactions on Microwave Theory and Techniques. 2016;64(10):3319-31.

[34] Shikhsarmast FM, Lyu T, Liang X, Zhang H, Gulliver TA. Random-Noise Denoising and Clutter Elimination of Human Respiration Movements Based on an Improved Time Window Selection Algorithm Using Wavelet Transform. Sensors. 2019;19(1):95.

[35] Tariq A, Zahid A, Khan U, Khan N, Khan F, editors. Implementation of Wavelet transform for monitoring of vital signs through IR-UWB Radar. 2017 International Conference on Communication, Computing and Digital Systems (C-CODE); 2017 8-9 March 2017.

[36] Yan J, Zhao H, Li Y, Sun L, Hong H, Zhu X, editors. Through-the-wall human respiration detection using impulse ultra-wide-band radar. 2016 IEEE Topical Conference on Biomedical Wireless Technologies, Networks, and Sensing Systems (BioWireleSS); 2016 24-27 Jan. 2016.

[37] Fear EC, Li X, Hagness SC, Stuchly MA. Confocal microwave imaging for breast cancer detection: localization of tumors in three dimensions. IEEE Transactions on Biomedical Engineering. 2002;49(8):812-22.

[38] Choi JW, Nam SS, Cho SH. Multi-human detection algorithm based on an impulse radio ultra-wideband radar system. IEEE Access. 2016;4:10300-9.

[39] Ahmed S, Khan F, Ghaffar A, Hussain F, Cho SH. Finger-Counting-Based Gesture Recognition within Cars Using Impulse Radar with Convolutional Neural Network. Sensors. 2019;19(6):1429.

[40] Ghaffar A, Khan F, Cho SH. Hand Pointing Gestures Based Digital Menu Board Implementation Using IR-UWB Transceivers. IEEE Access. 2019;7:58148-57.

[41] Khan F, Leem SK, Cho SH. Hand-Based Gesture Recognition for Vehicular Applications Using IR-UWB Radar. Sensors. 2017;17(4):833.

[42] Leem SK, Khan F, Cho SH. Detecting Mid-Air Gestures for Digit Writing With Radio Sensors and a CNN. IEEE Transactions on Instrumentation and Measurement. 2020;69(4):1066-81.

[43] Zheng T, Chen Z, Zhang S, Luo J. Catch Your Breath: Simultaneous RF Tracking and Respiration Monitoring with Radar Pairs. IEEE Transactions on Mobile Computing. 2022;PP.

[44] Chikhi S, Amine A, Chaoui A, Kholladi MK, Saidouni DE. Modelling and Implementation of Complex Systems: Proceedings of the 4th International Symposium, MISC 2016, Constantine, Algeria, May 7-8, 2016, Constantine, Algeria: Springer International Publishing; 2016.

[45] Samani A. An Introduction to Signal Processing for Non-Engineers: CRC Press; 2019.

[46] Microsoft. IR-UWB based Indoor Localization System [Available from: <https://www.microsoft.com/en-us/research/wp-content/uploads/2016/11/Mridula.pdf>.

[47] Fernandes JR, Wentzloff D, editors. Recent advances in IR-UWB transceivers: An overview. Proceedings of 2010 IEEE International Symposium on Circuits and Systems; 2010 30 May-2 June 2010.

[48] Khan F, Ghaffar A, Khan N, Cho SH. An Overview of Signal Processing Techniques for Remote Health Monitoring Using Impulse Radio UWB Transceiver. Sensors. 2020;20(9):2479.

[49] Sarkar A, Ghosh D. Detection of Multiple Humans Equidistant From IR-UWB SISO Radar Using Machine Learning. IEEE Sensors Letters. 2020;4(1):1-4.

[50] N.AS. Single-Chip Radar Sensors with Sub-mm Resolution -XETHRU 2017 [cited 2023 9 Jan]. Available from: <https://www.xethru.com/>.

[51] Foundation RP. Teach, Learn and Make with RaspberryPi - Raspberry Pi 2021 [cited 2023 6 Jan]. Available from: <https://www.raspberrypi.org/>.

[52] Piccardi M, editor Background subtraction techniques: a review. 2004 IEEE International Conference on Systems, Man and Cybernetics (IEEE Cat No04CH37583); 2004 10-13 Oct. 2004.

[53] Yim D, Cho SH. An equidistance multi-human detection algorithm based on noise level using mono-static IR-UWB radar system. 2015:131-4.

[54] Khan F, Azou S, Youssef R, Morel P, Radoi E. IR-UWB Radar-Based Robust Heart Rate Detection Using a Deep Learning Technique Intended for Vehicular Applications. Electronics (Basel). 2022;11(16):2505.

[55] Khan F, Leem SK, Cho SH. In-Air Continuous Writing Using UWB Impulse Radar Sensors. IEEE Access. 2020;8:99302-11.

[56] Liu T, Wu C-C, Huang K-C, Liao J-J. Effects of frequency and signal-to-noise ratio on accuracy of target sound detection with varied inferences among Taiwanese hearing-impaired individuals. Applied Acoustics. 2020;161:107176.

[57] Yoo S, Chung S, Seol D-M, Cho SH. Adaptive Clutter Suppression Algorithm for Detection and Positioning using IR-UWB Radar2018. 40-3 p.

[58] Barton DK. Modern Radar System Analysis: Artech House; 1988.

[59] Khan F, Azou S, Youssef R, Morel P, Radoi E, Dobre O, A. An IR-UWB multi-sensor approach for collision avoidance in indoor environments. IEEE Transactions on Instrumentation and Measurement. 2022.

[60] Nguyen V-H, Pyun J-Y. Location Detection and Tracking of Moving Targets by a 2D IR-UWB Radar System. Sensors. 2015;15(3):6740-62.

[61] Nar F, Okman OE, Özgür A, Çetin M. RmSAT-CFAR: Fast and accurate target detection in radar images. SoftwareX. 2018;8:39-42.

[62] Antonik P, Bowles B, Capraro G, Hennington L, Koscielny A. Intelligent use of CFAR algorithms. Interim Report, Jan - Sep 1992 Kaman Sciences Corp, Utica, NY. 1993.

[63] Hatem GM, Abdul Sadah JW, Saeed TR. Comparative Study of Various CFAR Algorithms for Non-Homogenous Environments. IOP Conference Series: Materials Science and Engineering. 2018;433(1):012080.

[64] Mahafza BR. Radar Systems Analysis and Design Using MATLAB Third Edition: Taylor & Francis; 2013.

[65] MATLAB. Constant False Alarm Rate (CFAR) Detection [cited 2023 13 Jan]. Available from: <https://www.mathworks.com/help/phased/ug/constant-false-alarm-rate-cfar-detection.html>.

[66] Hameed SW. Peaks Detector Algorithm after CFAR for Multiple Targets Detection. EAI Endorsed Transactions on AI and Robotics. 2022;1(1):e11.

[67] Levanon N. Radar Principles: Wiley; 1988.

[68] Hsu C-Y, Ahuja A, Yue S, Hristov R, Kabelac Z, Katabi D. Zero-Effort In-Home Sleep and Insomnia Monitoring using Radio Signals. Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies. 2017;1:1-18.

[69] Nijsure Y, Tay WP, Gunawan E, Wen F, Yang Z, Guan YL, et al. An impulse radio ultrawideband system for contactless noninvasive respiratory monitoring. IEEE transactions on bio-medical engineering. 2013;60(6):1509-17.

[70] Yoon BJ. Hidden Markov Models and their Applications in Biological Sequence Analysis. Current genomics. 2009;10(6):402-15.

[71] Tong JC, Ranganathan S. 5 - Computational T cell vaccine design. In: Tong JC, Ranganathan S, editors. Computer-Aided Vaccine Design: Woodhead Publishing; 2013. p. 59-86.

[72] Hao J, Yuan X, Yang Y, Wang R, Zhuang Y, Luo J. Visible Light Based Occupancy Inference Using Ensemble Learning. IEEE Access. 2018;6:16377-85.