UNIVERSITY OF CALIFORNIA,
IRVINE

Framework and Algorithms for Wearable Medical Applications

DISSERTATION

submitted in partial satisfaction of the requirements
for the degree of

DOCTOR OF PHILOSOPHY

in Electrical and Computer Engineering

by

SeungJae Lee

Dissertation Committee:
Professor Pai H. Chou, Chair
Professor Fadi J. Kurdahi
Professor Tony Givargis

2016

DEDICATION

I would like to dedicate my dissertation
to my beloved parents and family

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGMENTS

I would never have been able to complete my dissertation without the guidance of my advisor and committee members, and the understanding and support of my family and friends.

I owe my deepest gratitude to my advisor who made all of this possible. My research would not have been completed, or even started without his support and advice. His passion for the work and brilliant ideas have taught me how to analyze situations, discover approaches to solutions, and improve situations with solutions.

I would like to express my gratitude to my committee members, professor Fadi J. Kurdahi, and professor Tony Givargis. I am honored to have them in my committee.

I gratefully acknowledge the funding sources that made my Ph.D. work possible. This work was sponsored by the NIH STTR Phase I Grant R41HL112435, 1R41HL127868-01, 1R41GM113463-01A1, 1R41HD079051-01A1, Phase II Grant 2R42HL112435-03 through subcontracts by the QT Medical. The content is solely the responsibility of the authors and does not necessarily represent the official views of the sponsors.

I offer my regards and blessing to all of my colleagues in Professor Chou's group. It was my great pleasure to work with such young and talented people. I would like to give special thanks to Jun Luan and Chengjia Huo for providing part of the experimental data used in this work.

Finally, to my parents and family I owe my sincerest appreciation for their continued support and constant encouragement.

CURRICULUM VITAE

SeungJae Lee

EDUCATION

Ph.D., Electrical Engineering and Computer Science — **2016**
University of California, Irvine — *Irvine,California*

M.S., Electrical Engineering and Computer Sciences — **2006**
Korea Advanced Institute of Science and Technology — *Daejeon,Korea*

B.S., Electrical Engineering and Computer Sciences — **2004**
Korea Advanced Institute of Science and Technology — *Daejeon,Korea*

RESEARCH EXPERIENCE

Graduate Research Assistant — **2012–2016**
University of California, Irvine — *Irvine, California*

Researcher — **2006-2011**
Korea Telecom — *Daejeon, Korea*

TEACHING EXPERIENCE

Teaching Assistant — **2004–2006**
Korea Advanced Institute of Science and Technology — *Daejeon, Korea*

REFEREED CONFERENCE PUBLICATIONS

1. Jun Luan, Ting-Chou Chien, Seungjae Lee, Pai H. Chou, HANDIO: A Wireless Hand Gesture Recognizer Based on Muscle-Tension and Inertial Sensing, in *Proceedings of the Global Communications Conference: Selected Areas in Communications: E-Health (GC 15 - SAC - E-Health)*, San Diego, CA, USA, December 6-10, 2015.

2. Jun Luan, Seungjae Lee, Pai H. Chou, Fast Compressive Sensing Based on Dominant Frequency Estimation, in *Proceedings of the 12th IEEE International Conference on Mobile Ad hoc and Sensor Systems (IEEE MASS)*, Dallas, TX, USA, October 19-21, 2015.

3. Jun Luan, Seungjae Lee, Pai H. Chou, Low-Power Detection of Sternocleidomastoid Muscle Contraction for Asthma Assessment and Control in *Proceedings of the International Symposium on Low Power Electronics and Design,(ISLPED)*, Rome, Italy, July 22-24, 2015.

4. Seungjae Lee, Jun Luan, Pai H. Chou, ECG Signal Reconstruction from Undersampled Measurement Using A Trained Overcomplete Dictionary, in *Proceedings of the 4th International Conference on Convergence and its Application*, Jeju Island, Korea, November 12-14, 2014.

5. Seungjae Lee, Jun Luan, Pai H. Chou, New Approach to Compressing ECG Signals with Trained Overcomplete Dictionary, in *Proceedings of the 4th International Conference on Wireless Mobile Communication and Healthcare (MobiHealth 2014)*, Athens, Greece, November 3-5, 2014.

Patents

1. SeungJae Lee and Pai H. Chou, "A Method for Beat-by-beat QT Analysis with Multiple Lead ECG Recording", In Processing

ABSTRACT OF THE DISSERTATION

Framework and Algorithms for Wearable Medical Applications

By

SeungJae Lee

Doctor of Philosophy in Electrical and Computer Engineering

University of California, Irvine, 2016

Professor Pai H. Chou, Chair

Wearable embedded systems with sensing, communication, and computing capabilities have given rise to innovations in e-health and telemedicine in general. The scope of such systems ranges from devices and mobile apps to cloud backend and analysis algorithms, all of which must be well integrated. To manage the development, operation, and evolution of such complex systems, a framework systematic framework is needed. This dissertation makes contributions in two parts. First is a framework for defining the structure of a wide range of wearable medical applications with modern cloud support. The second part includes several algorithms that can be plugged into this framework for making these systems more efficient in terms of processing performance and data size. We propose a novel QT analysis algorithm that can take advantage of GPU as well as in a server-client environment, and we show competitive results in terms of both performance and energy consumption with or without parallelization. We also propose ECG compression techniques using trained overcomplete dictionary. After constructing the dictionary through learning process with a given dataset, the signal can be compressed by sparse estimation using the trained dictionary. We propose reconstructing ECG signal from undersampled data based on compressive sensing framework that can reconstruct the ECG signals precisely from fewer samples so long as the signal is sparse or compressible. Together, these algorithms operating in the context of our proposed framework validate the effectiveness of our structured approach to the framework for wearable

medical applications.

Chapter 1

Introduction

1.1 Motivation

Wearable devices refer to electronic technologies or computers that are incorporated into items of clothing and accessories that can be worn comfortably on the body. Generally, wearable technology will have some form of communication capability and will allow the wearer's access to information in real time. Some of wearable devices have computing power to solve the problem the wearer is facing. Another type of devices collect the signals or context by using built-in sensors or context engine. Examples of wearable devices include watches, glasses, contact lenses, e-textiles and smart fabrics, headbands, beanies, and caps. There are more invasive versions of the concept as in the case of implanted devices such as micro-chips or even smart tattoos. Whatever the device is worn on or incorporated into the body, the purpose of wearable technology is to create constant, convenient, seamless, portable, and easy to access the information in real time.

Medical-purpose wearable devices differ from general-purpose ones in several aspects[52]. First, user interaction tends to be much more limited. Second, the performance/power trade-off can often be made in terms of size of processor performance, data storage, and battery capacity. Third, there is an emphasis on signal processing that does not exist to the same degree in general-purpose wearable computing applications. Finally, wearable medical computers face more stringent requirements in terms of privacy, reliability, government regulations, and the manufacturer's legal responsibility. In this work, we will focus on the medical purpose wearable devices.

1.2 Technology

Wireless Body Sensor Networks (WBSN) are revolutionizing healthcare technology. These networks are comprised of wearable devices with sensors, which can detect physiological signals, gateways for supporting to access the networks, and back- and front-end servers for managing, analyzing, and displaying the information. The technology for WBSN and wearable devices is not so new. The core technologies are about sensors, apps, and network technology such as WiFi, Bluetooth, Zigbee and cellular networks. In these days, it tends to connect smartphones and tablets for gateways or user interfaces to contact users such as the doctors, caregivers, or the patient's guardian.

1.2.1 Applications

In [19], wearable medical devices can be classified in three categories: monitoring devices, medical aid devices, and rehabilitation assistance devices. A *wearable monitoring* system helps in managing the treatment of chronic diseases such as heart diseases, asthma, and diabetes, and monitoring vital signals from patient's or wearer's body. The target signals for monitoring can be electrocardiogram, blood oxygen level, respiration, and body fat. A *medical aid* device, which is designed to provide long-term assistance to patients who have disabilities. For example, consider hearing aids to help the visually impaired with obstacle avoidance and navigation. A wearable *rehabilitation assistance* device combines functions of monitoring and medical aid device for patients in rehabilitation. The monitoring function assists rehabilitation of patient to keep away from potentially dangerous situations or risks. Further, the medical aid device can help with temporary disabilities during rehabilitation.

1.2.2 Frameworks

In general, a framework is a real or conceptual structure intended to serve as a support or guide for building of systems or applications. Since the development of the system requires many considerations, frameworks can enable developers to build up their target system or application more efficiently. Some frameworks include actual programs, specific programming interfaces, or programming tools. Generally speaking, a framework may provide a set of functions within the system or application, how they interconnect, and how they communicate among the components in the framework. In this research work, we will focus on existing frameworks for wearable medical devices in Chapter 2, and present how to build up the system by using the frameworks.

1.3 Challenges

Designing wearable medical devices require several considerations. The technical challenge stems from its use under varying and demanding conditions[19]. For example, some devices are used in specific environments, such as hospitals, houses, or sports fields. Others can be used during specific periods of time, as while sleeping, in periods of high risk, or during exercise. The technical issues depend on the specific characteristics of the application [36].

Table 1.1: Issues to be considered when design wearable medical devices

	Issue	Description
Technical	Sensor	What kinds of sensors needed
	Data handling	Amount of data to be recorded, stored and transmitted
	Decision support	To help doctors or caregivers, what kind of analysis needed.
	Feedback	How to give feedback to patients
	Communication	How to transmit the data from sensor to server or service provider Consider interoperability issues with legacy system
	Physical design	Shape, size, and weight not to interrupt patients' movements
	Reliability	More reliable than general purpose devices
Non-technical	User acceptability	User friendliness of using the device
	Privacy & Security	Issues about the patient's rights and confidentiality

1.3.1 Technical issues

Sensors

Sensors in the wearable device measure physiological and kinesiological signal from the human body. They should have the characteristics of being non-invasive, reliable, compact, wearable, replaceable and capable of integration with the device. The choice of the sensors depends entirely on the target application. If the target application to build is a heart-rate monitoring system, it is necessary to determine the specifications of the sensor such as the resolution, the sampling rate. Then, we should choose one of the sensors that meet the requirements.

Data handling

In general, a wearable type device has limited power and computational capability. Due to the limitation, it is necessary to make a decision about the format and amount of data to be recorded, stored and transmitted.

Decision support

The main purpose of the wearable medical device is to help doctors or caregivers by providing useful information about the patient's status. It does not mean raw data from sensors, but the raw data should be processed to provide more useful information for decision making. Hence, we need to consider about what information needs to be offered to the doctor or the caregiver, and how to process it to obtain the useful information.

Feedback

Feedback is a means to delivering the information or the result of deciesion supports by the wearable medical device to the wearer, the patient, or caregivers. According to the purpose of the device, it informs a patient, a wearer, or a caregiver of the collected data itself or the result of decision supports by the device. For example, a glucose concentration monitoring system provides the injection of insulin by the decision support function. In this system, the control of the actuator requires feedback.

Communication

Communication is one of the core technologies in wearable medical devices. Generally, it involves the link between the sensor board and the back-end system and the link between the back-end system and the healthcare provider. There is standardization issue to set aside interoperability between a device and a device.

Physical design

The physical design issue deals with the physical shape, size, and the weight of a wearable and its ergonomics.

Autonomy

The autonomy of the wearable device is about the power requirements of the device, which should be met for specified periods of use.

Reliability

Medical-purpose electronics must be more reliable than general-purpose ones, since the patient's health or life is at stake. The reliablity of the wearable devices is particularly affected by factors related to the conditions of their use. If the operating environment cannot be controlled to a certain extent, then the devices may not operate reliably when exposed to extreme factors such as humidity and temperature. The medical device should be designed in consideration of the use environment for decreasing operational risks.

The reliability at the aspect of communications should also be considered. In general, wearable medical devices are assumed to have communication capabilities. It may store or transmit the collected data to the gateway, servers, or health providers. Moreover, the wearable medical devices are accompanied by the mobility of the wearer. To send the collected information to the destination reliably, it is necessary to set aside quality-assured and efficient communications.

1.3.2 Non-technical issues

User acceptability

The user acceptability is the issue related to the friendliness of using the device. This issue depends on the conditions of its use, complexity of setup and operating the device, and its comfort and wearability.

Privacy and Security

In the medical field, privacy is defined as the right and desire of a person to control the disclosure of personal health information [53].

To utilize wearable medical devices, health care providers must have access to the information to use it for improved health care. To keep the privacy of individuals for medical purposes, regulatory frameworks have been defined. According to HIPAA (The American Health Insurance Portability and Accountability Act) Privacy Protection rules [48], all medical records, billing, and patient accounts meet certain consistent standards with regards to documentation, handling, and privacy. HIPAA requires that all patients be able to access their own medical records, correct errors or omissions, and be informed how their personal health information (PHI) is shared or used. To use the personal health information for only medical purpose, health providers must have a broad meaning of confidence. Usually, confidentiality is defined as the controlled release of personal health information to a care provider or information custodian under an agreement that limits the extent and conditions under which that information may be used or released further.

Especially due to the characteristics of the electronic records, personal health information may be exposed to risks during storing, transmitting, and sharing. Security issue to keep those sensitive information secure and to utilize it confidentially is another problem. According to HIPAA, Security is defined as a collection of policies, procedures, and safeguards that help maintain the integrity and availability of information systems and control access to their contents. HIPAA also enforces to comply HIPAA Security Rules [27] to ensure the confidentiality, integrity, and availability of all e-PHI (Electronic Protected Health Information). Since e-PHI is a subset of the information covered by HIPAA Privacy Protection Rules, all PHI may not be the target for protection by HIPAA Security Rules. Thus, it has a limitation not to apply PHI transmitted orally or in writing.

Even though many regulations and rules surrounding the privacy and security issues in health care are appeared, still they have many things to be improved [43].

1.4 Contributions

Our research work comprises two parts: a framework for wearable medical devices and a set of algorithms for medical applications. In the framework part, we introduce the state-of-the-art in the field of the wearable medical devices and its applications. Particularly, we classify the frameworks for designing the applications by using wearable medical devices and sensor network systems. Further, we depict the issues to be considered, when the wearable medical devices and the systems for providing entire services are designed. Finally, we show a case study of developing a wearable medical device utilizing the framework.

In the algorithms part, we propose several useful algorithms for efficient sensing, compression, and decompression using signal-specific characteristic. We apply our algorithms to QT analysis of ECG signals as used in the case study for the framework. For evaluation, we implement our proposed algorithm for a client-server environment and an embedded environment for real-time processing. We show that porting to a modern GPU-based architecture can result in improved performance in processing time significantly while saving energy.

1.5 Dissertation outline

In Chapter 2, we review widely used frameworks for designing wearable devices and show a case study of an end-to-end system for utilizing wearable-type medical sensors. Further, we also introduce issues to be considered for designing medical applications.

Chapter 3 to Chapter 6 cover the algorithms. Chapter 3 proposes a signal reconstruction algorithm for an efficient sensing method. Chapter 4 proposes signal compression and decompression algorithms that utilize signal characteristics of electrocardiogram (ECG). Chapter 5 introduces a novel approach to QT analysis and its efficient implementation in a server-client environment. We also show an implementation of our proposed algorithm on an embedded system in Chapter 6. Particularly, we show that performance improvement and energy saving can be obtained by parallelization on the GPU.

Chapter 2

Frameworks for Medical Monitoring Systems

A monitoring system is not just a device but a full system from the front-end to the back-end in the cloud. The purpose of this chapter is to show a comprehensive view for categorizing existing frameworks. This chapter consists of three parts: categorization of frameworks, review of existing frameworks, and a case study. We categorize existing frameworks into three different planes to address what they pursue in common. We also review existing frameworks and their characteristics. Finally, we present an end-to-end system for ECG monitoring and analysis as a case study of our framework.

2.1 Generic Framework for Medical Systems

Health monitoring of physiological parameters through the use of wearable biosensors has been a research area of high interest in recent years [37]. To achieve the main goal of the system and to improve the performance and longivity, the device may have higher system

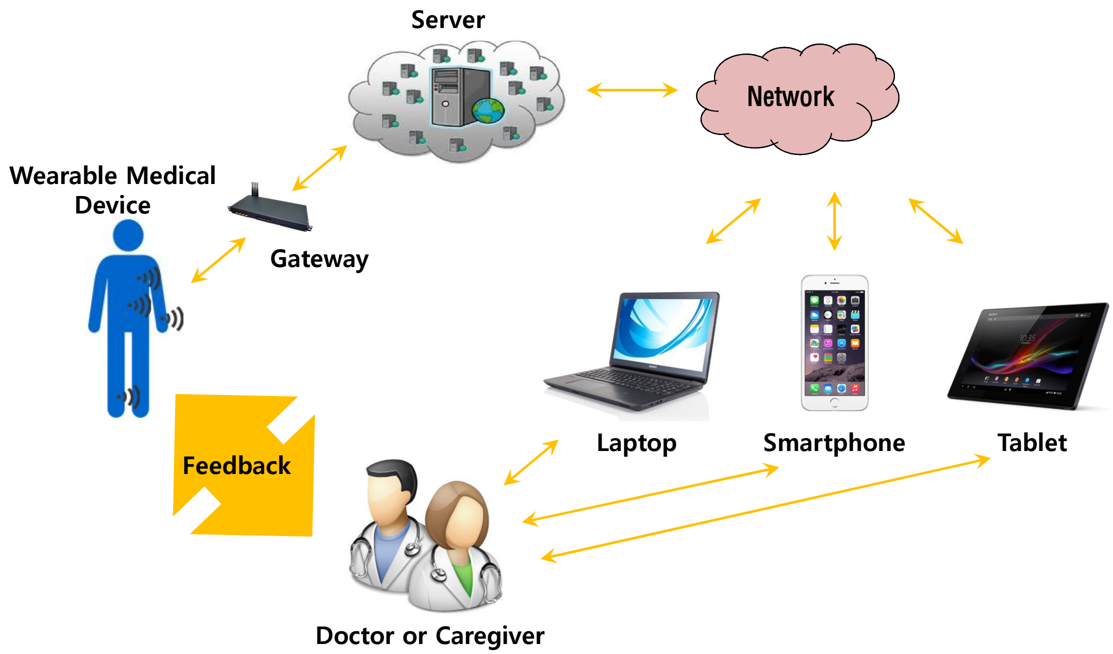

Figure 2.1: Generic framework for wearable medical devices and systems

complexity. Accordingly, many things should be considered during development and design. A framework can help the system developers and designers make their system more efficient. It provides a guide on what the developers should consider to build a system or an application.

Fig. 2.1 shows a generic framework for wearable medical systems. The generic framework comprises several functional components. The core components are a wearable medical device, gateway, servers, and channels to connect clients such as laptops, smartphones, and tablets over the network to the servers. The wearable medical device captures physiological signals. Through the gateway, the signals are transmitted to the servers, which manage, analyze, visualize, and share the information. Physicians or caregivers can access the processed data over the network.

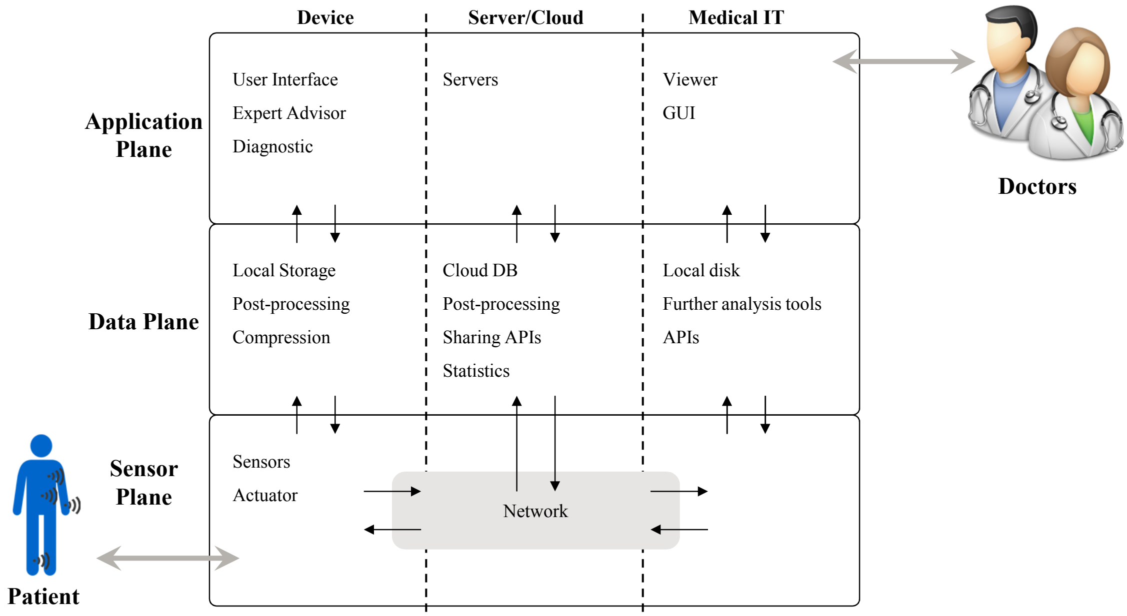

Figure 2.2: Classification of frameworks

2.2 Taxonomy of Frameworks

In this section, we categorize the functions of frameworks into our plane knowledge according to their functions. We also introduce the characteristics of the items in each plane to consider, when developers build up their systems in the corresponding plane.

Fig. 2.2 shows three planes: the sensor plane, the data plane and the application plane. Usually, target biosignals are from the patients' body, and the processed data through the system will be transferred to doctors, caregivers, or the patients themselves. In the sensor plane, the main goal is to capture the physiological signals from the target. The signals will be transmitted to the data plane through wireless or wired communications, then stored or shared. The signals are analyzed and reproduced as processed data in the application plane.

Table 2.1: Classification of Frameworks

	Sensor	Data	Application
Definition	- Plane for collecting raw data from sensors	- Plane for managing data from sensor plane	- Plane for processing the data to provide processed data or visualize raw/processed data
Feature	- Collecting the data from the target	- Storing the data	- User interface for visuallization, interaction, and presentation
	- Communication with the data plane	- Interfacing for storing and retrieving the data	- Analysis
	- Sensing component depends on the target application	- Compression and decompression	- Statistics
		- Management function - Categorizing, grouping, and sharing	
Issue	- Privacy : obtrusiveness problem - Power consumption - Physical design : size, material - User acceptability - Use environment - Physical design	- Security for accessibility	- Security for authentication and authorization

2.2.1 Sensor plane

We define the sensor plane to be the layer for collecting physiological or kinesiological parameters from the sensors. For a monitoring system, normally the sensors might be attached on a target, which can be a patient or a wearer of the device. The sensor plane includes sensing components, preprocessing on the sensor device such as filtering and compressing, and communication between the sensor and data planes. The sensing components are generally dependent on the target application. If the physician wants to monitor the concentration of glucose in a patient, the sensing component should measure the glucose level.

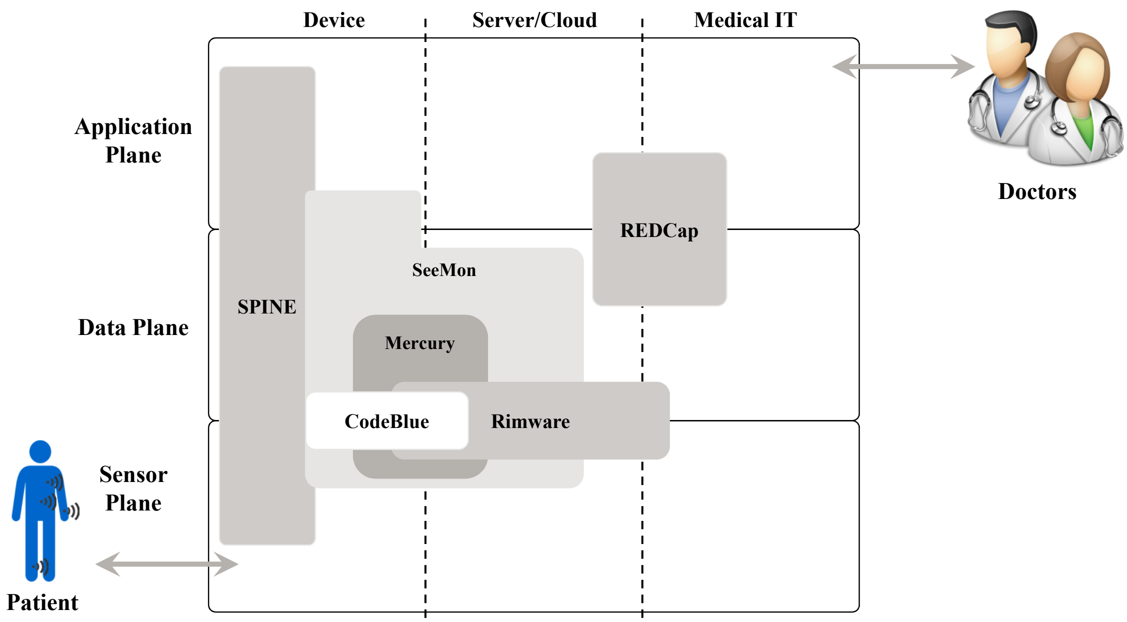

Figure 2.3: Positioning of the frameworks on our plane

Since most of the sensing devices are not equipped with powerful computational capabilities, only simple preprocessing techniques such as filtering and compressing are usually built in. Preprocessing can make the device more self-contained and can be useful even when the network connection is down, but preprocessing should minimize data distortion. In general, the sensor plane has at least one wired or wireless communication channel to transmit raw data to the data plane sensor plane.

In the sensor plane, considerations include the lifetime and physical design of the sensor device, user acceptability, privacy and security problem. The main considerations of the sensor plane is related to the sensor device directly. The sensor device should consider how much time to keep collecting the signal. If the sensor should catch the motions, it should be actuated when the motion occurs. If the sensor needs to collect electrocardiographic signals, it should keep on for some period of time.

According to the target application, developers should consider the lifetime of the device related to the power consumption. User acceptability of the device depends on the conditions of its use, its user friendliness both during setup and operation, its comfort and wearability, and the quality of the feedback by the users. When the system is designed, obtrusiveness of the sensor components, which is highly related to privacy of users, should be considered as well. Furthermore, security issue is also one of the important things to be considered, since the medical data is critical and sensitive for both patients and physicians.

2.2.2 Data plane

The data plane is the layer that handles the data related functions in the framework. Generally, the actions in regard to managing and interfacing the data the data transmission are performed in this plane.

Managing the data transmitted from the sensor plane includes all functions related to the data itself such as storing, compressing and decompressing, categorizing, grouping, and sharing. Some frameworks in this plane support device managing functions, which can activate/deactivate the sensors and control the specific parameters during sensing physiological signals.

Since the data plane is located between the sensor plane and the application plane, it plays a role as a bridge of the entire system. While the sensor plane or the application plane handles specific targeted data, the data plane should set aside generality. To support collecting the data from the multiple sensor devices, it should have the corresponding interfaces. At the aspect of the application, it may need to combine or reuse the data by the requirements

Due to the sensitiveness of the physiological data, it should provide those interfaces with keeping security for accessing the data. Since the data plane provides two channels for transmitting, one for the sensor plane and the other for the application plane, it should have different policies about accessing the data through each channel.

Between the sensor plane and the data plane, the key of security is transmitting the data securely. The security of this channel should support secure data transmission with keeping away from contaminating or sniffing the data. Furthermore, the data plane only allows registered sensor devices to transmit the data. Between the data plane and the application plane, the security policy should focus on confidentiality. Usually, the application plane consumes the data managed by the data plane. To keep the confidentiality of the entire system, the data plane should give permission to access to only authorized users. The features about the communication among the planes will be handled in Section 2.2.4.

2.2.3 Application plane

The application plane is for processing the data to provide analysis results or visualization of the raw and processed data. Normally, it comprises application engines and user interface. The application engine is a core component highly dependent on the kinds of analysis needs. The user interface at this plane is for visualization, interaction to give feedback, or control the device, and presentation in general. This plane also has communication channels with the data plane and sometimes the sensor plane. The confidentiality issue is the most important problem in this plane. Since many different users can access the sensitive data through the application plane, authentication and authorization are important to give permission to access the data to only the allowed users.

User interface is another key factor in the application plane. The application plane is the only plane that interacts with users who are not administrators or developers. According to the target application, the user interface should be designed to interact with users easily. User interaction encompasses all actions by users regarding the functions of the application, such as retrieving data, displaying results, or controlling the sensor devices. The user interface can be a channel to obtain and provide feedback. The users, usually physicians or caregivers, obtain the results of their requests, then give feedback to the wearers such as instructions or operations of the actuators for treatments through the user interface. Furthermore, applications sometimes require processing capability for a large number of requests by multiple users simultaneously. Hence, the performance factors such as the number of simultaneous processing requests and concurrent connected users in accordance with the requirements of the target application should be considered before designing the application.

2.2.4 Communication among Planes

Communication among the planes is one of the important features of the frameworks for the wearable medical systems. Communication technology is built into all levels, from devices in the sensor plane to the data plane and application.

Communication between Sensor and Data Planes

The sensor plane handles collecting raw data, which is captured by the sensor device and transmitted to servers in the data plane. For mobile and wearable devices, wireless communication is preferred [60], although it is subject to several constraints, including power, mobility, topology, data rate, and security.

First, the communication between sensor plane and data plane is power-constrained for battery-powered devices, and communication consumes relatively high power to sensing and processing in the device. Second, the mobility of the wearer impacts the communication range during monitoring [13].

Second, the choice of type of communication interface may depend on the mobility and range. Higher mobility may imply longer range, which may necessitate higher-power interfaces such as Wi-Fi. Lower mobility would allow the use of shorer-range, lower-power interfaces such as Bluetooth, Zigbee, or IrDA.

Third, the topology of the sensor network impacts the operational efficiency. With repect to the application, users may want to operate the entire system in a centralized manner. In this case, the star topology is helpful to control each sensor node. However, it has a critical problem, namely a single central point of failure. In contrast, mesh topology is more robust and does not have the central point of failure problem, but at the expense of resource redundancy. Some hybrid topologies have been proposed to address these problems, including extended star topology or partially mesh topology. The topology should be determined in consideration with the scale of network and expected usages.

Fourth, the data rate of the communication protocol should also be considered. The amount of data is dependent on an application. The signal to be captured at the sensor plane may have explosive quantity of data or only small amounts of data with respect to the application.

If the application requires only some feature data calculated on the sensor device, the communication channel does not have to support higher data rate. Applications that perform monitoring of the signal in real-time may require low latency and determinism, with or without high data rate. Table 2.2 shows the features of the widely used wireless communication standards. When we design an end-to-end system, the communication standard should be selected in consideration with the data rate.

HIPAA (The American Health Insurance Portability and Accountability Act) enforces to observe the security rules to keep the electronic protected health information secure by HIPAA security rule [27]. To build up a HIPAA compliant system, the system should comply the following recoomendations. First, to err on the safe side would be to combine two methods of encryption - send encrypted files over an encrypted connection. Second, when it comes to remote access, VPN (Virtual Private Network) should be used. Third, for web-based data access only SSL (Secure Socket Layer) connection is allowed. Fourth, following

Table 2.2: Wireless Communication Standards

	Range	**Data Rate**	**Power Consumption**	**Freq.**
Zigbee	10-75m	20-250kbps	30mW	868/915MHz 2.4GHz
Bluetooth	10-100m	1-3Mbps	2.5-100mW	2.4GHz
IrDA	1m	16Mbps	-	Infrared
MICS	2m	500kbps	25uW	402-405MHz
Wi-Fi(802.11g)	200m	54Mbps	1W	2.4GHz

the NIST (National Institute of Standards and Technology) standard, called the Advanced Encryption Standard (AES) for encryption is recommended for the data encryption [47].

Communication between Data and Application Planes

Communication from the data plane to the application plane uses generic networks like ethernet or cellular network using GSM, UMTS, CDMA and WiMAX. Generally, the application plane organizes its own methodology for displaying, visualizing, and analyzing the data on a generic platform such as laptops, tablets, or smartphones. Thus, the main point of the communications between the data plane and the application plane is data exchange.

There are two different types of data exchanging methods at the application plane: closed data exchanging and open data exchanging. A stand-alone application usually exchanges the data with the data plane through socket network programming. Thus, it doesn't have higher compatibilities with other applications or platforms, however, it has an advantage in security. In contrast, a web-based application has a great advantage in terms of compatibility. The application works at heterogeneous platforms, by using standardized data exchanging method such as XML over SOAP/HTTP. But, it is more vulnerable than the stand-alone application.

At the communication between the data plane and the application plane, the most important thing is security. Commonly, the data plane provides APIs for sharing and retrieving personal medical information, then application will retrieve the data through the APIs. The data plane should give permission to only the authorized/authenticated requests, and the communication channel should also be secured.

2.3 Frameworks

2.3.1 Rimware

Rimware[29] is middleware for IoT (Internet of Things)-Cloud integration. IoT devices are utilized as sensor nodes or actuators in certain applications. For more efficiency of the IoT devices, they are allowed to participate in multiple application domains by implementing multiple profiles. Profiles are application-specific protocols supported by many personal-area network protocols such as Bluetooth 4.0 Low Energy (BLE) technology. The use of profiles enable nodes to standardize on the meaning of the communication. Rimware plays a role as a middleware layer to enable profile-based IoT nodes to realize the full potential of inter-application participation. Especially, the IoT-Cloud environment needs middleware for several reasons. First, it is necessary to support secure, authenticated, and access-controlled connections. Second, the cloud side needs to know the capability of each node to provide an appropriate application to the device. Rimware performs actions to provide necessary features for both the device side and the cloud side.

Fig. 2.4 depicts how Rimware works among nodes, gateway, smartphones, and application specific domains. The core components of Rimware are adapters running on the gateway or apps running on the smartphones, knowledge base running in the cloud, and access controller running in the cloud. The adapters act as the interfacing process between a device and a cloud. The knowledge base in the cloud plays a role to map between BLE profiles and actions in the cloud. The access controller in the cloud enforces access control according

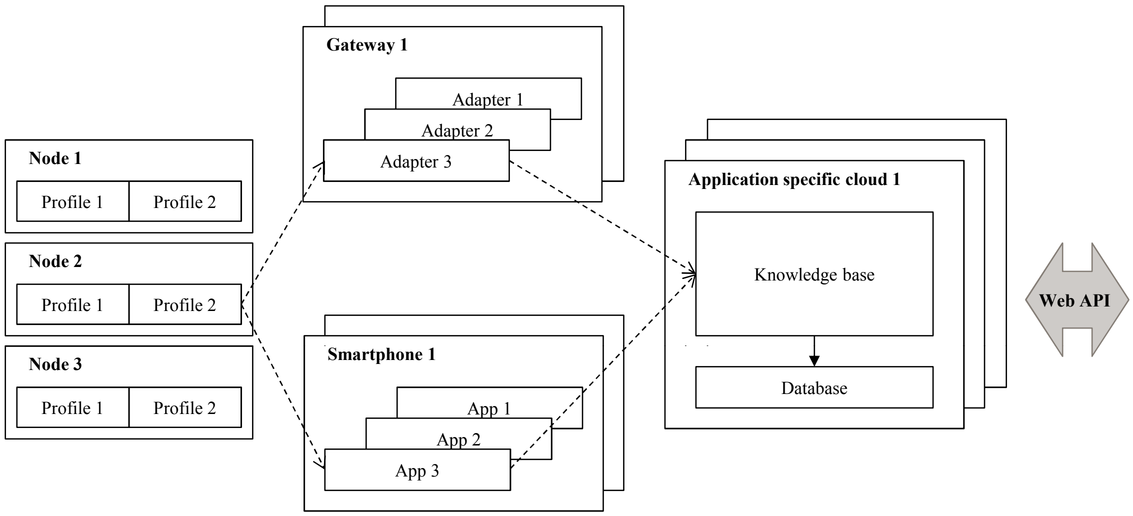

Figure 2.4: Overview of rimware

to the profile mapping by knowledge base. Rimware provides secure and authenticated connection establishment between a node and a cloud. Also, it allows a node to participate multiple application domains.

2.3.2 SeeMon

SeeMon (a Scalable and energy-efficient context Monitoring framework for personal sensor network) [31] is a framework for providing proactive services to mobile users by utilizing context-based sensor networks. Under sensor-enriched and resource-limited mobile environments, SeeMon can continuously monitor the changes of contexts on personal sensor networks.

Fig. 2.5 shows the architecture of SeeMon framework. Its four components inside are CMQ processor, Application broker, Sensor broker, and Sensor manager. CMQ is Context Monitoring Query, which is an intuitive monitoring query language that supports rich semantics for monitoring a wide range of contexts. CMQ processor stores registered queries and performs scalable context monitoring by evaluating numerous CMQs over data delivered by

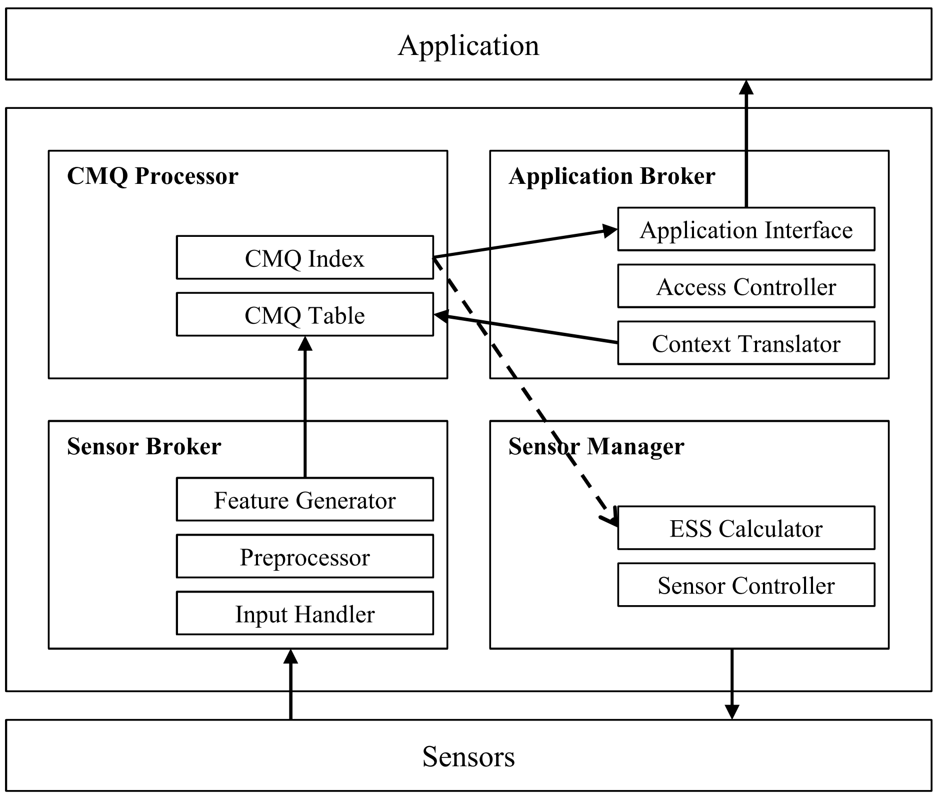

Figure 2.5: Architecture of SeeMon framework

the Sensor broker. Application broker provides interfaces between SeeMon and the application. Through the access controller, which is the inner component in Application broker, it manages privacy and security parameters in SeeMon. Sensor broker plays a role to deliver information from sensor nodes to CMQ processor. The Sensor broker is composed of input handler, preprocessor, and feature generator. The input handler collects information from the sensor node and the preprocessor removes noise and errors from the input data. The feature generator performs computations for generating feature data, which is used on CMQ processor for evaluating contexts. Finally, the Sensor manager provides a function to control the sensor based on the evaluation of CMQ processor. This feedback component is one of the features SeeMon has.

In sum, the goal of SeeMon is to provide an energy-efficient framework for context monitoring. It focuses on the continuous detection of the context changes. Moreover, it achieves a high degree of efficiency in computation and energy consumption by applying the bidirectional approach.

2.3.3 SPINE

SPINE (Signal Processing in Node Environment) [7, 22, 18] is an open-source software framework for designing and prototyping of WBSN (Wireless Body Sensor Network) applications. SPINE consists of efficient implementations of signal processing algorithms for analysis and classification of data from sensor nodes and application-level communication protocol. To support above features, SPINE provides libraries of protocols, utilities and processing functions, and a lightweight Java API to manage the sensor nodes or issue service requests.

Fig. 2.9 depicts the functional architecture of SPINE framework. It is composed of two software components: SPINE node and SPINE coordinator. SPINE node is designed in TinyOS environment and written in nesC language. It includes several utilities for signal

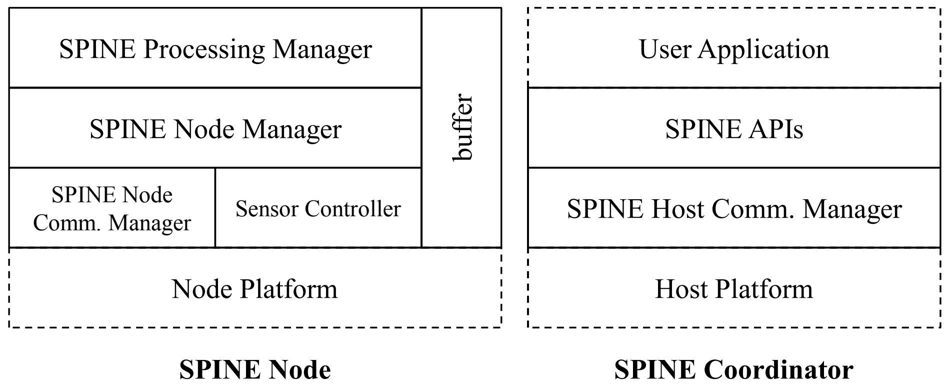

Figure 2.6: Functional architecture of SPINE framework

processing: data storage buffer, math functions, and common feature extractors. The sensor controller in SPINE node manages sensor data sampling and buffering on the sensor nodes. SPINE node ccommunicaton manager handles the communication between the coordinator and the sensor nodes through SPINE protocols.

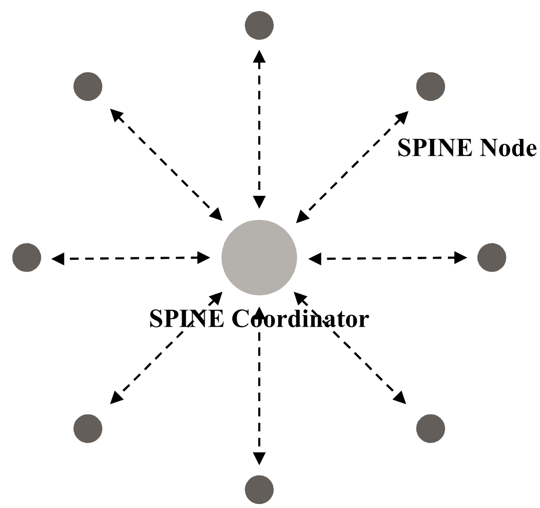

Figure 2.7: Topology of SPINE network

At SPINE coordinator side, there are two main components: SPINE APIs and SPINE host communication manager. SPINE host communication manager acts as a counterpart of SPINE node communication manager. To handle the data from SPINE-based network, the SPINE Coordinator provides APIs for user applications, defined as SPINE API.

SPINE-based network is organized as Fig. 2.7. SPINE coordinator acts as gateway to connect the BSN with wide area networks for remote data access. Since SPINE-based network is star topology, SPINE nodes can only be associated with a single SPINE coordinator. It has an advantage to implement and manage easily, but simultaneously, a restriction to extend.

2.3.4 Codeblue

CodeBlue is a hardware and software platform for medical sensor networks [39, 55]. It is designed for emergency situations in hospital. Thus, the main goal of this platform is to enhance first responders' ability to assess the patient's status by providing the information from sensors for catching vital signs.

CodeBlue is designed to provide routing, naming, discovery, and security for wireless medical sensors. For monitoring purposes, various devices such as PDAs, PCs, and other devices can be utilized as monitoring devices. CodeBlue is based on a publish/subscribe routing framework for data delivery, allowing sensing nodes to publish streams of signals, locations, and identities to which PDAs or PCs accessed by doctors can subscribe. By using ad hoc network concepts, the action of publish/subscribe occurs between any two devices. This *mesh* network topology is efficient in limited range such as in a hospital. To extend the coverage, some fixed nodes can be utilized to detect disappeared nodes and re-route the data to detected nodes.

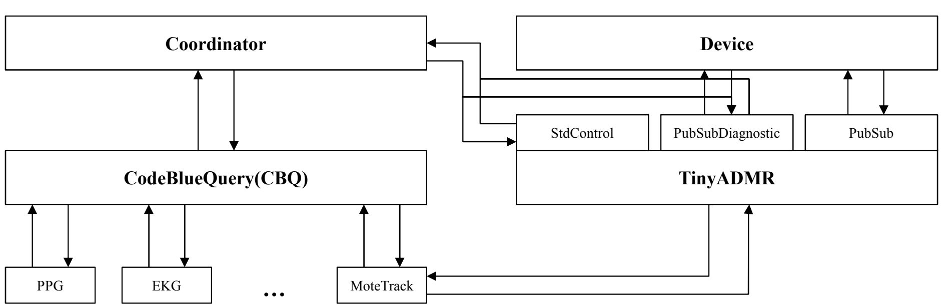

Figure 2.8: Architecture of CodeBlue framework

The architecture of CodeBlue framework is introduced in Fig. 2.8. The main components of the CodeBlue framework are a coordinator, CodeBlueQuery component, and an ADMR protocol layer. The coordinator plays a role as a command center. Usually, it receives messages, translates them and forwards commands to CodeBlueQuery. CodeBlueQuery collects the signals from the sensors and transmits them to the coordinator. Another core component is ADMR protocol. To implement publish/subscribe routing protocol, CodeBlue employs ADMR (Adaptive Demand-Driven Multicast Routing) protocol. TiyADMR component provides a PubSub interface. Through this interface, the sensor nodes and monitoring device can join in a particular channel they wish to be associated with.

CodeBlue framework shows that the medical care system for emergency can be implemented efficiently with low-power sensor nodes. By using publish/subscribe routing protocols, physicians can monitor multiple patients at the same time as long as they subscribe a particular channel, which includes their patients' sensor nodes. However, CodeBlue framework has several restrictions. Even though it supports multi-hop routing with several fixed nodes, It could not show efficiency, if the coverage is too wide. Further, CodeBlue framework doesn't handle any security or privacy problem. As medical information is sensitive to be exposed, privacy and security problem should be seriously considered.

2.3.5 REDCap

REDCap (Research Electronic Data Capture) [26] is a software solution to support clinical and transitional research. The main goal of REDCap is to provide intuitive and reusable tools for collecting, storing, and disseminating project-specific research data. Nowadays, REDCap is a standard to collect and share the research-related electronic data.

To use REDCap system, some infrastructural components such as a web server, a database server, a SMTP email server, and a file server, should be set up. Since REDCap system is a web-based application, it is easy to access the data through a web-based user interface. However, the security problem should be seriously considered. To keep the electronic data secure, it is recommended to set up the database server, the file server, and sometimes even web-server behind firewall. REDCap provides user authentication and role-based security to keep the data secure. Through the user privilege control, permission to access the data is given to a user, since many different types of the project-based data are handled in REDCap system.

A data export function is one of the important features in REDCap system. It also allows users to export the data as various format, such as PDF, CSV, JSON, or XML, if the users has permission to export them. Further, REDCap also provides APIs to import, export, and share the data programatically. REDCap API implements the use of tokens as a means of authenticating and validating all API request to exchange the information securely.

In sum, REDCap project provides the environment for researcher to collect, store, and share the electronic data. Even though the concept is simple, it is very helpful and efficient to share and reuse a wide and variety of research-related data.

2.3.6 Mercury

Mercury is a wearable sensor network platform for clinical neuromotor disorders such as Parkinson's Disease, epilepsy, or stroke [33]. As seen in Fig. 2.9, Mercury is composed of two main components: sensor nodes, and a base station. Sensor nodes capture and store the captured data, compute features, and respond to requests from the base station to download the data and change sampling and storage modes. The sensor nodes of Mecury is designed using the Pixie operating system [34], which supports a resource-aware programming model. Mercury can track energy and bandwidth in real-time by exploiting the abilities of Pixie OS. To control power consumption, Mecury provides an activity filter module in the sensor node.

The activity filter plays a role as an actuator of the sensing modules. If the accelerometer does not have enough meaningful movements, the activity filter drops the data from the accelerometer and disables other sensors. The sensor node has a control interface to operate the requests from the base station. This control interface can change the sampling rate of each sensor and store/transfer mode. Another component of the sensor node is a feature extractor. The feature extraction can be performed in the sensor node to save the bandwidth and energy during transmitting the data, since it is more efficient than transmitting the raw data. To communicate with the base station, Mercury also defines the reliable transfer protocol.

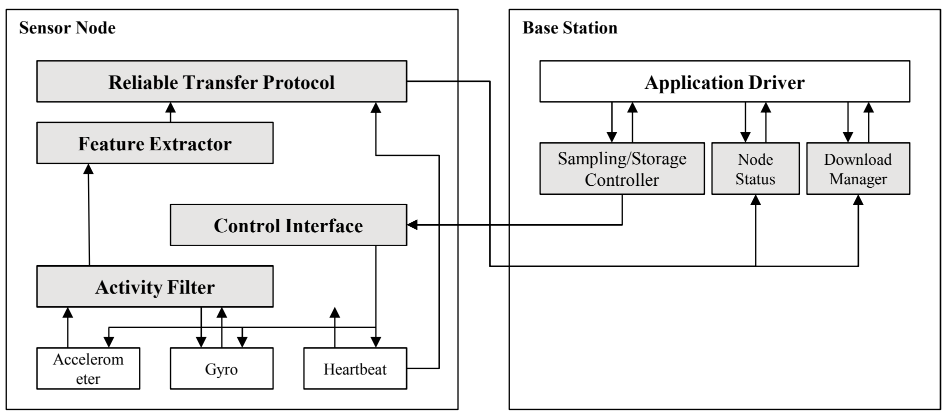

Figure 2.9: Architecture of Mercury

The core component of the base station is an application driver. The application driver runs on the base station and coordinates the sensor network's operation to manage data sampling, storage, and acquisition. However, this application driver is an application-dependent component, which means that the core functions should be determined by the application. Mercury provides only narrow APIs to handle the information from the sensors. More wide range of application-specific policies can be implemented on top of this narrow API according to target clinical requirements.

2.3.7 MedMon

MedMon (Medical security monitoring) is a framework for securing medical devices based on wireless channel monitoring and anomaly detection [63]. Wireless sensor network (WSN)-based medical devices can be hacked by external attackers. Since medical sensor, actuator, or controller handles sensitive data, it could be a severe threat to be exposed. MedMon protects the BAN (Body Area Network) for a personal health system by monitoring the wireless transactions and detecting abnormal behaviors.

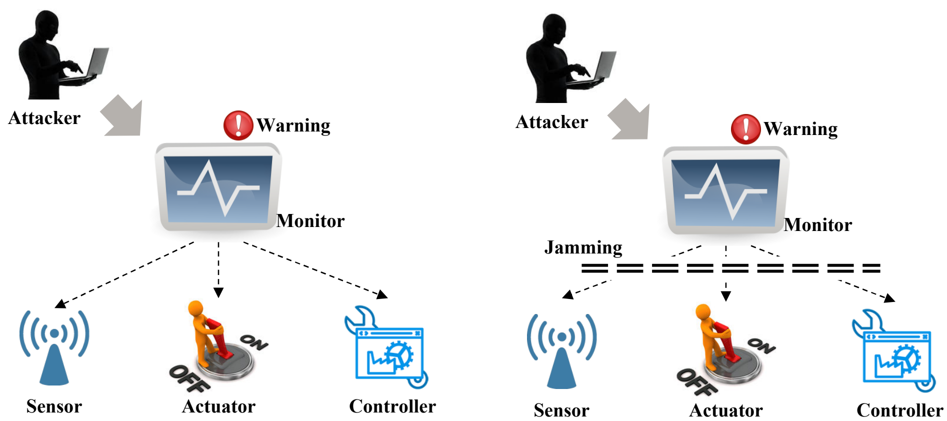

Figure 2.10: Operation of MedMon (left) Passive mode (right) Active mode

Fig. 2.10 shows the operation of MedMon. There are two modes: passive and active. If a potential attack is is detected at the monitoring component, the monitor can respond passively or actively. In passive mode, it provides a warning to the patient through an alarm or a vibration. If the detected attack may be dangerous, the monitor gives a warning and interferes with the transmission by sending jamming signals.

The main challenge of MedMon is how to determine if the transaction is normal or abnormal. MedMon classifies potential anomalies into two categories: physical and behavioral. The physical anomalies can be filtered out by security policies. To build up security policies, the monitor should collect physical characteristics of the transactions such as RSSI (received signal strength indicator), TOA (time of arrival), DTOA (differential time of arrival), and

AOA (angle of arrival). By setting up the normal range of each measurement, the monitor can determine if the transaction is a potential attack or not. The behavioral anomaly detection is based on historical information of the transactions. The monitor keeps records of the historical data and commands. When a new transaction is executed, it compares the transaction to the historical records to decide the abnormality.

The key benefit of MedMon is that it is applicable to existing medical devices without changes of hardware or software. This security framework is helpful to keep the integrity of the BAN. However, it does not cover the confidentiality of communication channels and availability during the duration of the jamming signal. Further, more comprehensive problem is strictness of the security policies. If the security policies are too strict, which means that the normal range of the measurements is too narrow, false-positive rate will increase. On the contrary, if the security policy is too loose, which means that the normal range of the measurements is too wide, the physical anomaly detection does not work well, as it increases the false-negative rate.

2.4 Case study: End-to-end system for ECG monitoring and analysis

In this section, we introduce a case study, which is a system for a medical monitoring system. The system is designed as an end-to-end system for ECG monitoring and analysis, which is based on our plane knowledge of the framework. The main purpose of the system is recording, storing, sharing, analyzing, and displaying the ECG signal with various types of device platforms. The architecture of our end-to-end system for ECG monitoring and analysis is shown in Fig. 2.11. As we introduce each functional plane concept in 2.2, the system is also composed of three planes.

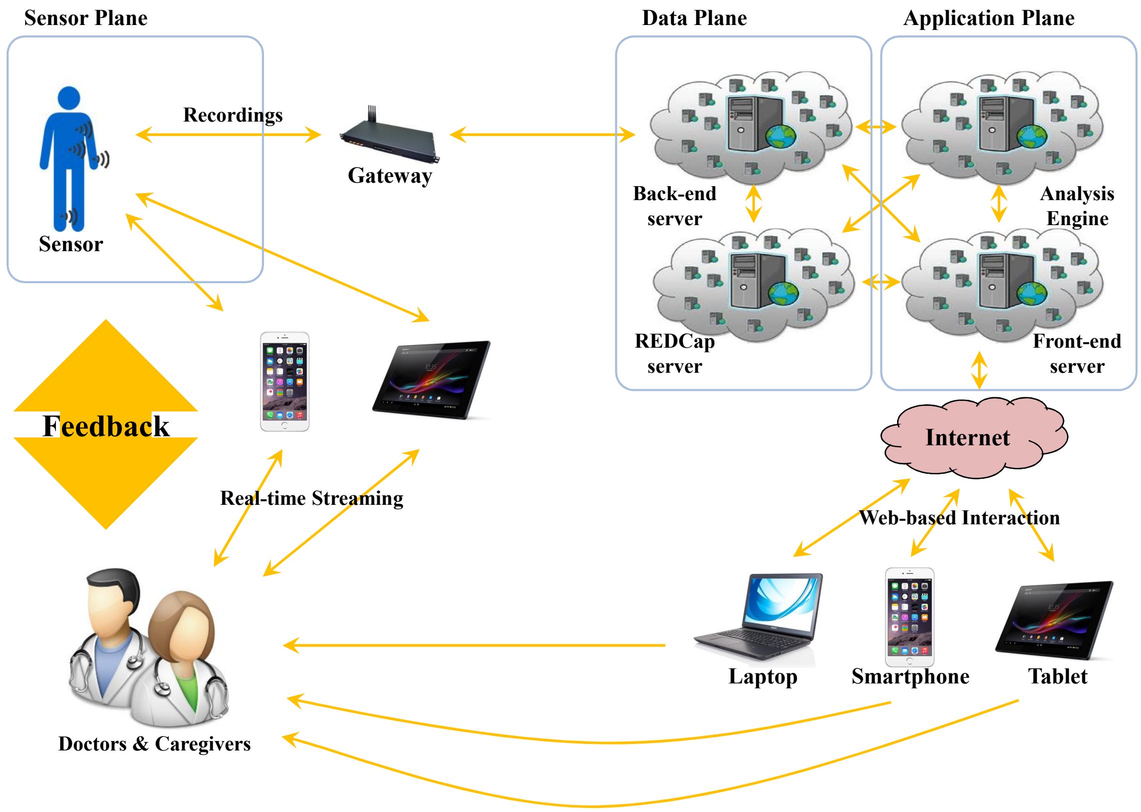

Figure 2.11: Architecture of ECG monitoring system

The wearable sensor device for ECG sensing can record the signal as raw format in the internal flash memory. It can communicate with a gateway using BLE (Bluetooth Low Energy Technology). Using an external BLE support monitoring device such as smart phone, laptop, or tablet, it can monitor the ECG signals in real time. The gateway plays a role as the proxy between the sensor plane and the data plane. It collects raw data and converts it into the formatted data such as DICOM. Then, it transmits the signal to the data plane.

There are several servers in the data plane and the application plane of our system. In the data plane, it is composed of a back-end-server and REDCap system. We employ REDCap system (Section 2.3.5) for storing, managing, and sharing the ECG recordings. The back-end server keeps the list of recordings, which are handled at the data plane, transmits, and updates newly coming data and its information. It also provides the information to the application server and the front-end server through its own APIs. To keep the data transmission secure, the token exchange based authorization is operated between the back-end server and REDCap system.

The application plane comprises the application engine and the front-end server. The function of the application engine is dependent on the application or the request of physicians of caregivers, which can be potential users. To analyze the data, it should collect the information from the back-end server or REDCap system through the APIs. The analysis results are transmitted to the front-end server. The front-end server plays a role of a contact point with users. It collects the information the users want to see, displays the raw data or the analysis results. It provides GUI for interacting with users or getting feedback from users. We implement the front-end server for a web-based application, thus, the users can access the information through browsers on various types of platforms such as laptops, smartphones, or tablets.

Our end-to-end system is still being improved. Especially, security problem should be improved. We should employ a secure channel establishment for exchanging protected health information. Basically, our system requires token-based authorization of users who want to access the information. To enhance the authorization for users, we need to employ the token timeout and re-issue the token to users.

2.5 Summary

In this chapter, we describe a framework for development of wearable medical devices and systems based on our proposed plane concepts. Our framework can encompass the state-of-the-art in the field of wearable medical devices and applications and give many considerations into the sensor plane, the data plane, and the application plane. Moreover, we introduce the state-of-the art of existing frameworks for wearable medical applications. Furthermore, we provide a use case for our framework in the form of an end-to-end system for ECG monitoring and analysis.

Chapter 3

ECG Signal Reconstruction from Undersampled Measurement

3.1 Motivation

Electrocardiogram (ECG) is one of the most significant types of signals for cardiac analyses and diagnostics. However, ECG signals sometimes yield false-positive or false-negative cases in the analyses of short-time recordings. To overcome these limitations, ECG signals should be recorded for a sizable time. Due to the nature of ECG signal, more storage and bandwidth are required to handle the signal. Furthermore, as the wearable and mobile types of health monitoring devices are introduced, the amount of signal data has seen explosive growth while pressures on energy efficiency increase.

The traditional approach to compressing the data is generally composed of two stages. The whole data is obtained in the first stage, and then a compression algorithm to reduce redundancies in the data is applied. In terms of compression performance, this approach can be efficient in analyzing the data to reduce redundancies in the signal. However, it is not good in terms of storing data to be obtained and power consumption of the sensing device, since the entire data must be obtained before processing.

The Compressive Sensing (CS) framework[11, 16, 12, 10] shows the underlying signal can be reconstructed precisely from fewer samples so long as the signal is sparse or compressible in a certain domain. According to Shannons sampling theorem, the signal should be sampled at least twice the maximum bandwidth to be reconstructed perfectly. However, CS framework shows that the signal can still be reconstructed with fewer samples than conventional sampling theory requires if the signal is sparse or compressible.

In this chapter, we proposed a new approach to reconstructing ECG signals with compressive sensing. Our proposed algorithm uses a trained overcomplete dictionary to be constructed by K-SVD[5]. A comparison is presented with conventional compressive sensing signal reconstruction method using MIT-BIH arrhythmia database[21, 2].

3.2 Background

3.2.1 Sparse approximation

Sparse approximation is the representation that accounts for most of signal information with a linear combination of a small number of elementary signals. Mathematically speaking, if the N-dimensional signal y is a sparse signal, which has k non-zero coefficients in the transformed domain, then the signal can be represented as

$$y = \mathbf{\Phi}\boldsymbol{\alpha} = \sum_{i=1}^{k} \alpha_i \phi_i \tag{3.1}$$

where Φ is called a dictionary, which is the set of elementary signals. Algorithms that find sparse approximation include matching pursuit (MP), orthogonal matching pursuit (OMP)

[61], and least absolute shrinkage and selection operator (LASSO) [59].

3.2.2 Dictionary learning

The approximation performance in terms of the approximation quality and the sparsity of the sparse vector a depends on the dictionary as well as signal itself. If the dictionary contains well-estimated atoms or basis vectors, then the sparse vector will have a small number of non-zero elements. Universally used dictionaries, such as those for discrete cosine transform, Fourier transform, and Wavelet transform, are orthogonal dictionaries. They have mathematical simplicity and few redundancies to represent dictionaries themselves, but they are not suitable to represent signals with few redundancies. Due to the limitations of the orthogonal dictionaries, researchers proposed overcomplete dictionaries by data-driven learning. K-SVD [5] is a popular algorithms for constructing overcomplete dictionaries by learning. Given signals $\boldsymbol{Y} = [\boldsymbol{y_1 y_2} \ldots \boldsymbol{y_N}] \in \mathbb{R}^{n \times N}$, the goal of K-SVD is to find the optimal dictionary $\boldsymbol{\Phi} \in \mathbb{R}^{n \times K}$, where $\boldsymbol{\Phi}$ is an overcomplete dictionary $(K > n)$ and α is a sparse code of given input signals Y, keeping the sparsity constraints; $\|\boldsymbol{\alpha}_i\|_0 \leq \boldsymbol{T}_0, \forall i$.

$$\min_{\Phi,a}\{\|\boldsymbol{y} - \boldsymbol{\Phi\alpha}\|_F^2\}, \text{ subject to } \|\boldsymbol{a}_i\|_0 \leq \boldsymbol{T}_0, \forall i \tag{3.2}$$

3.3 Proposed algorithm

Our proposed algorithm is a new approach to reconstructing a signal from compressive sensing system. Our proposed algorithm has a preprocessing stage for constructing a dictionary, which can make the signal sparse or compressible in the transformed domain. Fig. 3.1 shows the full procedure of the proposed algorithm.

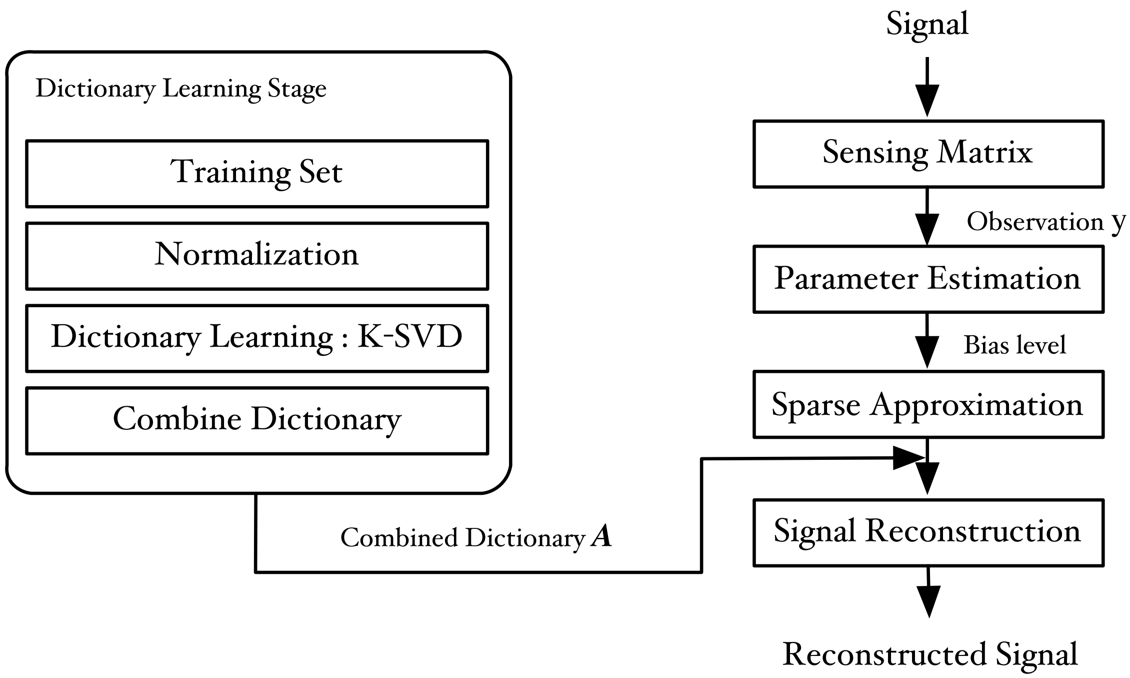

Figure 3.1: Procedure of the proposed algorithm

3.3.1 Dictionary learning stage

Fig. 3.1 shows the flow of our dictionary construction procedure for the overcomplete dictionary from the given dataset. For constructing a set of training data, our algorithm performs normalization. To remove the effects of the bias term, we also make the training data zero-mean. After normalization and removal of the bias term, a set of normalized signal vectors can be presented to $Y = [y_1, y_2, , y_k] \in \mathbb{R}^{n\times k}$ as a data set for training. Using the training dataset Y, the optimized dictionary to represent the data under the sparsity constraint is obtained by K-SVD dictionary learning. The trained dictionary performs well when representing signals with a few coefficients, but not signals with generality due to reflecting the characteristics of the signal used at the training stage. To compensate for this shortcoming, our proposed method uses the combined dictionary, which is made up of general basis matrices, such as DCT or wavelet basis matrix, as atoms of the dictionary.

3.3.2 Reconstruction stage

Our proposed algorithm assumes that the signal would be obtained by random sampling, which means that the signal is selectively digitized in a time window. The sensing mechanism can be represented as a sensing matrix, $\Phi \in \mathbb{R}^{m \times n}(m < n)$, which can select random samples. The observation can be presented using the sensing matrix Φ. The key point of our proposed algorithm is the signal model. We assume that the original signal can be presented as

$$x = A\alpha + c \tag{3.3}$$

The signal is composed of two parts: the signal shape and the bias term, which defines vector c. Our target ECG signal has a changeable baseline, so it is needs to separate the bias term to estimate the shape part more accurately. We can take the sensing matrix and our proposed signal model to derive the equation of observation y, thereby obtaining the following equation:

$$y = \Phi(A\alpha + c) = \Phi A\alpha + \Phi c \tag{3.4}$$

The observation y also has two parts: the shape (i.e., the zero-mean signal part) and the bias term. Since the sensing matrix is a simply random selection matrix, we can easily estimate the bias term by taking the average of the observation y for the parameter estimation step in Fig. 3.1. The bias term, called vector c, is

$$c = c \cdot \mathbf{1} \simeq E(y) \cdot \mathbf{1} \tag{3.5}$$

where **1** means a vector whose elements are all ones. After estimating the parameter and subtracting it from observation, the simply zero-mean signal part remains.

$$y' = y - E(y) \cdot \mathbf{1} = \Phi A \alpha \tag{3.6}$$

The zero-mean observation part can be a simple linear equation with ill-posed conditions. In this chapter, LASSO (Least Absolute Shrinkage and Selection Operator) is used as a solver for the convex problem. We also use MATLAB toolbox provided by sparselab[3] as a solver for l1 norm minimization problem.

3.4 Experimental results

For comparing the proposed algorithm with a conventional Compressive Sensing-based signal reconstruction algorithm, we have chosen the MIT-BIH arrhythmia database as the test signals. It provides 48 sets of 2-lead ECG signals at 360 11-bit samples/sec. Our metrics for evaluating signal reconstruction are percentage root-mean-squared distortion (PRD) and compression ratio (CR). PRD [30] shows the reconstruction error as a percentage and is defined as

$$PRD = \sqrt{\frac{\sum_{i=1}^{n}(x_i - \hat{x}_i)^2}{\sum_{i=1}^{n} {x_i}^2}} \times 100\% \tag{3.7}$$

where n is the number of samples, and x_i and $\hat{x}_i$ are respectively the original data and the reconstructed data from the proposed algorithm. CR is defined as

$$CR = \frac{b_{orig} - b_{comp}}{b_{orig}} \times 100 \tag{3.8}$$

Table 3.1: Data set for training and testing from MIT-BIH Arrhythmia Database

Records for Training	Records for Testing
100, 101, 102, 103, 104, 105, 106, 107, 108, 109, 111, 112, 113, 114, 115, 116, 118, 119, 121, 122, 123, 124, 200, 201	117, 202, 203, 205, 207, 208, 209, 210, 212, 213, 214, 215, 217, 219, 220, 221, 222, 223, 228, 230, 231, 232, 233, 234

where b_{orig} and b_{comp} represent the number of bits required for the original and compressed signals, respectively.

3.4.1 Dictionary construction

We have chosen 24 out of 48 signals from the MIT-BIH arrhythmia database for training our algorithm and constructing the dictionary. Another 24 signals except for the signals used in training are utilized to evaluate the performance of our proposed algorithm.Table 3.1 depicts the signals included in training dataset. User-specific parameters on the procedures of dictionary learning include sparsity, the size of dictionary, and the size of minimum signal block. The proposed algorithm uses $\mathbf{T_0} = 10$ as sparsity constraint for dictionary construction. It also uses 64 samples as a minimum signal block, and 5012 atoms as a dictionary size, which means that the trained dictionary is a 64×5012 matrix. After obtaining the trained dictionary, we combine it with 64×64 DCT basis and wavelet basis, and the final size of the combined dictionary is 64×5140.

3.4.2 Comparison of the reconstruction results

On Fig. 3.3, (a) and (b) show the results using conventional DCT and wavelet bases (daubechies4) with half of the randomly selected samples from original signal, i.e., $CR = 50\%$. Our method using the combined dictionary achieves $PRD = 1.34\%$ when $CR = 50\%$. Fig. 3.4 depicts the performance over the MIT-BIH arrhythmia database. It shows high CR leads to high

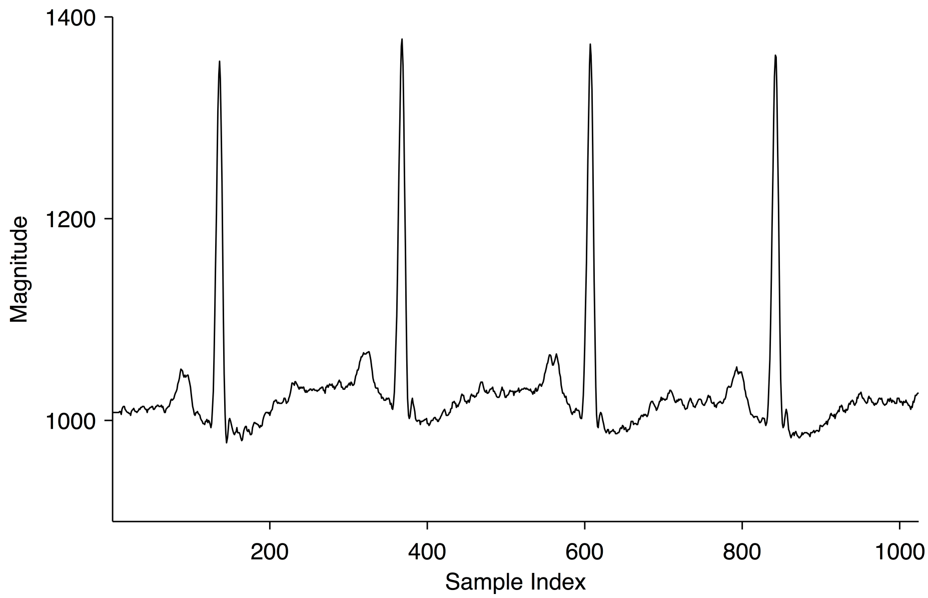

Figure 3.2: Original signal for comparison

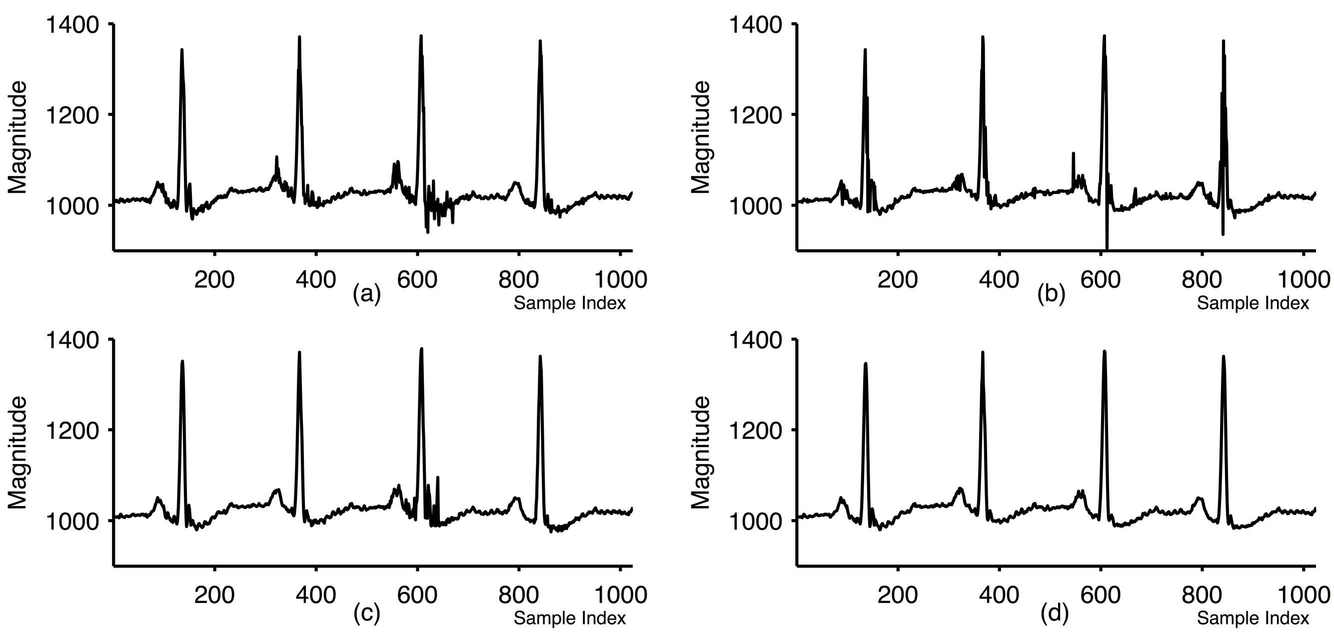

Figure 3.3: Reconstruction results) (a) DCT basis (b) Wavelet basis (Daubechies4) (c) Trained Dictionary by K-SVD (d) Combined Dictionary

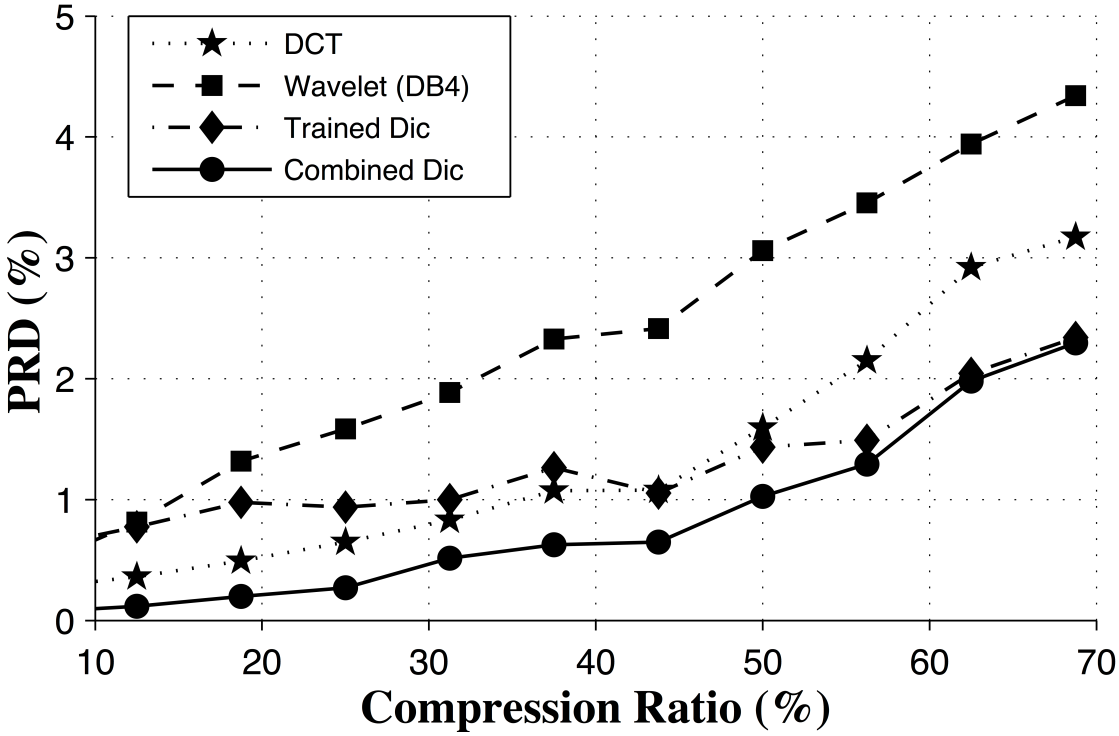

Figure 3.4: Comparison of Compression Performance between Conventional Methods

distortion. According to the dictionary or basis used into the reconstruction method, the performance is shown differently. Our proposed method shows less distortion than the other conventional methods do.

3.5 Summary

In this chapter, we propose reconstructing ECG signals from undersampled data by compressive sensing. The signal reconstruction from undersampled data is performed by finding a sparse solution using a given dictionary, and the overall performance depends on the dictionary utilized in the approximation process. Dictionaries constructed by our method achieve good approximation performance by combining the trained dictionary by K-SVD with universal types of dictionaries such as DCT or wavelet transform. Our algorithm shows good performance with low distortion in terms of the CR and PRD over MIT-BIH arrhythmia database. Since the algorithm is learning-based method, the performance can be improved if the training dataset is organized well.

Chapter 4

Lossy Compression for ECG Signal

4.1 Motivation

Electrocardiogram (ECG) data often require compression for storage, transfer over networks, and handling, especially given the trend towards miniaturization and wearable health-monitoring systems. ECG signal compression algorithm introduced in the past several decades can be lossless [62, 9], but lossy ones achieve better compression ratios by allowing small distortions in the parts of data that are not critical to the analysis of the signals. Lossy algorithms can be further categorized into direct signal compression and transformation compression. The former attempts to obtain redundancies on the signals to be compressed and extract meaningful parameters after analysis; the latter analyzes the signals in the transformed domain and retains the coefficients of the particular features in the data [49, 35, 28, 44].

We propose a new approach to ECG signal compression based on heartbeat and dictionary learning. Our algorithm is divided into two stages. The training stage performs length normalization by interpolation to construct the overcomplete dictionary. During the compression/decompression stage, it calculates the optimal coefficients, which can capture most of the information content of the ECG signals. We compare our results with those of several well-known ECG compression algorithms using the MIT-BIH arrhythmia database [21].

This chapter is organized as follows. We already introduce the concept of sparse representation and dictionary learning in Section 3.2, we introduce the background regarding sparse representation and dictionary learning. In Section 4.2, we describe our proposed ECG compression algorithm using overcomplete dictionary constructed by K-SVD. Section 4.3 presents experimental and comparison results using MIT-BIH arrhythmia database.

4.2 Proposed Algorithm

Data compression can be viewed as the process of reducing the redundancies in a signal. One type of redundancies due to the quasi-periodic nature of the ECG signals is shown in the correlation between adjacent beats. To reduce the redundancy, we propose the beat-based data compression algorithm with an overcomplete dictionary. Our algorithm is composed training step and compression/decompression step. In training step, the algorithm constructs a set of training data for the learning dictionary, which will be constructed using K-SVD algorithm.

4.2.1 Dictionary Construction

Fig. 4.1 shows the flow of our dictionary-construction stage for the overcomplete dictionary from the given dataset. Although ECG signals are repetitive, they do not necessarily have a fixed period and morphology, even though they may have regular patterns. An ECG signal between two adjacent peaks of QRS complexes may show several morphological patterns with different lengths. To use this information for compression, our algorithm performs

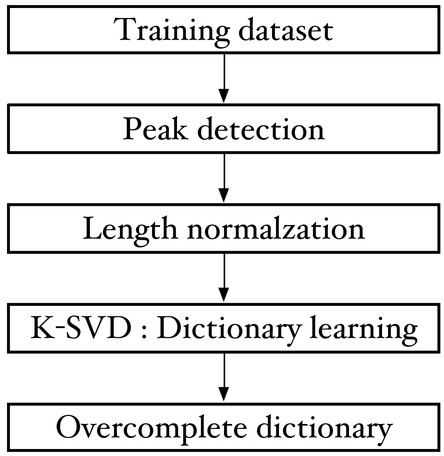

Figure 4.1: Dictionary construction process

peak detection and length normalization.

The proposed algorithm uses the Pan and Tomkins QRS peak detection method[50] to extract the patterns between the peaks of two adjacent QRS complexes. After the detection, length and magnitude normalizations are needed to regularize the peak-to-peak ECG signal for constructing a set of training data. To remove the DC component of the signals, the algorithm makes the extracted pattern zero-mean. Fig. 4.2 shows that each ECG signal has a different length. Fig. 4.3 depicts the result of normalization process.

After normalization, the k^{th} peak-to-peak data $\boldsymbol{d}_k$, whose length is not equal to n, could be represented as an n-dimensional column vector $\boldsymbol{y}_k$. A set of these normalized vectors could be presented to $\boldsymbol{Y} = [\boldsymbol{y}_1 \boldsymbol{y}_2 \dots \boldsymbol{y}_k]$, called the training dataset. Using $\boldsymbol{Y}$, the optimized dictionary to represent the data under the sparsity constraint is obtained by K-SVD dictionary learning.

4.2.2 Compression

Our proposed compression procedure is illustrated in Fig. 4.5. It compresses beat-based data, so the target data should be extracted before compression. During dictionary construction, QRS complexes should be detected to distinguish adjacent beats.Then, $\boldsymbol{X}$, which is a selected

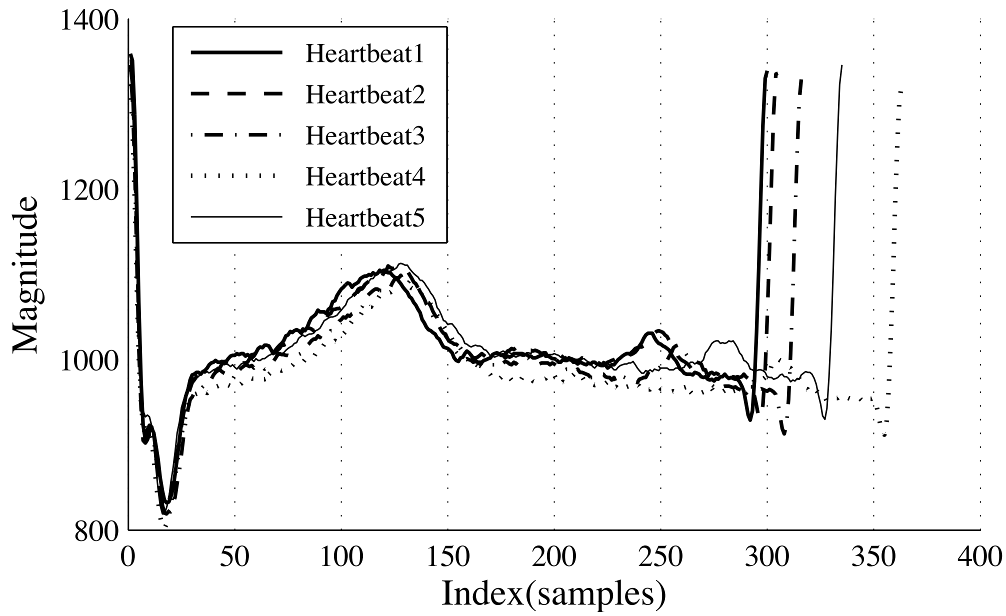

Figure 4.2: Original ECG signal (Record 231 from MIT-BIH Arrhythmia Database)

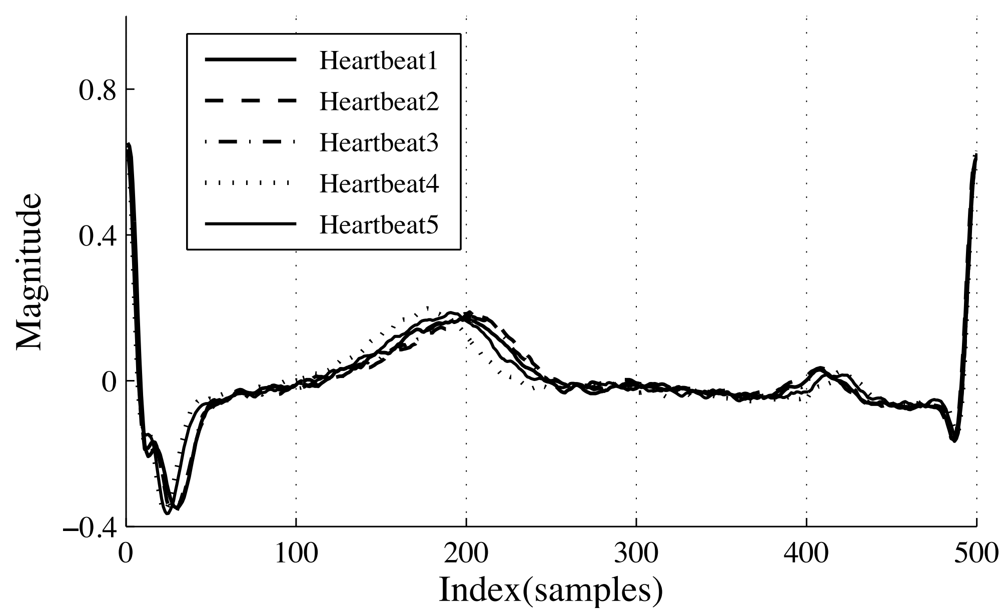

Figure 4.3: After normalization

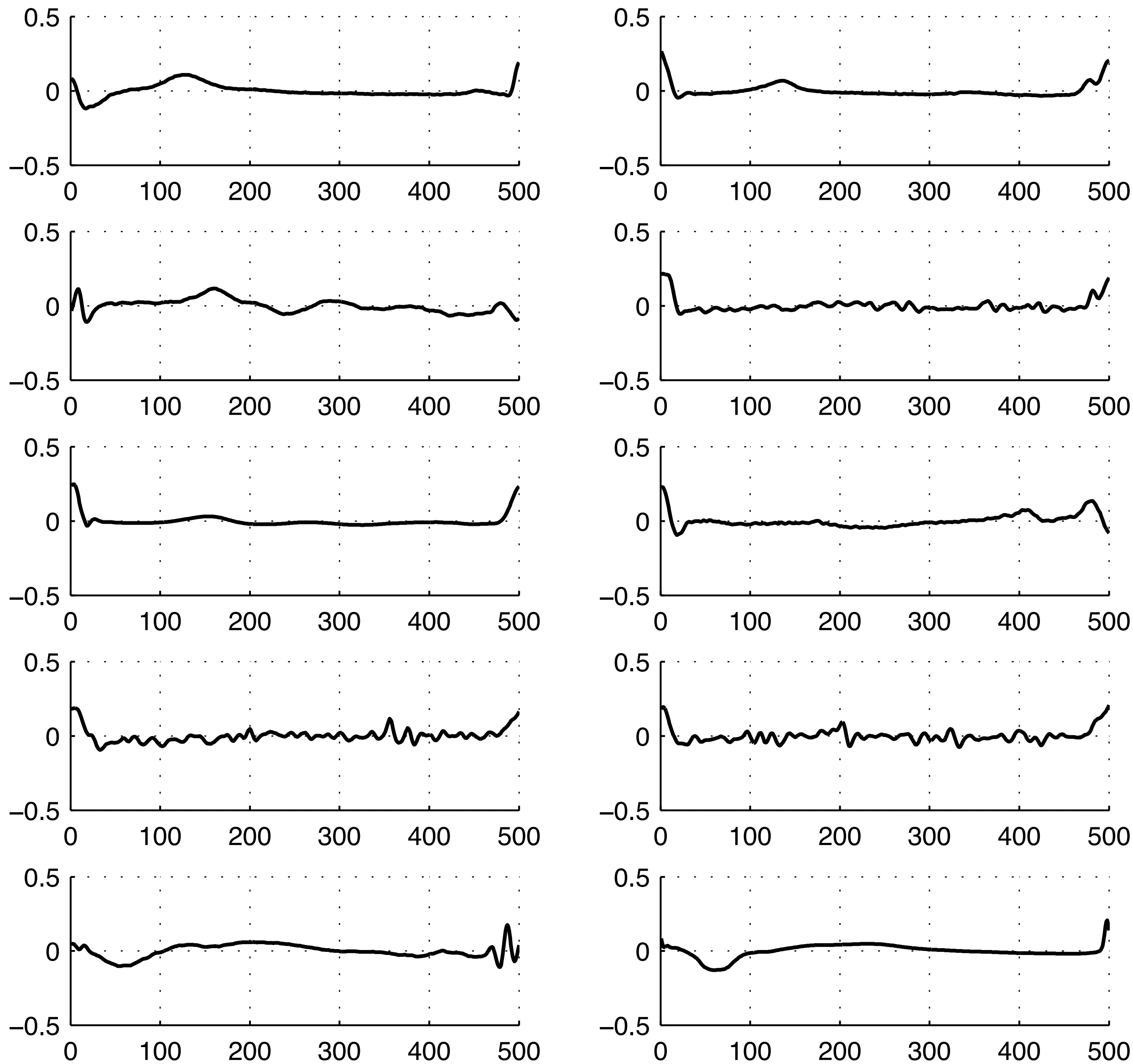

Figure 4.4: Atoms of Learned Dictionary by K-SVD

target beat to be compressed, should be length-normalized. To recover the signal it the decompression step, the length of the beat should be stored.

In the sparse coding stage, the algorithm finds the sparse vector using the given overcomplete dictionary. From Eq. (3.1), the sparse code in $\boldsymbol{\alpha}$ is not a unique vector, since the given dictionary is overcomplete. To obtain the appropriate coefficients, which can represent the signal with a small amount of error, sparsity constraint should be included in the equation.

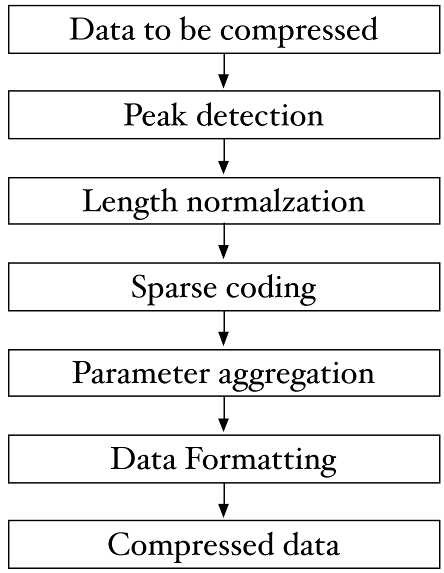

Figure 4.5: Compression process

Sparse solutions with overcomplete dictionary can be found using MP, OMP, and LASSO, as discussed in Section 3.2. We use OMP, a greedy algorithm, for its simplicity and efficiency.

In the parameter aggregation stage, the proposed compression algorithm needs to store information about the heartbeats. The beat-related information to be stored is in terms of the average value of the signal in a beat and the length of the beat. If the dimension of the constructed dictionary is $n \times K$, then the sparse vector $\boldsymbol{\alpha}$ is $K \times 1$. Due to the overcomplete requirements, K is much larger than n. Under the sparsity constraint, most of the elements in sparse vector $\boldsymbol{\alpha}$ are zero. Only a small number of elements have non-zero coefficients. To reconstruct the sparse vector $\boldsymbol{\alpha}$ during decompression, the index of the non-zero elements should be stored.

4.2.3 Decompression

The decompression step is exactly the reverse procedure of compression except for the sparse coding stage. For the decompression, the sparse decoding stage is replaced by the sparse coding. Since the elements of sparse vector means the score of respective atoms of the given dictionary, the decoding merely requires the linear combination of the relevant atoms. After sparse decoding, the algorithm restores the original length of signal using stored length information.

4.3 Experimental results

For comparing the results of the proposed algorithm with those of conventional ECG compression methods, we have chosen the MIT-BIH arrhythmia database [21] as the test signals, which are widely used for evaluating the performance of ECG compression algorithms. The database provides 48 sets of ECG signals, which are all two-lead ECG data. Each lead is recorded at 360 samples/sec with 11-bit resolution. To quantify the performance of the proposed algorithm, we employ three widely used metrics: compression ratio (CR), percentage root-mean-squared distortion (PRD), and quality score (QS).

4.3.1 Measurement of Performance

Compression ratio (CR) is defined as

$$CR = \frac{\textit{the number of bits of original signal}}{\textit{the number of bits of compressed signal}} \tag{4.1}$$

Another widely used metric is percentage root-mean-squared distortion (PRD), which shows the reconstruction error as a percentage. It is defined as

$$PRD = \sqrt{\frac{\sum_{i=1}^{n}(x_i - \hat{x}_i)^2}{\sum_{i=1}^{n} {x_i}^2}} \times 100\% \tag{4.2}$$

where n is the number of samples, and x_i and $\hat{x}_i$ are the original data and the reconstructed data from our proposed algorithm, respectively.

The quality score (QS)[17] is the ratio between the CR and PRD:

$$QS = \frac{CR}{PRD} \tag{4.3}$$

High value of QS means good performance on lossy compression. It is useful to choose the best method while taking the compression reconstruction errors into consideration as well.

4.3.2 Parameters for Dictionary Construction

Since our proposed algorithm is learning-based, we should construct data sets for training purpose as explained Section 4.2. We have chosen 24 out of 48 signals from the MIT-BIH arrhythmia database. Of the 48 signals, 24 are utilized by our K-SVD based algorithm to construct the dictionary, while the rest are used to evaluate the performance of our proposed algorithm. Table 4.1 depicts the signals included in training dataset, totaling 51,492 beats per lead.

Parameter selection is a key for good quality of dictionary construction. Several user-specific parameters on the procedures include sparsity, the size of dictionary, and the size of length normalization. The proposed algorithm uses $\boldsymbol{T_0} = 10$ as sparsity constraint for dictionary construction. To satisfy the overcomplete requirement, we set the dictionary size to 1024. Furthermore, the algorithm normalizes the peak-to-peak signals to 500 samples by linear interpolation. The size of the normalized peak-to-peak signals should be larger than the original peak-to-peak signals for preserving the signals after the processing.

The magnitudes of the coefficients are determined by the dictionary. The coefficients are determined by projecting the normalized data on the atoms of the dictionary. Since the magnitude of each atom of the dictionary is unit-length, each element in the atoms dictionary is quite small.

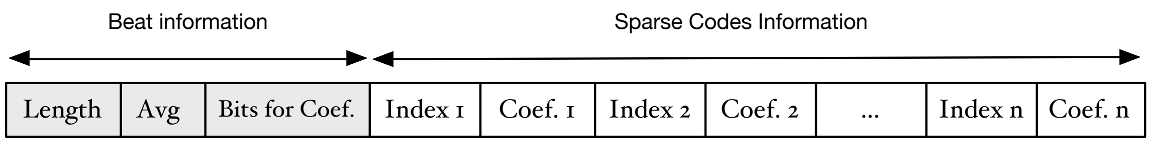

Figure 4.6: Compressed data format

Table 4.1: Data set for training and testing from MIT-BIH Arrhythmia Database

Records for Training	**Records for Testing**
100, 101, 102, 103, 104, 105, 106, 107, 108, 109, 111, 112, 113, 114, 115, 116, 118, 119, 121, 122, 123, 124, 200, 201	117, 202, 203, 205, 207, 208, 209, 210, 212, 213, 214, 215, 217, 219, 220, 221, 222, 223, 228, 230, 231, 232, 233, 234

More bits should be allocated to represent the coefficients; but to avoid over-allocation, we scale up the dictionary. Experimentally, we use 10 as a scaling factor.

4.3.3 Data format

Fig. 4.6 shows our proposed data format after parameter aggregation for representing one heartbeat. In the beat information header, we allocate 11 bits for representing the average peak-to-peak value and 9 bits for the length. Due to the total dictionary size, we allocate 10 bits for each index, which is the location of the atom with respect to the coefficient value. The size of the coefficient field varies between 8-10 bits in these experiments.

Table 4.2: Performance comparison with conventional algorithms

Algorithm	SPIHT[35]	AZTEC[14]	CORTES[4]	Hilton[28]	Djohan[15]	Fira[17]	Ours
PRD	1.18%	28%	7%	2.6%	3.9%	0.61%	0.55%
CR	8:1	10:1	4.8:1	8:1	8:1	12.74:1	13.79:1
QS	9.3	3.57	0.68	3.076	2.05	20.88	24.75

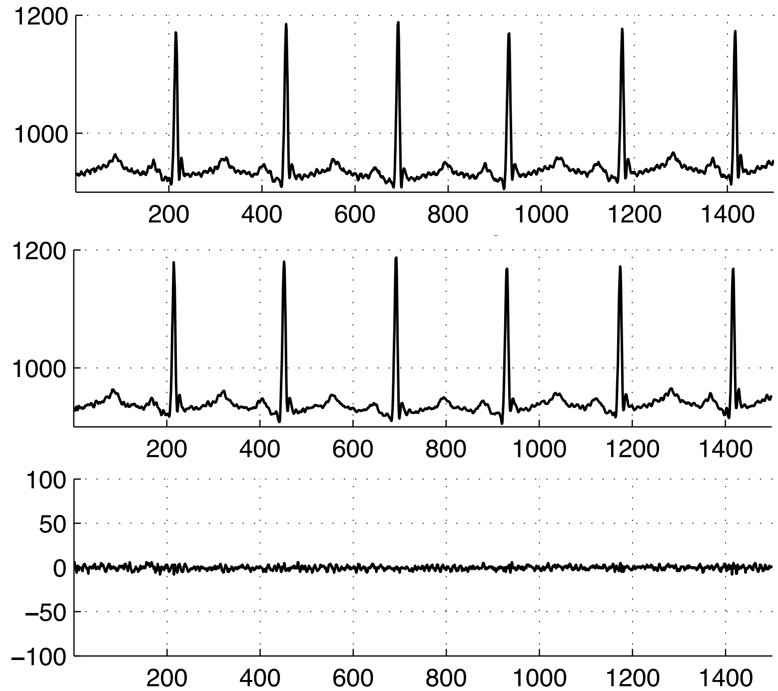

Figure 4.7: Signal reconstruction with the proposed algorithm(Record 205 from MIT-BIH arrhythmia database)

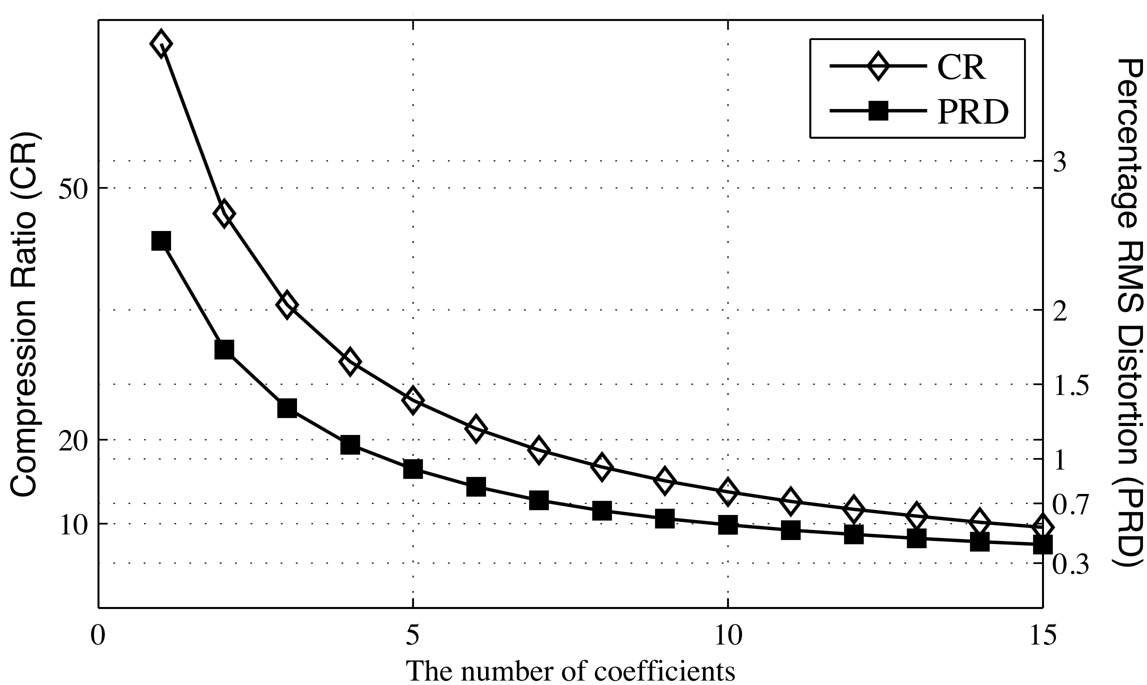

Figure 4.8: Average of CR and PRD over testing dataset(MIT-BIH arrhythmia database)

4.3.4 Evaluation

To recall, our metrics are CR, PRD, and QS. For the purpose of evaluating performance, the first 10 heart beats of the signal are used to the compression. Table 4.1 shows the training dataset for constructing dictionary and the testing dataset for evaluation.

Fig. 4.7 shows that the reconstructed signal contains small deviations from the original signal. The original signal is Record 205 from MIT-BIH arrhythmia database. With Record 205, $CR = 11.23, PRD = 0.30\%, QS = 37.43$ are achieved. The highest compression ratio ($CR = 18.27, PRD = 0.69\%$) is achieved for Record 117. The lowest compression ratio ($CR = 7.99, PRD = 0.75\%$) is achieved for Record 213. The proposed algorithm has high CR in the range of 7.99-18.27 and low PRD in the range of 0.28-1.52(%).

The CR and PRD have close relationship in lossy compression. In general, high CR implies high PRD, and low CR leads to low PRD. Since the proposed algorithm is a beat-based compression algorithm, the CR is determined by the number of coefficients that represent a heart beat. Fig. 4.8 shows the relationship between CR and PRD over entire testing dataset.

Table 4.2 shows the comparison results with the conventional methods, such as SPIHT [35], AZTEC [14], CORTES [4], Hilton [28], Djohan [15] and Fira [17]. Our proposed algorithm, which uses 10 coefficients to represent each heart beat, has better PRD performance than other conventional methods do. Our algorithm has an average PRD of 0.55%, CR of 13.79:1, and QS of 24.75 over the testing dataset.

4.4 Summary

We propose a new two-stage algorithm for ECG signal compression based on dictionary learning from training datasets. The dictionary construction stage uses the K-SVD algorithm with given set of training data, which is zero-mean and length-normalized. The compression/decompression stage first extracts the peaks of QRS complexes by using Pan and Tomkins method and then finds the best-matched sparse codes with the trained dictionary. The algorithm has been tested with MIT-BIH arrhythmia database and compared with existing methods in terms of the compression ratio (CR), percentage of root mean squared distortion (PRD), and quality score (QS). Our algorithm shows good compression performance with low distortion. Since the algorithm is a learning-based method, the performance can improve with the training dataset.

Chapter 5

QT Analysis Algorithm and its implementation

5.1 Motivation

The electrocardiogram (ECG) is one of the most significant types of signals for cardiac analysis and diagnostic purposes. ECG signals consist of a recurrent wave sequence: the P wave, the QRS complex, and the T wave. In ECGs, most of the clinically useful information can be found in intervals, amplitudes, or wave morphology. Of particular interest is the QT interval, which is defined as the time interval from the onset of the QRS complex to the offset of the T wave, and which is known as a good indicator of an increased risk for life-threatening ventricular arrhythmia and sudden cardiac death. The process of measuring the prolongation of the QT interval is called *QT analysis*, which can be a good marker and reference for diagnostic purposes. However, there is no standard for measuring the prolongation of the QT interval on an ECG signal [51, 23], due to subjective measurement.

Conventional recording-based QT analysis methods are generally divided into two stages:

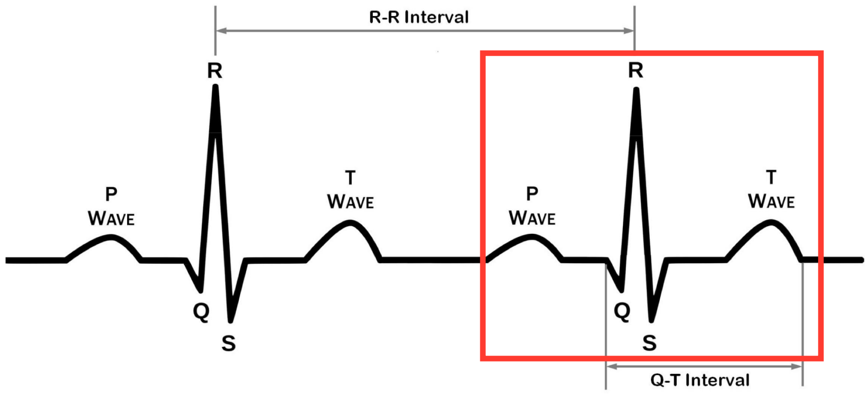

Figure 5.1: Corrected QT Interval

one for recording and transferring, and the other for analysis based on the recorded signal. To automatically measure the prolongation of the QT interval, the algorithm should find the fiducial points, namely the R peaks (for calculating heart rate), the QRS onset, and T offset (for measuring the QT interval). It is relatively easy to find the QRS complex due to its salient characteristics, but it is not so easy to detect the other fiducial points, especially the onset of the QRS complex and the offset of the T wave. To find the fiducial points, recent research employs a temporal search interval before and after the detected QRS location to search for the T waves using adaptive filtering [58], low-pass differentiation [32], or wavelet transform [42].

In this chapter, we propose a QT-analysis algorithm based on beat-by-beat analysis and deploy our proposed algorithm at a server, which can provide entire analysis for doctors and caregivers. This system comprises several components for analyzing, displaying and interfacing. At implementation section, we introduce each components and how they work.

5.2 Proposed algorithm

5.2.1 QT interval

An electrocardiogram (ECG) measures the electrical activity of the heart. It senses electrical impulses generated by the polarization and depolarization of the cardiac tissue and renders it as a waveform. It is an important instrument for cardiac analysis and diagnosis. As Fig. 5.1 shows, an ECG signal is shown as a mixed signal of several different waves. A typical ECG tracing of the cardiac cycle consists of a P wave, a QRS complex, a T wave, and a U wave. The U wave is usually dominated by the P and T waves and is normally invisible.

Our focus is the QT interval, which has been shown as a good marker for life-threatening cardiac risks. It is measured by calculating the time difference between the QRS onset and the end of the T wave. To measure the QT interval, signal segmentation should first be performed, but this is not easy due to the characteristics of the component waves of ECG. Since the QT interval varies with heart rate, it should be normalized for comparing the QT values over time at different heart rates. In general, Bazzett's formula [6] is widely used as a method for calculating *the corrected QT interval*, denoted QT_C:

$$QT_C = \frac{|\mathrm{QT}|}{\sqrt{|\mathrm{RR}|}} \tag{5.1}$$

where |RR| means the length of the R-R interval between the current R peak and the previous R peak, while |QT| is the length from the onset of Q to the end of T, as shown in Fig. 5.1. To monitor the beat-by-beat QT_C, we need to segment the fiducial points such as the QRS onset, the T offset, and the R peak in every heart beat. Our proposed method shows an effective way to segment each fiducial point and calculate QT_C for beat-by-beat analysis for in-device processing.

5.2.2 Making templates

To find fiducial points from a given set of signals, we need prior knowledge about the patterns of interest. These are called templates, which are reference patterns we want to find. Our method suggests making a template with multiple channels. Let $S_i(j)$ denote the j^{th} sample from the i^{th} channel. Template T can be defined as

$$T_s = \{T_1, T_2, \ldots, T_n\} \tag{5.2}$$

$$T_i = \{S_i(j-k), S_i(j-k+1), \ldots, S_i(j), \ldots, S_i(j+k)\}^T \tag{5.3}$$

where n is the number of channels used in analysis, and the length of the proposed model is $2k+1$. To suppress the effect of noise, we take a center weighted mask using a Gaussian function. Let G_i denote the one-dimensional Gaussian weighted vector for the i^{th} channel whose size is $2k+1$.

$$k(n) = \exp\left(-\frac{1}{2}\left(\frac{\|x-k\|}{\sigma}\right)^2\right) \tag{5.4}$$

$$G_i = \{k_1, k_2, \ldots, k_n\}^T \tag{5.5}$$

The Gaussian mask, G, can be defined as

$$G = \{G_1, G_2, \ldots, G_n\} \tag{5.6}$$

where $G_1 = G_2 = \cdots = G_n$. The final template T we use in the analysis can be defined as

$$T = \text{Mask}(G, T_s) \tag{5.7}$$

where $\text{Mask}(A, B)$ returns the elementwise product of A and B.

5.2.3 Preprocessing

Filtering

ECG signals can be easily contaminated by various types of noise. The purpose of the preprocessing step is to remove the noise that is not a valid ECG signal. Baseline drift is one of the phenomena commonly seen in ECG signals.

With band-pass filtering, the artifacts due to baseline drift and high frequency noise can be removed. The proposed method utilizes a band-pass filter with cut-off frequencies of 0.5 Hz and 100 Hz, since the majority of ECG signals are located in the lower-frequency area under 100 Hz.

QRS-Complex Detection

The ECG signal is a pseudo-periodic signal. For beat-by-beat analysis, we need to segment every beat. The QRS Complex is a dominant signal in a beat of the ECG signal. It could be a good separator, which distinguishes adjacent beats. Every fiducial points should be located between the former and latter R peaks; furthermore, the R-R interval should be found to calculate the corrected QT interval, according to Bazzett's formula. Due to this characteristic, our QT analysis entails finding the QRS complex. In this chapter, we employed the *Pan and Tomkins QRS detection method* [41, 25], which is the most popular method for finding the QRS complex.

5.2.4 Detection step

We already have the information about each fiducial point, the QRS onset, and the T wave offset. The templates depict these patterns, which can explain the fiducial points, which are located in the center of the templates. At the detection step, t he proposed algorithm finds the best matching location with respect to the given template. Due to the characteristics of ECG signal, there exist only one matching point in a beat signal, which is the signal between consecutive R peaks. Our proposed algorithm utilizes the result of the peak detection step as a beat separator for beat-by-beat analysis. Let $R = \{r_1, r_2, \ldots, r_n\}$ denote the result of the peak detection. We can split the signal as a single R-to-R beat, which consists of the signal between consecutive R peaks. In the proposed algorithm, the expanded signal as much as the length of a given template will be utilized for handling the boundary. Thus, the expanded k^{th} single R-to-R beat signal B in the i^{th} channel can be defined as

$$B_i = \{S_i(r_k - \frac{w}{2}), S_i(r_k - \frac{w}{2} + 1), \ldots, S_i(r_{k+1} + \frac{w}{2})\}^T \tag{5.8}$$

$$B = \{B_1, B_2, \ldots, B_n\} \tag{5.9}$$

where n is the number of channels in analysis, and w is the length of a given template.

To search for the best matching location compared with a given template, we need to compare the similarity with a given template at each location. As an indicator for investigating the similarity between the signal and a given model, our algorithm utilizes the *Pearson correlation coefficient*, defined as

$$r_{XY} = \frac{\sum_{i=1}^{m} (X_i - \overline{X})(Y_i - \overline{Y})}{\sqrt{\sum_{i=1}^{m} (X_i - \overline{X})^2} \sqrt{\sum_{i=1}^{m} (Y_i - \overline{Y})^2}} \tag{5.10}$$

The sliced signal B_s at a location k, which has exactly the same length as a given template, can be defined as

$$B_{i,s} = \{B_i(k), B_i(k+1), \ldots, B_i(k+w-1)\}^T \tag{5.11}$$

$$B_s = \{B_{1,s}, B_{2,s}, \ldots, B_{n,s}\} \tag{5.12}$$

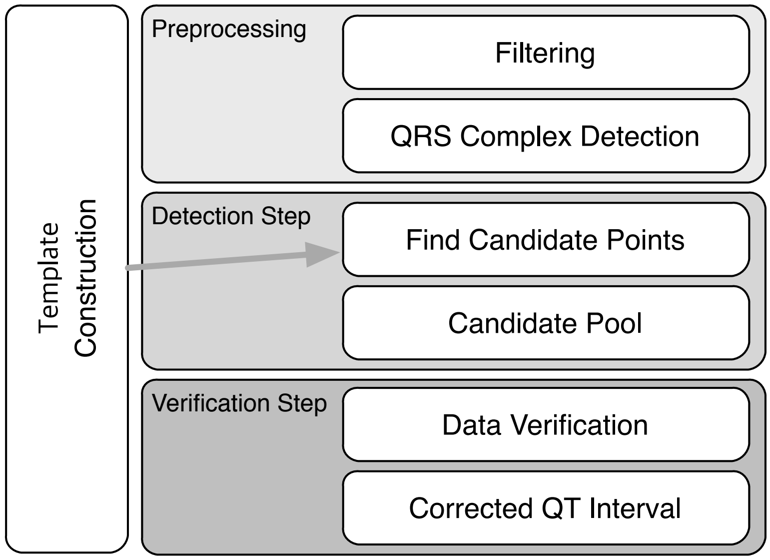

Figure 5.2: Procedure of the proposed algorithm

where w is the length of the given template, and n is the number of channels in analysis. After the elementwise multiplication with a Gaussian mask, which is defined in Eq. (5.6), all these signals will be investigated by calculating the Pearson correlation coefficient with a given template. We can regard the location, which has the highest value of the Pearson correlation coefficient that meets the threshold constraint, as the best matching location in a single beat. According to the proposed method, we can find sets of the searched fiducial points, Q_{on} and T_{off}.

5.2.5 Data verification

Reorganization of Dataset

We should reorganize the dataset according to the result of the above method. A set can be composed from the searched fiducial points (i.e., two consecutive R peaks, a QRS onset, and a T offset), which are utilized to compute the corrected QT interval in Eq. (5.1). The

i^{th} set of the fiducial points can be defined as

$$QT_c^i = \{r_i, r_{i+1}, Q_{\text{on}}(i-1), T_{\text{off}}(i)\} \tag{5.13}$$

With a single set of QT_c^i, we can get the i^{th} corrected QT interval.

Decision Rule for Data Verification

To verify the validity of the searched fiducial points, we need to investigate the points that meet the requirement related to the characteristics. According to the constraints on the human signal, we already have the information regarding the general range of the intervals. Our method uses the following tests to show the validity of the given set from Eq. (5.13).

a) Test for the range of heart rate In [20], the maximum heart rate can be calculated by the formula, $HR_{\text{max}} = 220 - Age$. In other words, the heart rate of most human beings is less than 220 bpm. Generally, normal range for adult heart rates is 60 bpm to 100 bpm, according to [8, 56]. However, younger people have faster heart rates than adults. The article shows the possible range of human heart rates. The proposed algorithm sets the valid range of the heart rate to 40-200 bpm.

b) Test for the validity of Q_{onset} and T_{offset} In a single beat signal, T_{offset} should be located prior to Q_{onset}. If the assumption is not met by the given points, the proposed method rejects the points put into analysis.

Another test for showing the validity of each point is the comparison of the correlation coefficient value with the given template and the searched location. The proposed method has a previously set threshold value. Even though the cross correlation value of a certain location is the maximum value in a given beat signal, it will not be accepted as long as the value is greater than the threshold.

c) Test for the range of the corrected QT interval Bazett's Formula from Eq. (5.1) is the length of the QT interval normalized by the square root of the length of the RR interval. Many researchers have tried to show the possible range of the corrected QT interval [40, 54]. Experimentally, the range should be between 0.3 and 0.6. The given dataset must also meets the condition to be proved as a valid dataset.

5.3 Implementation

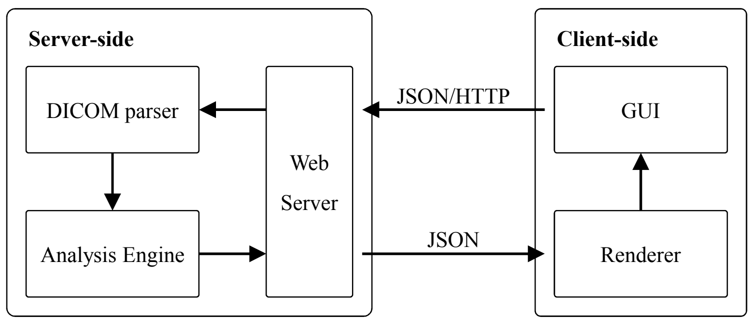

Figure 5.3: Server structure for QT analysis

In this section, we introduce our system to deploy proposed algorithm. Fig. 5.3 shows the structure of our system. The system comprises two parts : server side and client side. It is web-based application, thus, the client side is made of browser based interfaces over HTTP.

5.3.1 DICOM

DICOM(Digital Imaging and Communications in Medicine)[46] is the international standard for medical images and related information. It defines the formats for medical images such as X-ray, CT, ultrasound and MRI, that can be exchanged with the data and quality necessary for clinical use. According to DICOM supplement 30[45], waveform data such as electrocardiographic and hemodynamic signals, can be manipulated inside DICOM format. Our DICOM parser is a component in server side. When the DICOM format is put into server over the network, the web server catches the file and puts it into DICOM parser for processing.

At our application, we employ DICOM standard for manipulating electrocardiographic signals as inputs to the system by users, and keeping the compatibility with recordings from widely used recording devices.

5.3.2 Data flow

According to Fig. 5.3, client-side puts the data, which is formatted as DICOM, through GUI, then the data is transmitted to the server side. We use JSON over HTTP as a data exchanging protocol, since it is simple and easy to implement. At DICOM parser, the transmitted data is translated to digitized signal by parsing metadata included DICOM format. This digitized signal can be the input of our proposed algorithm in 5.2, which is implemented at the server component named analysis engine. The processing results are dumped to the web server and the web server makes them JSON-formatted data, t hen send it to the client side. We implemented the server side with python 2.7 and tornado web server and the client side with jQuery and HTML5.

5.3.3 Web GUI

For the client side, we need to figure out several GUI pages for interfacing user action and visualizing the input and processed data. Fig. 5.5 shows the GUI page for transferring input DICOM formatted recording. By clicking the button, we can put DICOM formatted recording into server. After completing the data transfer, the signal, which is the result of baseline correction at 5.2.3, is rendered. Fig. 5.5 shows the rendering signal after baseline correction.

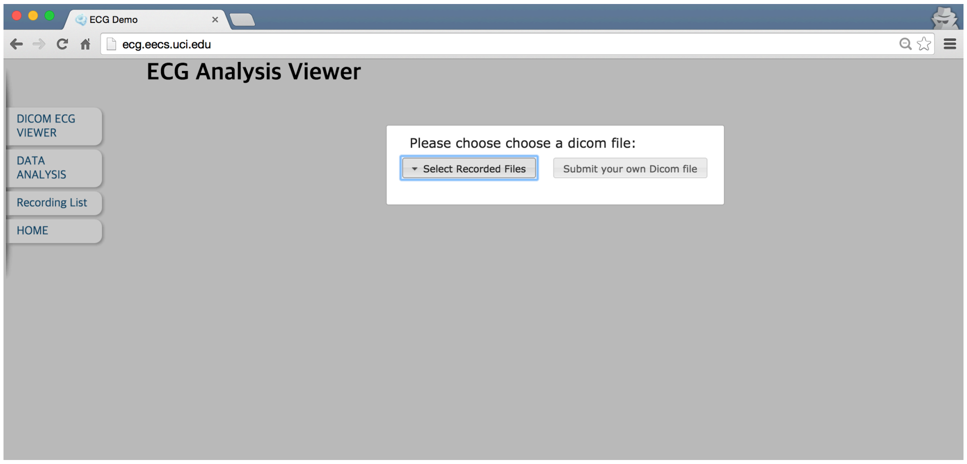

Figure 5.4: DICOM upload at GUI

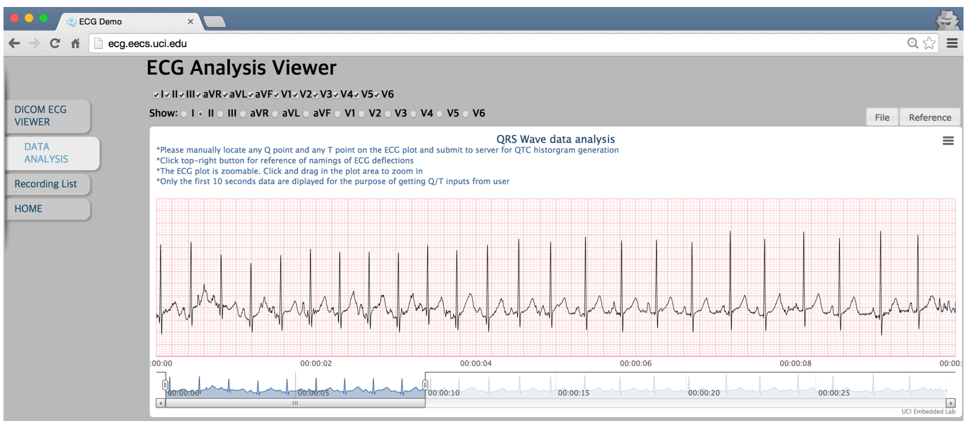

Figure 5.5: Rendering ECG signal at GUI

According to our analysis scenario, the models about the fiducial points, which is described in 5.2 should be set up. Fig. 5.6 depicts the GUI page for fiducial point selection. By intervention of users through this page, the models for QRS onset and T wave offset are automatically generated.

After generating models for searching for each fiducial points, we need to set up initial parameters as Fig. 5.7 . Our proposed algorithm utilizes single channels for searching for the R peaks. Thus, users have to choose which channel is included in peak detection. Another parameter, threshold, is a minimum value to meet the similarity level between models and signals. If the candidate fiducial points has less crosscorrelation value than threshold, our proposed algorithm ignore that points in analysis. We also determine the number of bins for

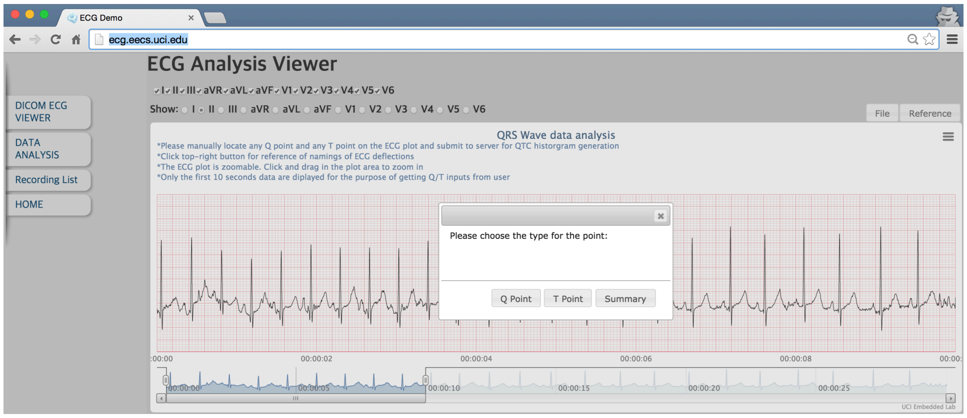

Figure 5.6: Fiducial point selection for making templates at GUI

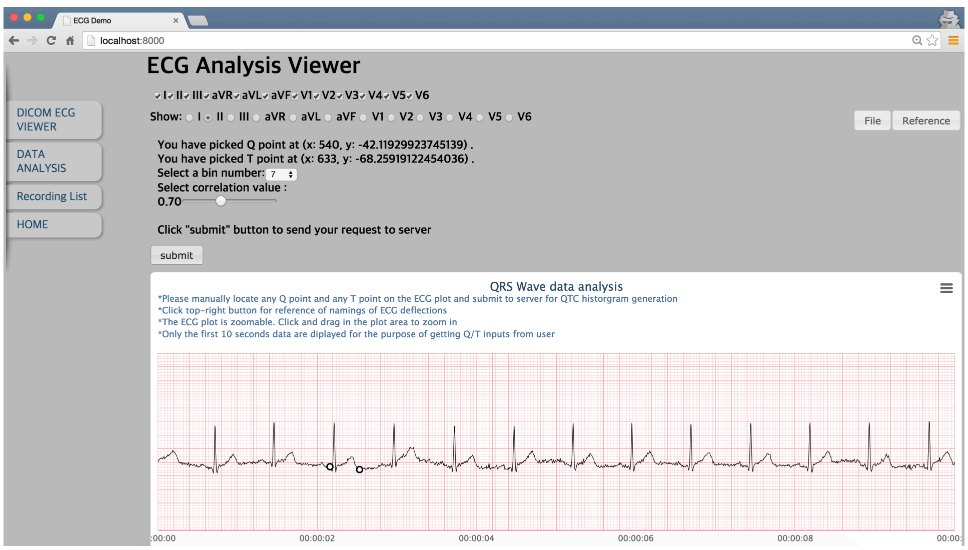

Figure 5.7: Setting up initial parameters

result of QT analysis. Our GUI will visualize the QT analysis result as displaying distribution like histogram. Fig. 5.8 shows the QT analysis result with histogram.

Our GUI also provides functions to mimic the standard format for helping doctors to analyze. Fig. 5.9 depicts that the signals are visualized like real standard paper format. In order to provide convenient method to measure intervals for manual analysis by doctors, Caliper function is included in the GUI page. The red circle in Fig. 5.10 is interval measuring result of two different points on signal.

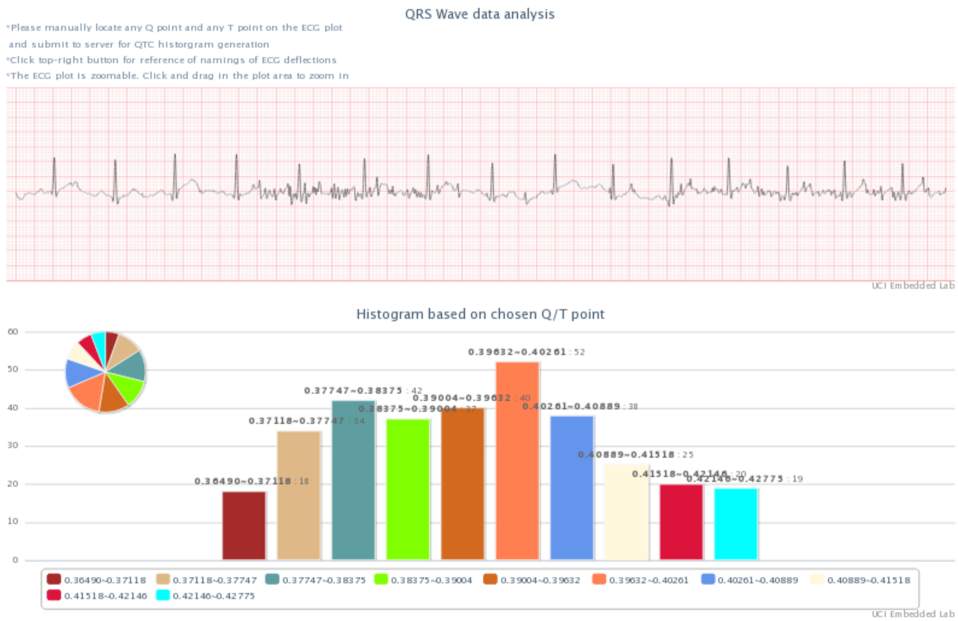

Figure 5.8: Graphical QT analysis result on client side

5.4 Summary

QT analysis is a significant measurement as a good marker of heart-threatening disease. But due to the features of the component waves of ECG signal, subjectiveness of measurements, lack of references, and large volume of data, it is not easy to analyze automatically.

In this chapter, we proposed a new algorithm for beat-by-beat QT analysis algorithm. Further, we built a web-based system for QT analysis by utilizing our proposed algorithm. Our system also provided several useful tools for manual/automatic QT analysis. To meet compatibility with the existed recording device, we employed DICOM recording format.

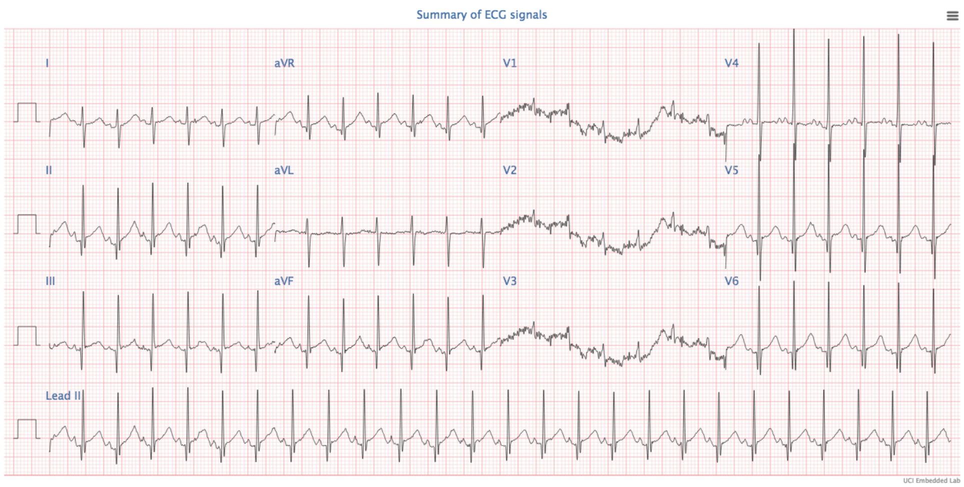

Figure 5.9: ECG snapshot following to standard ECG format

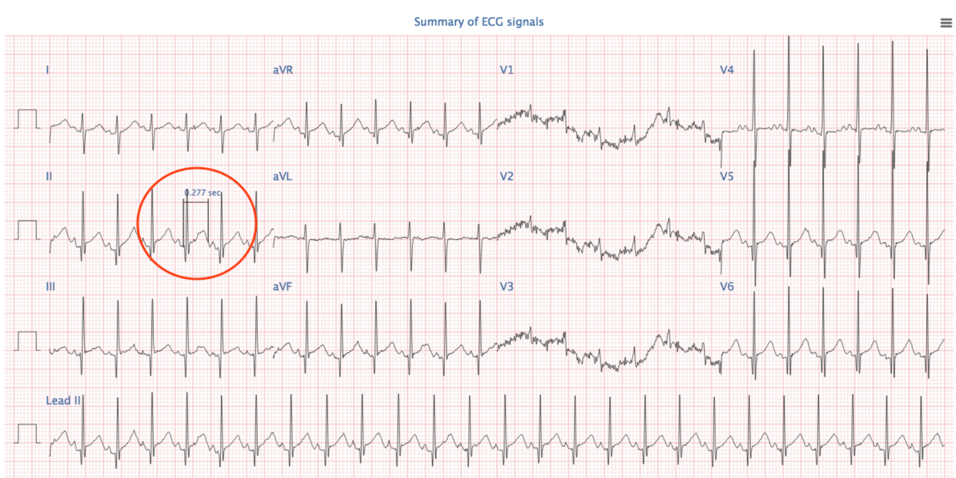

Figure 5.10: Measuring time intervel by caliper function

Chapter 6

In-device QT Analysis

6.1 Motivation

Existing methods for determining the prolongation of the QT interval can be generally classified into two types: global analysis and beat-by-beat analysis. In global analysis, the prolongation of the QT interval can be computed from the representative beats of each lead by calculating the median or mean. In contrast, beat-by-beat analysis utilizes every beat of each lead to compute the prolongation of the QT interval. Global analysis is more robust to noise artifacts than the beat-by-beat analysis but is not suitable for a real-time monitoring system. On the other hand, beat-by-beat analysis is affected by noise from the muscles, devices, or the environment, and **its computational complexity is too high for most embedded systems today** [24]. The existing analysis methods, mathematical modeling based analysis[38], and SVD-based analysis [57] have a runtime complexity of at least $O(n^3)$. In this chapter, we propose a QT-analysis algorithm based on beat-by-beat analysis for real-time prolongation of the QT interval. The lower complexity of $O(n^2)$ and its robustness to noise make it suitable for implementation on wearable embedded systems. Experimental results show that the proposed algorithm achieves real-time performance while consuming low power.

6.2 Data Stream Model

We assume that the signals from the ECG monitoring system are composed of a set of streams, each of which generates an infinite sequence of items with a particular period. Each stream is called a channel. The set of input channels is denoted by $\mathbf{S} = \bigcup_{i=1,2,...} \{S_i\}$, where i is the index of the respective channel. Each stream S_i is recorded in every time interval h_k and generates a new item I_i^k each moment in time kh_i, where $k = \{0, 1, 2, ...\}$. Our system records the signals with a certain sampling rate, f, for all channels. Thus, h_k is defined as $1/f$ sec. I is the set of all items $\mathbf{I} = \bigcup_{i,k}\{I_i^k\}$.

To analyze the data, the system should collect the signals from each channel, which is called a data slice, for several seconds, and then process them until the next data slice has arrived. The timing constraint from the current data slice to the next data slice is called the deadline, D. If the deadline is too short, the analysis result will be less accurate due to the lack of data. On the other hand, if it is too long, the system's response may exceed the deadline.

6.3 Implementation on single core architecture

6.3.1 Serialized implementation

In Section 5.2, we proposed new algorithm for QT analysis. Using the proposed algorithm, we show the implementation for real time QT analysis on embedded system. We already assumed the data stream model mentioned in Section 6.2. According to reflect data stream model, ECG sensor will put the signals of 12 leads into memory. Our proposed algorithm waits for a deadline, analyzes the signals at the point, and then obtains the analysis result before the next data slice coming in.

Algorithm 1 depicts the algorithm for implementation on single core embedded platform.

Algorithm 1: Pseudo code for serialized implementation

```
input  : 12-lead ECG signal
output: Array of corrected QT Intervals(QT_c)
initialization
q_tmpl ← LoadModel(q)           /* load model information for QRS onset */
t_tmpl ← LoadModel(t)           /* load model information for T offset */
numData ← sampling rate×input interval          /* length of data */
ptr ← 0
QRSDet(Init)                          /* Initialize QRS Detector */
while ptr < maxIter do
    dataPtr ← ptr × dataNum                          /* data pointer */
    for i ← 0 to numLead do
        Sig[i] ← loadData(dataPtr, dataNum, i)   /* load the i_th lead data */
    end

    for i ← 0 to numLead do
        h_Filtered[i] = filter(Sig[i], 'highpass')          /* Filtering */
        l_Filtered[i] = filter(Sig[i], 'lowpass')
    end

    QRSDet(l_Filtered['II'])                         /* QRS detection */

    FindFiducialPoint(q)                  /* Finding fiducial points */
    FindFiducialPoint(t)

    VerifyCandidate()                          /* Data verification */
    CalcQTc()                   /* Calculate corrected QT intervals */
    ptr++
end
```

6.3.2 Experimental result

Experimental environment

The algorithm is ported to an ARM7-based single-core embedded platform. The signal used in the analysis is recorded using a wearable 12-lead ECG board based on the TI ADS1298 connected to standard wet electrodes at 250 Hz. To show the performance with respect to different sampling rates, the signal is expanded to 500 Hz and 750 Hz by linear interpolation. We initially set the template size at 200 ms, deadline at 3 s, and threshold value at 0.5.

Performance evaluation

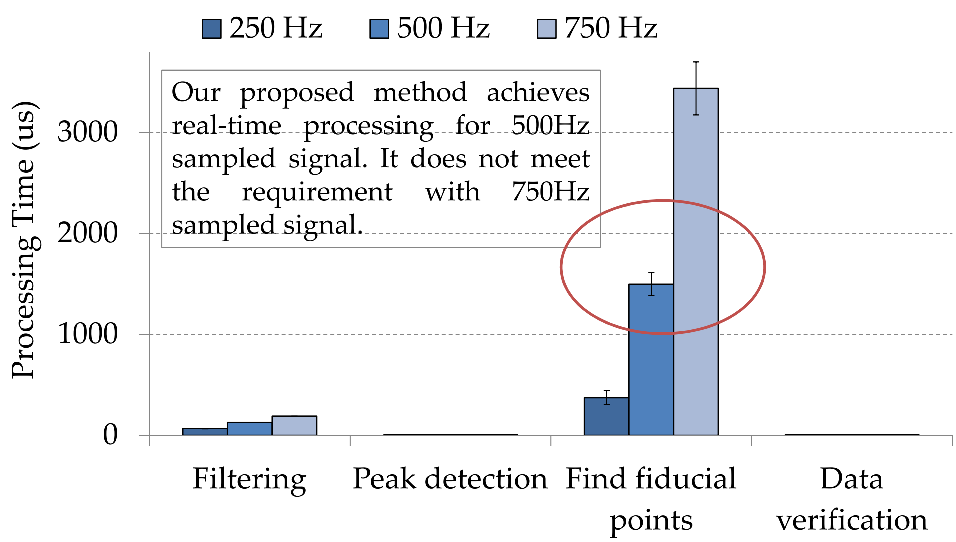

Figure 6.1: Processing time of each stage over the different sampling rates

Fig. 6.1 shows the processing time of each analysis stage on the target platform with respect to the different sampling rates. Fig. 6.2 displays the ratio of the processing time for each stage. According to the figures, most of the processing time is spent on searching for fiducial points. The complexity of searching for fiducial points is $O(n^2)$, which means this takes more processing time for a higher sampling rates. This processing time can't meet the timing

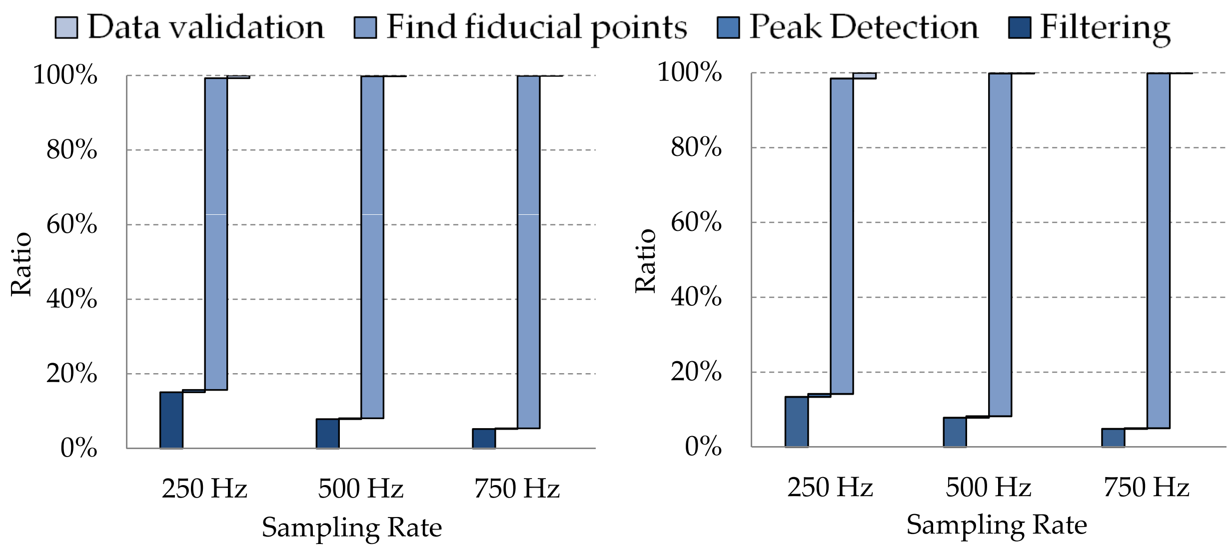

Figure 6.2: The ratio of each stage in entire processing time (left) and energy (right)

requirement at a 750 Hz sampling rate. In this case, parallelization or further optimization is needed.

Energy consumption evaluation

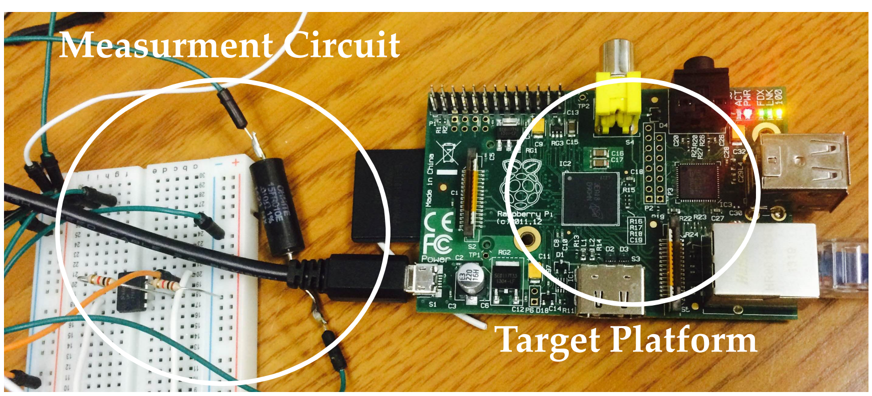

Figure 6.3: Prototype board with energy measurement circuit.

Table 6.1 shows the energy consumed at each stage. Since the target embedded platform on which the proposed algorithm runs is single-core, the energy consumed scales linearly with processing time. Fig. 6.2 also shows that the searching stage consumes most of the energy, and that energy scales linearly with the sampling rate.

Table 6.1: Average and standard deviation of the energy consumed in each stage

	250Hz		500Hz		750Hz	
	avg. (μJ)	std.	avg. (μJ)	std.	avg. (μJ)	std.
Filtering	1.2211	0.4842	3.0588	0.5614	4.3718	1.3677
Peak Detection	0.0732	0.0290	0.1433	0.0257	0.1674	0.0528
Find Fiducial Points	7.6976	1.3725	35.7051	2.8731	85.8586	6.6277
Verification	0.1392	0.0590	0.0744	0.0732	0.0890	0.0507

6.4 Implementation using CUDA

According to Section 6.3.2, the significant bottleneck would be the stage for finding fiducial points. For real-time processing, the stage should be optimized or parallelized. Since our proposed algorithm is correlation-based algorithm, it has a large amount of convolutional computations, which can be applied to pipeline for parallelization on GPU. In this section, we will introduce parallelized implementation of our proposed algorithm using CUDA. We also evaluate the performance after parallelization.

6.4.1 Overview of CUDA

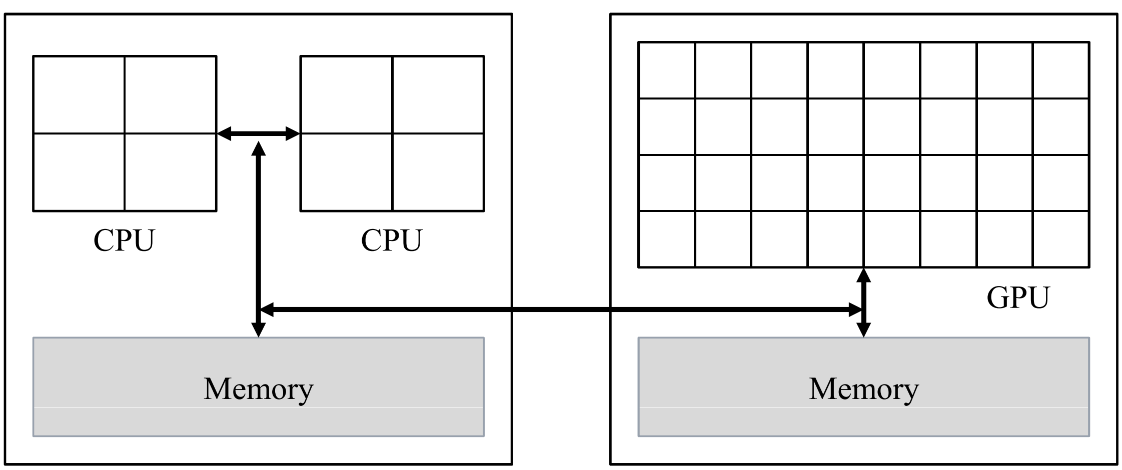

Figure 6.4: Architecture of GPGPU

GPU is the processor applied in graphic processing. Since computation for graphical processing is complicated and repetitive, it has lots of simple cores in parallel processing to enhance performance. When the needs of powerful computing is increased, the GPU can be used in various applications, not even graphical computations, as the concept of GPGPU(General Purpose GPU). The concept of GPGPU is shown at Fig. 6.4. GPGPU has both latency processor(CPU) and throughput processor(GPU). Generally, a CPU consists of a few cores optimized for sequential serial processing, while a GPU has a massively parallel architecture consisting of many simple cores designed for handling multiple tasks simultaneously.

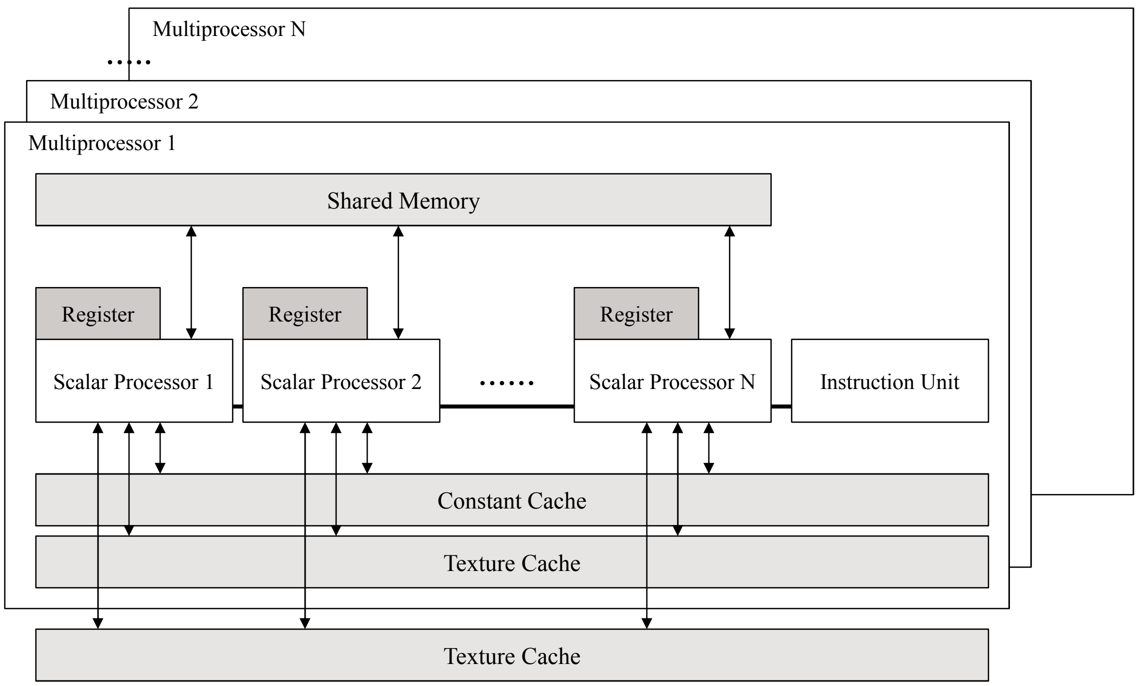

Figure 6.5: Architecture of CUDA

CUDA(Compute Unified Device Architecture) is one of the well-known parallel computer architecture developed by NVIDIA. Fig. 6.5 shows the CUDA programming model and architecture[1]. Device defines as a GPU chip, which is composed of several MPs(Multiprocessors) or a single MP. Each MP is composed of SPs(Scalar Processors), register, instruction unit, shared memory, and cache. Each SP operates independently. In CUDA programming model, a task to be processed on the GPU is divided into grid, block, and thread. The thread is a minimum task, which is executed on a SP. The block is composed of several threads, and the grid is composed of several blocks. The block is executed on a MP, and the grid is executed on the device. The threads from the same block can cooperate with each other by sharing data of synchronizing their executions through shared memory. However, if the threads are from different blocks, there is no way to cooperate with each other.

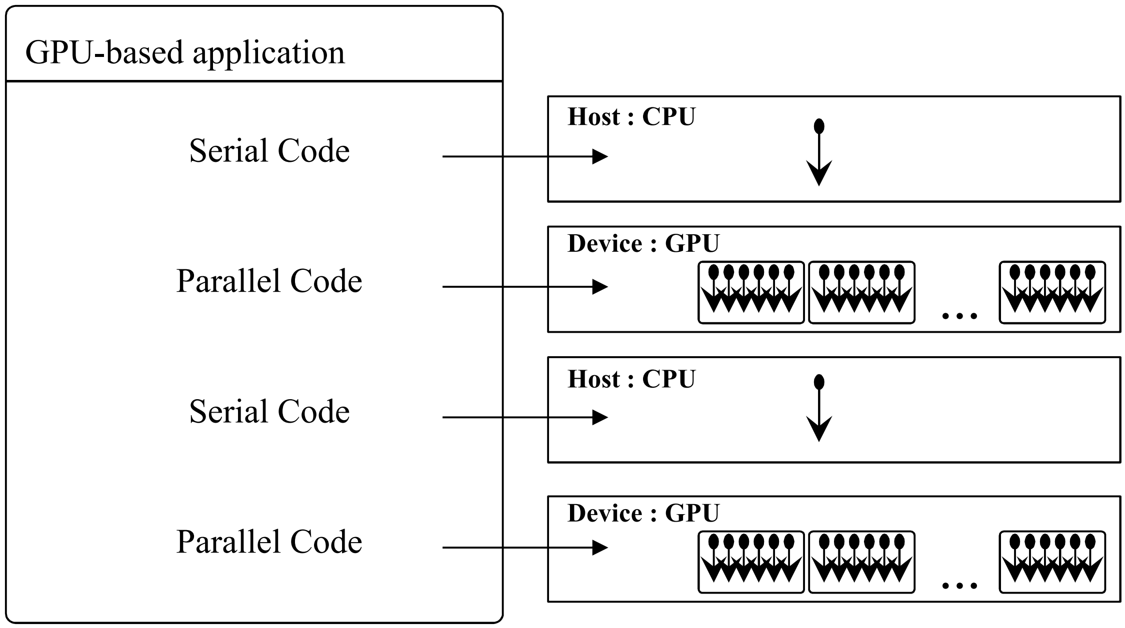

Figure 6.6: Execution flow of parallelized algorithm

The execution flow of parallelized algorithm on the GPU is shown at Fig. 6.6. A GPU-based application is composed of serial codes and parallel codes. The serial code is executed on host(CPU), and the parallelized code is executed on the device(GPU). Generally, the parallelized portion of an application is called as a kernel. Only one kernel is executed at a time on the device, and it has many threads executed each kernel. As mentioned above, the threads in the same kernel are independently executed at SP. Thus, general strategy of parallelization is that the entire algorithm should be separated to independent modules, which has no feedback or relationship during execution.

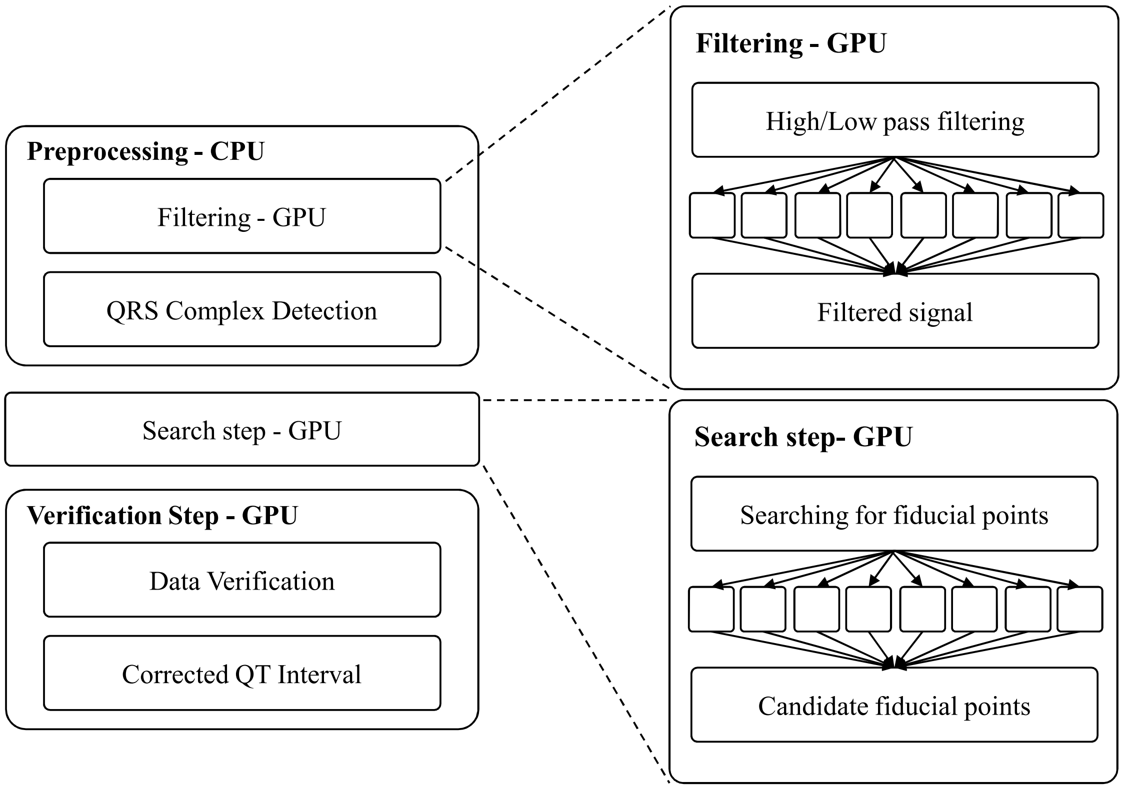

Figure 6.7: Algorithm parallelization

6.4.2 Parallelized implementation

According to Section 6.3, the serialized implementation faces a limitation for processing with a large amount of data. If the data is getting larger, the computational complexity is increased exponentially. In order to solve this problem, we make the proposed algorithm parallelized using GPGPU. Fig. 6.7 shows the strategy of our parallelization. As seen in Fig. 6.2, most of processing time is occupied by filtering and finding fiducial point step. Since these steps comprise repetitive calculations, we can expect the performances of our proposed algorithm to be improved. Algorithm 2 depicts pseudo code of our parallelized method for implementation on GPGPU.

The filtering and searching for fiducial steps are windowed calculation. Each windowed data block is independent from other blocks, and further repetitive, since there is no feedback

Algorithm 2: Pseudo code for GPU parallelization

```
input  : 12-lead ECG signal
output: Array of corrected QT Intervals(QT_c)
initialization
q_tmpl ← LoadModel(q)              /* load model information for QRS onset */
t_tmpl ← LoadModel(t)                /* load model information for T offset */
numData ← sampling rate×input interval                  /* length of data */
ptr ← 0
QRSDet(Init)                                    /* Initialize QRS Detector */
while ptr < maxIter do
    dataPtr ← ptr × dataNum                                /* data pointer */
    for i ← 0 to numLead do
        Sig[i] ← loadData(dataPtr, dataNum, i)     /* load the i_th lead data */
    end

    cudaMemcpy(Sig, Sig_Dev)                  /* memcopy from host to device */
    /* Execute kernel for filtering                                         */
    h_cuFiltered, l_cuFiltered ← cudaFilterKernel(Sig_Dev)
    cudaMemcpy(h_cuFiltered, h_Filtered)      /* memcopy from device to host */
    cudaMemcpy(l_cuFiltered, l_Filtered)
    cudaDeviceSynchronize()

    QRSDet(l_Filtered['II'])                                /* QRS detection */

    cudaMemcpy(l_Filtered, cuFiltered)        /* memcopy from host to device */
    /* Execute kernel for searching for fiducial points                     */
    cudaCandT, cudaCandQ < −cudaSearchKernel(cudaFiltered)
    cudaMemcpy(cudaCandT, CandT)              /* memcopy from device to host */
    cudaMemcpy(cudaCandQ, CandQ)
    cudaDeviceSynchronize()

    VerifyCandidate()                                   /* Data verification */
    CalcQTc()                             /* Calculate corrected QT intervals */
    ptr + +
end
```

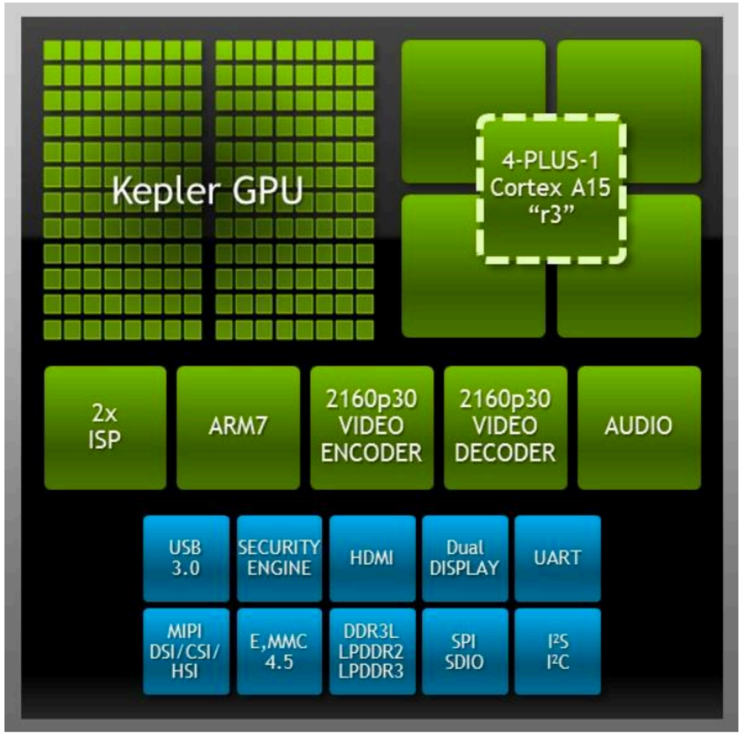

Figure 6.8: NVIDIA Tegra K1 Mobile Processor on Jetson TK1

from the other blocks during calculation. These indicates that these steps are suitable for parallel implementation on the GPU. As seen in Algorithm 2, these steps are designed to CUDA kernel.

6.4.3 Experimental results

We used Jetson TK1 for evaluating the proposed parallelized algorithm. This board is based on the hybrid processor Tegra K1. It consists of a quad-core ARM Cortex A15 CPU and a NVIDIA Kepler core with 192 computational cores. Fig. 6.8 is shown to the architecture of Tegra K1 on Jetson TK1 evaluation board.

We compare the performance of the sequential implementation and parallelized implementation. Mainly, the performance metrics are the running time and energy consumption of each parallelized step. The dataset used in analysis is recorded at 250Hz by standard 12 lead ECG recording system, and it would be expanded to 500Hz, 750Hz and 1kHz through linear interpolation.

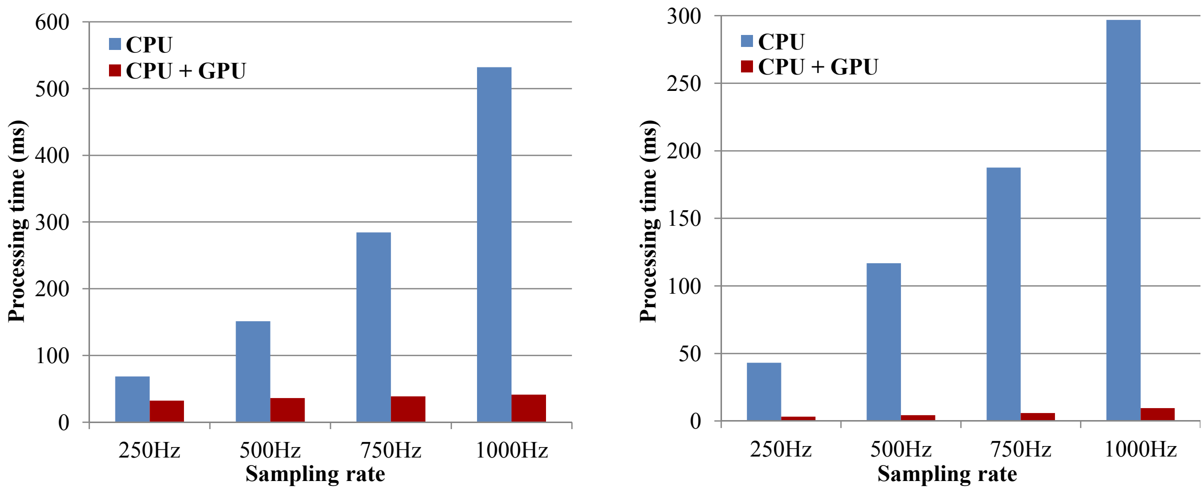

Figure 6.9: The comparison of CPU implementation and GPU parallelization for filtering stage(left) and searching for fiducial points stage(right)

Fig. 6.9 shows the performance of each step. The left figure on Fig. 6.9 depicts the running time of the filtering step. According to the result, the processing time on the CPU only, is proportional to the square of the sampling rate, since the filtering has convolutional computations. On the contrary, we would get better performance after parallelization with the CPU and the GPU. The right figure on Fig. 6.9 shows the running time of searching for the fiducial point stage. Our proposed method is based on the comparison of the Pearson correlation coefficients, which is also convolutional compuation, the running time on CPU implementation is also proportional to the square of the sampling rate. According to the performance comparison, we would get benefits about the running time after parallelized implementation. As getting larger volumes of data, the running time of the serialized code on the CPU only gets exponential increase. On the contrary, the running time of the parallelized implementation looks more stable. Even though the data size is doubled, the total running time is increased at a minimum.

Fig. 6.10 shows the average power consumption for filtering stage and searching for fiducial

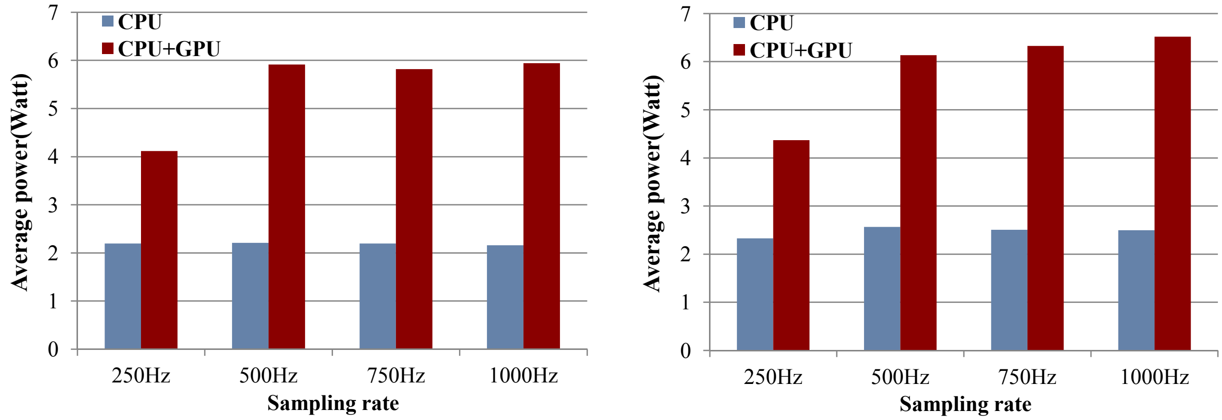

Figure 6.10: The comparison of average power consumption between CPU implementation and GPU parallelization for filtering stage(left) and searching for fiducial points stage(right)

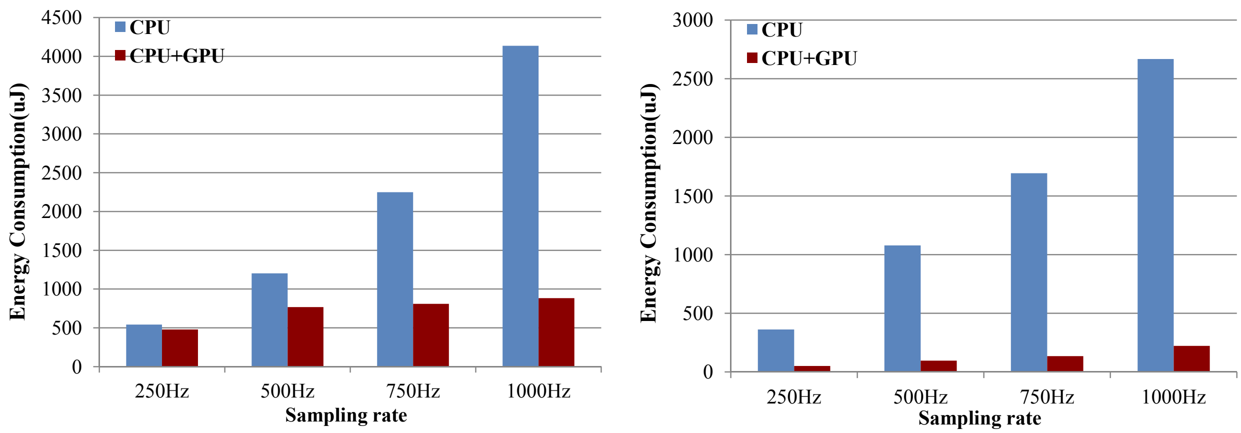

Figure 6.11: The comparison of energy consumption between CPU implementation and GPU parallelization for filtering stage(left) and searching for fiducial points stage(right)

point stage. To calculate the power consumption, we measure the power consumed in each stage, then subtract the base power, which is consumed in default. According to Fig. 6.10, the average power consumption of the parallelized implementation using the CPU and the GPU is higher than that of the implementation on the CPU only. However, energy consumption of the parallelized implementation is much less than the serialized implementation on the CPU only, since the parallelized implementation obtains much gain from processing time using throughput processor. Fig. 6.11 explains that the energy consumption of the parallelized implementation is much lower than that of the serialized implementation. Table 6.2 shows the consumed energy of each stage. According to the characteristic of the CPU and GPU architecture, the CPU and GPU architecture needs more power for operating than the CPU based architecture. Even though the higher base power of the CPU and GPU architecture, the total energy consumed at each step is lower than the CPU only. It is because the running time is much shorter than the CPU only implementation.

Table 6.2: Comparison of energy consumption(μJ)

	Architecture	250Hz	500Hz	750Hz	1kHz
Filtering	CPU	541.5564	1203.8120	2247.7450	4137.362
	CPU+GPU	478.7691	767.6780	810.0001	883.4907
Find fiducial points	CPU	361.8818	1079.3800	1693.9180	2667.817
	CPU+GPU	50.4509	95.6088	135.5463	222.8812

6.5 Summary

In this section, the in-device implementation is discussed. There are two different types of approaches introduced : One is the serialized implementation and the other is the parallelized implementation. We also implement our proposed QT analysis algorithm on single core architecture, however, it would not show enough performance for real-time processing. Since the size of data to be processed is increasing if the higher sampling rate is used in the device, parallelization using throughput based processor is one of the good solutions to achieve the real-time processing.To achieve the real-time processing, we use the CPU and GPU co-designed platform for parallelization of the algorithm. We show that the timing performance using the CPU and GPU together is much better than the implementation on single core architecture. Furthermore, we can obtain another benefit, which is energy saving. Even though the average power consumption of the implementation on the CPU and GPU is higher than that of the implementation on the CPU only, the energy consumption is much less than the CPU only implementation due to the timing performance.

Chapter 7

Conclusion and Future Works

7.1 Conclusions

This dissertation describes a framework for development of wearable medical devices and systems. Our framework can encompass the state-of-the-art in the field of wearable medical devices and applications. It organizes the many considerations into the sensor plane, data plane, and application plane. Furthermore, we provide a use case for our framework in the form of an end-to-end system for ECG monitoring and analysis.

This work also proposes algorithms for several aspects of the ECG application. First, we propose a novel approach for QT analysis and its implementation, and we show a real-time implementation of our proposed algorithm to be feasible on an embedded system as well as in a server-client environment. For effective data handling, we also propose a new compression/decompression algorithm for ECG signals using a trained dictionary. It utiliizes patterns learned with a given dataset and helps improving the compression ratio with low distortion. We also explore reducing sensing power by compressive sensing and support the reconstruction of the signal from undersampled data. Using our collection of algorithms, we expect ECG devices in general to be able to reduce energy consumption during data collection, a feature especially important for battery-powered wearable systems.

7.2 Furture Work

Much work remains to take our proposed framework to its full realization. Many components in each of the planes may be in existence but need to be integrated. Directions for future work include security and privacy, performance and feature enhancements of the algorithms, and expanding the application base. Ultimately, the goal of the framework is to build up a full database of medical data that can be accessible to medical practitioners globally, and it requires not only raw data but more importantly annotations either manually by medical professionals or automatically by algorithms. The annotated data will enable improvement to our learning-based algorithm depicted in Chapters 3 and 4 in both performance and accuracy.

At the same time, our proposed analysis algorithm in Chapter 5 can still be further improved. Its current limitation is that it focuses on the normal type of signal only. Many abnormal cases in ECG signal such as monophasic, biphasic, and triphasic cases currently cannot be handled reliably. Someone with a cardiovascular disease such as arrhythmia may have a varying type of signal. Future work therefore must consider such abnormal cases.

Bibliography

[1] CUDA zone for developers. https://developer.nvidia.com/cuda-zone.

[2] MIT-BIH arrhythmia database. http://www.physionet.org/physiobank/database/mitdb/.

[3] Sparselab toolbox for matlab. http://sparselab.stanford.edu.

[4] J. P. Abenstein and W. J. Tompkins. A new data-reduction algorithm for real-time ECG analysis. *IEEE Transactions on Biomedical Engineering*, (1):43–48, 1982.

[5] M. Aharon, M. Elad, and A. Bruckstein. K-SVD: An algorithm for designing overcomplete dictionaries for sparse representation. *IEEE Transactions on Signal Processing*, 54(11):4311–4322, 2006.

[6] H. C. Bazett. An analysis of the time-relations of electrocardiograms. *Heart*, 7:353–370, 1920.

[7] F. Bellifemine, G. Fortino, R. Giannantonio, R. Gravina, A. Guerrieri, and M. Sgroi. Spine: a domain-specific framework for rapid prototyping of wbsn applications. *Software: Practice and Experience*, 41(3):237–265, 2011.

[8] D. Bernstein. Evaluation of the cardiovascular system: history and physical evaluation. *Nelson Textbook of Pediatrics. 19th ed. Philadelphia, PA: Elsevier Saunders*, 2011.

[9] British Standards Institute Staff. *Health Informatics. Standard Communication Protocol. Computer-Assisted Electrocardiography.* B S I Standards, 2005.

[10] E. J. Candè and M. B. Wakin. An introduction to compressive sampling. *Signal Processing Magazine, IEEE*, 25(2):21–30, 2008.

[11] E. J. Candès, J. Romberg, and T. Tao. Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information. *Information Theory, IEEE Transactions on*, 52(2):489–509, 2006.

[12] E. J. Candes and T. Tao. Near-optimal signal recovery from random projections: Universal encoding strategies? *Information Theory, IEEE Transactions on*, 52(12):5406–5425, 2006.

[13] H. Cao, V. Leung, C. Chow, and H. Chan. Enabling technologies for wireless body area networks: A survey and outlook. *Communications Magazine, IEEE*, 47(12):84–93, 2009.

[14] J. Cox, F. Nolle, H. Fozzard, and G. Oliver. AZTEC, a preprocessing program for real-time ECG rhythm analysis. *IEEE Transactions on Biomedical Engineering*, (2):128–129, 1968.

[15] A. Djohan, T. Q. Nguyen, and W. J. Tompkins. ECG compression using discrete symmetric wavelet transform. In *Proc. IEEE 17th Annual Conference on Engineering in Medicine and Biology Society*, volume 1, pages 167–168. IEEE, 1995.

[16] D. L. Donoho. Compressed sensing. *Information Theory, IEEE Transactions on*, 52(4):1289–1306, 2006.

[17] C. M. Fira and L. Goras. An ECG signals compression method and its validation using NNs. *IEEE Transactions on Biomedical Engineering*, 55(4):1319–1326, 2008.

[18] G. Fortino, R. Giannantonio, R. Gravina, P. Kuryloski, and R. Jafari. Enabling effective programming and flexible management of efficient body sensor network applications. *Human-Machine Systems, IEEE Transactions on*, 43(1):115–133, 2013.

[19] D. I. Fotiadis, C. Glaros, and A. Likas. Wearable medical devices. *Wiley Encyclopedia of Biomedical Engineering*, 2006.

[20] S. Fox 3rd, J. P. Naughton, and W. L. Haskell. Physical activity and the prevention of coronary heart disease. *Annals of Clinical Research*, 3(6):404–432, 1971.

[21] A. L. Goldberger, L. A. Amaral, L. Glass, J. M. Hausdorff, P. C. Ivanov, R. G. Mark, J. E. Mietus, G. B. Moody, C.-K. Peng, and H. E. Stanley. Physiobank, physiotoolkit, and physionet components of a new research resource for complex physiologic signals. *Circulation*, 101(23):e215–e220, 2000.

[22] R. Gravina, A. Guerrieri, G. Fortino, F. Bellifemine, R. Giannantonio, and M. Sgroi. Development of body sensor network applications using spine. In *Systems, Man and Cybernetics, 2008. SMC 2008. IEEE International Conference on*, pages 2810–2815. IEEE, 2008.

[23] C. L. Green, P. Kligfield, S. George, I. Gussak, B. Vajdic, P. Sager, and M. W. Krucoff. Detection of qt prolongation using a novel ecg analysis algorithm applying intelligent automation: Prospective blinded evaluation using the cardiac safety research consortium ecg database. *American heart journal*, 163(3):365, 2012.

[24] M. Hadjem, O. Salem, and F. Naït-Abdesselam. An ecg monitoring system for prediction of cardiac anomalies using wban. In *e-Health Networking, Applications and Services (Healthcom), 2014 IEEE 16th International Conference on*, pages 441–446. IEEE, 2014.

[25] P. S. Hamilton and W. J. Tompkins. Quantitative investigation of qrs detection rules using the mit/bih arrhythmia database. *Biomedical Engineering, IEEE Transactions on*, (12):1157–1165, 1986.

[26] P. A. Harris, R. Taylor, R. Thielke, J. Payne, N. Gonzalez, and J. G. Conde. Research electronic data capture (redcap)a metadata-driven methodology and workflow process for providing translational research informatics support. *Journal of biomedical informatics*, 42(2):377–381, 2009.

[27] J. Hash. *An introductory resource guide for implementing the Health Insurance Portability and Accountability Act (HIPAA) security rule.* PhD thesis, National Institute of Standards and Technology, 2005.

[28] M. L. Hilton. Wavelet and wavelet packet compression of electrocardiograms. *IEEE Transactions on Biomedical Engineering*, 44(5):394–402, 1997.

[29] C. Huo, T.-C. Chien, and P. H. Chou. Middleware for iot-cloud integration across application domains. *Design & Test, IEEE*, 31(3):21–31, 2014.

[30] S. Jalaleddine, C. G. Hutchens, R. D. Strattan, W. Coberly, et al. Ecg data compression techniques-a unified approach. *Biomedical Engineering, IEEE Transactions on*, 37(4):329–343, 1990.

[31] S. Kang, J. Lee, H. Jang, H. Lee, Y. Lee, S. Park, T. Park, and J. Song. Seemon: scalable and energy-efficient context monitoring framework for sensor-rich mobile environments. In *Proceedings of the 6th international conference on Mobile systems, applications, and services*, pages 267–280. ACM, 2008.

[32] P. Laguna, R. Jané, and P. Caminal. Automatic detection of wave boundaries in multilead ecg signals: Validation with the cse database. *Computers and biomedical research*, 27(1):45–60, 1994.

[33] K. Lorincz, B.-r. Chen, G. W. Challen, A. R. Chowdhury, S. Patel, P. Bonato, M. Welsh, et al. Mercury: a wearable sensor network platform for high-fidelity motion analysis. In *SenSys*, volume 9, pages 183–196, 2009.

[34] K. Lorincz, B.-r. Chen, J. Waterman, G. Werner-Allen, and M. Welsh. Resource aware programming in the pixie os. In *Proceedings of the 6th ACM conference on Embedded network sensor systems*, pages 211–224. ACM, 2008.

[35] Z. Lu, D. Y. Kim, and W. A. Pearlman. Wavelet compression of ECG signals by the set partitioning in hierarchical trees algorithm. *IEEE Transactions on Biomedical Engineering*, 47(7):849–856, 2000.

[36] A. Lymberis. Smart wearables for remote health monitoring, from prevention to rehabilitation: current r&d, future challenges. In *Information Technology Applications in Biomedicine, 2003. 4th International IEEE EMBS Special Topic Conference on*, pages 272–275. IEEE, 2003.

[37] A. Lymberis and A. Dittmar. Advanced wearable health systems and applications-research and development efforts in the european union. *Engineering in Medicine and Biology Magazine, IEEE*, 26(3):29–33, 2007.

[38] J. P. Madeiro, W. B. Nicolson, P. C. Cortez, J. A. Marques, C. R. Vázquez-Seisdedos, N. Elangovan, G. A. Ng, and F. S. Schlindwein. New approach for t-wave peak detection and t-wave end location in 12-lead paced ecg signals based on a mathematical model. *Medical engineering & physics*, 35(8):1105–1115, 2013.

[39] D. Malan, T. Fulford-Jones, M. Welsh, and S. Moulton. Codeblue: An ad hoc sensor network infrastructure for emergency medical care. In *International workshop on wearable and implantable body sensor networks*, volume 5, 2004.

[40] M. Malik, P. Färbom, V. Batchvarov, K. Hnatkova, and A. Camm. Relation between qt and rr intervals is highly individual among healthy subjects: implications for heart rate correction of the qt interval. *Heart*, 87(3):220–228, 2002.

[41] M. S. Manikandan and K. Soman. A novel method for detecting r-peaks in electrocardiogram (ecg) signal. *Biomedical Signal Processing and Control*, 7(2):118–128, 2012.

[42] J. P. Martínez, R. Almeida, S. Olmos, A. P. Rocha, and P. Laguna. A wavelet-based ecg delineator: evaluation on standard databases. *Biomedical Engineering, IEEE Transactions on*, 51(4):570–581, 2004.

[43] M. Meingast, T. Roosta, and S. Sastry. Security and privacy issues with health care information technology. In *Engineering in Medicine and Biology Society, 2006. EMBS'06. 28th Annual International Conference of the IEEE*, pages 5453–5458. IEEE, 2006.

[44] G. Nave and A. Cohen. ECG compression using long-term prediction. *IEEE Transactions on Biomedical Engineering*, 40(9):877–885, 1993.

[45] P. NEMA. DICOM suppl. 30, waveform interchange, national electrical manufacturers association (nema), washington d.c., usa, 1999.

[46] P. NEMA. Iso 12052, digital imaging and communications in medicine (dicom) standard, national electrical manufacturers association (nema), rosslyn, va, usa, 2015.

[47] A. NIST. Advanced encryption standard. *FIPS Publication*, 197, 2001.

[48] U. D. of Health, H. Services, et al. Summary of the hipaa privacy rule. *Washington, DC: Department of Health and Human Services*, 2003.

[49] S. Olmos, M. MillAn, J. Garcia, and P. Laguna. ECG data compression with the Karhunen-Loeve transform. In *Computers in Cardiology*, pages 253–256. IEEE, 1996.

[50] J. Pan and W. J. Tompkins. A real-time QRS detection algorithm. *IEEE Transactions on Biomedical Engineering*, (3):230–236, 1985.

[51] G. K. Panicker, D. R. Karnad, R. Joshi, S. Shetty, N. Vyas, S. Kothari, and D. Narula. Z-score for benchmarking reader competence in a central ecg laboratory. *Annals of Noninvasive Electrocardiology*, 14(1):19–25, 2009.

[52] D. Raskovic, T. Martin, and E. Jovanov. Medical monitoring applications for wearable computing. *The Computer Journal*, 47(4):495–504, 2004.

[53] T. C. Rindfleisch. Privacy, information technology, and health care. *Communications of the ACM*, 40(8):92–100, 1997.

[54] A. Sagie, M. G. Larson, R. J. Goldberg, J. R. Bengtson, and D. Levy. An improved method for adjusting the qt interval for heart rate (the framingham heart study). *The American journal of cardiology*, 70(7):797–801, 1992.

[55] V. Shnayder, B.-r. Chen, K. Lorincz, T. R. F. Jones, and M. Welsh. Sensor networks for medical care. In *SenSys*, volume 5, pages 314–314, 2005.

[56] D. Simel. Approach to the patient: history and physical examination. *Goldmans cecil medicine, 24th edn. Elsevier Saunders, Philadelphia*, 2011.

[57] V. Starc and T. T. Schlegel. Real-time multichannel system for beat-to-beat qt interval variability. *Journal of electrocardiology*, 39(4):358–367, 2006.

[58] N. V. Thakor and Y.-S. Zhu. Applications of adaptive filtering to ecg analysis: noise cancellation and arrhythmia detection. *Biomedical Engineering, IEEE Transactions on*, 38(8):785–794, 1991.

[59] R. Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 267–288, 1996.

[60] K. Townsend, J. W. Haslett, T. K. Tsang, M. N. El-Gamal, K. Iniewski, et al. Recent advances and future trends in low power wireless systems for medical applications. In *System-on-Chip for Real-Time Applications, 2005. Proceedings. Fifth International Workshop on*, pages 476–481. IEEE, 2005.

[61] J. A. Tropp. Greed is good: Algorithmic results for sparse approximation. *IEEE Transactions on Information Theory*, 50(10):2231–2242, 2004.

[62] T. A. Welch. A technique for high-performance data compression. *Computer*, 17(6):8–19, 1984.

[63] M. Zhang, A. Raghunathan, and N. K. Jha. Medmon: Securing medical devices through wireless monitoring and anomaly detection. *Biomedical Circuits and Systems, IEEE Transactions on*, 7(6):871–881, 2013.

Appendices

Chapter

Compressive Sensing Framework

A Introduction to compressive sensing(CS)

Digital world is driving the development and deployment of new kinds of sensing system with ever-increasing fidelity and resolution. The pioneering work of Nyquist, Shannon and Whittaker on sampling continuous-time band-limited signals leads new types of sensing technoloies. Digitization has enabled the creation of sensing and processing systems that are more robust, flexible, cheaper and consqequently, more widely-used than their analog technology.

The Shannon/Nyquist sampling theorem specifies that to avoid losing information when capturing(or digitizing) a signal, one must sample at leat two times faster than the signal bandwidth. However, in many applications, the Nyquist sampling rate is too high that too many samples results, which means that the result includes redundancies, making compression a necessity prior to storage or transmission.

Compressed sensing or compressive sensing(CS)[10] was introduced as a new sensing framework, which is more efficient and less redundant than the Shannon/Nyquist sampling method.

A.1 Sparse representation

Sparse approximation is the representation that accounts for most of signal information with a linear combination of a small number of elementary signals. Mathematically speaking, if signal x, which can be represented as an $N \times 1$ column vector in $\mathbb{R}^N$. Any signal in $\mathbb{R}^N$ can be represented in terms of a basis of $N \times 1$ vectors $\{\psi_i\}_{i=1}^N$. For simplicity, let the basis be orthonormal. Using $N \times N$ basis matrix $\Psi = \{\psi_1|\psi_2|\ldots|\psi_N\}$ with the vectors $\{\psi_i\}$ as columns, a signal x can be expressed as

$$x = \mathbf{\Psi s} = \sum_{i=1}^{N} s_i \psi_i \tag{A.1}$$

where $\mathbf{s}$ is the $N \times 1$ column vector of weighting coefficients $s_i = \langle x, \psi_I \rangle = \psi_i^T \mathbf{x}$. Clearly, x and s are equivalent representations of the signal, with x in the time or space domain and s are in the Ψ domain. The signal x is K-*sparse* if it is a linear combination of only K basis vectors. That is, only K of the s_i coefficients in A.1 are nonzero and $(N-K)$ are zero. The cas of interest is when $K << N$. The signal x is *compressible* or *sparse* if the representation A.1 has just a few large coefficients and many small coefficients.

A.2 Restricted Isometry Property (RIP)

Restricted Isometry Property(RIP) is one of the important notion in CS framework. Generally, measurement y can be modelled as

$$y = Ax + z \tag{A.2}$$

where A is an $m \times n$ matrix($m < n$), known as a sensing matrix, and z is an unknown error term for describing noisy environment. For the robust signal recovery from undersampled measurements in noisy environment, RIP condition was introduced by D. Donoho, E. Candes ans T. Tao in 2004.

Definition 1. *Let A be an $m \times n$ matrix and let $1 \leq K \leq n$ be an integer. if there exsits δ_K of a matrix A as the smallest number that*

$$(1 - \delta_K)\|x\|_2^2 \leq \|Ax\|_2^2 \leq (1 + \delta_K)\|x\|_2^2 \tag{A.3}$$

holds for all K-sparse vector x, we could say that a matrix A satisfies RIP condition of order K.

As a noiseless case, the reconstructed signal can be defined $\hat{x}$ subject to $A\hat{x} = y$. If $\hat{x}$ is a sparse solution, $\hat{x}$ can be solved by the following linear program.

$$\min_{\hat{x} \in \mathbb{R}^n} \|\hat{x}\|_1 \text{ subject to } A\hat{x} = y \tag{A.4}$$

If we consider the noisy environments, the measurement can be handled like A.2. Hence, the reconstruced solution $\hat{x}$ can be solved by the following linear program.

$$\min_{\hat{x} \in \mathbb{R}^n} \|\hat{x}\|_1 \text{ subject to } \|A\hat{x} - y\| = \epsilon \tag{A.5}$$

Where ϵ bounds the amount of noise in the signal. D.Donoho, E. Candes, and T. Tao proved that the solution $\hat{x}$ can be recovered if the matrix A obeys RIP condition and the isometry constant $\delta_{2K} < \sqrt{2} - 1$.

A.3 Coherence

Definition 2. *The coherence between the sensing basis* Φ *and the representation basis* Ψ *is defined as*

$$\mu(\mathbf{\Phi}, \mathbf{\Psi}) = \sqrt{n} \cdot \max_{1 \le k,j \le n} |\langle \phi_k, \psi_j \rangle| \tag{A.6}$$

According to A.6, the coherence means the largest correlation between any two elements of Φ and Ψ. The range of the coherence is defined as $\mu(\mathbf{\Phi}, \mathbf{\Psi}) \in [1, \sqrt{n}]$. If the coherence between the bases is 1, we have *maximal incoherence*. Compressive sampling is mainly concerned with low coherence pairs, which means that the pairs are largely incoherent with each other. Random matrices are known as largely incoherent with any fixed basis. Thus, CS framework utilizes random matrices as a sensing matrix for compressive sampling.