Politecnico di Torino

Master's degree program in Electronic Engineering
(Embedded Systems Track)

Localization and identification of people in closed spaces with low-power and low-cost environmental and wearable sensors

Master's Degree Thesis

Supervisor:	Candidate:
Prof. Mihai Teodor LAZARESCU	Haifeng Yang
Prof. Luciano LAVAGNO	s301567@studenti.polito.it

November 2025

Acknowledgements

I would like to thank my supervisor, Professor Lazarescu Mihai Teodor, and Professor Lavagon Luciano.

Abbreviations

Abbreviation	**Full Form**
RR	Respiratory Rate
HR	Heart Rate
BMP	Breath Per Minute
PSD	Power Spectrum Density
PCA	Principal Component Analysis
ROI	Region Of Interest
FFT	Fast Fourier Transform
TOF	Time Of Flight
PCA	Principal Component Analysis
UKF	Unscented Kalman Filter
KF	Kalman Filter
EKF	Extend Kalman Filter
SNR	Signal Noise Ratio
AR	Autoregressive
MA	Moving Average
ARMA	Autoregressive Moving Average
MA	Moving Average
ESPRIT	Estimation of Signal Parameters via Rotational Invariance Technique
MUSIC	MUtiple SIgnal Classification
SPAD	Single Photon Avalanche Diode
NVM	Non-Volatile Memory
CRLB	Cramér Rao Lower Bound
SVD	Singular Value Decomposition
MLE	Maximum Likelihood Estimation

List of Figures

List of Tables

Contents

CHAPTER 1

Introduction

1.1 Introduction

In this study, we aim to identify the person in a closed room by testing vital signs. By testing for critical signs, we may be able to determine the person's location. A human's body's respiratory signal can be tested by observing its appearance outside. According to the modern power spectrum density analysis, we can detect multiple people's respiratory rates in the same field of view. Such as the Multiple Signal Classification technique.

Furthermore, respiratory rate detection plays a crucial role in the medical field, as it reflects an individual's health status, such as physical activity levels. While the respiratory rate does not necessarily decrease with age, vital lung capacity often declines. The decrease in body metabolic level indicates reduced oxygen consumption. Most human cells contain mitochondria, and energy production from mitochondria relies on oxygen. Medical studies show that modern diseases are related to reduced mitochondrial activity level, causing some diseases such as diabetes and emotional depression. The lower lung capacity of elderly people leads to a lower SNR of the signal, which increases the challenge of our design.

Monitoring lung activity level can help infer the oxygen consumption, such as estimating the body's metabolic level. Furthermore, monitoring respiratory rate helps in routine healthcare. Compared to other vital signs like blood glucose or blood pressure, respiratory activity can be observed more easily through external appearance. A non-invasive and convenient method for measuring respiratory rate would be both practical and economical.

Currently, devices such as spirometers are commonly used for respiratory detection. However, accurately measuring respiratory rate remains challenging. Medical monitoring can be a typical application scenario, e.g., in Intensive Care Units and among older adults. Furthermore, fatigue is typically associated with low RR, making our study suitable for use as a driving fatigue monitoring system.

1.2 Paper structure

The rest of this paper's structure is as follows: Chapter 2: Review the respiratory monitor and TOF technique, and related signal processing algorithm(The Least square estimate, Kalman filter, PCA). Related state-of-the-art and discussion.

- Chapter 3 and Chapter 4: Comparison between different sensors, and details in sensors' characteristics, investigating the state of the art with potential innovation.
- Chapter 5: Details system hardware configuration, software program flow, and central algorithms.
- Chapter 6: Presents experimental results under different conditions, analyzing the performance of the parametric method, FFT method, and eigen data analysis for frequency.
- Chapter 7: Discusses system limitations and their reasons, providing some hypotheses for improvement.

CHAPTER 2

Literature Review

This study focuses on detecting respiratory signals with the VL53L sensor. We chose to detect the respiratory rate as a vital sign, so as to identify people. In both academic and commercial fields, the VL53L sensor is typically chosen for hand gesture motion pattern identification or SLAM projects(Simultaneous Localization and Mapping), such as sweeper collision avoidance. In [3], it is relevant to our topic. There are worthwhile materials covering random processes, joint probability, linear systems, and space vectors. The classic least squares estimation algorithm is practical for tracking moving objects and smoothing images.

Understanding the equations in [3] is also valuable to comprehend the algorithm. Data fitting relies heavily on prediction, even though our current time-of-flight (TOF) system is only in detection mode and does not implement system control that would alter detection, such as in a motor speed control system. However, one-step ahead prediction is used to estimate the parameters in the Burg method. Within the data buffer, the Burg method compares the predicted value with the actual value to determine optimal parameters.

Reviewing the literature provides great inspiration for our project, such as works on dynamic state estimation. In this section, we sort out some commonly used techniques related to our project.

The Least square estimator

The Least Squares Algorithm was invented by Gauss when he observed the movement of celestial bodies and discovered patterns. Based on this algorithm, there is a parameter estimation method that minimizes the sum of squared residuals between the tested and fitted values. It does not rely on probabilistic assumptions or specific signal characteristics such as the mean, making it particularly suitable for data fitting applications.

Weiner filter

The Wiener filter [3, chapter 3], developed by mathematician Norbert Wiener, was one of the earliest methods used in system control. It forms the foundation of several other filters, like adaptive filters. It inspired subsequent algorithms, such as the Kalman filter, and is itself rooted in least squares estimation.

Before the Wiener filter, people often use filters as tools for smoothing signals. The Wiener algorithm's innovation introduces the concept of mathematical expectation into the computation of filter equations. The method applies the autocorrelation matrix and deviation to FIR filters. The total deviation corresponds to the noise term added into the FIR filter in digital signal processing. In detail, the total deviation is related to the total power.

The filter is no longer simply a frequency sorter, but rather a statistical optimizer. Through the solution of minimization of the mean square error (MMSE) to find the best coefficients, this technique originates from LSE but advances toward optimal estimation rather than mere smoothing.

Meanwhile, the autocorrelation matrix is the outer product of the sample vector with two different time-shifted versions. The one-step-ahead prediction inspired the Levinson-Durbin algorithm to compute the number of taps in the filter efficiently. The Levinson-Durbin algorithm utilises the symmetric diagonal property of the autocorrelation matrix. Mathematically, inverting a matrix is computationally intensive. The Levinson-Durbin algorithm cleverly avoids this inversion by recursively updating the prediction coefficients.

Based on the Wiener-Khinchin theorem, the power spectrum is the Fourier transform of the autocorrelation. Finite data length estimation can only provide finite lags of autocorrelation, which may lead to decreased frequency resolution. The Burg method proved that the M-th order autoregressive model power spectrum is precisely equal to the maximum entropy power spectrum [4]. To find the best coefficients in the AR model, the Burg method adopted the Levinson-Durbin algorithm to avoid matrix inversion.

Kalman filter [3, chapter 5]

The Kalman filter is a recursive algorithm that estimates the state of a linear dynamic system from a series of noisy measurements, widely used in navigation, such as the Apollo moon landing program. The Kalman filter performs prediction and update.

Based on RLS, the Kalman filter has three advanced components:

- 1, dynamics model;
- 2, processing noise estimation;
- 3, state estimation.

These components make the Kalman filter more sophisticated, evolving from a parameter estimator to a robust dynamic state estimator.

We considered the Kalman filter as a potential framework for our design at the beginning, which is inspired by [5]. However, we finally dropped the KF for the following reasons:

- People use the Kalman filter for aerospace control systems. The design is perfect for processing complex dynamic systems with specific state equations. The states can include position, velocity, and acceleration.
- We only have to produce a scalar quantity, which is respiratory frequency. We do not necessarily need to focus on the precise position, velocity, or acceleration of chest wall movement. Constructing a complex state equation for a simple frequency detection system would be excessive and could introduce instability and systematic errors.

Respiratory signals exhibit strong periodic characteristics, whereas the KF state model is typically adapted to track random movements, such as those of satellites. In contrast, our system is simple: the frequency changes slowly, and the rate of change is within a small margin. The respiratory rate will not change from 0.1 Hz to 2 Hz within 0.1 seconds.

Extended and unscented Kalman filter

The signals in nature being tested are to some degree linear, but not perfectly linear [3]. When the signal is highly nonlinear, using the standard Kalman filter may be insufficient for object tracking. The Extended Kalman Filter (EKF) and Unscented Kalman Filter (UKF) are variants of the standard Kalman filter designed to handle nonlinear signals. The EKF linearises the signal by shifting its nominal value. While the UKF samples deterministic sigma points and sums up these weighted points, allowing the mean value to adapt to non-Gaussian estimates. Thereby, the UKF provides a reasonable mean estimation.

In tracking periodic signals, the UKF can be faster than the MUSIC method [5] due to its iterative propagation of variance and mean values. However, the UKF does not achieve as high a frequency resolution as the MUSIC method. The UKF can also incorporate additional elements through the propagation rule. In [5], they put the frequency into the state equation. This approach makes the project convenient to summarise breathing signals [5]. Furthermore, they skipped the step of transforming the time into frequency.

Using PCA as a noise reduction method

In the digital signal processing (DSP) domain, PCA is referred to as the Karhunen–Loève transform (KLT), as they are mathematically equivalent. In signal processing, PCA tends to focus on cancelling correlation. In PCA, the principal components form a new set of orthogonal basis vectors. From the perspective of the KLT, the random process is transformed into a linear combination of random variables, treating the signal as a random variable. In the section 5.3.5.4, we discuss the random process can be transformed into a linear combination of random variables.

Linearity in Kalman filtering

From the summaries of [3, chapter 1], we learn that the linear theory of real-time detection.

In TOF-based chest activity detection, people often use filtering techniques in separating useful respiratory signals from noise and motion artifacts. The Kalman filter represents a classical Bayesian approach for state estimation. It offers a smooth solution to describe the problem under the assumption of linear dynamics and Gaussian noise.

In such a framework, the state of the system is modelled as a linear process corrupted by Gaussian process noise. And the TOF measurements are assumed to be linear observations with Gaussian measurement noise. For example, the state of the system can be the chest displacement over time. This formulation enables recursive estimation of the hidden respiratory state, updating predictions as new TOF data become available. However, the sensor captures the chest motion and produces an output that is not always perfectly linear.

Some studies employed nonlinear extensions such as the Extended Kalman Filter (EKF) or the Unscented Kalman Filter (UKF), such as [5].

From a broader Bayesian theory perspective, the Bayesian filter can be interpreted as updating a posterior probability distribution of the hidden respiratory state given the noisy TOF observations. Whereas the Unscented Kalman Filter (UKF) relies on deterministic sigma point sampling.

In this view, the Kalman filters are a special case of Bayesian filtering under Gaussian assumptions. More general Bayesian filters provide the flexibility to handle nonlinear and non-Gaussian conditions, though at a higher computational cost, such as particle filters.The key step in a Bayes filter is to estimate the probability distribution of the system state. The Bayes filter is very computationally expensive. However, the particle filter is an alternative, less computationally costly solution based on its nonlinearity processing property and a similar probability estimate strategy.

Linear system theory in TOF base detection

In this study, we operate within the framework of a linear system [3], utilizing a sensor that measures the chest wall and converts it to a frequency signal. The expansion and contraction of the chest lead to the periodic distance variance. This variance could be modeled as a linear time variant output system. The regularly breathing signal can be linearized approximately, allowing it to adapt some classical signal processing approaches effectively. The PSD estimation naturally involves the autocorrelation function. And the AR model relies on linear system assumptions for abstracting the principal frequency.

CHAPTER 3

Critical Discussion of the State-of-the-Art

We discuss some critical techniques and compare different sensors after viewing the relevant literature in this chapter.

3.1 Direct measurement versus indirect measurement

In general, infrared light-based time of flight sensors have two types of processing: intensity reflection (as in VL53L) and high-frequency modulated infrared [6] [7]. The TOF technique can be categorized into two main classes: direct and indirect [8].

- Direct TOF: The infrared light is emitted in the form of an impulse, directly measuring the photon's traveling time from transmission to reception. Based on the equation: $distance = c * t/2$, where c is the speed of light, and t is the traveling time, such that the depth data is calculated.
- Indirect TOF: The infrared light is emitted in the form of a sinusoidal-like signal. By measuring the phase difference between the modulated light wave and the received light wave, we can infer the flying time. And then we can convert the time into distance.

3.2 Sensor comparison and technique

TOF VS Stereo Vision

Stereo vision requires an algorithm to abstract from left and right images and then calculate depth through triangulation. This process involves massive computation and is time-consuming. Similar to human eyes, this camera relies heavily on ambient light. In contrast, the TOF sensor directly measures the time of flight and does not rely on ambient light to identify objects. However, ambient infrared light can sometimes cause a low SNR.

Colour cameras often adjust the entire or partial lens group to alter the focal length. This adjustment is similar to adjusting the thickness of a convex lens. In this way, we have a clear picture with detailed texture. The depth sensor does not need to change the focus length. The power consumption of focus adjusting has dropped, and the processing time can be eliminated. This power and processing time can be replaced by efficient data processing in a time-constrained scenario inference. The measurement precision of a TOF sensor depends only on the accuracy of its time measurement. In contrast, the precision of a binocular camera decreases as the object distance increases, because its precision is inversely proportional to distance.

Since TOF obtains depth information rather than RGB, it is more acceptable from a privacy perspective. Individuals who are detected are more likely to accept providing only depth data rather than full-color video in private spaces like toilets and bedrooms.

The sunlight naturally contains infrared components. These infrared light components become a significant source of noise for TOF systems. This noise can submerge the signal, which leads to a low SNR and reduced measurement accuracy. Optical filters can be used in this case, but their use introduces another problem: crosstalk, which requires calibration.

TOF VS mmWave Radar

TOF uses infrared light with a wavelength shorter than that of radar. Angle resolution and depth resolution of TOF are usually higher than those of radar. For measuring chest wall movement, the TOF is more sensitive to a minor position variance than the radar.

The TOF sensor directly outputs depth map information and does not require complex Doppler or phase demodulation like radar does. TOF sensors consume less power. Some consumer-grade TOF modules (VL53L7CX) are small and low-priced. They can be easily embedded into small phones and IoT devices, making them suitable for wearable devices due to their size.

Radar is not affected by ambient infrared light, has strong penetration ability, and can penetrate clothes and thin quilts. The measuring distance of radar is longer than the TOF, reaching up to tens of meters, while the TOF can only be 1 to 5 meters. For this reason, radar is usually adapted to autonomous driving.

Infrared TOF VS Ultrasonic sensors

Both of them are based on the time-of-flight technique. An ultrasonic sensor sends a sonic wave at a high frequency and counts the traveling time to determine the distance:

$distance = sonicspeed(340m/s) * time/2$. The difference in infrared light is the speed of light is $3 * 10^8 m/s$, $distance = 3 * 10^8 m/s * time/2$.
This leads to some key differences as follows: Tab 3.1:

Table 3.1: Comparison of Ultrasonic and Infrared light

character	**Ultrasonic**	**Infrared light**
precision	low(1-3 cm)	high(mm)
range	far(2cm-700cm)	within 500cm
measure speed	10 Hz	15-60 Hz
requires a medium to travel	yes	no
size	large	small
cost	low	high
object surface characteristics influence	less affected by the material and surface	more affected by the color and surface of the object

Ultrasonic can partially penetrate the object. Whereas, infrared light may not penetrate the thin clothes to the chest wall.

Wireless radio frequency(RF)

The wireless radio frequency is a continuous-wave form at a frequency of 9000 MHz. This signal is modulated waves, and the receive side is Software Defined Radio [1]. The people being tested are positioned between the transmitter and receiver, as illustrated in Figure 3.1. The frequency of the signal on the reception side leads to the variance of the radio wave amplitude. They were detecting the RR in this way.

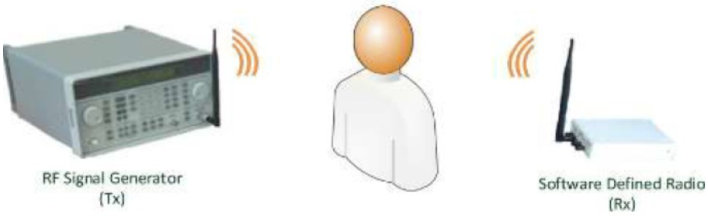


Figure 3.1: Radio frequency method testing respiratory[1]

Passive infrared sensor (PIR)

The PIR sensor can detect any object with a temperature above absolute zero (-273.15°C). Those object naturally radiates electromagnetic waves, and the energy of the waves is proportional to the temperature. Our study aims to detect the human body. Based on the previous description, the PIR sensor is a good candidate for our project. However, the PIR sensor cannot provide the distance information. Furthermore, the PIR sensor can only know the temperature of the vital sign. In this case, temperature information becomes less prompt when monitoring patients' vital signs in hospital wards. In contrast, by testing the respiratory signal, we can detect vital signs more likely in real time.

Commercial Depth Camera Case

Commercial-grade depth cameras are widely used in respiratory testing cases [9], such as the Kinect developed by Microsoft. Additionally, mobile phones have seen advancements in depth camera technology, notably with the Apple iPhone. Both of them utilized structured light technology.

Apple uses this depth camera technology for facial identification, e.g., phone unlocking and digital payment purchases. For sensitive applications like phone payment identification, spoofing is significantly mitigated by a depth sensor, which provides a natural physical defence. Unlike colour cameras, which some criminals exploit by using photos to replace faces and trick phones into making payments, the depth sensor provides enhanced security. Although the merchant systems introduce verification steps (e.g., blinking, turning the head motion). In contrast, the attackers have adapted their strategy by correspondingly offering video replays.

Depth cameras offer advantages in capturing 3D data. The high and low contours of the face are fully described, such as the nose and chin. Additionally, phones are usually brought out to some place outside, the scenario of a low-light environment is unavoidable, and a depth camera does not need to consider this drawback. Furthermore, in the case of image data leakage, criminals can only obtain 3D point cloud images, which reduces the potential misuse.

Consumers tend to prefer products with contactless technology, as demonstrated by the success of devices such as the Kinect and Apple iPhone. Furthermore, contactless devices reduce friction between moving physical parts, thereby extending the lifespan of the device.

In the area of the virtualisation process, a depth camera offers a convenient solution. When combining two images, those containing depth information can be easily merged.

In contrast, people are challenged in merging colour images because they don't have distance data.

Surface normals and 3D geometry provide inherently additive information compared to colour cameras. Surface normals and 3D geometry offer angular resolution in practical applications. RGB cameras do not have geometric data, but compensate by capturing more detailed surface textures. Human eyes rely on motion to perceive the shape and distance of objects, whereas depth sensors can determine these attributes even when the object is static. In scenarios involving colour cameras, distance is inferred based on movement. For example, objects closer to our eyes appear to move faster, while those farther away seem to move more slowly.

3.3 General TOF technique

Time-of-Flight (TOF) sensors measure the travel time of light or sound waves to estimate distance. In respiratory monitoring, TOF sensors can capture chest movements by analysing depth variations over time. Some respiratory rate detection algorithms are based on RGB-D cameras, which combine RGB images for edge detection and depth sensors for motion tracking. We can find the description of RGB-D cameras in the manual of the ST company.

People often use Principal Component Analysis (PCA) to extract the dominant motion component related to breathing, such as [9]. In [10], they also applied the Durbin–Watson test to estimate the residual.

3.4 ROI and transmission rate

Artifacts in measurement primarily arise from two sources:
- A single pixel capturing both foreground and background information.
- Object motion during the acquisition interval across a pixel matrix.

As noted in the VL53L manual [6], the Region of Interest (ROI) in the pixel matrix is configurable (e.g., 8×8 or 4×4). Using configuration 4×4 allows faster data transfer at a maximum of 60 frames per second, which is particularly useful in applications like automotive collision avoidance [11]. When the object is detected and begins to move closer to the sensor, the system can estimate its velocity and switch the ROI from 8×8 to 4×4. This adjustment increases the frame rate, such that we can perform higher frequency inference.

CHAPTER 4

Identification of an Innovative Gap

Traditional respiratory monitoring methods have some limited aspects. Examples include sensors that operate in a contact manner. Furthermore, performing contact measurement introduces potential errors. Therefore, developing signal processing with the TOF sensor is a significant point in the innovative gap, which enhances the sensitivity and robustness in a practical environment.

Multiple signal classification methods have been adapted to wireless radio frequency (RF) signals to monitor respiration rate [12]. Such approaches achieve high resolution frequency. In [12], they used radio frequency in their sensing system. compared to our case, [12] has a relatively high SNR.

4.1 Theoretical frequency resolution of FFT and parametric model

The Fourier-based methods are fundamentally limited by the observation window. This method results in a resolution proportional to $1/T$ where T is the data length. Unlike FFT, AR models can achieve superior resolution, particularly for short data segments. This higher resolution is not derived from the data length but from the signal's statistics and the model order.

Assume the AR(p) model according to the p value proceed the prediction:

$$x[n] = -\sum_{k=1}^{p} a_k x[n-k] + u[n]$$

Where a_k are the AR coefficients, $u[n]$ is white noise. Such that the power spectral density(PSD) is:

$$P_{\mathrm{AR}}(f) = \frac{\sigma^2}{|1 + \sum_{k=1}^{p} a_k e^{-j2\pi fk}|^2}$$

Where σ^2 is the noise variance.

The resolution of the AR model depends on the number of orders and signal-to-noise ratio (SNR). The latter section will also investigate SNR Chapter 6.6, which will prove that the resolution of frequency depends on the number of orders rather than the observation time. Resolution proximately can be described as:

$$\Delta f \propto \frac{1}{p \cdot \mathrm{SNR}}$$

From the equation above, the higher the number order or the higher the SNR, the better the resolution.

4.2 Machine learning

From a machine learning perspective, this study shares common aspects with methods such as polynomial regression. However, a notable limitation of machine learning is its high dimensionality and large number of parameters [13, introduction]. In contrast, classical digital signal processing techniques provide strong analytical interpretation, with clear connections to underlying physical models. Such as autoregressive (AR) modeling, Fast Fourier analysis, as well as some modern power spectrum density analysis [14], for example subspace method.

Machine learning demands significant computational resources and memory storage for inference. Furthermore, machine learning also poses implementation challenges in resource-constrained environments. In other words, machine learning achieves greater robustness than classical DSP approaches if a sufficient data set and a powerful computation device are provided. This fact is true especially when machine learning is dealing with complex and noisy real-world signals.

In the 1950s, the perceptron model was invented, inspired by the working mechanism of biological neurons. Related to DSP, the perceptron model behaves like a self-adaptive filter. The self-adaptive filter receives multiple input samples, assigns each input a specific weight, and then sums the terms to pass a threshold function. In this way, producing the inference. The training of the perception model adjusts the weights based on the output error. This phenomenon aligns with the processing of the least mean error of the self-adaptive filter.

In the history of machine learning, a period of stagnation occurred due to computational limitations. This stagnation was not relieved until the introduction of the GPU, which revitalized the field. Indeed, the technique of integrated chip manufacture has made a significant contribution to the success of machine learning.

CHAPTER 5

Methodology

5.1 Well-Motivated Choice of Methods and Approach

The main challenge of this project is detecting tiny frequency variations corresponding to the change of respiratory rate.

In our study, we frequently alternate between two phases of signal processing: spatial signal denoising and temporal frequency abstraction. In spatial signal denoise, we investigated improving the SNR of the signal. The reason for this fact is that we don't have a clear estimation of the upper bound, which is relative to the SNR of the spatial correlation. We have tried many approaches, such as the Pearson coefficient analysis and the PCA algorithm. However, after many attempts, we found that the spatial SNR has an upper bound due to its inherent characteristic property. The reason is that the respiratory frequency is low, which is close to the environmental frequency.

5.2 Methods

The sensing system detects small changes in the frequency of signals. The project is challenging, particularly in scenarios with noise. Non-parametric methods require a trade-off between frequency resolution and observation time, for example, the FFT method. The respiratory signal frequency range is 0.2 to 0.8 Hz. The observation time required to achieve sufficient frequency resolution with FFT is often impractical. Furthermore, non-parametric methods are also influenced by leakage of the power spectrum.

To overcome this limitation, we adopted the method of autoregressive model parametrization. Unlike FFT, the frequency resolution of AR modeling is not strictly limited by the observation time, but rather by the number of orders. This advantage enables the system to estimate respiratory frequency with a relatively higher resolution from a shorter data window, thereby achieving more accurate tracking of respiratory frequency over time. This model can produce a sharp peak of PSD, even in a noisy environment, and can also identify the relevant respiratory signal.

There are some similarities between the AR model and the FIR(Finite Impulse Response filter). From the perspective of pattern recognition, people majoring in electronics often regard an FIR merely as a device for selecting frequencies. But actually, an FIR filter can be viewed as a classical identification system. The FIR parameters can be treated as convolution weights, not only tuning the output response, but also representing the characteristics of the input signal itself in a broader sense.

5.3 Implementation

We employed an STM32 microcontroller and a VL53L7CX sensor with an 8x8 pixel array. In our study, we have been processing log data rather than the sensor's sampling stream in real time. Although processing log data can be more efficient for the preliminary checking of the algorithm, it needs more techniques to restore the actual behavior of the time series. For example, we can record the raw data of breathing specified as different RR, then apply different algorithms to these data to summarize the frequency. However, the raw data needs additional timestamp information, leading to a complex code structure. Otherwise, the time series analysis program doesn't know what the data points correspond to the time points. This situation leads to an uncertainty of two results. Either leads to a much longer processing time because the time interval is set too small, or to a shorter processing time but with a result with large error. This processing time is even longer than processing in real time with online sensor data.

For online data processing, execution time and memory are critical considerations for implementation on the STM32 platform.

5.3.1 Design and analysis system

The digital signal processing and testing procedure is outlined as follows:

- The system first detects chest wall distance using the STM32 embedded with the VL53L7 sensor. Data frames are transmitted via serial communication at a baud rate of 921600. The sampling rate of the sensor is 15Hz with respect to the 8x8 resolution.

- On the PC side, data is collected and processed using PyCharm, which provides graphical analysis capabilities for vector and matrix operations, constructing spatial and temporal processing frameworks.
- The software structure employs two synchronized threads: one thread for receiving data from STM32, the other thread specifically for data analysis.

The Numpy library significantly contributes to the project. The Numpy library helps us construct a compact and modularized organization with its powerful vector and matrix operations.

The initial algorithm development and modeling were conducted on a PC for greater convenience and flexibility. However, we still keep in mind that the ultimate goal of this study is to achieve respiratory pattern recognition directly on the STM32 with the depth sensor, thereby enabling pattern inference without relying on PC resources.

5.3.2 Calibration

Part-to-part variations result in different sensor outputs, so we necessarily calibrate the parameters. The calibration procedure is performed automatically when the sensor starts ranging. The procedure reads the data from NVM and then compensates for the measurement. The user can use the API to rewrite the calibration data into NVM for customized calibration.

RefSPAD

The ST Company calibrated the RefSPAD and stored the data in the NVM before shipping from the factory. The RefSPAD calibration only requires a few tens of milliseconds.

Offset

The calibration of offset and crosstalk should be performed in a dark environment without infrared interference. Offset calibration must be completed before crosstalk calibration.

- Place a gray target in front of the sensor at a distance of 140 mm.
- Start ranging and record the measured distance.
- Compare the measured distance with the actual distance of 140 mm.

- Normalize the ratio between the measured value and the standard value of 140 mm, and store this ratio in the NVM.

Crosstalk

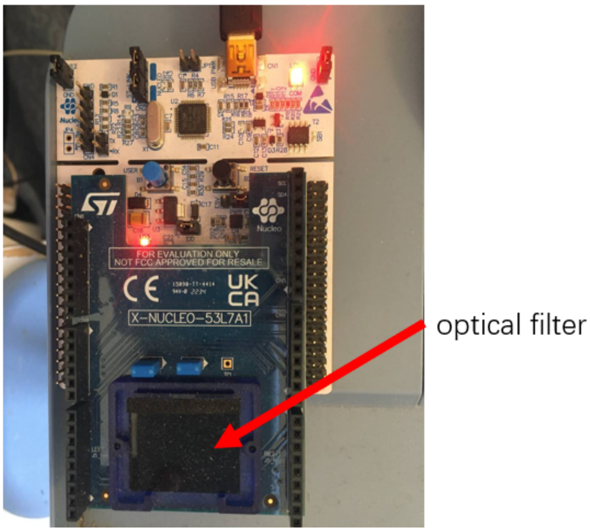
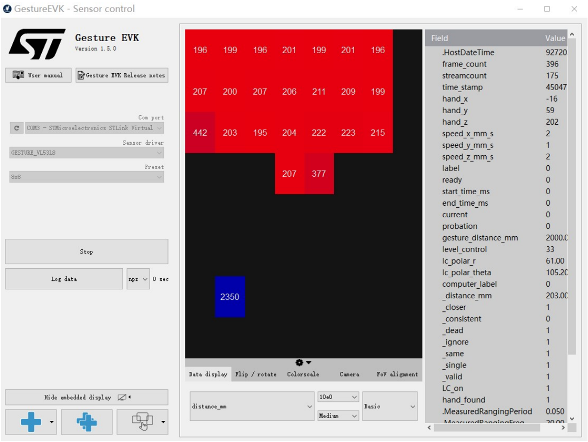


Figure 5.1: The optical filter on the PCB

Time-of-Flight (TOF) sensors typically incorporate a optical filter to reduce interference from ambient light. The optical filter shows as Figure 5.1. However, this filter can introduce crosstalk. Calibration can be performed to mitigate the crosstalk effect. After completing the offset calibration, multiple measurements are taken. The unit for crosstalk calibration is cps/SPAD (kilo counts per second per SPAD).

The ST company offers numerous demo codes, including hand gesture detection. It provides firmware not only for the microcontroller but also for the PC end graphic analyzer, as shown in Figure 5.2, which uses a heat map to display depth information intuitively. The PC application Gesture EVK can also perform crosstalk calibration Figure 5.3, and the calibration data can be recorded in NVM. The ST company also provided the STSW-IMG035 demo code for a Boolean valid pixel array. Another way to calibrate cross-talk is by sending a series of data to a microcontroller, as shown in Figure 5.4.



Field	Value
.HostDateTime	92720
frame_count	396
streamcount	175
time_stamp	45047
hand_x	-16
hand_y	59
hand_z	202
speed_x_mm_s	2
speed_y_mm_s	1
speed_z_mm_s	2
label	0
ready	0
start_time_ms	0
end_time_ms	0
current	0
probation	0
gesture_distance_mm	2000.0
level_control	33
lc_polar_r	61.00
lc_polar_theta	105.20
computer_label	0
_distance_mm	203.00
_closer	1
_consistent	0
_dead	1
_ignore	1
_same	1
_single	1
_valid	1
LC_on	1
hand_found	1
.MeasuredRangingPeriod	0.050

Figure 5.2: Pixel array as a heat map

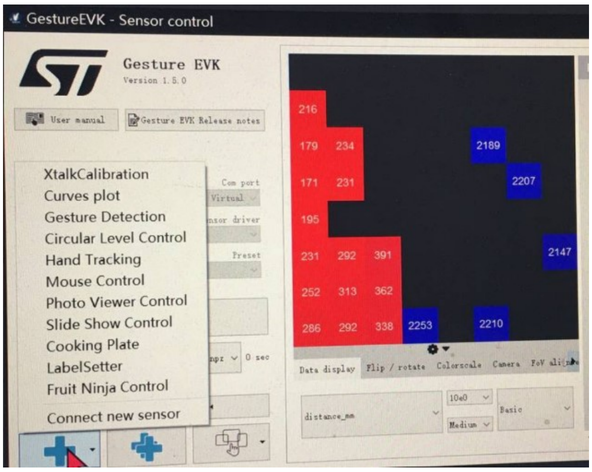
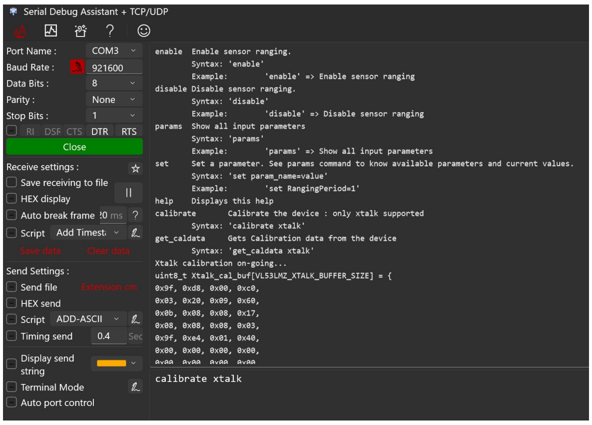


Figure 5.3: On the PC side, the Gesture EVK application is for crosstalk calibration.



Port Name : COM3
Baud Rate : 921600
Data Bits : 8
Parity : None
Stop Bits : 1
Script ADD-ASCII
Timing send 0.4

```
enable  Enable sensor ranging.
        Syntax: 'enable'
        Example:        'enable' => Enable sensor ranging
disable Disable sensor ranging.
        Syntax: 'disable'
        Example:        'disable' => Disable sensor ranging
params  Show all input parameters
        Syntax: 'params'
        Example:        'params' => Show all input parameters
set     Set a parameter. See params command to know available parameters and current values.
        Syntax: 'set param_name=value'
        Example:        'set RangingPeriod=1'
help    Displays this help
calibrate       Calibrate the device : only xtalk supported
        Syntax: 'calibrate xtalk'
get_caldata     Gets Calibration data from the device
        Syntax: 'get_caldata xtalk'
Xtalk calibration on-going...
uint8_t Xtalk_cal_buf[VL53LMZ_XTALK_BUFFER_SIZE] = {
0x9f, 0xd8, 0x00, 0xc0,
0x03, 0x20, 0x09, 0x60,
0x0b, 0x08, 0x08, 0x17,
0x08, 0x08, 0x08, 0x03,
0x9f, 0xe4, 0x01, 0x40,
0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00
```

```
calibrate xtalk
```

Figure 5.4: Series port communication on the PC side for crosstalk calibration

5.3.3 Spatial denoise

In this study, our essential task is to test the periodic movement of the chest wall. The movement is a relative distance rather than an absolute distance. So we only performed crosstalk for initialization, leaving the offset to the latter process. Furthermore, offset calibration is more complicated because there is no concrete demo code online for our specific version. We perform the detrending in the latter procedure, which can cancel out the offset error.

5.3.3.1 Anti artifact

Motion artifact arises from clothing movement or motion across multiple sample frames (the sample frequency is 15 Hz). In general, noise can be categorized into two parts: noise from a sensor(white noise) and process noise. Artifact belongs to process noise. We should adopt methods to mitigate artifacts.

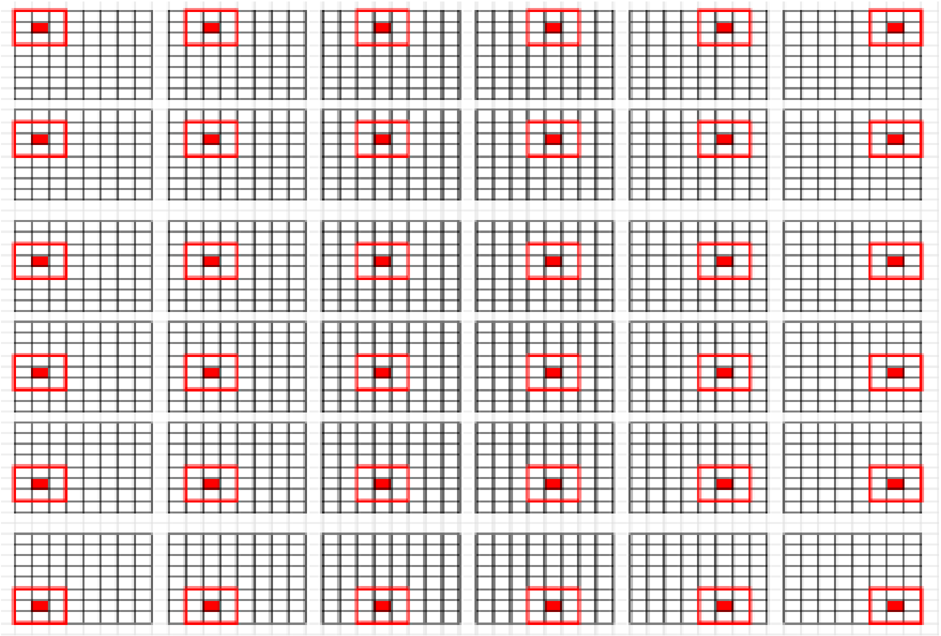


Figure 5.5: Analysing order

First, the program selects 9 pixels as shown in the red box in Figure 5.5 from the 8×8 pixel array. The selection order moves one pixel at a time from left to right and top to bottom.

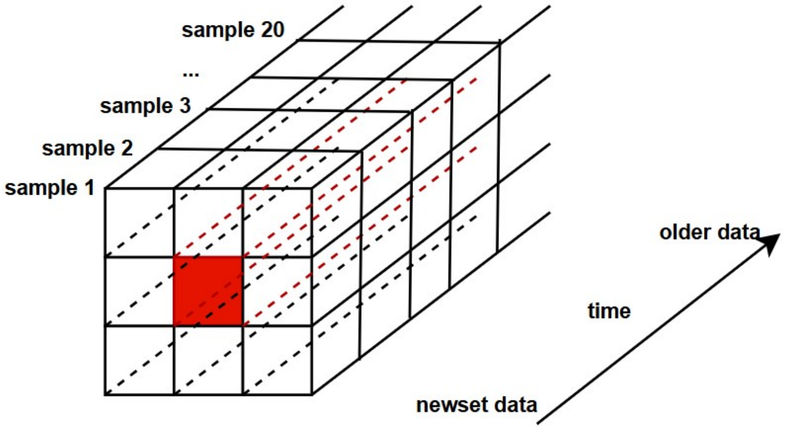


Figure 5.6: One region performs Pearson coefficient analysis

Next, one such region is selected containing 9 pixels as shown in Figure 5.6. Each pixel records 20 consecutive samples. The center pixel within this region performs Pearson correlation calculations with the surrounding 8 pixels. Each Pearson correlation calculation is performed on eight groups of historical data in left-to-right and top-to-bottom order. This process yields eight correlation coefficients. When the average of all eight coefficients is higher than 0.7, the position of this center pixel and its depth data are recorded.

There are up to 36 possible center pixels that can satisfy this criterion in each frame. After analyzing the frame, this current version of the program averages the depth data of the qualifying center pixels and records it. Whereas in the time series, weighted mean filtering is performed, as Figure 5.7 shows.

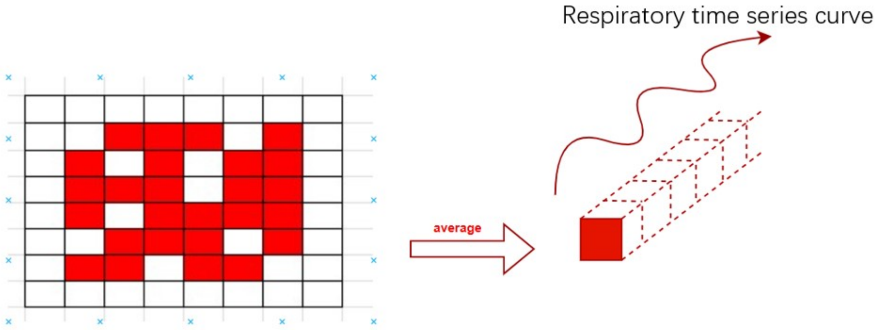


Figure 5.7: Averaging those pixels is sufficient for the threshold of the Pearson Coefficient.

Comparison with Pearson coefficients analysis and PCA

The PCA algorithm can be used to generate a time-series signal suitable for fitting the autoregressive model. Predictions can then be made based on this derived series and validated against newly acquired data.

Both Pearson correlation and Principal Component Analysis (PCA) rely on covariance structures for data analysis. However, the PCA often proves more efficient in capturing dominant variation patterns. Some studies have successfully employed PCA on point cloud data to extract respiratory signals, such as [15].

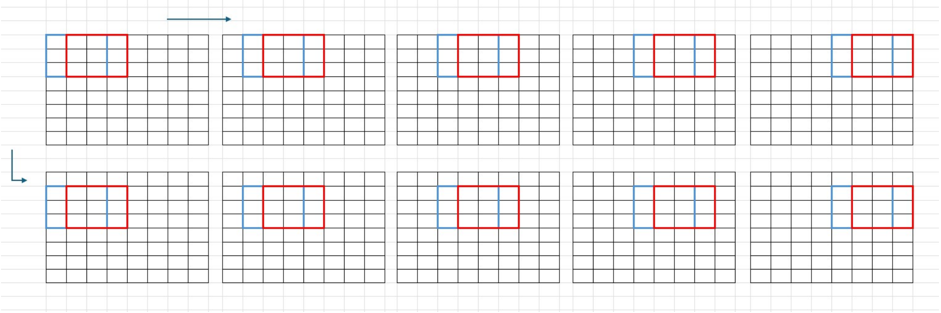


Figure 5.8: Analyzing order

Another way to spatially perform Pearson correlation

The previous section proposed a method combining time series and spatial correlation to filter out the artifact of motion. Alternatively, we can also perform pure spatial Pearson correlation analysis.

One frame with an example of 2 rows (the remaining six rows are not shown), as Figure 5.8 shows. The order of the Pearson correlation analysis proceeds from left to right and top to bottom. The pairs marked with red and blue boxes are analyzed and produce a correlation coefficient. Each row produces five coefficients. With six rows in total, that makes 30 coefficients. All the correlations described above are within a single frame, so this is not a time series analysis. The data is recorded in text form and was processed using a simple moving average analysis.

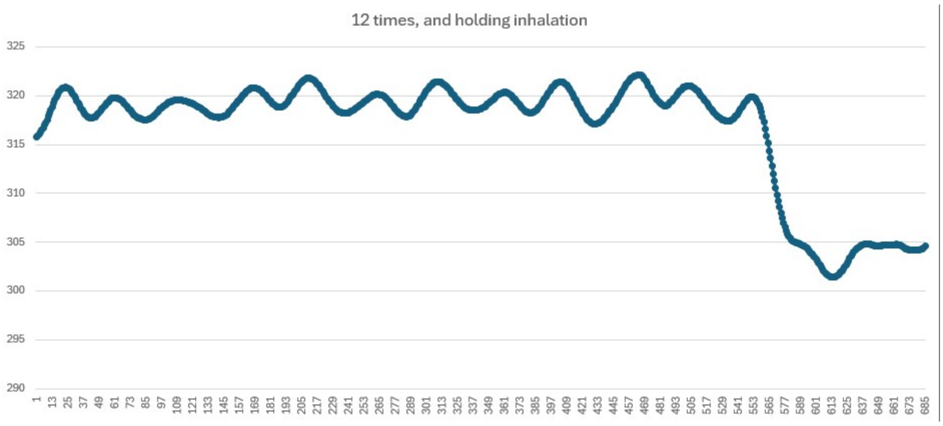


Figure 5.9: Ranging result of 12 breathing cycles

At the beginning of our exploratory testing, we were counting breaths from the log file diagram for confirmation. The Figure 5.9 shows the best measurement among others. We recorded the distance between the chest and the sensor regardless of the ambient light intensity. Then we smoothed the data using a moving average window with five samples. There were 12 breaths during testing, with a final inhalation hold of about 10 seconds. The curve looks relatively recognizable as a respiratory signal.

For the method of spatial Pearson correlation analysis we implemented, we only potentially solved the motion artifact. However, the essential challenge lies in detecting the respiratory frequency. The solution can be found through power spectrum analysis. Furthermore, the PSD provides the analysis of the signal with respect to its surrounding noise. The ratio between useful and non-useful frequencies helps assess whether our design implementation is effective. We have tried using Welch power spectrum analysis, but the output did not change with faster breathing—it remained the same.

5.3.3.2 Data fitting

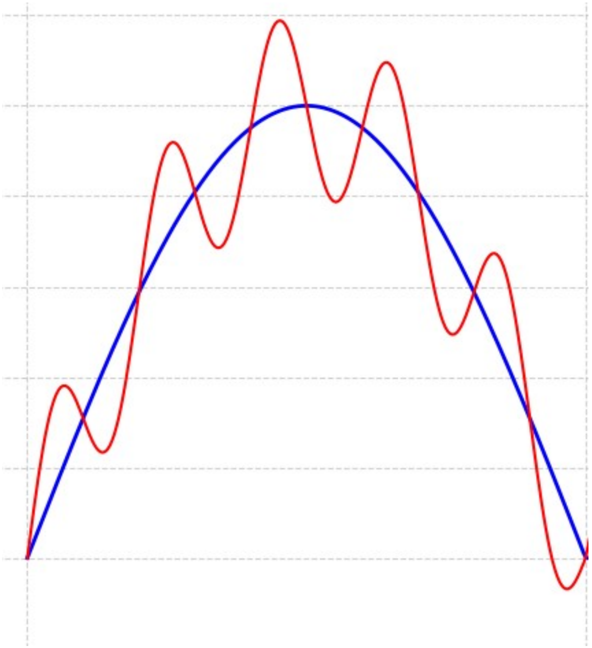


Figure 5.10: Fitting a line for the respiratory signal

Then we begin to think about how to improve time series analysis. The first thing we tackled was how the system distinguishes the distance between breath motion and body movement.

The curve of the time series data points (red curve) is not guaranteed to oscillate around a fixed horizontal line (blue curve) because the chest potentially moves toward or away from the sensor, as shown in Figure 5.10. In such cases, performing spectrum analysis to identify the frequency with maximum density in the respiratory signal becomes challenging. The solution is to fit a curve to the raw data around which the respiratory curve fluctuates. In other words, we need to find a fitted curve and subtract it from the original data points. In this way, we obtain a new curve with a period based on inhalation and exhalation.

The positive and negative peaks correspond to exhalation and inhalation, and these two peaks approximately have a mean of zero. The challenge lies in finding this curve. In time-series analysis, some implementations of the UKF algorithm also account for the extremely low-frequency component of the signal. They refer to the extremely low-frequency that we previously described as the direct current component in [5], which should be blocked.

Estimate the parameters of the trend[3, chapter 3] [2, chapter 4]

$$L = \sum_{i=1}^{n} (y_i - f(x_i))^2$$

Let's define $x_{1,1}, x_{1,2}, ..., x_{1,n-1}$ as n points sample originally, $x_{2,1}, x_{2,2}, ..., x_{2,n-1}$, represents the n points sample that sliding the window with one time unit after the pervious one, $x_{m,1}, x_{m,2}, ..., x_{m,n-1}$ represents the n points sample that sliding window with m time unit after the first one. L is the deviation of $f(x)$, we need to find $f(x)$ that the L is always the minimum. In Figure 5.11, the matrix is $m \times n$, where n represents n time points sampling data. In other words, the model needs n sample points as a buffer for initialization. The buffer length is n, forming a first-in-first-out queue. And m represents m samples after the initial sampling. The graphic illustration is shown as Figure 5.11.

$$\begin{aligned}
h_1 &= \theta_0 + \theta_1 x_{1,1} + \theta_2 x_{1,2} + \ldots + \theta_{n-1} x_{1,n-1} \\
h_2 &= \theta_0 + \theta_1 x_{2,1} + \theta_2 x_{2,2} + \ldots + \theta_{n-1} x_{2,n-1} \\
&\vdots \\
h_m &= \theta_0 + \theta_1 x_{m,1} + \theta_2 x_{m,2} + \ldots + \theta_{n-1} x_{m,n-1}
\end{aligned}$$

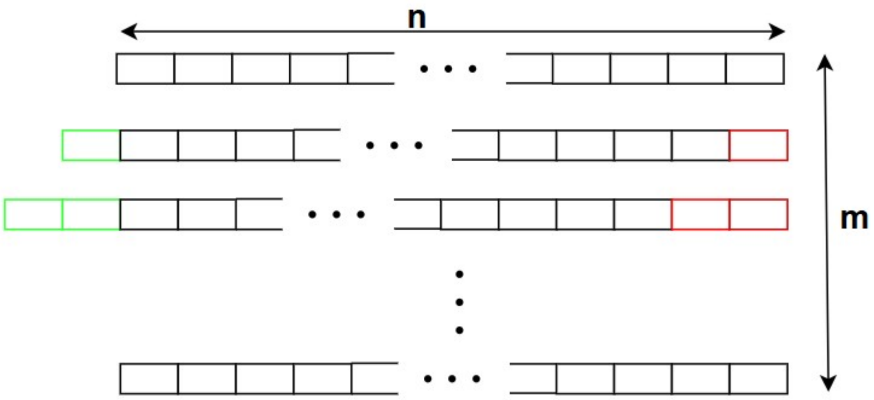


Figure 5.11: Data buffer as matrix form

In matrix form is $\mathbf{h} = \mathbf{X}\theta$, where $\mathbf{h}$ is a vector with m elements, representing the curve we want to find that describes the trend of the original curve. This is the ideal curve, θ is a vector with n elements representing the parameters of the model, $\mathbf{X}$ is an $m * n$ matrix, m represents the number of samples, and n represents the n points

sampled.
Let's define Y as the real output value. The target loss function is expressed as a matrix:

$$\mathbf{J}(\theta) = \|\mathbf{h} - \mathbf{Y}\|^2 = \|\mathbf{X}\theta - \mathbf{Y}\|^2 = (\mathbf{X}\theta - \mathbf{Y})^T(\mathbf{X}\theta - \mathbf{Y})$$
$$\mathbf{J}(\theta) = \theta^{\mathbf{T}}\mathbf{X}^{\mathbf{T}}\mathbf{X}\theta - \mathbf{2}\theta^{\mathbf{T}}\mathbf{X}^{\mathbf{T}}\mathbf{Y} + \mathbf{Y}^{\mathbf{T}}\mathbf{Y}$$

Take the derivative and make it equal to 0:

$$\frac{\partial \mathbf{J}(\theta)}{\partial \theta} = \mathbf{2}\mathbf{X}^{\mathbf{T}}\mathbf{X}\theta - \mathbf{2}\mathbf{X}^{\mathbf{T}}\mathbf{Y} = \mathbf{0}$$

$$\theta = (\mathbf{X}^{\mathbf{T}}\mathbf{X})^{-\mathbf{1}}\mathbf{X}^{\mathbf{T}}\mathbf{Y}$$

Where θ is the parameter of the linear equation describing the trend of the data. Then the newly coming sample data is as follows:

$$\mathbf{v_{m+1}} = x_{m+1,1}, x_{m+1,2}, ..., x_{m+1,n-1}$$

In this way, we plug in new sample data with parameters θ obtained:

$$\mathbf{h_{m+1,trend}} = \theta_0 + \theta_1 x_{m+1,1} + \theta_2 x_{m+1,2} + \ldots + \theta_{n-1} x_{m+1,n-1}$$

Where $\mathbf{h_{m+1,trend}}$ is the data point describing the trend of the original data.
The newly sampled data subtrcut the data of trend: $\mathbf{d_{m+1,trend}} = \mathbf{v_{m+1}} - \mathbf{h_{m+1,trend}}$.
And $\mathbf{d_{m+1,trend}}$ is the point that oscillates around the horizontal line, instead of the random curve caused by random body movement.

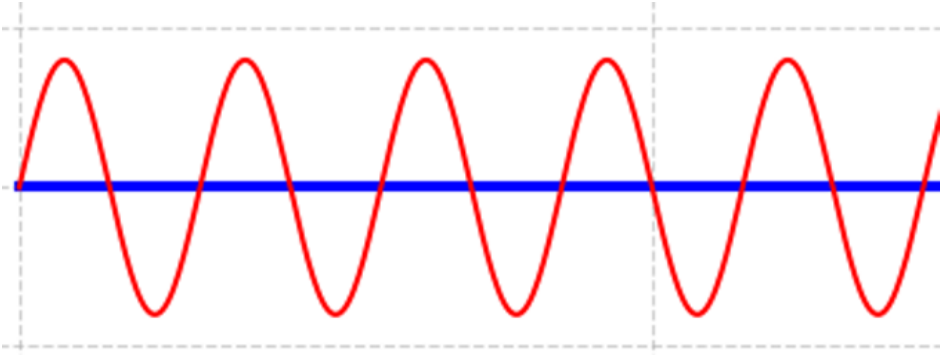


Figure 5.12: The respiratory curve oscillates on the straight horizontal line

As Figure 5.12 shows, the data points after subtracting the trend line result in a signal fluctuating around a horizontal line. Up to this step, there is a simple manner to detect respiration by recording the timestamps when the curve crosses the horizontal line. In this way, determine the period, and then convert it to frequency.

5.3.3.3 Mathematical meaning

When fitting a line to data points, it is often impossible to have all points lie exactly on the line. Instead, we can set the goal to minimize the sum of the distances from each point to the line. From a geometric perspective, this is equivalent to finding the vector that represents the projection of the data onto the plane defined by the points.

5.3.4 Spectrum analysis

The required output resolution of respiratory frequency is approximately 0.05 Hz to ensure a satisfactory product experience. The typical respiratory frequency range is 0.20 Hz to 1 Hz, while the sampling frequency is only 2 Hz. Using FFT, it is not possible to achieve 0.05 Hz resolution with 10 seconds of sampling data. Instead of performing an FFT, we use a parametric method to describe the time series. The previous section discussed the model describing the trend of the observation, which aims to minimize the extremely low frequency of body movement. In this section, we aim to find a model that describes the similarity between pattern versions—i.e., the correlation of each sample, rather than the trend of the samples. This is the reason for using two models to estimate the observation.

In Section 2, we mentioned the Kalman filter, and it indeed has fast inference. Additionally, the study of [5] reveals that the modified unscented Kalman filter had the most satisfying experimental results among others. The other algorithms include the MUSIC, ESPRIT, and FFT. However, the sensor they used has a high SNR, so noise is not considered. In our case, with the TOF sensor, we should use the AR model or the subspace system identification method.

Relationship between autocorrelation and power spectrum density

The PSD can help us find the lost periods. The respiratory curve may accidentally coincide with some interference if we begin directly from the time domain and convert the period to frequency. The system might produce twice the time period. PSD can avoid this problem.

The PSD and autocorrelation function are closely related. Analyzing PSD is essentially analyzing correlation [16]. The Wiener-Schinchin theorem:

$$r_{xx}(m) = E[x(m)x(m+k)] \tag{5.1}$$

$$P_{XX}(f) = \sum_{m=-\infty}^{\infty} r_{xx}(m)e^{-j2\pi fm} \tag{5.2}$$

Where $r_{xx}(m)$ is the autocorrelation function. The autocorrelation function describes the similarity between a signal and a later moment of itself. Whereas $P_{XX}(f)$ is the power-spectral density, describing how the power is distributed across the frequency domain. Power density spectrum is the Fourier transform of the autocorrelation function.

System identification[17, chapter 1]

The PSD is an intuitive way to analyse the system. Furthermore, we have a more general way to describe the input signal, which is by identifying the system. The model of a system is constructed from its parameters, which reveal the system's frequency. Additionally, the parameters can also tell us the dynamic characteristics, such as convergence and the speed of response.

As the filter we are familiar with, the Finite Impulse Response (FIR) with its parameters is used merely for frequency domain analysis. However, in this study, the parameters in the model are dynamically changed, which are updated along with each sample.

5.3.5 PSD derived from the AR model

Based on the autocorrelation of each time point, we can formulate a model similar to a Finite Impulse Response (FIR) filter, characterized by parameters and an additive residue term. The input time series data is convolved with the parameters. This model is autoregressive [18]. The autoregressive moving average(ARMA) process has more terms of noise compared to the AR model.

$$x[n] + a_1 x[n-1] + \cdots + a_p x[n-p] = e[n] + b_1 e[n-1] + \cdots + b_q e[n-q]$$

Where $e[n] + b_1 e[n-1] + \cdots + b_q e[n-q]$ are the noise of ARMA(p,q). In general, the ARMA provides a more accurate representation of a physical process than AR. However, the AR model is more acceptable in practical scenarios. In our study, we first consider the AR model rather than the ARMA, due to its simplicity and stability.

Assume time series data $X_1, X_2, ..., X_T$. First compute the mean:

$$\overline{X} = \frac{1}{T} \sum_{t=1}^{T} X_t$$

Estimate the σ^2 of white noise w_t: AR(p) defined as:

$$X_t = w_t + \phi_1 X_{t-1} + \cdots + \phi_p X_{t-p}$$

w_t is white noise, mean 0, square deviation is σ^2.
through autocovariance:

$$k_x(0) = \sigma^2 + \phi_1 k_x(1) + \cdots + \phi_p k_x(p)$$

Such that:

$$\sigma^2 = k_x(0) - \sum_{i=1}^{p} \phi_i k_x(i)$$

Where $k_x(h)$ is lag h autocovariant deviation(i.e, $k_x(h) = \mathrm{Cov}(X_t, X_{t+h})$. In the signal analysis domain, the covariance is noted as k.

Estimate the parameter ϕs [1]

Compute the expectation of X_{t-1} and X_t:

$$E[X_{t-1}X_t] = E[x_{t-1}(w_t + \phi_1 X_{t-1} + ... + \phi_p X_{t-p})]$$

Yule-Walker equation:

---

[1] [16, Chapter 9][19, Chapter 12]

$$\begin{aligned}
k_x(1) &= \phi_1 k_x(0) + \phi_2 k_x(1) + \cdots + \phi_p k_x(p-1) \\
k_x(2) &= \phi_1 k_x(1) + \phi_2 k_x(0) + \cdots + \phi_p k_x(p-2) \\
&\vdots \\
k_x(p) &= \phi_1 k_x(p-1) + \phi_2 k_x(p-2) + \cdots + \phi_p k_x(0)
\end{aligned}$$

In matrix form

$$\mathbf{K} = \boldsymbol{\Gamma}\boldsymbol{\phi}$$

Where: $\mathbf{K} = \begin{pmatrix} k_x(1) \\ k_x(2) \\ \vdots \\ k_x(p) \end{pmatrix}$ auto covariant, vector $\boldsymbol{\phi} = \begin{pmatrix} \phi_1 \\ \phi_2 \\ \vdots \\ \phi_p \end{pmatrix}$ parameters to to estimate.

$\boldsymbol{\Gamma} = \begin{pmatrix} k_x(0) & k_x(1) & \cdots & k_x(p-1) \\ k_x(1) & k_x(0) & \cdots & k_x(p-2) \\ \vdots & \vdots & \ddots & \vdots \\ k_x(p-1) & k_x(p-2) & \cdots & k_x(0) \end{pmatrix}$ is Toeplitz matrix.

Linear algebra solution:

$$\hat{\boldsymbol{\phi}} = \hat{\boldsymbol{\Gamma}}^{-1}\hat{\mathbf{K}}$$

Plug in the parameter to the following equation to obtain the PSD:

$$P(f) = \frac{\sigma^2}{|1 - \sum_{k=1}^{p} \phi_k e^{-j2\pi f k}|^2}$$

A pure AR model does not include an MA term. In contrast, the Least Squares Modified Yule-Walker method estimates the AR parameters based on the residuals and also estimates the MA parameters. In [19, Chapter 8, experiment], the Least Squares Modified Yule-Walker has satisfying experimental results, which are comparable with the SVD-ESPRIT method.

5.3.5.1 Parameter Estimation Based on the Burg Method

As we have derived the Yule–Walker equation from the correlation function and the Toeplitz matrix. We can see that the calculation is dense in the inverting matrix, so we introduced the Burg method. The Burg method contains the Levinson–Durbin algorithm for estimating the parameters. The Levinson–Durbin algorithm relies on forecasting based on historical data. The Levinson–Durbin algorithm is a highly effective approach that can be adapted to estimate the parameters of the linear model. For the details of the Burg method, refer to [18][20].

5.3.5.2 Levinson Durbin iteration

The core of the Burg method is forward and backward prediction, which replaces the inverse operation of the autocorrelation matrix. Here is a brief description of this algorithm [16, Chapter 8] [2, chapter 5][20].

The Levinson-Durbin algorithm begins by setting the expectation: $E^0 = r_{xx}(0)$ $r_{xx}(0)$ is the correlation function of x in lag zero. At this time, the error is the signal itself and makes no prediction either.

Recursive loop: For i =1 to P

- 1. Numerator of the Reflection Coefficient:

  $$\Delta^{(i-1)} = r_{xx}(i) - \sum_{k=1}^{i-1} a_k^{(i-1)} r_{xx}(i-k)$$

  Which estimates the residue between the current signal $x(n)$ and the past signal $x(n-1)$.

- 2. Compute the Reflection Coefficient $k_i = -\frac{\Delta^{(i-1)}}{E^{(i-1)}}$ k_i is the reflection coefficient. In this way, to normalize the residual correlation from the previous step into a form of the current power prediction error. The absolute value of k_i is always less than one. This is the reason for the stability of the AR model.

- 3. Update the i-th coefficient for the current order $a_i^{(i)} = k_i$
  The last coefficient of the current model order i equals the reflection coefficient k_i.

- 4. Update all other coefficients for the current order
  $a_j^{(i)} = a_j^{(i-1)} - k_i a_{i-j}^{(i-1)} \quad (1 \leq j \leq i-1)$
  Following the rule of the coefficient update. By using the reflection coefficient k_i to compute the other coefficients from the previous model order (i-1). This is the interpretation of minimizing both forward and backward prediction errors.

- 5. Update the Prediction Error Power

$E^{(i)} = (1 - k_i^2)E^{(i-1)}$
Because $|k_i| < 1$, the error power E^i decreases as the model order i increases. By adding a new term to the model, the prediction error power is definitely decreased. The reduction is proportional to the factor of$(1 - k_i^2)$. When the iteration is finished, we obtain the p-order AR coefficients.

5.3.5.3 MLE method estimate AR(1) parameters

This study explored the MLE (maximum likelihood estimation) method for parameter estimation. Its estimation involves linearization, because the practical application may be unstable. This method relies more heavily on probability theory than the Yule-Walker method. The details about how the MLE method estimates AR(1) parameters can refer to [21, Chapter 5.2, pages 118-125], which involves AR coefficients ϕ and error deviation σ^2.

The AR(1) process with a mean, under the assumption of normal distribution. Since the AR process has Markov characteristics, we can use the joint likelihood function. The method involves complicated nonlinear equations. Through the transformation of parameters, we can convert the AR(1) model to the form of a standard linear regression. After the transformation, we can use the least squares to estimate the parameters.

5.3.5.4 Derivation of AR model

The AR model is a special case of a linear filter model. The signal z_t can be expressed by a linear combination of the observations. Understanding this concept can help estimate the parameters of an AR model [22, Introduction]. In the data analysis area, the input signal series is also referred to as a regressor. Here we note it as a_t, and ϕ is the parameter.

The following equations derive the formal linear filter model from the AR model. The linear filter model is defined as:
$z_t = \mu + a_t + \phi_1 a_{t-1} + \phi_2 a_{t-2} + \ldots$
The AR model is defined as:
$\hat{z}_t = \phi_1 \hat{z}_{t-1} + \phi_2 \hat{z}_{t-2} + \ldots + \phi_p \hat{z}_{t-p} + a$
By eliminating the $\hat{z}_{t-1}$ on the right, have:
$\hat{z}_{t-1} = \phi_1 \hat{z}_{t-2} + \phi_2 \hat{z}_{t-3} + \ldots + \phi_p \hat{z}_{t-p-1} + a_{t-1}$
The same for $\hat{z}_{t-2}$, and so on, have the form of a linear filter model, with the ϕ having a large exponent then the system can be convergent, $\hat{z}_t$ is the combination of random shock a_t:
$\hat{z}_t = \phi^{m+1} \hat{z}_{t-m-1} + a_t + \phi a_{t-3} + \phi^2 a_{t-2} + \ldots + \phi^m a_{t-m}$ This derivation shows that a finite-order AR model can be equivalently expressed as an infinite-order Moving

Average (MA) model.

5.3.5.5 Conclusion

We can see that the PSD analysis method overlaps with the probability theory of statistics literature. We have the derivation of the relation between them in section 5.3.5.4 of this paper. A summary is that the useful signal is the sum of weighted noise, which is not intuitive. In practice, the most important job of our study is estimating the parameters of the model. Different methods for estimating the AR model parameters are shown in Table 5.1.

Table 5.1: Comparison of AR parameter estimation methods

Method	**Advantages**	**Disadvantages**
Yule–Walker	Directly based on autocorrelation. This is a simple implementation.	The autocorrelation matrix may be ill-conditioned. The performance is not always stable.
MLE (Least Squares)	Statistically efficient under Gaussian noise	Higher computational cost; does not guarantee model stability.
Burg	Guarantees a stable AR model. Good spectral resolution.	More complex algorithm

5.3.6 Practical cases of filter

From previous sections, we can find that the PSD analysis is important to our study. The parametric spectral analysis relies on the knowledge of random processes, probability theory, and corresponding mathematical derivation. In this section, we discuss some practical and widely used cases based on the theory.

5.3.6.1 Lattice filter [2, chapter 6]

We can implement lattice filters as all-pole or all-zero structures derived from the Levinson–Durbin algorithm. In detail, the lattice filter is constructed by cascading stages of forward and backward prediction errors. They provide an efficient realization of linear prediction filters, similar to FIR filters [19, Chapter 11]. Furthermore, the lattice filters are well-suited for VLSI implementation due to their robustness and structural regularity.

A basic element of a lattice filter can be duplicated in a manufacturing procedure, which constructs its whole structure. Every basic block contains only two multipliers and two adders. This simplicity reduces the design's complexity and shortens the design period. The regular layout of the basic block implies highly efficient use of area and higher performance. In a lattice filter, the reflection coefficient magnitude is always less than one. This property ensures its stability.

In digital integrated chips, fixed-point arithmetic is used for its hardware efficiency. Operations like truncation manage the bit of growth but introduce errors, while scaling is essential to avoid overflow. A lattice filter whose stability is suitable for VLSI compared with other direct form IIR filters.

The time-series distance data from the TOF sensor can be modeled using a lattice filter. We can estimate the system frequency by analyzing the reflection coefficients of this model, obtained through a system identification process. Where the system's frequency is the respiration rate in our case.

On the other hand, if we assume the model is direct from the FIR filter as

$y[n] = b_0 * x[n] + b_1 * x[n-1] + b_2 * x[n-2] + ... + b_N * x[n-N]$

These inputs are variant delay versions. In most cases, the observations are correlated, so that tuning one coefficient will affect others. In other words, the terms are not orthogonal. While the lattice filter transforms the non-orthogonal function to a new orthogonal basis function[23, chapter 4], which is the reason for stability. This transformation is called orthogonalization.

In the AR model, we initially assume it is a standard Markov process, i.e., the current state depends only on the previous state. However, the presence of noise severely corrupted this assumption in fact. The degree of corrupted situation is proportional to the distribution of noise. The stronger the noise in the model, the lower the orthogonality between terms in the AR model.

A lattice filter can be recognized as a recursive estimate method that efficiently updates the state vector; its recursive relationship is:

$$\begin{aligned} f_m(n) &= f_{m-1}(n) + k_m b_{m-1}(n-1) \\ b_m(n) &= b_{m-1}(n-1) + k_m f_{m-1}(n) \end{aligned}$$

Where f_m represents forward prediction, and b_m represents backward prediction. Essentially, it describes a dynamic system state that evolves from the previous state. This one-step prediction is a Markov characteristic property.

In this form, the Lattice filter is similar to the Kalman filter. They both only focus on the previous state. In contrast, a non-perfect AR model must consider more previous states.

5.3.6.2 Akaike Information Criterion

The AIC equation for finding a suitable number of orders as [19, chapter 12] in:

$$AIC(p) = \ln \hat{\sigma}_{wp}^2 + \frac{2p}{N}$$

Where AIC(p) represents the score of the model, the lower, the better. In the first term $\ln \hat{\sigma}_{wp}^2$, where $\hat{\sigma}_{wp}^2$ is the residue of the model, representing the noise of the model. When p increases, the number of parameters increases, and the fitting ability increases. Hence, $\hat{\sigma}_{wp}^2$ decreased, such that $\ln \hat{\sigma}_{wp}^2$ decreases along with p increases.

The second term is a penalty term. When the model becomes complex, this term imposes a penalty on the entire equation. Its purpose is to prevent overfitting. When p increases, $\frac{2p}{N}$ linearly increases. A larger number of order models might fit the training data perfectly but also learn the details of the noise, leading to poor performance on new data. The AIC aims at finding an optimal model order by choosing the proper balance between p and $\hat{\sigma}_{wp}^2$.

Initially, setting a small p and then increasing p improved the fit. Along with $\hat{\sigma}_{wp}^2$ decreased, the score of AIC(p) decreased. When the model has successfully learned the pattern from data, increasing p no longer has a greater improvement in fit than before, but only causes $\frac{2p}{N}$ to increase dramatically. In this way, it finds a suitable number of orders.

5.3.6.3 Adaptive Model Order Selection Based on SNR

From the previous section, we found that the number of orders relies on the noise term $\hat{\sigma}^2_{wp}$. However, the term $\hat{\sigma}^2_{wp}$ has a strong link with the SNR. In [19, chapter 12], the experiment analyzed the SNR, defined SNR as: $10log_{10}\frac{A^2}{\sigma^2}$, where A is the amplitude of the sinusoidal signal generated by an artificial simulation. This form of SNR representation is the signal itself, which cannot provide the fitting ability of our algorithm. Instead, we perform the SNR criteria analysis in another way, which we discuss in the section 6.3 of this paper. But we were inspired by the rule of choosing an appropriate number of orders. When the distance between the chest wall sensors is too great, it leads to low SNR.

In other words, the system dynamically increases the number of orders when the system detects that the σ^2 becomes higher. Physiologically, the respiratory amplitude is lower in older people than in younger people, and the lower amplitude also leads to lower SNR.

5.3.6.4 Eigen value analysis in spectrum estimation

In the previous sections 5.3.5, the computation is relatively low because we did not involve the computation of the eigenvalue. Whereas we often involve eigen abstraction from frequency in practice. We will use linear algebra in time series analysis, such as singular value decomposition. Although the method based on eigenvalues is data-driven, it is still within the DSP domain. Through obtaining the eigen data of an autocorrelation matrix, we can find the principal frequency in the spectrum [19, Chapter 12]. We assume that the signal is a linear combination of sinusoids and corrupted by white noise. Under this assumption, we can use this method. Whereas the AR model is a linear combination of weighted observations corrupted by white noise.

For a matrix A, if there exists a vector v and a scalar λ, satisfying $Av = \lambda$, then λ is the eigenvalue, and v is the corresponding eigenvector. The main idea is to divide the signal space being analyzed into two: the signal space is spanned by sinusoids, and the noise space is orthogonal to the signal subspace.

The method based on eigenvalues has a higher frequency resolution, significantly exceeding the assumption of AR model PSD estimation. The method also guarantees a sharp peak at the primary frequency. However, this method assumes the signal is a combination of sinusoids and cannot capture the details of the power at specific frequencies. This method can provide a pseudo-power spectrum density if we specify the number of sinusoids in the signal.

By analyzing the eigen data in the correlation matrix, we can identify multiple eigenvalues to focus on. And then we can sort out the principal components of the eigenvalue in descending order. Furthermore, we can detect all individual respiratory rates simultaneously if the field of view includes multiple people [12]. Otherwise, if there is only one person in the field of view, and the testing distance is close enough, this algorithm could likely detect the heart rate. Both methods are suitable for short observation intervals, particularly when compared to the long-time records of the FFT method.

Multiple signal classification [2, Chapter 8.6, pages 451-465]

Multiple signal classification (MUSIC) is a concrete case of eigenvalue analysis in spectrum estimation. It does not directly estimate the power but calculates the pseudo-spectrum.

$$P_{\mathrm{MUSIC}}(f) = \frac{1}{\sum_{k=M+1}^{N} |\mathbf{e}^H(f)\mathbf{v}_k|^2}$$

Where $\mathbf{v}_k$ is the noise subspace eigen vector, $\mathbf{e}(f)$ is the frequency steering vector.

The Eigen data-based method initially provides frequency components without power density. In this way, it has the advantage of fewer spurious peak value occasions. In contrast, the pseudo power density spectrum produces more spurious peak frequencies.

ESPRIT Algorithm

Estimation of signal parameters via the rotational invariance technique (ESPRIT) is also a high-resolution frequency technique[24]. ESPRIT and MUSIC are both subspace-based methods for estimating frequency. The number of signal sources also needs to be indicated, but they are fundamentally different in some respects. ESPRIT does not provide pseudo-spectrum. For some applications, ESPRIT performs faster than MUSIC and is less computationally intensive than MUSIC. The brief description of the ESPRIT procedure is [19, Chapter 12]:

- 1. Construct two vectors, with the vector having one lag after the previous one:

$$Y(n) = [y(n), y(n+1), \ldots, y(n+M-1)]^T$$

$$Z(n) = [y(n+1), y(n+2), \ldots, y(n'M)]^T$$

These two vectors relate to the rotation invariance with the matrix Φ.

- 2. Compute the autocorrelation matrix R_{yy}, and the cross-correlation matrix R_{yz}.
- 3. Compute their eigenvalues.
- 4. Eigenvalues lie on the unit circle corresponding to the sinusoids' frequency.
- 5. Corresponding eigen vectors estimate the power.

Compared to the ESPRIT method, the AR mode relies on linear prediction. However, there is a close relationship with the AR model and the ESPRIT method. The ESPRIT does not consider any additional noise terms, as ARMA does [19, Chapter 12], which only considers one noise term, just like AR. However, in practice, the SVD-ESPRIT treated the moving-average term in the same way as the ARMA [25].

5.3.6.5 The SVD-ESPRIT algorithm

The SVD-ESPRIT algorithm is also a subspace system identification method, but it is more generally used in engineering. In [25, chapter 5], the Kalman filter was described as the preliminary review of the subspace method for system identification. From this review, we can conclude that the Kalman filter is the foundation of the subspace method in system identification. In detail, this method can be simply described as the Kalman filter estimating the parameters of the autoregressive model. Additionally, this method used the SVD to estimate the parameters, which can be found in [25, chapter 6].

$$x(t+1) = Ax(t) + Bu(t) + w(t)$$

$$y(t) = Cx(t) + Du(t) + v(t)$$

In the Kalman filter, there are two important equations shown above. Where $w(t)$ is sensor noise, and $v(t)$ is process noise, x is the state, y is the output. In particular, where in the equation u is the input, we can recognize it as our time-series distance data in our case. We can recognize the process noise as infrared lights come from the environment. In the end, the A, B, C and D matrices are parameters that describe the AR model, which we aim to estimate.

We will use the SVD-ESPRIT method for the experiments in the following chapter, which yield relatively satisfying results among others. The algorithm includes Multivariable Output Error State Space (MOESP) to identify the parameters.

CHAPTER 6

Experimental Results

In this chapter, we present the results of the experiment. We first use the criteria based on the signal-to-noise ratio to assess the reliability. Then, we quantified the system's reliability using a metronome-based method to estimate the error in breaths per minute. By visualizing a diagram of the power spectrum density for an intuitive reliability estimation. Furthermore, a time-variant breathing test was performed. During robust testing, the measurement system required additional time to readjust when body movement occurred. Various parameter tests were also conducted, including those involving the number of orders and distance.

6.1 Fitting performance

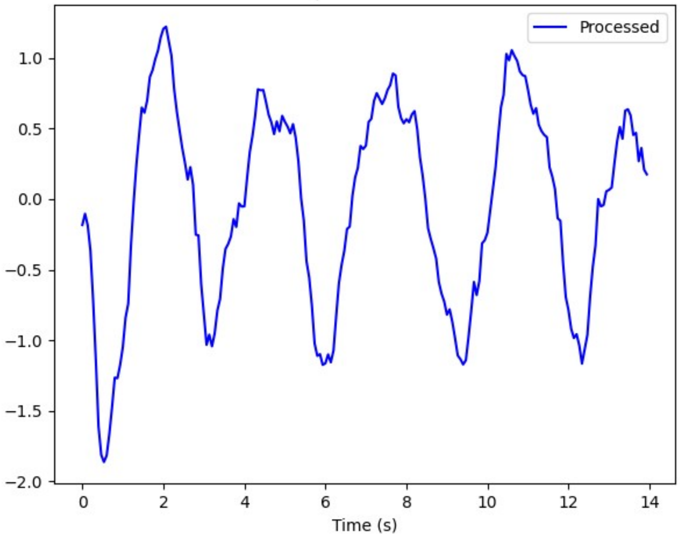


Figure 6.1: Normal Breath mode

Figure 6.1 shows that the Y axis is the normalized distance between the chest wall and the sensor. After subtracting trend data, as the previous chapter explained, the curve oscillates around the zero point. The picture also implies that the signal's amplitude is small, at only 2mm.

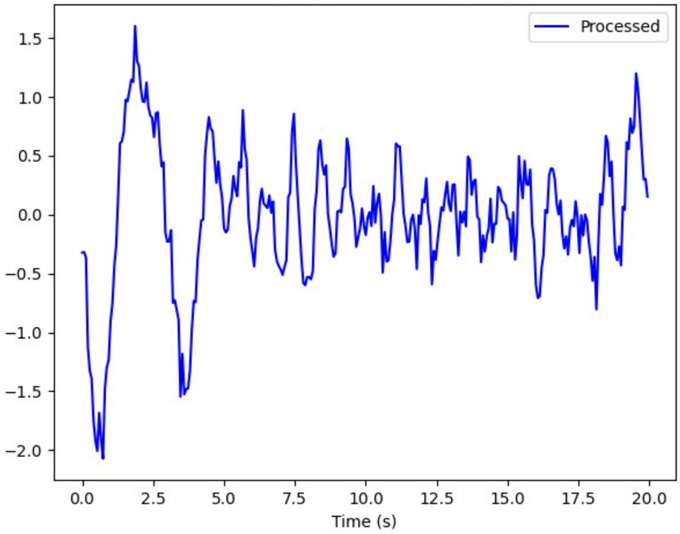


Figure 6.2: Normal switch to high-frequency breathing mode.

Figure 6.2 recorded the switching from normal breathing rate to high frequency mode. The high-frequency mode is obviously more difficult to recognize, but the curve still crosses the zero point every period, because the body does not move horizontally.

Comparing the low frequency at the earlier part of the figure and the high frequency of the latter part, we conclude that the high frequency of RR presents relatively lower SNR.

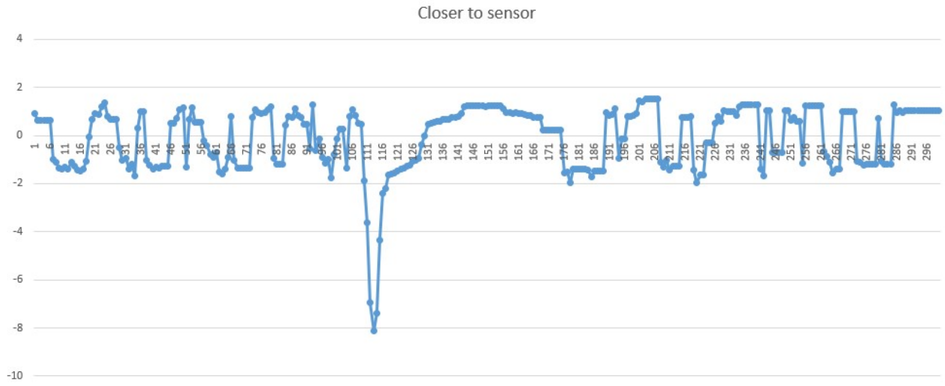


Figure 6.3: Body getting closer to the sensor

Figure 6.3 recorded a normal breathing rate, but with the body slightly closer to the sensor. The sensor could not determine whether the distance change was caused by breathing itself or the body leaning towards the sensor. Sensors distinguish them by testing whether the zero point is absent or present after one period of time. As can be seen from Figure 6.3, the system cannot tell the actual breathing rate for sample points 116 to 186 in this interval. Precisely in this interval, the body closes the sensor. From this interval, we can still see a step-shaped curve, this time with a length of approximately one or two breathing periods. However, the system can detect the normal breath again after point 186 because a new zero point is detected.

Furthermore, the picture also shows the ratio between the average distance and the signal amplitude. The average distance has changed by 10mm, and the breathing amplitude is 2mm. This fact implies the low SNR.

For a precise time interval during which the system failed to produce RR, refer to section 6.9.

6.2 A rough comparison of AR model and FFT

Based on the AR model, the number of parameters can't be too large or too small. We should choose a reasonable data buffer length according to the number of parameters. At the beginning of our study, the length of the data buffer is equal to the number of parameters.

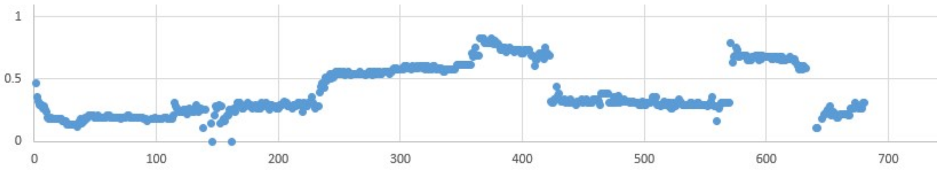


Figure 6.4: AR model 8th-order analysis of the respiratory signal.

As the Figure 6.4 shows, the respiration frequency oscillated from 0.2 Hz to 0.8 Hz, and the frequency resolution is higher than the model of the FFT output. The frequency of 0.8 Hz is extremely high compared to a practical case, which corresponds to approximately 48 breaths per minute. The following test of the FFT method also has a high-frequency input up to 0.8 Hz.

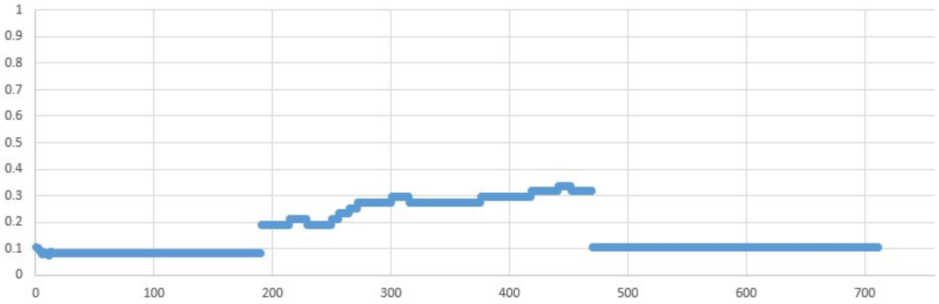


Figure 6.5: FFT method analysis of respiratory signal

In contrast, the FFT-based spectrum analysis reveals no significant change at a higher respiratory rate. The FFT-based method is Welch's method in detail, which uses overlapping frequency windows for adjacent components. The test results show a frequency of only 0.4 Hz, far from the actual frequency of 0.8 Hz, as shown in Figure 6.5. Figure 6.6 illustrates our testing process using a TOF sensor.

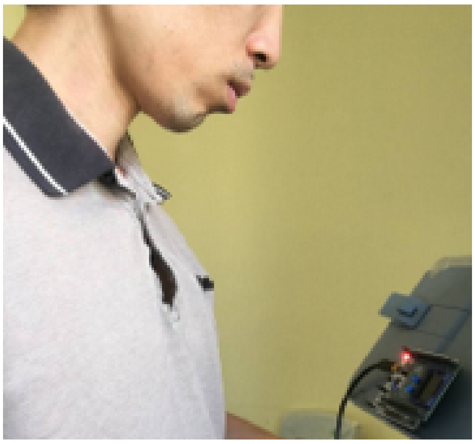


Figure 6.6: Testing with VL53L sensor

6.3 Signal-to-noise ratio calculation based on peak frequency

We extract the SNR of the respiratory signal [26].

$$\mathrm{SNR_{dB}} = 10\log_{10}\left(\frac{P_s}{P_n}\right)$$

Where the P_s is the power of the signal and the P_n is the power of the noise. The power of the signal P_s can be expressed as:

$$P_s = \int_{f_0-\frac{B}{2}}^{f_0+\frac{B}{2}} S(f)df$$

Where B is the bandwidth, we set the bandwidth to 0.3 Hz. The bandwidth represents the lower and upper bounds of the breathing signal's frequency, which can be treated as tolerance errors in the frequency. And the $S(f)$ is the power spectral density of the respiratory signal. Whereas the power of noise P_n equals the total power minus the power of the signal.

6.4 Signal to noise ratio comparison

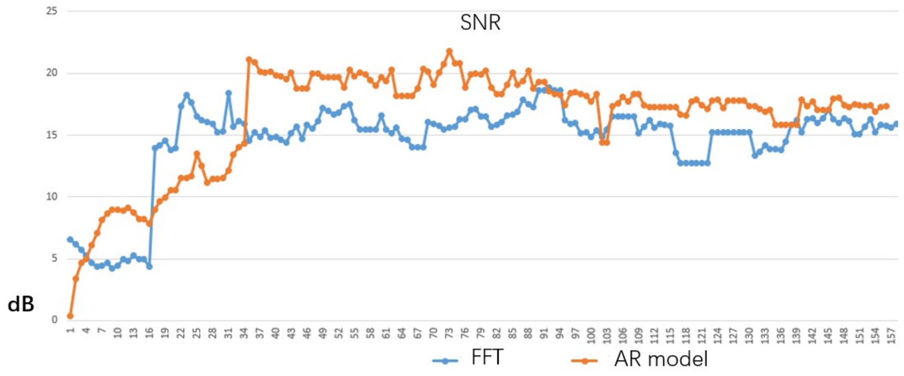


Figure 6.7: 20cm SNR of the AR method compared with the FFT method

This experiment was performed with the two different algorithms, the FFT and AR model, at the same distance of 20 cm. We can reveal these two characteristics by comparing the two SNRs. The AR model, after a period of adaptation, has a clear advantage. Initially, the AR model's SNR is lower than the FFT's because the AR model requires a specific samples to initialize. The Burg algorithm requires particular data to converge and remain stable. The data of initialization enables the algorithm to estimate the parameters accurately. However, after that, the SNR curve of the AR model surpasses the FFT, and almost always remains in a high position at a single point.

The sharp, clear peak of PSD estimation reveals more energy centers on the actual respiratory signal. The FFT method contains more noise power in the bandwidth due to the spectrum leakage and poorer frequency resolution when calculating the power of the respiratory signal.

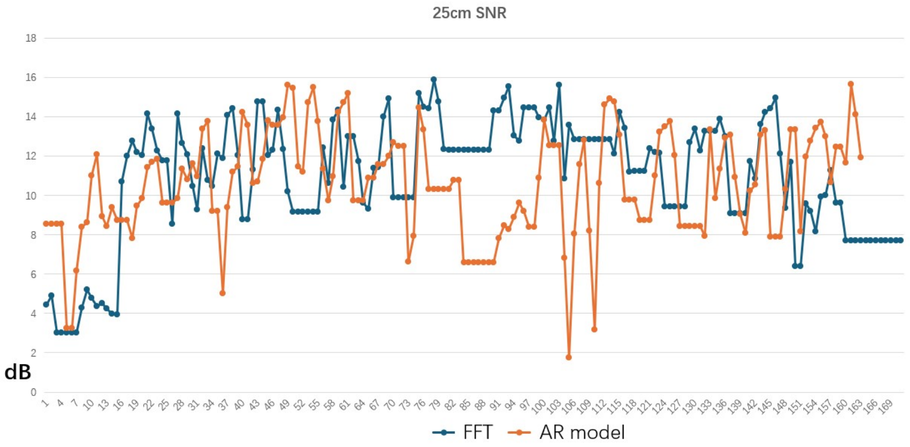


Figure 6.8: 25 cm SNR of AR method VS FFT

Compared to the previous experiment, Figure 6.8 shows the sensor positioned slightly farther from the chest wall, 5cm beyond the previous setup. The average SNR is now lower than before, at 12 dB. The AR model method PSD computation no longer has an obvious advantage over the FFT method. The reason is that as the testing distance increases, the signal strength decreases, and the noise remains strong, resulting in a reduced SNR. From Figure 6.9 and Figure 6.10, we can see that the AR model PSD has no side lobes and is smooth, while the FFT method has side lobes and the power density is obviously oscillating.

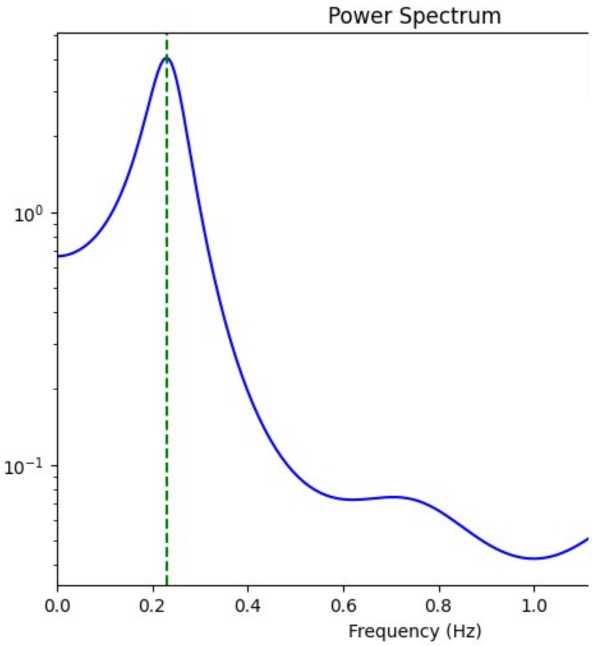


Figure 6.9: PSD of AR method

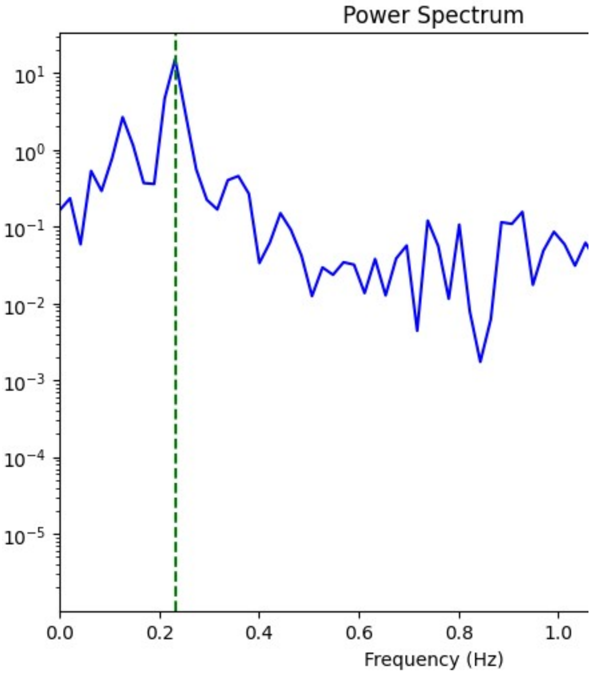


Figure 6.10: PSD of the FFT method

6.5 Refinement of spatial anti-artifact

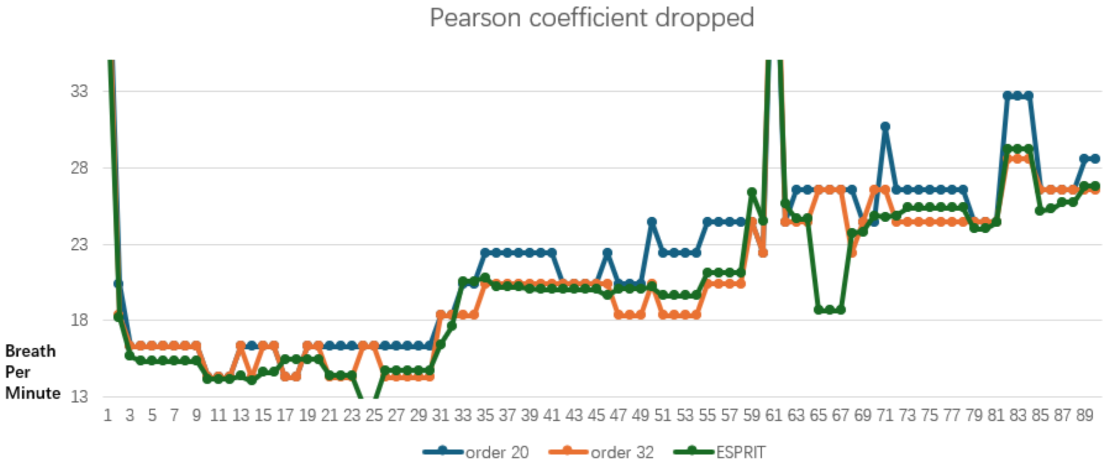


Figure 6.11: Average distance of 17cm

We note that in Section 5.3.3.1, we drop pixels with a low value of Pearson coefficients. Dropping those pixels with low Pearson coefficients leads to a smooth curve in the estimation of RR. However, the error becomes large when the RR changes over time, as the Figure 6.11 shows. Based on the framework, we have made a slight adjustment to the use of correlation coefficients, resulting in better results. The modification is that we don't drop any pixels that have low correlation coefficients. But using the correlation coefficients, multiply the pixel distances. This modification also indicates that we don't perform the average of these distances.

The tests we have already provided are all using the method that we have slightly modified, which we described above.

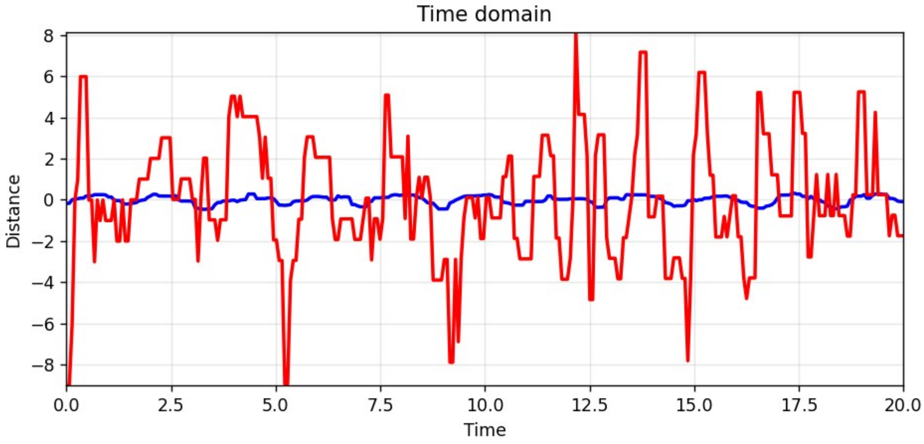


Figure 6.12: Comparison between Pearson coefficients filter and original data, blue one is output of filter and red curve is raw data.

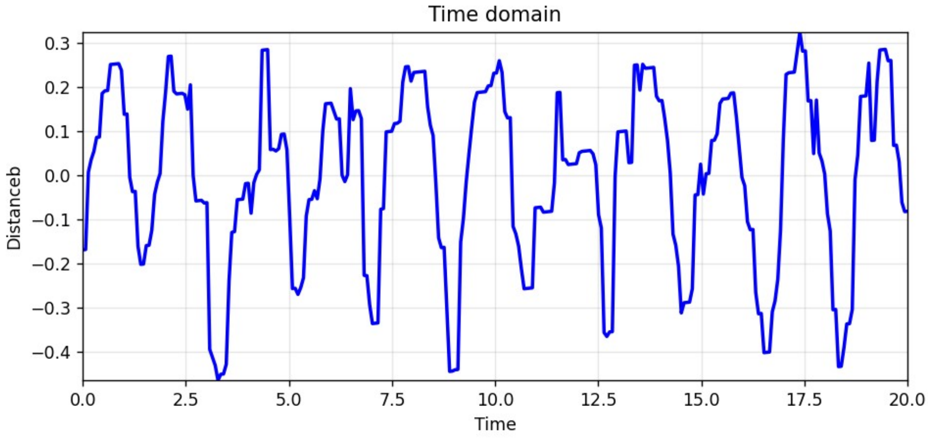
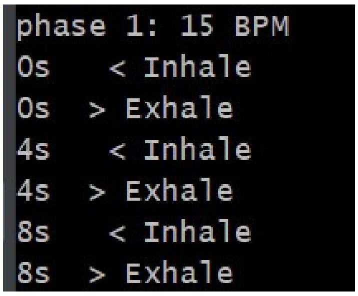


Figure 6.13: Zoom in the blue curve from Figure 6.12

By setting the average distance of 30cm, we have the result of Figure 6.12 and 6.13.

6.6 Testing frequency convergency rate with metronome

Some studies use a spirometer as a reference standard by computing the Pearson correlation between the outputs of the device and spirometer readings to validate accuracy. To precisely test the errors in our system, we need a method to estimate the deviation and the convergence of the respiratory rate provided by the measurement system. Furthermore, we require a standard answerer to examine our design, which we have chosen to be a metronome method [5].



```
phase 1: 15 BPM
0s    < Inhale
0s   > Exhale
4s    < Inhale
4s   > Exhale
8s    < Inhale
8s   > Exhale
```

Figure 6.14: Terminal printing indicates inhalation and exhalation.

In detail, a synchronization pace is provided via terminal output to indicate inhalation and exhalation during RR testing in Python, with variants at 15, 20, and 25 BMP. The terminal output is shown in Figure 6.14. Each phase of RR endures for 2 minutes. This indication was also recorded as post-analysis. In this way, we can determine the system's detection speed of the new RR and its convergence to the point, while also easily observing the error fluctuation. We test the AR model with the Burg method for its stability instead of the traditional Yue-Walker method.

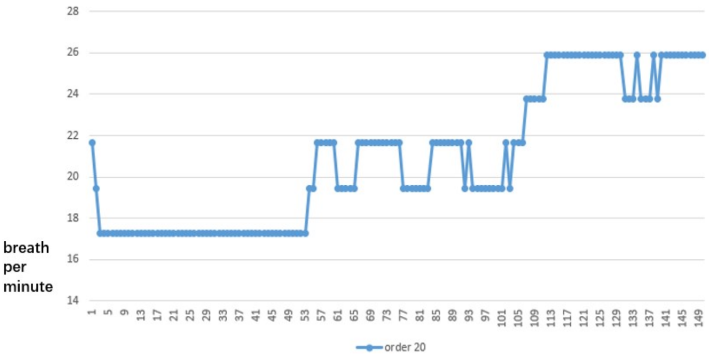


Figure 6.15: Burg method-based AR model with the order 20

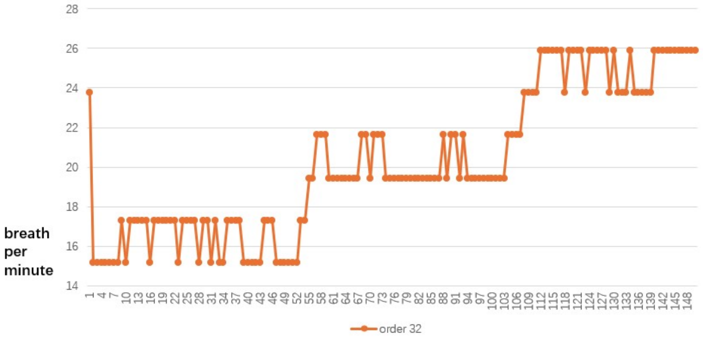


Figure 6.16: Burg method-based AR model with the order 32

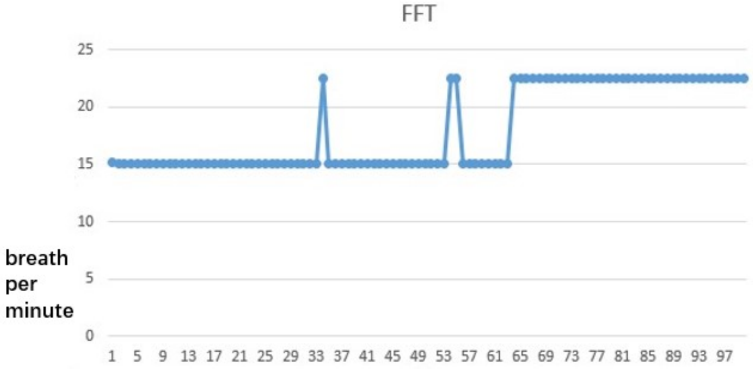


Figure 6.17: FFT method can't distinguish between 15 BMP and 20 BMP

Figure 6.18: ESPRIT method

We then compare different frequency estimation methods (FFT, AR model based on Burg, ESPRIT), focusing on their convergence rates and deviations. Setting the sensor in front of the chest wall at a distance of 17cm, the following experiment results were obtained, shown as Figures 6.15, 6.16. The ESPRIT method exhibited relatively higher frequency resolution than others, as shown in Figure 6.18. Whereas the FFT method has not obvious difference between the phase 15BPM and 20BPM, as Figure 6.17 shows.

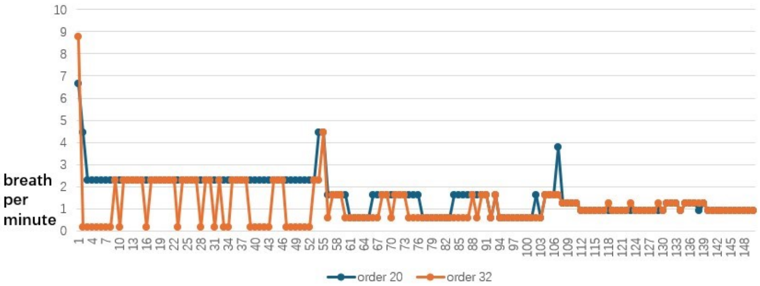


Figure 6.19: Absolute deviation based on the Burg method AR model of 32 vs 20 order

Comparing these two different orders based on the Burg method AR model is shown in the Figure 6.19. Order 20 exhibits a relatively lower deviation than order 32 at the beginning of the 15 BMP phase, likely due to the need for more observation to adjust order 32. However, in the 25 BMP phase, 32 orders surpassed 20 orders with errors less than 2 BMP.

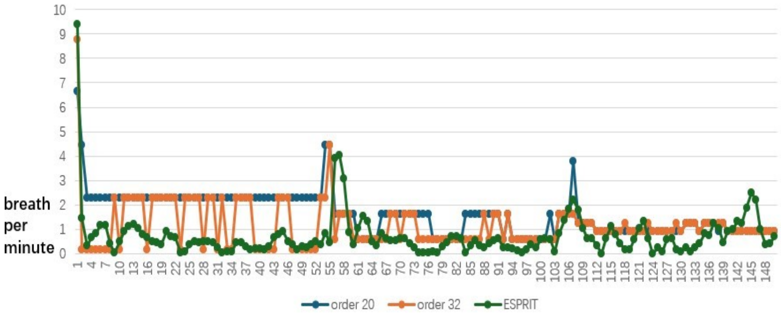


Figure 6.20: Deviation comparison among AR models ESPRIT

According to Figure 6.20, we can see that for the range of all these three different RR phases. The ESPRIT method has the fewest deviation, up to 0.89 BMP. The Burg method with 20 orders has a deviation of 2.159 BMP, whereas 32 orders have a deviation of 2.09 BMP. We can conclude that the resolution of an AR model strongly relies on the number of orders.

6.7 Testing in evening

We know there is infrared light present as a component of daily light. Because there is less interference from the environment in the evening, the test shows less deviation in the data.

Average distance 30cm

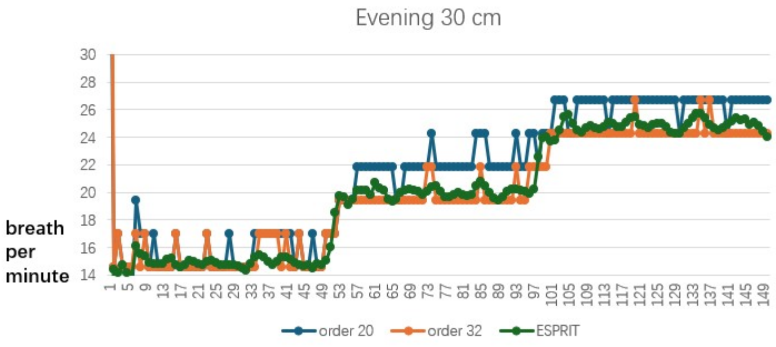


Figure 6.21: Comparison in the evening

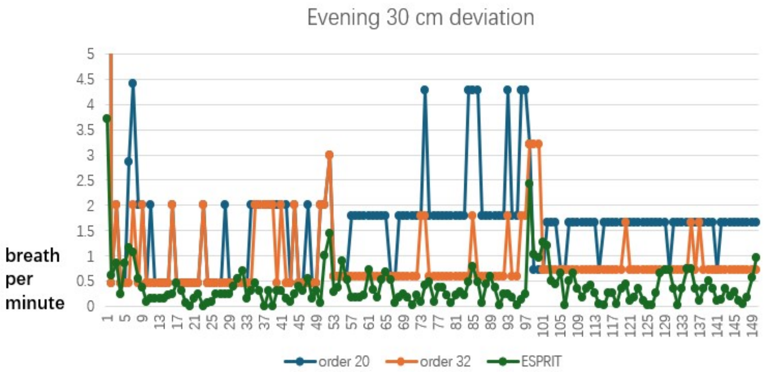


Figure 6.22: Deviation comparison in the evening

From Figure 6.23 and Figure 6.1, the average distance is 30 cm, and the test is performed in the evening. We observed that the deviations in the evening are relatively lower than those in the day.

Average distance 17cm

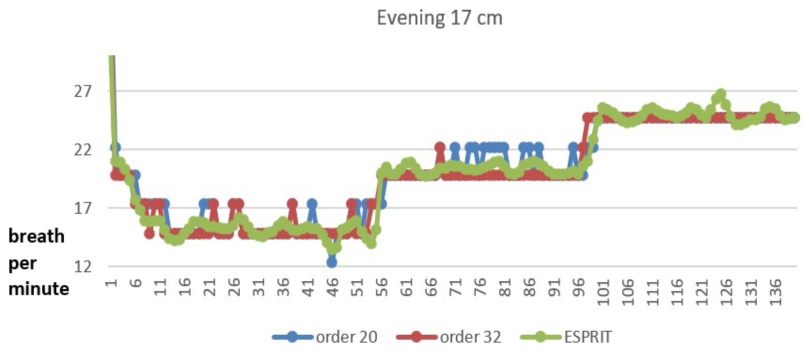


Figure 6.23: Comparison in the evening with an average distance of 17cm

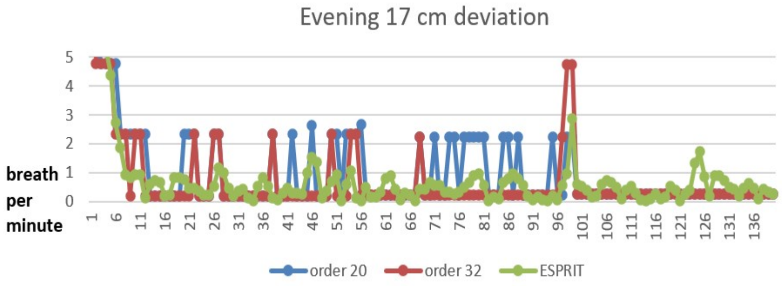


Figure 6.24: Deviation comparison in the evening with average distance 17cm

Table 6.1: Deviation of the test in the the day and evening

	order 20	**order 32**	**ESPRIT**
day 17cm(BPM)	2.16	2.09	0.89
evening 30cm(BPM)	1.84	1.24	0.39
evening 17cm (BPM)	1.08	0.76	0.76

In detail, the deviations of different models are summarised in Table 6.1. We conclude that ambient light indeed has an impact on the sensor. In particular, we don't filter the valid bits from the firmware or use the SPAD data.

6.8 Testing with time stamp

In the previous section, we didn't record a timestamp which can only reflect the stability of the output. In this section, we record the timestamp and the switching points between different RRs.

The average distance is 31cm, and each phase endured for 60 seconds.

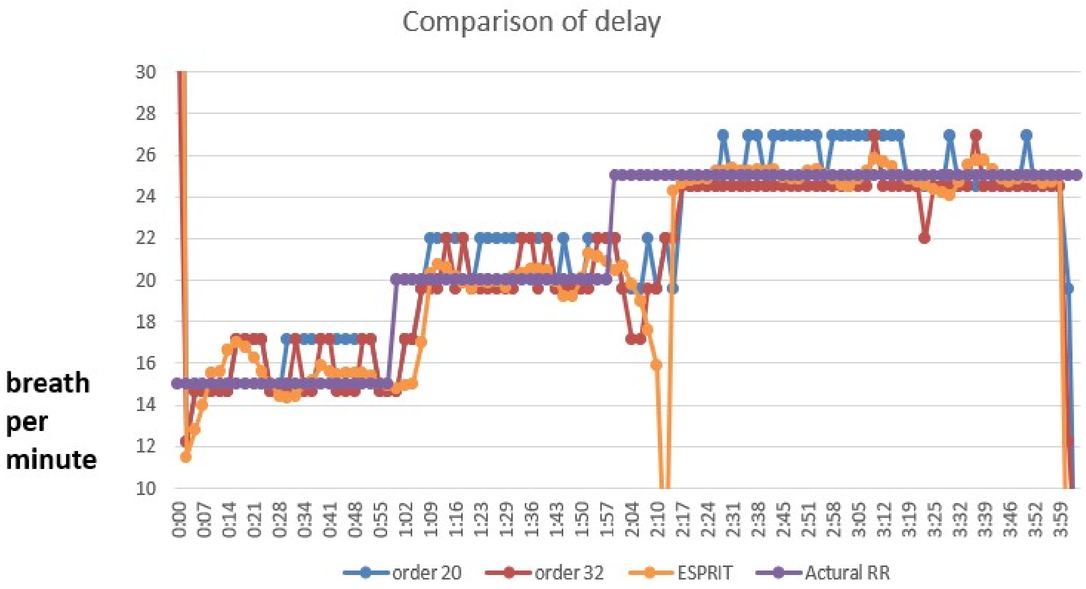
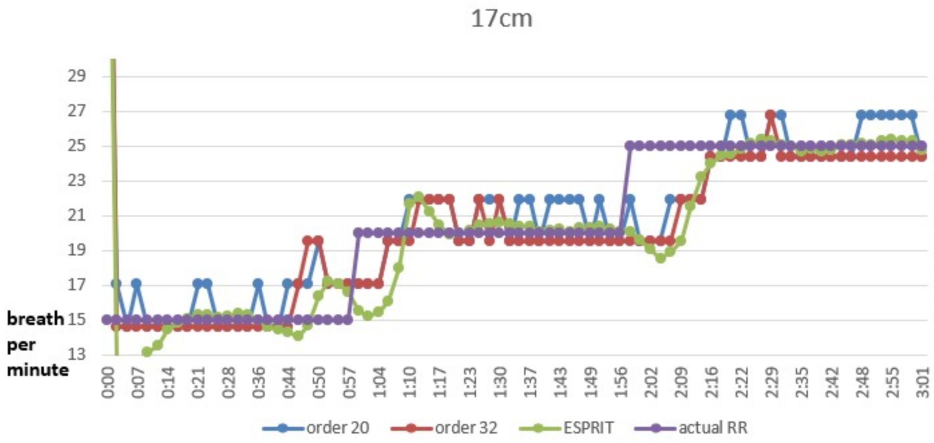


Figure 6.25: Delay compared to actual RR

As shown in Figure 6.26, we tested three phases of RR. Although the 32-order method has lower absolute errors, its frequency resolution is lower than that of the ESPRIT method. The problem with the ESPRIT lies at the switching point between the 20BMP and 25BMP phases, which corresponds to the time point at 2:10. At that point, the tester needs to adjust the RR. Occasionally, the tester may drop to a lower frequency for one or two breathing periods. This fact leads to a sharp drop in frequency at the switching point.

Table 6.2: Comparison of delay

Model **Phase**	**order 20**	**order 32**	**ESPRIT**
15 bmp	5 s	5 s	7 s
20 bmp	9 s	6 s	9 s
25 bmp	18 s	18 s	16 s
Absolute error	2.5 bmp	1.6 bmp	2.1 bmp



	20th order	32th order	ESPRIT
Absolute error(bmp)	2.013015	1.824439	1.633171

Figure 6.26: Delay compared to actual RR 17cm

Table 6.3: Comparison of delay 17cm

Model Phase	order 20	order 32	ESPRIT
15 bmp	3 s	3 s	13 s
20 bmp	7 s	11 s	11 s
25 bmp	20 s	18 s	18 s
Absolute error	2.0 bmp	1.8 bmp	1.63 bmp

From the Table 6.3 we can see that the ESPRIT method is the best. The result is better than the previous experiment, shown as Figure 6.26. The reason is that we don't have the abnormal frequency at the turning point between phases 20 and 25 BMP. This fact also implies that the tester following the metronome has a smooth switch at the point between the two phases of RR. However, the ESPRIT method takes the longest time to infer the RR at a phase of 15 BMP. The 15 BMP phase is the first phase of the program. The fact is that the ESPRIT needs more initial process time than the AR model.

6.9 Timestamp record of body movement

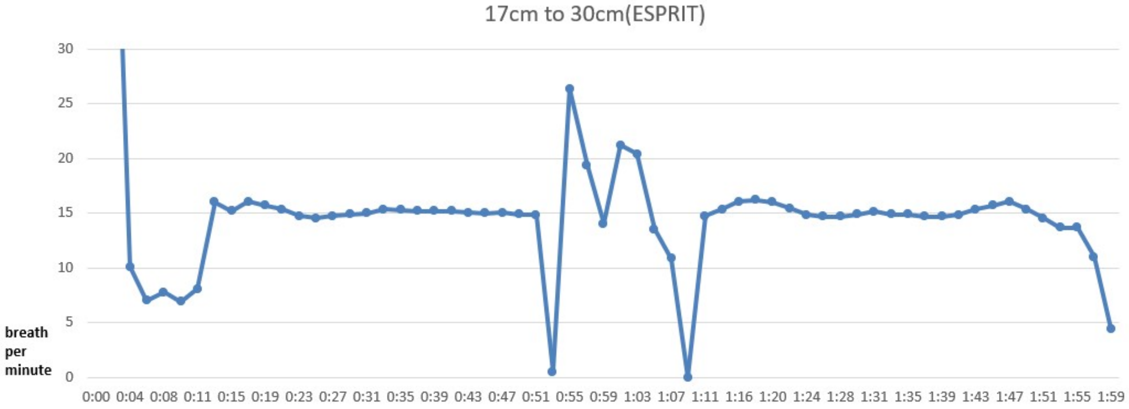


Figure 6.27: Body move from distance 17cm to 30cm

We performed robust testing with our design. First, we tested the ESPRIT algorithm. To determine the system's output speed of the RR when the body is moved away from the sensor. Initially, we placed the sensor 17cm in front of the chest. Then the tester's body moved to a distance of 30cm during the test. The metronome indicates the RR at 15 BPM to the tester throughout the test. As we can see from Figure 6.27, the body moved at 51 seconds, then the curve was stable again at 1:07. The system has a large oscillation around 15 BPM lasting for a time interval of 16 seconds.

6.10 Resolution of 4x4 setting

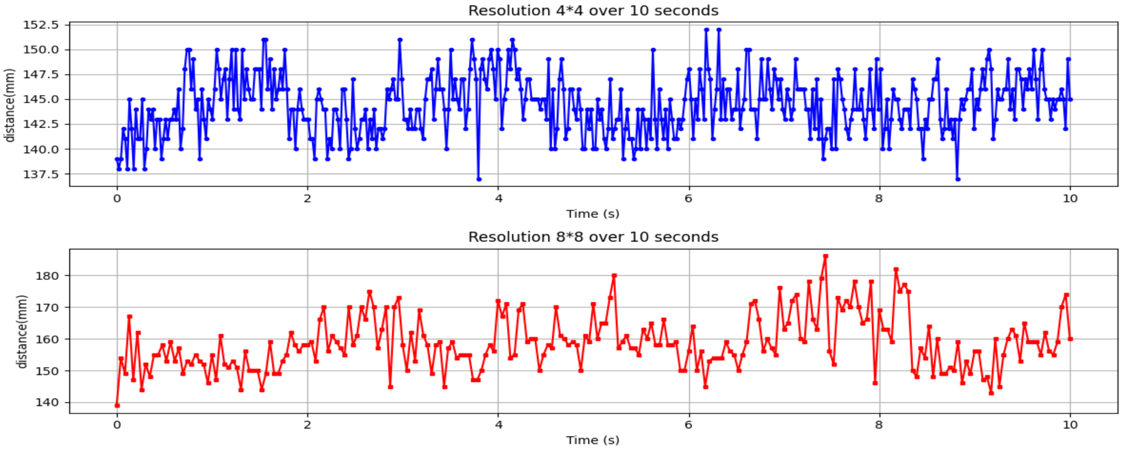


Figure 6.28: VL53L7 setting resolution 4*4 vs 8*8

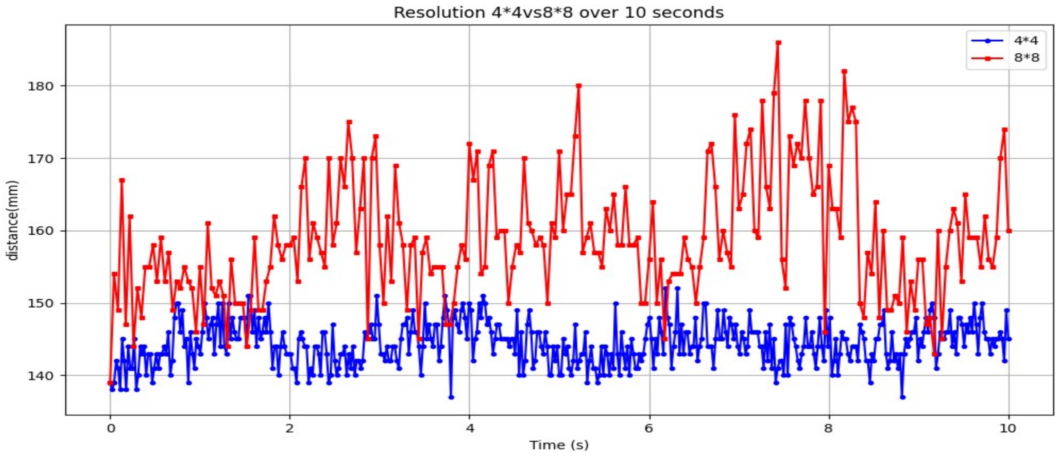


Figure 6.29: Put together VL53L7 setting resolution 4*4 vs 8*8

To achieve high-frequency sampling, we set the resolution to 4x4, as shown in Figures 6.28 and 6.29. The blue curve represents the 4x4 resolution, which shows denser points within the 10-second test. In detail, we select the pixel in the middle of the array, and the distance is about 160mm. In addition, we can have an overview of the noise compared to the signal. The noise amplitude sometimes exceeded that of the signal.

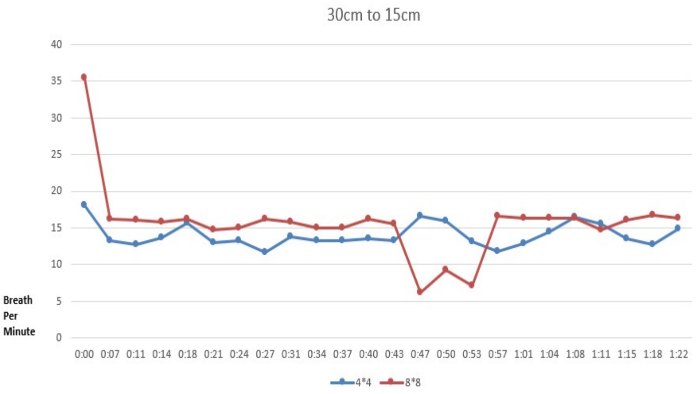


Figure 6.30: Compare resolution 4*4 vs 8*8 with body moved

By setting the resolution of 4*4, the tester moved from 30cm to 15cm during the test, and we got the result as Figure 6.30. We can see that the setting of the resolution of 4*4 presented no noticeable change during the test, remaining 15 BPM.

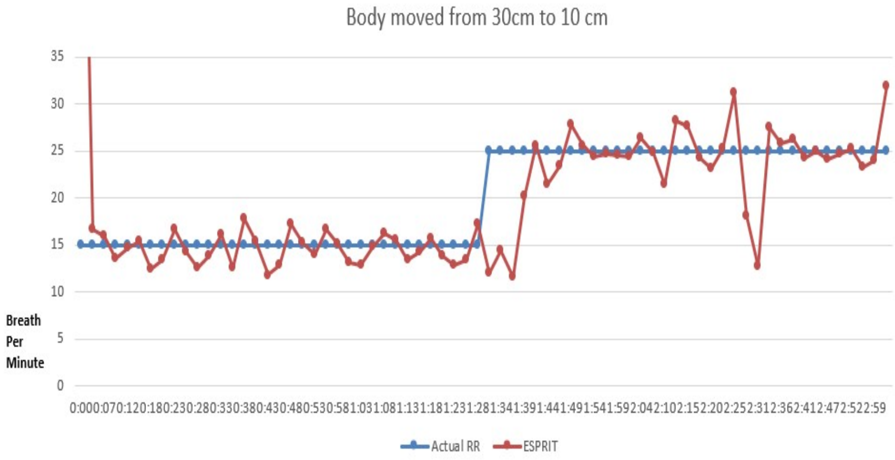


Figure 6.31: Phase of 15BMP to 25BMP, both with body moved

We set the resolution to 4*4 and tested the phase of 15BMP as well as 25BMP. We moved from 30cm to 15cm during both of these phases. The phase of 15BMP maintained a steady output. In contrast, the phase 25BMP experienced a 7BMP drop at 2:31. This abnormal time interval, lasting 10 seconds, occurred during the 25BMP phase when the body moved, as illustrated in Figure 6.31.

So we can conclude that the higher sampling rate is suitable for low RR, but still has errors with high RR. Although we have changed the resolution from 64 to 16, we don't see a drop in steadiness.

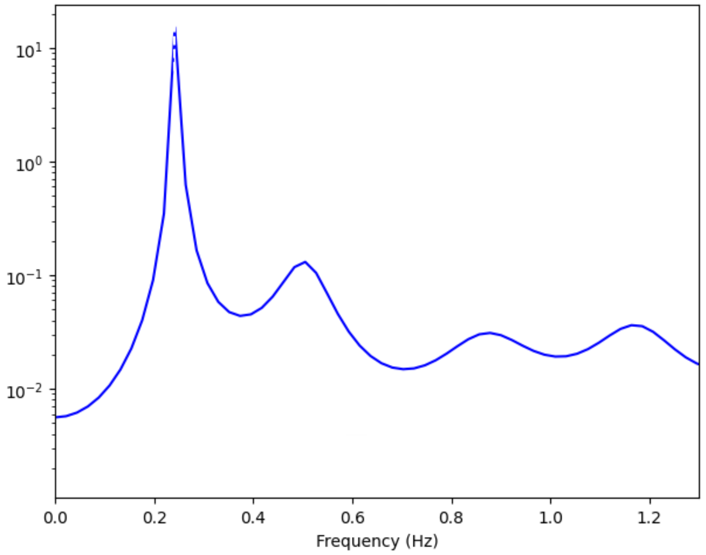


Figure 6.32: The PSD of distance setting at 20cm, Burg method with 32 order

As Figure 6.32 shows, the peak of the power is proximate at 0.25 Hz. The frequency with respect to the RR is 15 BPM. We can see the curve is steep at frequencies lower than 0.25 Hz, while the higher frequencies have some peaks.

6.11 Robust testing with long distance

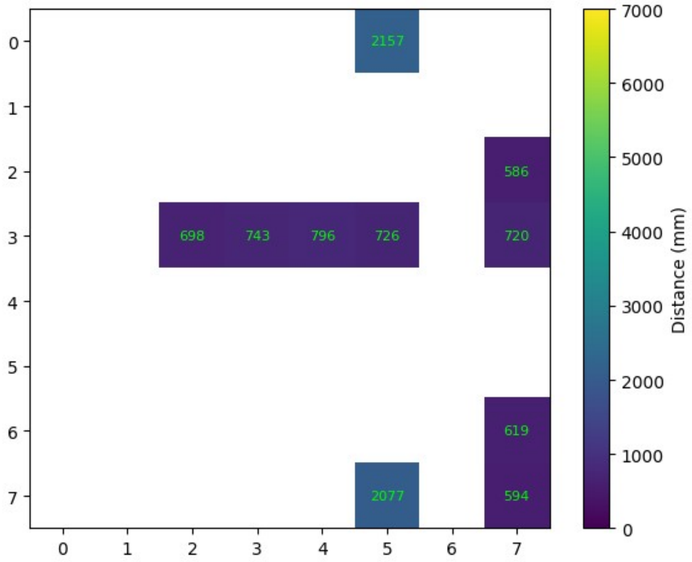


Figure 6.33: Average distance of 1m heat map

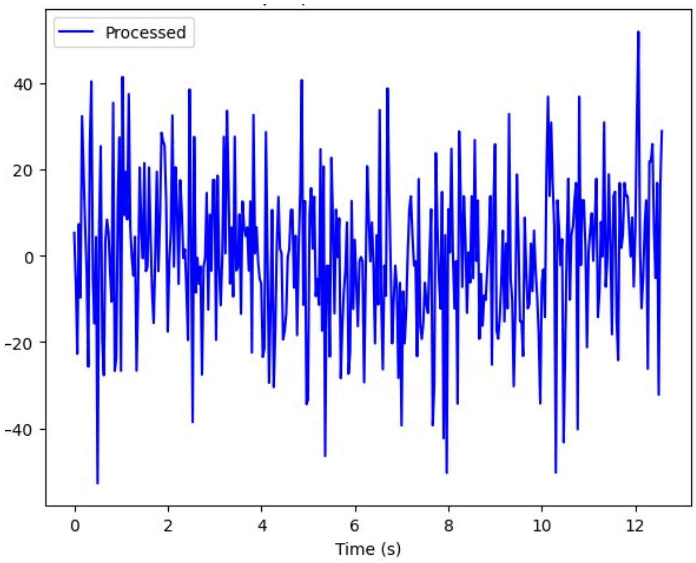
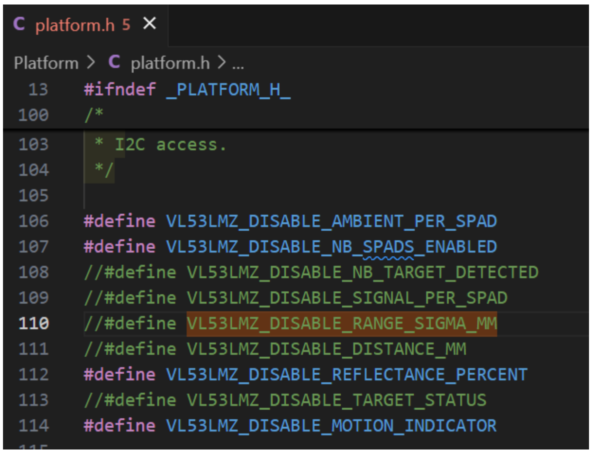


Figure 6.34: Average distance of 1m time domain signal with noise

Long distance usually introduces background and foreground in the FOV. This fact makes our design more challenging, because a longer distance leads to lower SNR and fewer valid pixels. As Figure 6.33 shows, there are only a few pixels with a valid bit equal to 1. Also, the Figure 6.34 shows an overview of the noise affecting the respiratory signal, where the chest wall to the sensor is a 1m distance. In particular, the amplitude of the noise is approximately 20mm, while the amplitude of the signal is approximately 1mm from the previous experiments. This fact implies the challenge of the long-distance RR detection.

According to the case, we need to address both the background and the foreground. We need to filter out the background because its high uncertainty. In detail, we modify the firmware of the STM32 to filter out the invalid bits of the pixels. In the previous section, we used Pearson coefficients to normalize the distance, which considers all the pixels to be valid. Whereas, in this case we should modify the approach to compute the Pearson coefficients only on the pixels that belong to the foreground.

Standard deviation and SPAD



```
#ifndef _PLATFORM_H_
/*
 * I2C access.
 */

#define VL53LMZ_DISABLE_AMBIENT_PER_SPAD
#define VL53LMZ_DISABLE_NB_SPADS_ENABLED
//#define VL53LMZ_DISABLE_NB_TARGET_DETECTED
//#define VL53LMZ_DISABLE_SIGNAL_PER_SPAD
//#define VL53LMZ_DISABLE_RANGE_SIGMA_MM
//#define VL53LMZ_DISABLE_DISTANCE_MM
#define VL53LMZ_DISABLE_REFLECTANCE_PERCENT
//#define VL53LMZ_DISABLE_TARGET_STATUS
#define VL53LMZ_DISABLE_MOTION_INDICATOR
```

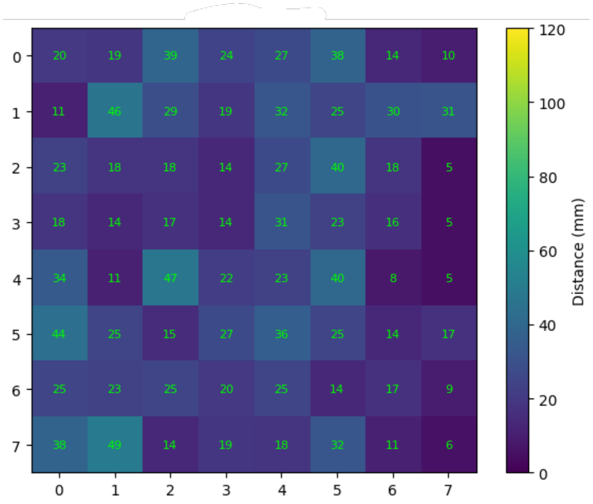
Figure 6.35: By commenting the code to activate the SPAD analysis

Listing 6.1: Call from main code

```
for(int i=0;i<64;i++){
printf("␣range_sigma_mm%d=%d␣",i,
    RangingData.range_sigma_mm[i]);
}
printf("\n");
```

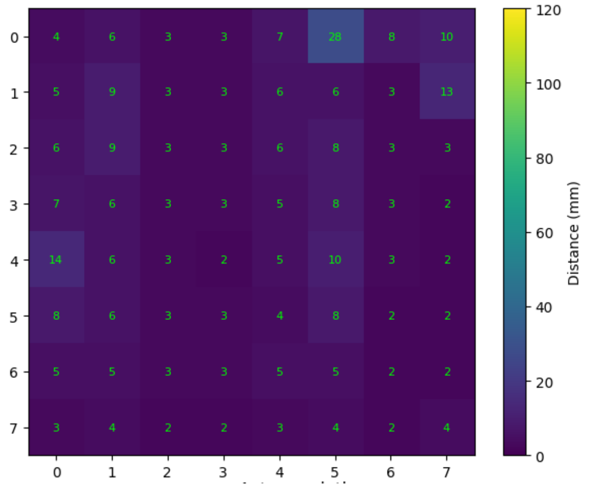
As Figure 6.35 shows, we can comment out the lines of code to activate the output from STM32, such as SPAD and sigma. In detail, the firmware version is $STSW-$

$IMG035\_F401\_v1.5.2$. Then we add some lines of code to print the standard deviation, shown as the List 6.1. From the results of the experiments, we conclude that the higher the value of SPAD, the better. The lower the standard deviation, the better, which we denote as σ, pronounced sigma.



	0	1	2	3	4	5	6	7
0	20	19	39	24	27	38	14	10
1	11	46	29	19	32	25	30	31
2	23	18	18	14	27	40	18	5
3	18	14	17	14	31	23	16	5
4	34	11	47	22	23	40	8	5
5	44	25	15	27	36	25	14	17
6	25	23	25	20	25	14	17	9
7	38	49	14	19	18	32	11	6

Figure 6.36: The average distance is 1m. We obtained a standard deviation.



	0	1	2	3	4	5	6	7
0	4	6	3	3	7	28	8	10
1	5	9	3	3	6	6	3	13
2	6	9	3	3	6	8	3	3
3	7	6	3	3	5	8	3	2
4	14	6	3	2	5	10	3	2
5	8	6	3	3	4	8	2	2
6	5	5	3	3	5	5	2	2
7	3	4	2	2	3	4	2	4

Figure 6.37: The average distance is 20cm, we obtained a standard deviation

According to the Figure 6.36 and Figure 6.38, we can find that the longer the distance, the larger the standard deviation.

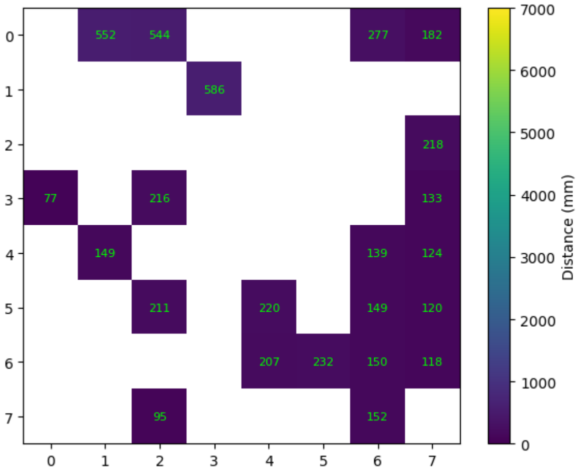
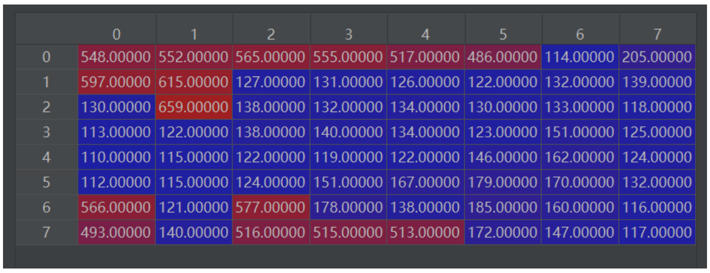


Figure 6.38: By setting the threshold of the deviation to 15mm, and the average distance is 70cm, we obtained the distance heatmap.

6.12 Background and foreground, based on the average distance

Noise usually presents as the longer distance data in the pixels, while the signal presents as the shorter distance. Such that we can calculate the average distance, then based on the average distance, split the area of noise and signal as shown in Figure 6.39. Theoretically, the area of signal can have reliable data.



	0	1	2	3	4	5	6	7
0	548.00000	552.00000	565.00000	555.00000	517.00000	486.00000	114.00000	205.00000
1	597.00000	615.00000	127.00000	131.00000	126.00000	122.00000	132.00000	139.00000
2	130.00000	659.00000	138.00000	132.00000	134.00000	130.00000	133.00000	118.00000
3	113.00000	122.00000	138.00000	140.00000	134.00000	123.00000	151.00000	125.00000
4	110.00000	115.00000	122.00000	119.00000	122.00000	146.00000	162.00000	124.00000
5	112.00000	115.00000	124.00000	151.00000	167.00000	179.00000	170.00000	132.00000
6	566.00000	121.00000	577.00000	178.00000	138.00000	185.00000	160.00000	116.00000
7	493.00000	140.00000	516.00000	515.00000	513.00000	172.00000	147.00000	117.00000

Figure 6.39: The blue area represents the signal, and the red area represents the noise.

Result of ignored near object around sensor

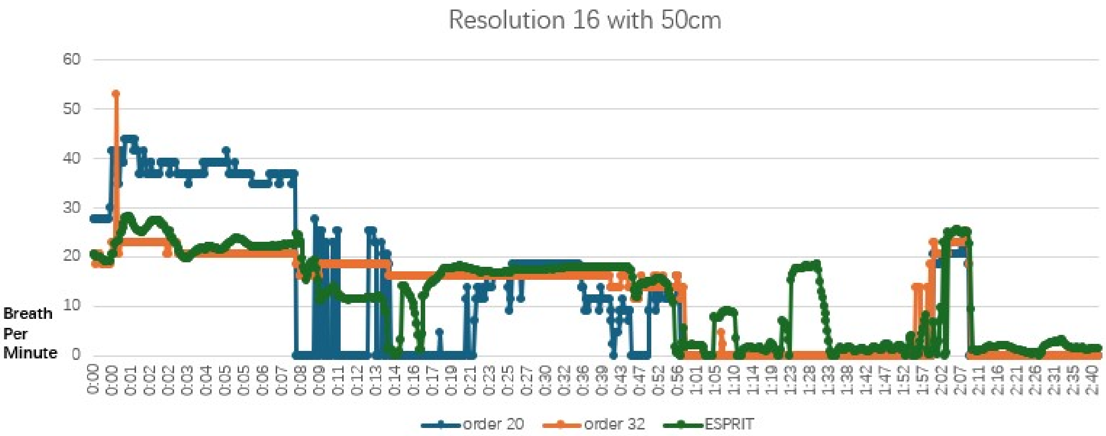


Figure 6.40: Average distance is 50cm with resolution of 16

We have been trying those approaches to increase the SNR. However, the results of the experiment are not as satisfying as those of the short distance. The phases at 20BPM and 25BPM exhibit large errors, as shown in Figure 6.40. The test only has the phase of 15BPM with relatively stable output. We can observe that the phase of 25BPM is estimated lower than the actual RR. The reason is that the objects around the sensor have a low frequency.

Result of considering the near object around the sensor

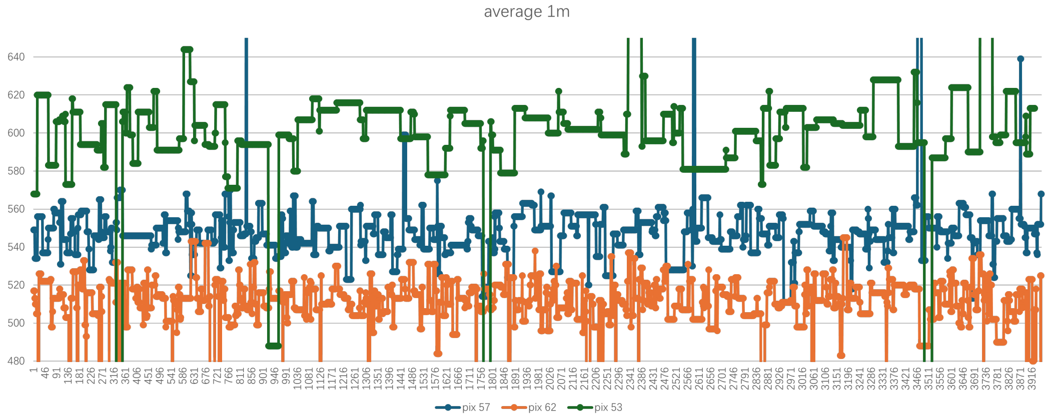


Figure 6.41: The picture illustrates three pixels of foreground.

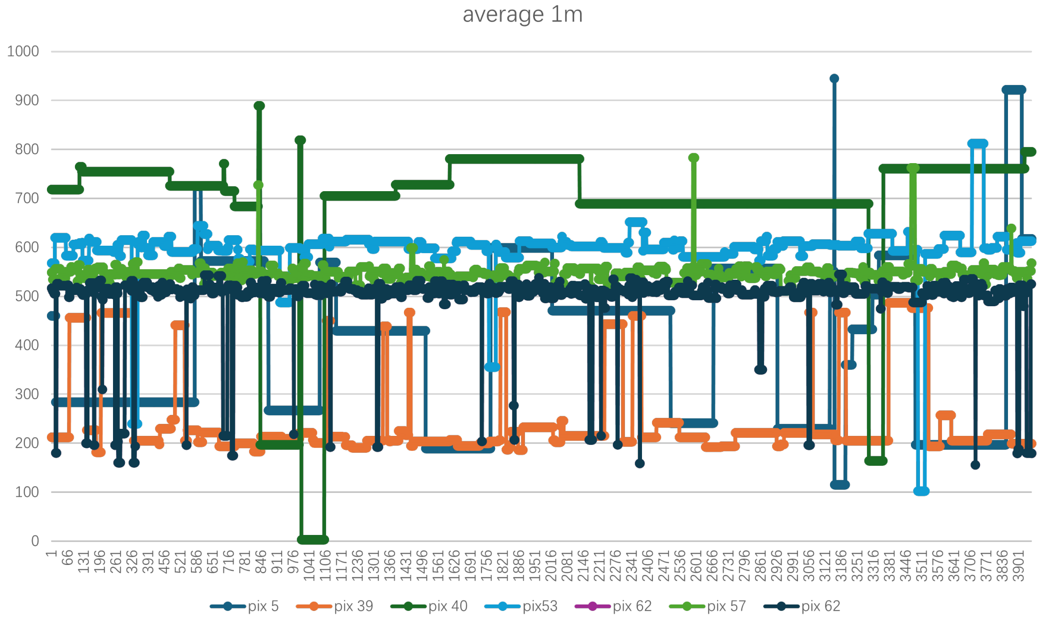


Figure 6.42: The picture illustrates both the pixels of the background and foreground.

As we increased the testing average distance, the FOV introduced background. Those pixels belonging to the background definitely have lower SNR compared to shorter average distances, which is illustrated in Figure 6.41. The figure is plotted from the raw data. Furthermore, we put the plots both about the background and foreground as Figure 6.42 shows. We can see that the background has less variation than the foreground, whose pixels 5, 39, and 40 are related to the background.

Result of long distance

We have been trying to abstract the RR from the scenario with background and foreground in the FOV. However, the problem might be more than background and foreground. Because the objects around the sensor can introduce errors, confusing the foreground. Those objects can be the desk or anything near the sensor.

Based on the assumption above, we sampled the distance between 45cm and 65cm. Such that a certain number of pixels less than 45cm will not be taken into account, nor the pixels longer than 65cm. This evidence can also be found in Figure 6.42.

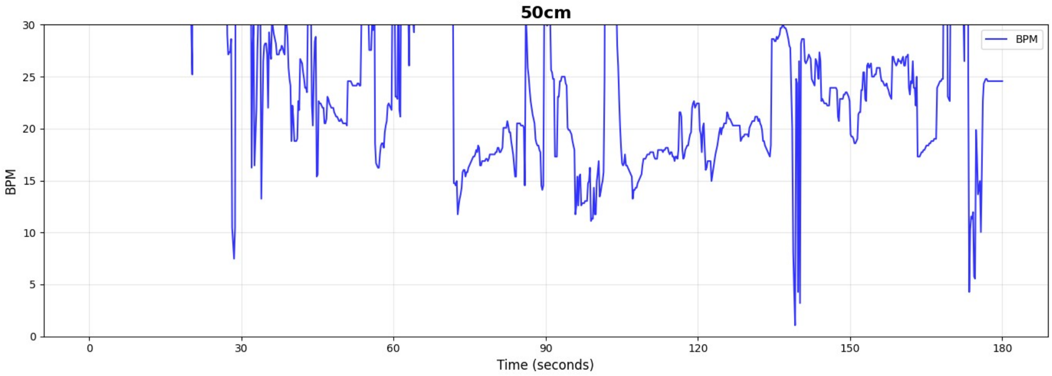


Figure 6.43: Experiment based on the above setting

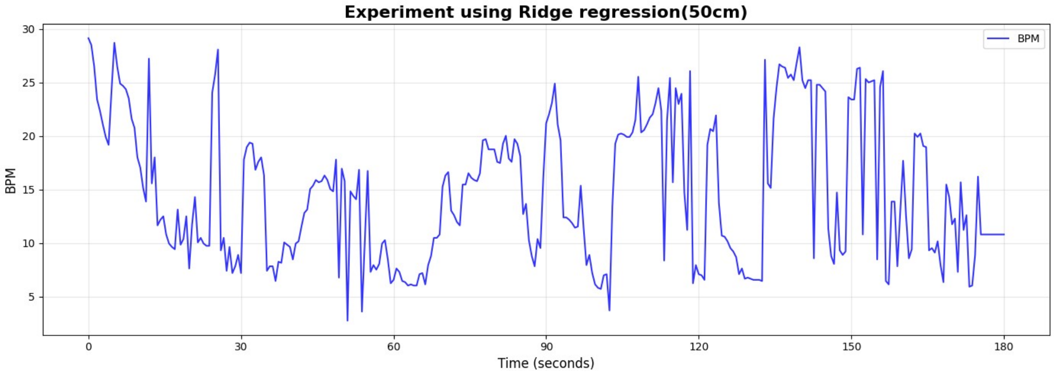


Figure 6.44: Experiment using Ridge regression

As we can see from the Figure 6.43, the first phase of RR is terribly corrupted by noise. The program with only the Pearson coefficients filter estimated the RR in the phase of 20 BPM to be lower than the actual RR. While the 25 BPM also has oscillation. Furthermore, we added the Ridge regression to fit the data, giving the result as the Figure 6.44 illustrated. We can see that the phase of 15BPM is better than before. The phase of 20BPM in the earlier part is relatively close to the actual RR, but the latter part is worse. The 25BPM with oscillation from 25 to 6 covers almost all the phase.

Figure 6.45: Ridge regression RR with background frequency

We have also extracted the frequencies of the background for comparison. Such that the result at least distinguishes background and chest wall, although the RR is inevitably oscillating. As we can see from Figure 6.45, the curve of RR is almost always higher than the frequency of the background during the whole test.

Adaptive filter with penalty term and Ridge regression

As we mentioned in Section 4.2, we observe that the adaptive filter has an advantage in terms of recursive adaptive parameters. Furthermore, the adaptive filter with a forgetting factor or leaky factor is similar to Ridge regression and the linear methods for regression, such as Lasso [27, chapter 3]. We can use a one-step-ahead prediction method to adjust the parameters in real time, which is a method similar to detrending for eliminating the extremely low frequency caused by body movement. Especially in this low SNR case, we tend to rely on probability theory and statistical analysis to track the trajectory of time series data.

$$\hat{\beta}^{\text{Ridge}} = \arg\min_{\beta} \left\{ \sum_{i=1}^{N} (y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j)^2 + \lambda \sum_{j=1}^{p} \beta_j^2 \right\} \tag{6.1}$$

$$\mathbf{w}(m+1) = \mathbf{w}(m) + \mu \cdot e(m) \cdot \mathbf{y}(m) \tag{6.2}$$

$$\mathbf{w}(m+1) = \alpha\mathbf{w}(m) + \mu \cdot e(m) \cdot \mathbf{y}(m) \tag{6.3}$$

In an adaptive filter, we have weighted coefficients noted as **w** in equations 6.2 and 6.3 [16, chapter 4]. We can see that equation 6.1 represents the Ridge regression with a shrinking factor λ, which is similar to the equation 6.3, the leaky factor α, and forgetting factor μ. Where in equation 6.1, x_i is the sampling data, y_i is the output, β_i is the parameter of the model.

6.13 Summary

In general, we have an assumption ahead of our experiment that the respiratory signal is stationary with additive white noise.

According to the experiment with the Burg method, we set it as the 4th order. We saw that the peak of PSD is less stable than for the 8th order. Then we changed the order to 12, and we don't observe overfitting. The reason is that the Burg method does not utilize an inverted matrix, which remains stable in this context. However, with a lower number of orders, the printing timestamp reveals a shorter parameter estimation time. For the stability estimation, the section 6.6 has an apparent assumption based on the metronome testing method. Finally, to achieve comparable experiment results with the ESPRIT algorithm, we set the order to 20 and 32 in the AR model during the test with the metronome.

The advantage of a higher-order model is that it can separate the frequencies that are close to each other, but it might tend to overfit. The model takes the noise into account, but the noise does not make a good contribution to the eigen data. The AR model with a higher order is likely to cause spurious peaks, which will make the peaks sharper but unstable, thereby reducing reliability.

Some studies investigate finding the proper number of orders, such as CRLB and AIC. Cramér Rao Lower Bound(CRLB) sets a lower bound for all unbiased estimated variances. In our study, we encountered the order of the autoregressive model to 8 at the beginning of our attempt based on intuition. To perform a comparison, the experiments involving the FFT method should be conducted in the same environment, with the same light intensity, and using the same measuring distance.

CHAPTER 7

Discussion

7.1 Limitations and Possible Causes

Our design has made progress in detecting frequency, but we still need some improvement in practical application. Such computations may be large in an embedded system. Accordingly, we can implement the Lattice filter algorithm in a VLSI manner, which optimally utilizes the stability advantage of the Burg method. Furthermore, the detection distance needs to be improved.

The speed of convergence should also be taken into account. By introducing VLSI, there should be some probability of improvement in the storage of the algorithm, because VLSI has a better specification of the program.

In terms of the location of the object being tested, we can fuse more sensors into our project, such as Millimeter wave (mmWave) radar. Radar might not be able to obtain the vital signs of people, which can leave this job to the sensor of VL53L. In general, our design investigates the detection of vital signs, specifically respiratory.

Because our academic background is in electronic engineering of embedded systems, the advanced digital signal processing might be challenging. We have the background benefits in improving the efficiency of the algorithm, but the job of selecting a proper algorithm has been the most important job in our design. During our exploration of the study, we found that the research is strongly related to system dynamics and system identification. Nevertheless, instead of machine learning, our design is based on digital signal processing, which also aligns with the consideration of storage in an embedded system. Our study also has a deficiency in the exploration of statistical theory, which can be found in the exploration of Ridge regression.

7.1.1 VL53l SPAD and standard deviation σ

For improving the detection distance, we can alternatively turn to hardware enhancement.

Interference in Single Photon Avalanche Diodes (SPADs) may stem from ambient light, which affects the intensity of the infrared light emitted by the sensor. In cases of high ambient light conditions, the sensor receives a very high intensity, which is the sum of the partial infrared light emitted by the sensor and the ambient light. This high mixed intensity causes the signal intensity to be obscured within the total intensity, resulting in a lower signal-to-noise ratio.

The SPAD data for each pixel can be utilised as a criterion for reliability. For example, we can set a threshold to filter out the pixels with low SPAD. Replacing these pixels with the interpolation around the other pixel. The testing distance is too long if there are a lot of pixels lower than the threshold, which is necessary to inform the user to adjust the measuring distance.

The alternative approach is as follows. First, to normalize the SPAD data, construct a normalized SPAD array by multiplying the ranging data, and then obtain the weighted sum of the pixels' ranging distance.

According to the manual [6], we know that a single photon is the sensitivity of the detector, which is capable of detecting the arrival of individual photons. SPAD represents the infrared light intensity received by each pixel during a single measurement period. The standard deviation is denoted by σ. The sensor performs multiple continuous measurements and obtains multiple single-photon intensities. Based on the intensities, the program computes the standard deviation σ. A lower value of σ represents a more stable measuring signal, lower noise. This fact also results in more reliable quality measurements and more precise distance calculations.

The standard deviation σ of SPAD data could be helpful when processing the depth data of pixels. Instead of simply averaging all the pixels, there is a more sophisticated method. We can normalize the SPAD to a weighted matrix. After that, we multiply the weight and the distance, and sum those two factors. This method is particularly good for the edge of the pixel array, which is less reliable than the center pixels. Another approach to utilizing SPAD data is to drop those low SNR pixels, and then interpolate the values and replace them.

7.1.2 Singularity in Ordinary Least Squares

In the domain of algorithm, we have some detail in special cases. We derived the analytical solution for Least Squares Estimation at 5.3.3.2:

$$\theta = (X^T X)^{-1} X^T Y$$

After obtaining the solution and having the data buffer, we can compute the parameters. In practical application, there may be some cases where $(X^T X)^{-1} X^T Y$ is singular (non-invertible). When $X^T X$ is a singular matrix, the least square estimation method has no solution. The reason may be that the column vector is linearly relevant, i.e., the data has redundant eigenvalues. In general, the larger the data set, the higher the probability of a matrix becoming singular, and it is non-inevitable.

Ridge Regression

Traditional Ordinary Least Squares (OLS) becomes unstable when handling singular or ill-conditioned matrices. Instead, Ridge regression is an improved form of least squares regression that introduces some bias into the estimates. Ridge regression trades unbiasedness for a significant reduction in variance. This trade-off enhances stability and generalization, making ridge regression a widely used estimation technique.

While in the section of 6.12, we derived the Ridge regression as an approach of smoothing the parameter of the model instead of smoothing the raw data. These two usages of Ridge regression are different. The latter one is smoothing the parameter based on the maximum a posteriori of the parameter. This approach assumes the parameters have a Gaussian distribution [28, chapter7].

The Ridge regression we used for smoothing the parameters is before the step of the time series analysis. This fact implies that the Ridge regression is after the Direct component cancellation. In our design, we set the shrinkage factor with a fixed number, which can be processed with an approach by finding the scores to obtain an optimum one. The score is called the loglikelihood.

7.1.3 Pseudo-inverse

Moore–Penrose Pseudoinverse is also an approach for solving the problem in the LSE, but it differs in motivation and application scenario from Ridge Regression. When the matrix $X^T X$ is not invertible, then the solution of OLE

$$\hat{\beta} = (X^T X)^{-1} X^T y$$

is not exist.

$$\hat{\beta}_{Pseudo\ inverse} = X^{+} y$$

We note X^{+} as the Pseudo-inverse. The pseudo-inverse can always find a solution with the smallest norm. In contrast, the Ridge Regression has an additional penalty term.

Similar to PCA, Ridge Regression, and Pseudoinverse are techniques for handling large datasets, such as image pixels, where redundant data can lead to matrix singularity.

7.1.4 Zero-Crossing Detection VS parametric PSD

We propose two different methods of extracting respiratory frequency: Zero-Crossing Detection and Parametric PSD method.

An extra low-frequency component is present when the body is moving slowly. Typically, we can use polynomial fitting to mitigate this effect. As Section 5.3.3.2 mentioned, we can detect the period when the respiratory curve crosses the newly fitted curve. The time interval between two zero points is period T, then converted to frequency, f=1/T. The advantage is that the performance is highly real-time, and the algorithm is simple. This fact is perfect for respiratory detection and friendly for processors. However, this approach has a disadvantage, which is poor anti-noise. Any high-frequency components could cross the zero point.

We perform the parametric PSD procedure after the fitting process of extremely low frequency cancellation. This order of steps is based on the assumption that the signal is proximate stationary. And based on this assumption, we build an autoregressive model. This model assumes that the linear combination of the previous observations with an additive white noise term can represent the current observation. The AR model can produce high-resolution frequency, suitable for short data sets. However, the AR model has relatively poor real-time performance and inevitable initialization.

7.1.5 System Feasibility and Application

Based on the previous technical assumption, the system for testing RR is feasible, although limited in distance and FOV, with the background having no quantified experimental data. However, at least we have verified the high-resolution frequency recognition. Therefore, we could consider refining the system's feasibility and application. For example, testing at the bedside involves monitoring elderly people while they sleep, a scenario with minimal interference.

7.2 Refinements to Address Limitations

Our design tends to low power consumption in general by utilizing the sensor of VL53L, which eventually obtains the vital sign in a closed space. If we don't have the constraint of low power consumption, we can use the sensor with a higher SNR, such as radio frequency and Rada. The fact is obvious that VL53L is challenged when the FOV is introduced into the background. The sensor fusion can perhaps be used by combining the VL53L and the sensor of Rada.

7.2.1 Testing distance

Since our design is limited in distance, we can consider a wearable manner. The printed circuit board can be embedded with a wireless module that transfers data to the server.

7.2.2 Active noise cancellation with adaptive filter

Since the SNR is low at long distances, we can utilize active noise cancellation. Some sophisticated techniques can denoise based on the signal and the noise in these two areas, which is called active noise cancellation.

In system identification, the model can be described in terms of complexity and interpretability. The general adaptive model is close to the nonlinear models, because the flexibility is higher than the ordinary least squares estimation.

7.2.3 Pseudo-Random Binary Sequence (PRBS)

The PRBS technique can be used for modulating the infrared light. We sense the intensity of infrared light comes from the sensor VL53L7, which is always at the highest emission rate. However, we can skip some time intervals of emission based on the PRBS technique, such that the emission carries the modulated information. In this way, we potentially have a higher SNR at the receive side. The emission of infrared light may not be modifiable, which requires searching the manual of the sensor.

7.2.4 PCA algorithm with surface normals

The original method captures the breathing signal by recording the most reliable pixels with their Z-axis data in the middle of the array. Typically, the pixels in the middle of the array are more correlated with the breathing signal, thereby enabling its recognition. However, in this way, the pixels in the array do not provide surface normal information [11], i.e., the angle resolution is not used.

The cloud points lie on the object's surface, whose area is often proportional to the distance between the object and the sensor. Calculating the deviation of cloud points' covariance to refer to distance is more sophisticated. When the chest is parallel to the sensor, the deviation of the point cloud's projection on the XOY plane can adequately describe the breathing signal. In cases where the chest is not parallel to the sensor, Principal Component Analysis should be used to identify the largest covariances among these cloud points, which are related to the chest expansion and contraction.

The description above is an intuitive way to comprehend the working mechanism of PCA in our study. Dimensions greater than three cannot be graphically visualised. Instead, we can describe a system with a higher dimension by using mathematical equations.

7.2.5 Sensor fusion

We can use an IMU(Inertial measurement unit) to detect the body on a large-scale swing. When the body swings, the respiratory identification system should pause and introduce a compensation algorithm. An RGB camera can also be included for long-distance measuring scenarios, specifically identifying regions of interest for the human trunk and thorax.

Bibliography

[1] C. Uysal and T. Filik, “Contactless respiration rate estimation using music algorithm,” in *2017 10th International Conference on Electrical and Electronics Engineering (ELECO)*, 2017, pp. 606–610.

[2] M. H. Hayes, *Statistical Digital Signal Processing and Modeling*, 1st ed. New York: John Wiley & Sons, 1996.

[3] D. Simon, *Optimal State Estimation: Kalman, H-infinity, and Nonlinear Approaches*. Hoboken, NJ: John Wiley & Sons, 2006.

[4] J. P. Burg, “Maximum entropy spectral analysis,” Ph.D. dissertation, Stanford University, Stanford, CA, 1975.

[5] C. Uysal, A. Onat, and T. Filik, “Non-contact respiratory rate estimation in real-time with modified joint unscented kalman filter,” *IEEE Access*, vol. 8, pp. 99 445–99 457, May 2020.

[6] *UM3038: VL53L7CX Time-of-Flight 8x8 Multizone Ranging Sensor User Manual*, STMicroelectronics, 2022, accessed: 2025-09-01. [Online]. Available: https://www.st.com/resource/en/user_manual/um3038-vl53l7cx-timeofflight-8x8-multizone-ranging-sensor-stmicroelectronics.pdf

[7] J. Rumiński, “Analysis of the parameters of respiration patterns extracted from thermal image sequences,” *Biocybernetics and Biomedical Engineering*, vol. 36, no. 4, pp. 731–741, 2016. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0208521616301905

[8] C. Niclass, M. Gersbach, B. S. Rueger, M. W. Fishburn, and E. Charbon, “Time of flight: Principles, challenges, and performance,” *Sensors*, vol. 19, no. 3, p. 690, 2019. [Online]. Available: https://www.st.com/content/dam/technology-tour-2017/session-1_track-4_time-of-flight-technology.pdf

[9] S. Bernhard, A. Züfle, A. Martin, and B. Eskofier, "Respiration rate estimation with depth cameras: An evaluation of parameters," in *2020 42nd Annual International Conference of the IEEE Engineering in Medicine & Biology Society (EMBC)*, 2020, pp. 567–570.

[10] M. Martinez and R. Stiefelhagen, "Breath rate monitoring during sleep using near-ir imagery and pca," in *Proceedings of the 21st International Conference on Pattern Recognition (ICPR2012)*, 2012, pp. 3472–3475.

[11] C. Anqi, N. Xiuhua, and Z. Long, "Principal component analysis(pca) normal estimation based on tof sensor," in *2023 8th International Conference on Intelligent Informatics and Biomedical Sciences (ICIIBMS)*, vol. 8, 2023, pp. 291–294.

[12] C. Uysal and T. Filik, "Rf-based noncontact respiratory rate monitoring with parametric spectral estimation," *IEEE Sensors Journal*, vol. 19, no. 21, pp. 9841–9849, 2019.

[13] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. Cambridge, MA, USA: MIT Press, 2016, http://www.deeplearningbook.org. [Online]. Available: http://www.deeplearningbook.org

[14] S. Kay and S. Marple, "Spectrum analysis—a modern perspective," *Proceedings of the IEEE*, vol. 69, no. 11, pp. 1380–1419, 1981.

[15] J. Kempfle and K. Van Laerhoven, "Respiration rate estimation with depth cameras: An evaluation of parameters," in *Proceedings of the 5th International Workshop on Sensor-Based Activity Recognition and Interaction*, ser. iWOAR '18. New York, NY, USA: Association for Computing Machinery, 2018. [Online]. Available: https://doi.org/10.1145/3266157.3266208

[16] S. V. Vaseghi, *Advanced Digital Signal Processing and Noise Reduction*, 2nd ed. Chichester, UK: John Wiley & Sons, 2000.

[17] L. Ljung, *System Identification: Theory for the User*, 2nd ed. Upper Saddle River, NJ: Prentice Hall PTR, 1999.

[18] K. Kajihara, S. Izumi, S. Yoshida, Y. Yano, H. Kawaguchi, and M. Yoshimoto, "Hardware implementation of autoregressive model estimation using burg's method for low-energy spectral analysis," in *2018 IEEE International Workshop on Signal Processing Systems (SiPS)*, 2018, pp. 199–204.

[19] J. G. Proakis and D. G. Manolakis, *Digital Signal Processing: Principles, Algorithms, and Applications*, 3rd ed. Upper Saddle River, NJ: Prentice Hall, 1996.

[20] C. Collomb, "Burg's method, algorithm and recursion," 2009. [Online]. Available: https://api.semanticscholar.org/CorpusID:124069813

[21] J. D. Hamilton, *Time Series Analysis.* Princeton, NJ: Princeton University Press, 1994.

[22] G. Box, *Box and Jenkins: Time Series Analysis, Forecasting and Control.* London: Palgrave Macmillan UK, 2013, pp. 161–215. [Online]. Available: https://doi.org/10.1057/9781137291264_6

[23] G. Strang, *Introduction to Linear Algebra*, 4th ed. Wellesley, MA: Wellesley-Cambridge Press, 2009.

[24] R. Roy, A. Paulraj, and T. Kailath, "Esprit–a subspace rotation approach to estimation of parameters of cisoids in noise," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 34, no. 5, pp. 1340–1342, 1986.

[25] T. Katayama, *Subspace Methods for System Identification*, ser. Communications and Control Engineering. Springer-Verlag London, 2005.

[26] A. Papoulis and S. U. Pillai, "The power spectrum," in *Probability, Random Variables, and Stochastic Processes*, 4th ed. McGraw-Hill, 2002, ch. 9, pp. 385–420.

[27] T. H. T. Friedman, "The elements of statistical learning data mining, nference, and prediction," 2009. [Online]. Available: https://www.sas.upenn.edu/~fdiebold/NoHesitations/BookAdvanced.pdf

[28] S. Boyd and L. Vandenberghe, *Convex Optimization.* Cambridge University Press, 2004.