UNIVERSIDADE REGIONAL DE BLUMENAU

CENTRO DE CIÊNCIAS EXATAS E NATURAIS

CURSO DE CIÊNCIA DA COMPUTAÇÃO – BACHARELADO

CARDIOREADER: SISTEMA DE IDENTIFICAÇÃO DE BATIMENTOS CARDÍACOS

ANDERSON MORDHORST

BLUMENAU
2013

2013/2-01

ANDERSON MORDHORST

CARDIOREADER: SISTEMA DE IDENTIFICAÇÃO DE BATIMENTOS CARDÍACOS

Trabalho de Conclusão de Curso submetido à Universidade Regional de Blumenau para a obtenção dos créditos na disciplina Trabalho de Conclusão de Curso II do curso de Ciência da Computação — Bacharelado.

Prof. Aurélio Faustino Hoppe, Mestre - Orientador

BLUMENAU
2013

2013/2-01

CARDIOREADER: SISTEMA DE IDENTIFICAÇÃO DE BATIMENTOS CARDÍACOS

Por

ANDERSON MORDHORST

Trabalho aprovado para obtenção dos créditos na disciplina de Trabalho de Conclusão de Curso II, pela banca examinadora formada por:

Presidente:	Prof. Aurélio Faustino Hoppe, Mestre – Orientador, FURB
Membro:	Prof. Dalton Solano dos Reis, Mestre – FURB
Membro:	Prof. Francisco Adell Péricas, Mestre – FURB

Blumenau, 13 de dezembro de 2013.

Dedico este trabalho à minha família e amigos, que sempre ofereceram apoio e motivação para a conclusão deste trabalho.

AGRADECIMENTOS

À minha família, por toda a motivação e incentivo ao longo de toda a minha vida. Agradeço em especial à minha mãe Rosângela por todo o seu esforço em garantir um ensino de qualidade para a minha pessoa.

Aos meus amigos e professores, por fornecerem o conhecimento que ajudou a conclusão desse trabalho.

Ao meu orientador, Aurélio Faustino Hoppe, por ter acreditado na viabilidade e conclusão deste trabalho e na minha capacidade de fazê-lo, prestando todo o auxílio que necessitei durante todo o desenvolvimento deste projeto.

A mente que se abre a uma nova idéia jamais voltará ao seu tamanho original.

Albert Einstein

RESUMO

Este trabalho apresenta um sistema de identificação de batimentos cardíacos utilizando um aplicativo *desktop* como emissor de dados e um Android como receptor, conectados através de uma comunicação *bluetooth*. O sistema proposto consegue filtrar os sinais eletrocardiográficos usando algoritmos de filtro passa-alta e passa-baixa e identifica os batimentos cardíacos existentes através de algoritmos de limiar fixo e adaptativo. Os resultados obtidos neste trabalho demonstram que o sistema de identificação de batimentos cardíacos proposto pode ser aplicado em vários cenários diferentes.

Palavras-chave: Eletrocardiograma. Sinal vital. *Bluetooth*. Android. *Health care*.

ABSTRACT

This work presents a system for the identification of heartbeats using a desktop application as a data transmitter and an Android as a receiver, connected via bluetooth communication. The proposed system can filter electrocardiographic signals using algorithms high-pass filter and low-pass filter and identifies existing heartbeats using algorithms fixed and adaptive thresholding. The results of this study demonstrate that the proposed system for the identification of heartbeats can be applied in many different scenarios.

Key-words: Eletrocardiogram. Vital sign. Bluetooth. Android. Health care.

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

LISTA DE SIGLAS

ADT – *Android Development Tools*

ECG – Eletrocardiograma

MIT-BIH - *Massachussetts Institute of Technology – Beth Israel Hospital*

MLII – *Modified lead* II

OMAP - *Open Multimedia Application Platform*

RF – Requisito Funcional

RNF – Requisito Não Funcional

SMS - *Short Message Service*

SPP – *Serial Port Profile*

UML - *Unified Modeling Language*

SUMÁRIO

1 INTRODUÇÃO

De acordo com Correia (2008, p. 1), a área da saúde é muito grande e necessita de tecnologia para facilitar e agilizar o trabalho dos seus profissionais. A alta necessidade de informação, tanto sobre seus pacientes como sobre tratamentos e medicamentos, são vitais para que o profissional tome decisões no seu dia-a-dia. Contudo, como já é esperado, o volume que estas informações têm é muito grande. Assim sendo, a tecnologia da informação vem sendo cada vez mais inserida em todas as áreas da medicina, auxiliando em novas descobertas e em novas formas de oferecer uma assistência médica de melhor qualidade.

Segundo Amaral (2001, p. 113), para oferecer uma assistência médica de qualidade faz-se necessário o uso do atendimento médico domiciliar (*Home Health Care*). Nesta alternativa de atendimento o profissional da área da saúde atende o paciente em casa, provendo uma redução de custos, diminuindo o risco de infecções hospitalares e garantindo um ambiente mais confortável ao paciente.

Em um ambiente *Home Health Care* são levantadas informações do paciente, tais como pressão sanguínea, peso, taxa de açúcar e batimentos cardíacos. Segundo Murad (2009, p. 1), atualmente a monitoração de sinais biológicos produzidos pelo ser humano tem sido uma preocupação constante da medicina moderna, aproximando a medicina e a computação de forma a produzir novos métodos de diagnóstico e tratamento. Dentre os métodos mais usados estão os diagnósticos baseados em eletrocardiograma, um exame de baixo custo não-invasivo.

Segundo Carvalho (2005, p. 1), um sistema de monitoramento remoto dos sinais vitais diminui os gastos médicos e possibilita ao profissional fornecer um diagnóstico mais preciso. Isso acontece porque é possível acompanhar os dados históricos do paciente e não somente os dados coletados no momento da consulta.

Diante do cenário descrito, este trabalho apresenta um sistema de identificação de batimentos cardíacos divido em dois módulos: emissor e receptor. O módulo emissor consiste em um aplicativo desenvolvido em Java sob o sistema operacional Windows 7 responsável por carregar informações da base de dados MIT-BIH *Arrhytmia Database* (Physionet, 2012) e transmití-las usando comunicação *bluetooth*. O módulo receptor consiste em um aplicativo Android responsável por analisar as informações recebidas, filtrar o sinal recebido e identificar os batimentos cardíacos existentes no sinal eletrocardiográfico.

1.1 OBJETIVOS DO TRABALHO

O objetivo deste trabalho é implementar um sistema de identificação de batimentos cardíacos a partir de sinais eletrocardiográficos.

Os objetivos específicos do trabalho são:

a) disponibilizar a comunicação *bluetooth* entre o dispositivo emissor de dados médicos e o dispositivo receptor usando o protocolo SPP (*Serial Port Profile*);
b) reduzir os ruídos existentes no sinal aplicando filtros passa-alta e passa-baixa;
c) identificar a quantidade de batimentos cardíacos existentes em um sinal eletrocardiográfico;
d) disponibilizar uma interface com informações do usuário sobre sua saúde cardíaca utilizando sistema operacional Android.

1.2 ESTRUTURA DO TRABALHO

Estre trabalho está subdividido em quatro capítulos, sendo o primeiro a apresentação da justificativa para seu desenvolvimento e seus objetivos.

O segundo capítulo aborda a fundamentação teórica, explicando os conceitos gerais sobre o funcionamento do coração, arritmias cardíacas e funcionamento do exame de eletrocardiograma.

O terceiro capítulo trata do desenvolvimento do sistema de identificação de batimentos cardíacos, onde são listados seus requisitos, bem como sua especificação através de diagramas de caso de uso, diagramas de classe e de atividade. Também é descrita a implementação, apresentando técnicas e ferramentas utilizadas, operacionalidade e por fim, apresenta-se os resultados obtidos.

As conclusões deste projeto e sugestões para trabalhos futuros são apresentadas no quarto capítulo.

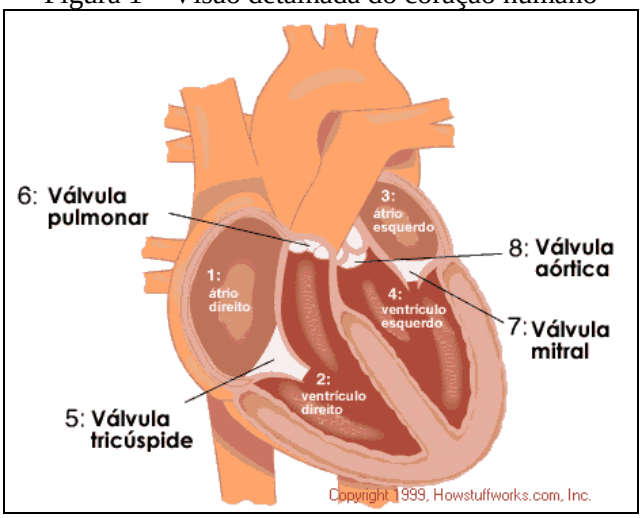
2 FUNDAMENTAÇÃO TEÓRICA

Na seção 2.1 é apresentado o funcionamento do coração, útil para o entendimento das seções seguintes. A seção 2.2 apresenta o funcionamento do exame de eletrocardiograma, técnica adotada para análise do coração humano. A seção 2.3 apresenta as arritmias cardíacas mais comuns, a seção 2.4 apresenta técnicas de identificação de batimentos cardíacos, usadas no sinal gerado pelo exame de eletrocardiograma. Por fim, a seção 2.5 apresenta os trabalhos correlatos relacionados à sistemas de identificação de batimentos cardíacos.

2.1 FUNCIONAMENTO DO CORAÇÃO

Segundo Sampaio (2011, p. 28), o coração é um órgão muscular que tem como função principal impulsionar o sangue através dos vasos sanguíneos, fazendo com que este circule por todo o organismo. O coração é formado por quatro câmaras internas (Figura 1): dois átrios e dois ventrículos, sendo que cada um possui em sua saída uma válvula unidirecional responsável pela circulação de sangue em apenas uma direção (SAMPAIO, 2011, p. 28).

Figura 1 – Visão detalhada do coração humano



Fonte: Sampaio (2011, p. 28).

Basicamente o funcionamento do coração se dá por dois movimentos: sístole e diástole. A sístole consiste em uma contração dos átrios, enviando o sangue aos ventrículos. Em seguida há a contração dos ventrículos, expulsando o sangue do coração e enviando-o para o corpo. Após a sístole ocorre a diástole, que consiste no relaxamento das quatro câmaras, ocasionando na entrada de sangue ao coração. Estes dois movimentos formam o ciclo cardíaco que é controlado por impulsos elétricos que originam do sistema nervoso.

Segundo Vier (2008, p. 19) a atividade elétrica do coração se encontra nas células miocárdicas. Quando as fibras do músculo do coração se encontram em repouso ou relaxamento, estas levam a carga negativa no interior da célula e a carga positiva no seu lado

externo. Este processo denomina-se polarização. A troca de carga negativa por positiva denomina-se despolização e seu efeito é a contração da fibra muscular do coração. Para se registrar tais atividades elétricas, utiliza-se o eletrocardiograma (VIER, 2008, p. 19).

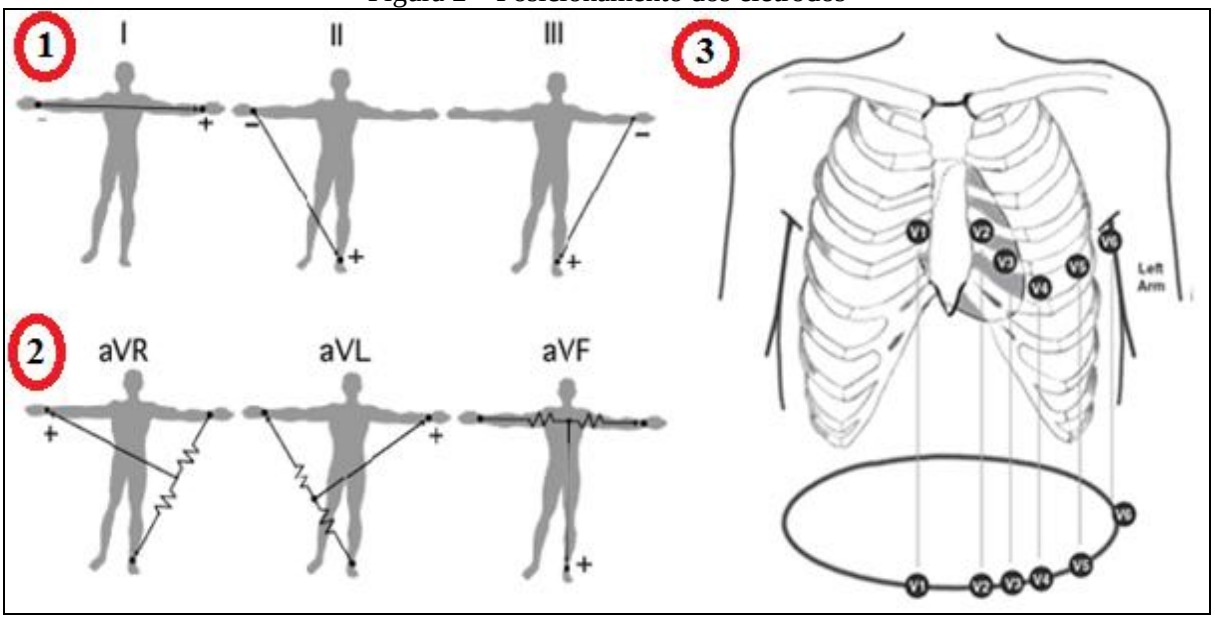
2.2 ELETROCARDIOGRAMA

Segundo Azevedo (1999, p. 13) o eletrocardiograma (ECG) é um procedimento técnico não-invasivo, reproduzível e de baixo custo muito utilizado para efetuar a avaliação da função cardiovascular. É com ele que se pode diagnosticar as moléstias próprias do coração, congênitas ou adquiridas.

Segundo Sampaio (2011, p. 29), o ECG é na verdade, uma medida dos pulsos elétricos que controlam o coração e não dos batimentos do coração propriamente ditos. Este exame consiste no levantamento de todas as atividades elétricas oriundas do coração, sendo captados por eletrodos espalhados pela superfície corporal do usuário. A captação dessa atividade elétrica é possível graças a constiuição do corpo humano, sendo 60% composto de água e por possuir sais minerais, combinação com excelente condução elétrica (AZEVEDO, 1999, p. 27).

Para a leitura correta do sinais vitais, os eletrodos do eletrocardiograma devem ser posicionados em locais específicos. Segundo Vier (2008, p. 21) o eletrocardiograma padrão dispõe de 12 derivações diferentes de posicionamento. A Figura 2 demonstra o posicionamento de cada eletrodo.

Figura 2 – Posicionamento dos eletrodos



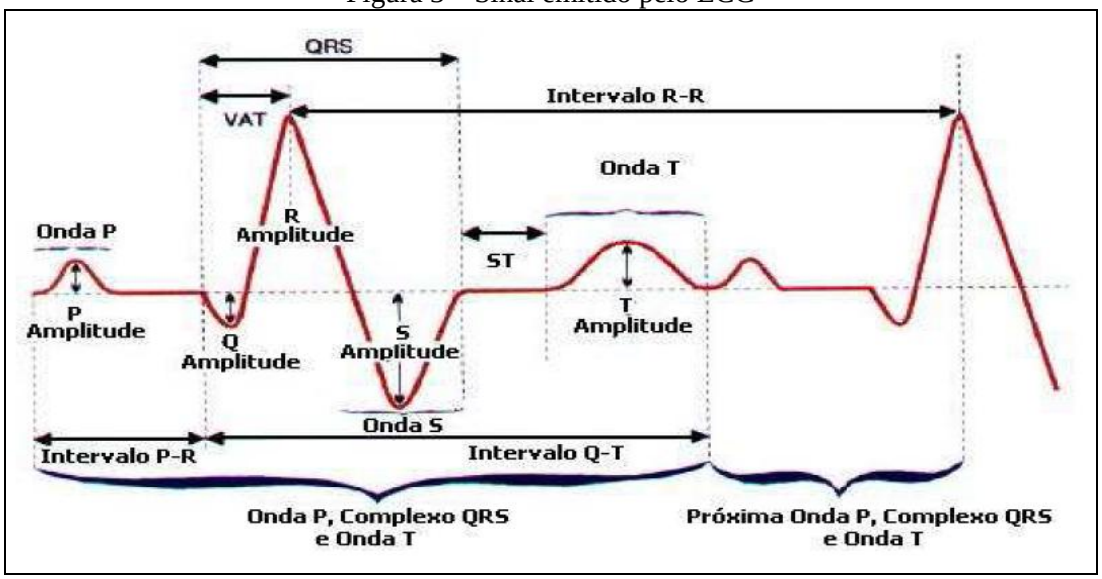
Fonte: adaptado de Aguiar (2006, p. 30).

Na Figura 2 pode-se visualizar que no peito do paciente são posicionados 6 eletrodos, nomeados de V1 a V6 (item 3). Estes eletrodos formam a derivação unipolar precordial de Wilson.

Segundo Azevedo (1999, p. 34) sendo o coração um órgão tridimensional, a atividade elétrica emitida pelo mesmo seria melhor analizada se captados em pelo menos dois planos. Aguiar (2006, p. 32) afirma que a derivação unipolar precordial de Wilson não registra apenas o sinal emitido pela área cardíaca próxima aos eletrodos, mas sim o evento do ciclo cardíaco como um todo, captando assim o evento em toda a sua extensão (razão ao qual levou o uso desta derivação neste projeto).

Após o posicionamento dos eletrodos, inicia-se a leitura e interpretação dos dados obtidos pela atividade cardíaca do paciente. A Figura 3 demonstra o gráfico gerado pela leitura cardíaca.

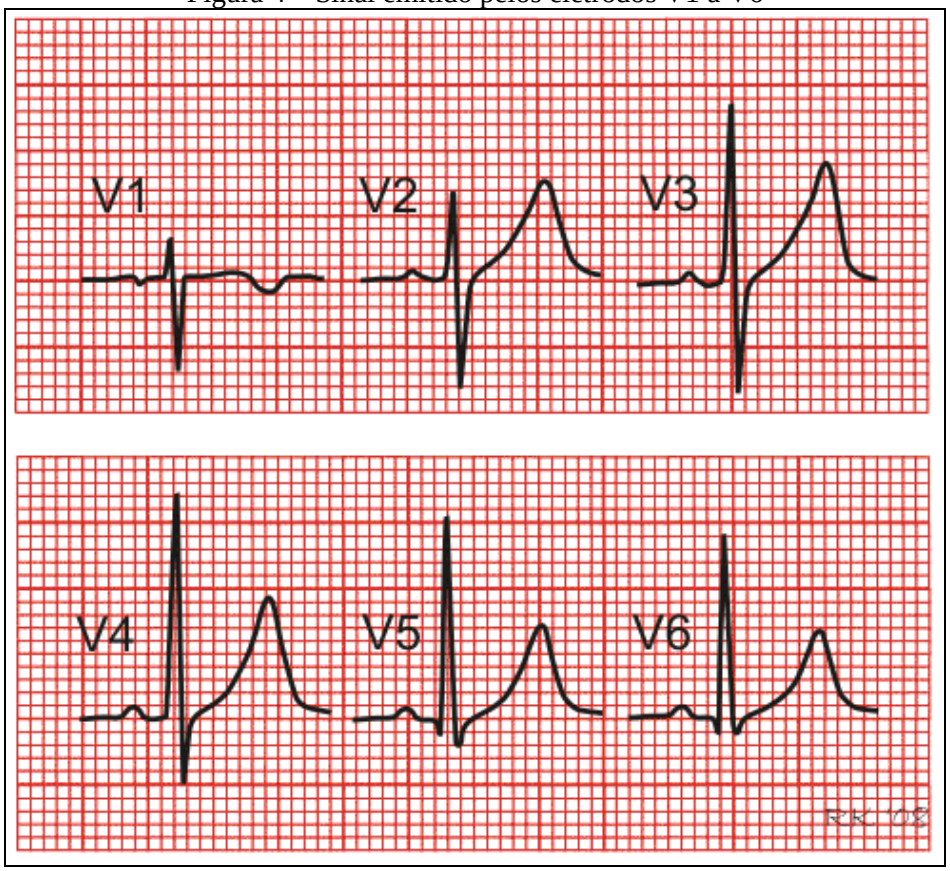
Figura 3 – Sinal emitido pelo ECG



Fonte: Sampaio (2011, p. 29).

O sinal demonstrado na Figura 3 é composto por várias ondas. A onda inicial P é gerada na propagação da atividade elétrica nos átrios, iniciando a contração do coração (início da sístole). Em seguida são geradas ondas Q, R e S, resultantes da atividade elétrica nos ventrículos e sua contração (fim da sístole). Por fim, a onda T é gerada com o relaxamento das quatro cavidades do coração (diástole). Segundo Klabunde (2008), a leitura através da derivação unipolar precordial de Wilson gera um gráfico diferente para cada eletrodo. A Figura 4 demonstra cada gráfico gerado para os eletrodos V1 à V6.

Figura 4 – Sinal emitido pelos eletrodos V1 à V6

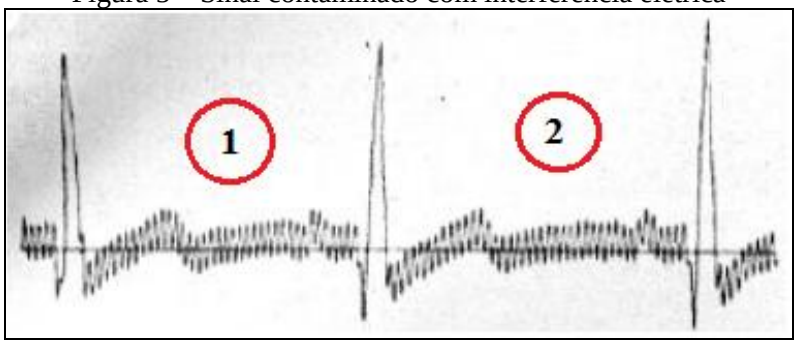


Fonte: Klabunde (2008).

A Figura 4 demonstra os gráficos gerados pelos eletrodos de V1 à V6. É possível visualizar a diferença nas ondas P, Q, R, S e T em cada gráfico.

Durante a captação dos sinais vitais, segundo Born (2000, p. 12), alguns desses sinais podem não representar o sinal vital propriamente dito. Estes sinais são denominados artefatos ou ruídos. Um dos mais comuns ruídos presentes no sinal eletrocardiográfico é a interferência elétrica, com frequência em torno de 60 Hz (Hertz). Essa interferência pode ser causada por indução magnética ou efeito eletrostático (BORN, 2000, p. 12). A Figura 5 demonstra um sinal de eletrocardiograma contaminado com tal interferência. Na Figura 5 pode-se visualizar a interferência causada por atividade elétrica (itens 1 e 2).

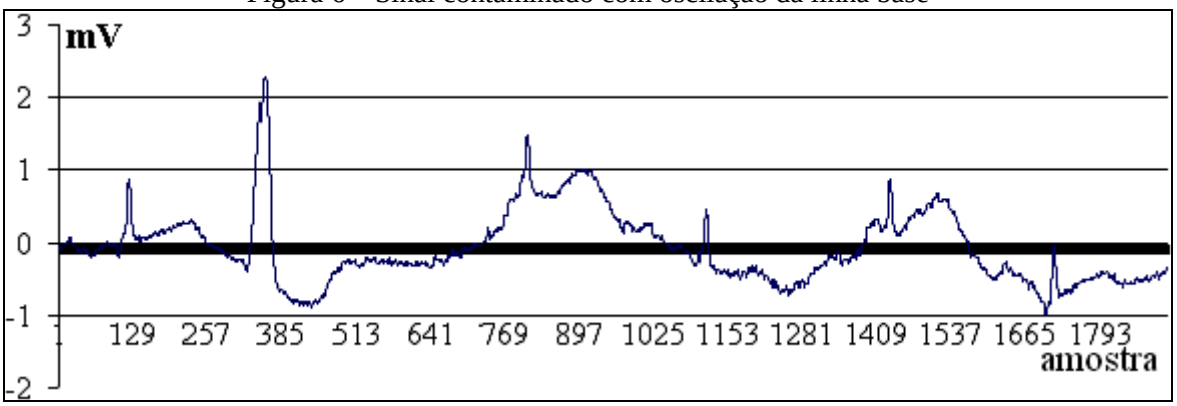
Figura 5 – Sinal contaminado com interferência elétrica



Fonte: adaptado de Born (2000, p. 12).

Outro ruído muito comum em sinais eletrocardiográficos é a oscilação da linha base do sinal vital. Segundo Volpato (2005, p. 14) este tipo de interferência pode ser causado devido a respiração ou movimentos musculares do paciente, sendo compreendida em uma faixa entre 0 Hz e 5 Hz. A Figura 6 demonstra um sinal contaminado com tal interferência.

Figura 6 – Sinal contaminado com oscilação da linha base



Fonte: Volpato (2005, p. 14).

Pode-se visualizar na Figura 6 que a onda do sinal vital oscila em torno da linha base (linha 0), com variação entre 0.5 mV (Milivolts) e -0.5 mV.

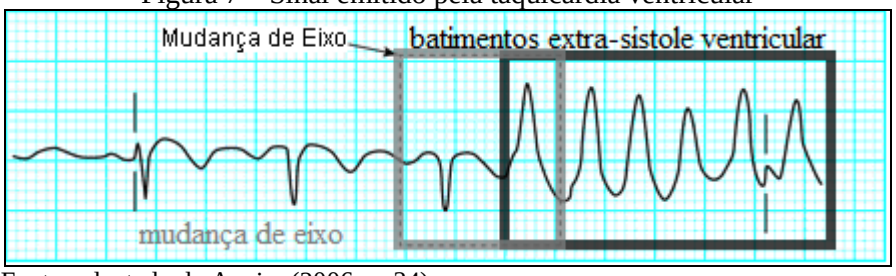
2.3 ARRITMIAS CARDÍACAS

Segundo o Instituto de Arritmias Cardíacas (2009) as arritmias cardíacas são perturbações que alteram a frequência cardíaca e o ritmo dos batimentos do coração. As arritmias cardíacas podem levar a morte e constituir por isso um caso de emergência médica. Estas podem ser classificadas pelo ritmo excessivamente rápido (taquicardia), lentos (bradicardia) ou irregulares. Nas subseções a seguir são descritas duas das principais arritmias cardíacas

2.3.1 Taquicardia Ventricular

De acordo com Aguiar (2006, p. 34) a taquicardia ventricular ocorre tanto em pessoas com doenças no músculo do coração (infartados, doença de Chagas, coração dilatado ou hipertrofiado) quanto em pessoas normais, sendo classificadas como idiopáticas. Essa arritmia cardíaca é definida quando ocorre três ou mais batimentos extra-sístole ventricular (batimentos cuja sequência de ativação cardíaca é alterada). A Figura 7 demonstra um sinal eletrocardiográfico em que pode-se visualizar a ocorrência da taquicardia ventricular.

Figura 7 – Sinal emitido pela taquicardia ventricular



Fonte: adaptado de Aguiar (2006, p. 34)

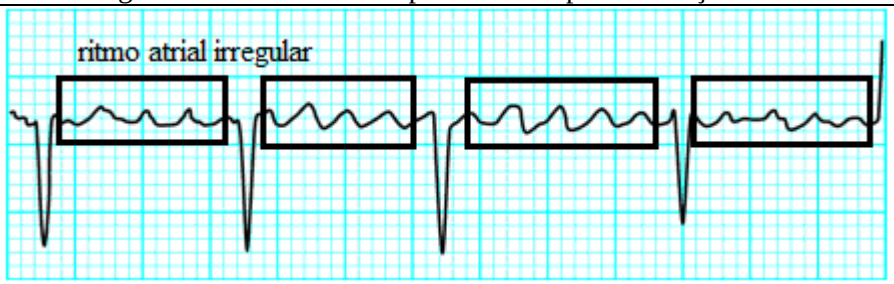
Na Figura 7 pode-se observar o momento em que há a mudança de eixo do sinal elétrico, seguido dos batimentos extra-sístole ventricular, caracterizando uma taquicardia ventricular.

2.3.2 Flutter e Fibrilação Atrial

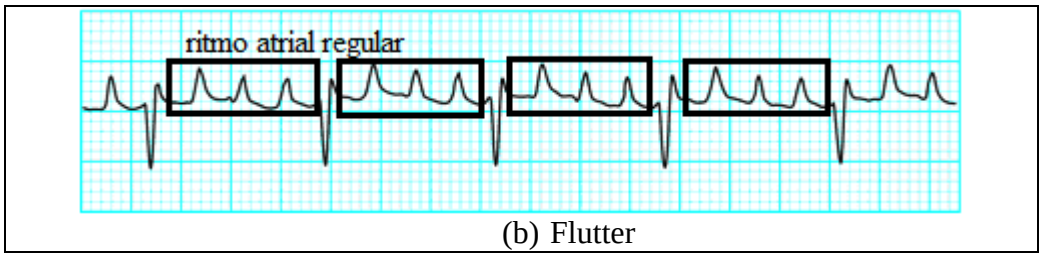
A fibrilação atrial e o flutter, de acordo com Aguiar (2006, p. 34) são arritmias que tem como características padrões de descargas elétricas muito rápidas resultando na contração dos átrios e dos ventrículos mais rápida do que a normal, ocasionando na ineficácia do batimento cardíaco.

Ainda segundo Aguiar (2006, p. 34) essas contrações são tão rápidas que fazem as paredes atriais tremularem, impedindo o bombeamento eficaz do sangue. A diferença entre o flutter e a fibrilação atrial está no ritmo em que ambos acontecem, sendo que o ritmo do flutter é regular. A Figura 8 demonstra o momento em que acontece ambas as arritmias.

Figura 8 – Sinais emitidos pelo flutter e pela fibrilação atrial



(a) Fibrilação atrial



(b) Flutter

Fonte: adaptado de Aguiar (2006, p. 34).

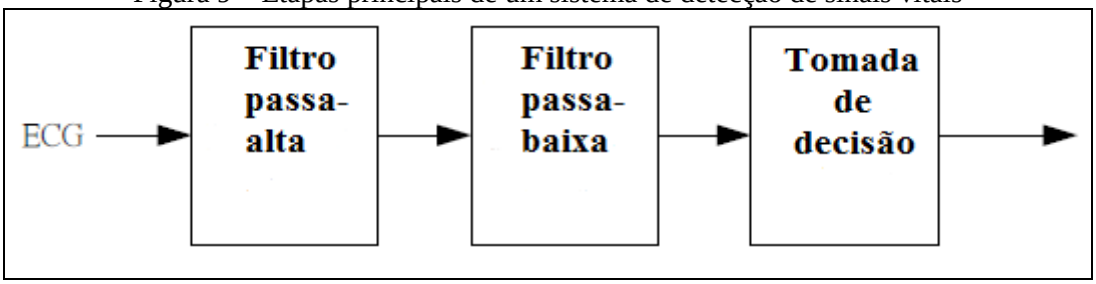
Na Figura 8 pode-se visualizar o momento em que ocorre a fibrilação atrial (Figura 8a) e o momento em que ocorre o flutter (Figura 8b).

2.4 PROCESSO DE IDENTIFICAÇÃO DOS BATIMENTOS CARDÍACOS

Segundo Chen e Chen (2003, p. 585) a detecção de batimentos cardíacos serve como fundamento para uma variedade de algoritmos de análise de sinais vitais automatizados. Esta técnica é muito importante para a análise dos sinais eletrocardiográficos, tendo em vista que tal informação possui variação de tempo e pode estar corrompido por sinais indesejados (ruídos), itens que dificultam a análise correta do ciclo cardíaco.

De acordo com Chen e Chen (2003, p. 585) a maioria dos sistemas de detecção de batimentos cardíacos são compostos por 3 etapas: aplicação de um filtro passa-alta, aplicação de um filtro passa-baixa e a tomada de decisão. A Figura 9 demonstra as 3 etapas.

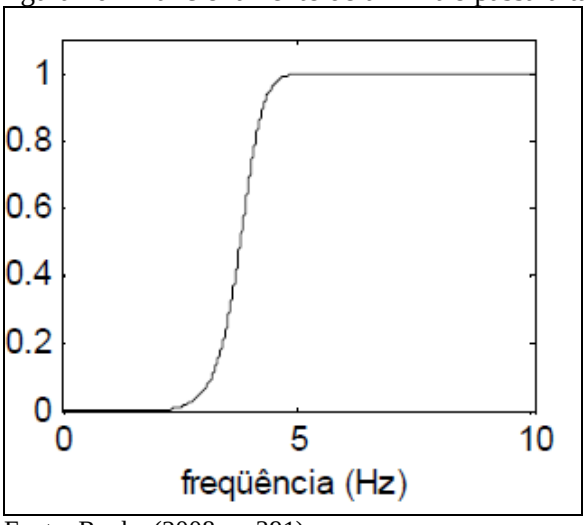
Figura 9 – Etapas principais de um sistema de detecção de sinais vitais



Fonte: adaptado de Chen e Chen (2003, p. 585).

Conforme visto na Figura 9, a primeira etapa consiste na aplicação do filtro passa-alta. Segundo Volpato (2005, p. 15) os filtros passa-alta funcionam da seguinte forma: seleciona-se uma determinada faixa de frequência para corte. Frequências inferiores a selecionada são eliminadas ou atenuadas. Para Chen e Chen (2003, p. 585), o filtro passa-alta foi utilizado para suprimir as ondas P e T (indesejadas) e para remoção da oscilação das ondas sobre a linha base. A Figura 10 demonstra o funcionamento de um filtro passa-alta.

Figura 10 – Funcionamento de um filtro passa-alta

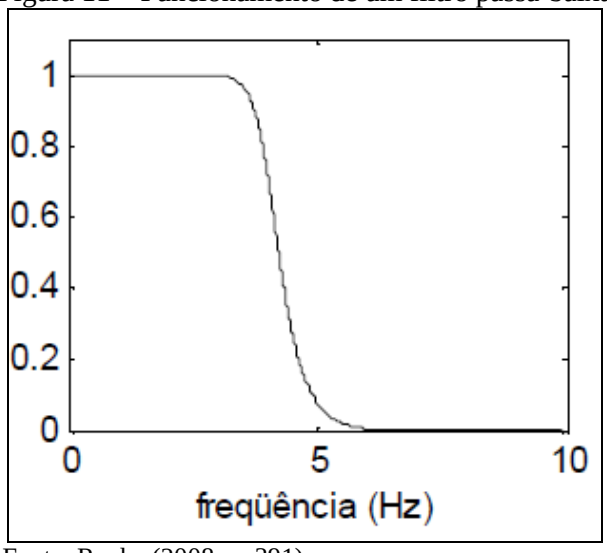


Fonte: Rocha (2008, p. 391).

Conforme visto na Figura 10, valores inferiores a 5 Hz (exemplo) são atenuados, permitindo apenas a passagem de valores acima desta frequência.

A segunda etapa consiste na aplicação do filtro passa-baixa. Segundo Volpato (2005, p. 15) os filtros passa-baixa funcionam selecionando uma determinada frequência de corte. Qualquer sinal acima da frequência escolhida desta forma é atenuado ou eliminado. Para Chen e Chen (2003, p. 585), o filtro passa-baixa foi utilizado para atenuar um intervalo de frequência cardíaca dentro da faixa escolhida (neste caso, ruídos causados por interferência elétrica), não alterando nenhum sinal fora desse intervalo. A Figura 11 demonstra o funcionamento de um filtro passa-baixa.

Figura 11 – Funcionamento de um filtro passa-baixa



Fonte: Rocha (2008, p. 391).

Conforme visto na Figura 11, a frequência de corte escolhida possui 5 Hz. Valores acima dessa frequência são atenuados.

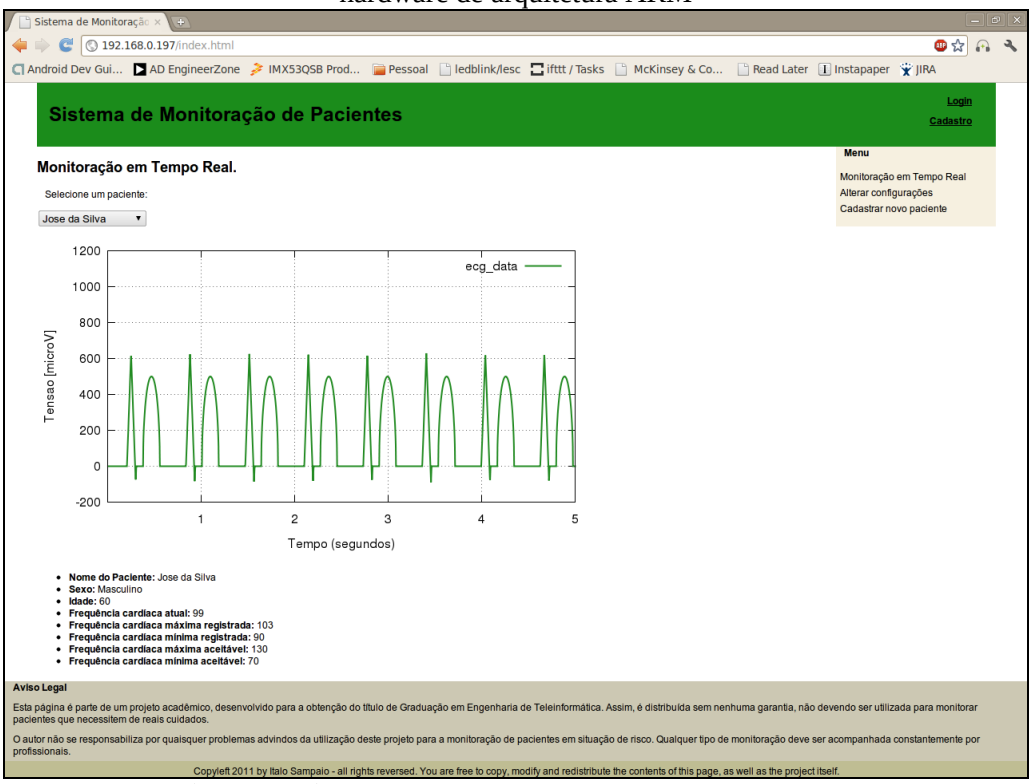
A terceira etapa a ser aplicada para detecção de batimentos cardíacos é a tomada de decisão. Chen e Chen (2003, p. 585) adotaram um limiar adaptativo para essa etapa do algoritmo. Durante a leitura do sinal cardíaco filtrado há a busca pelo maior valor encontrando dentro de uma faixa, estabelecida em testes. No momento em que se encontra esse valor, o limiar é atualizado.

2.5 TRABALHOS CORRELATOS

Esta seção tem como objetivo investigar na literatura trabalhos relacionados sobre coleta de informações cardiológicas, descrevendo equipamentos e técnicas utilizadas. Dentre os trabalhos selecionados estão o "Sistema de monitoramento remoto de pacientes implementado em hardware de arquitetura ARM" (SAMPAIO, 2011), "Sistema de sensorização móvel e controle baseado em ZigBee para bicicletas elétricas" (FERREIRA et al., 2012) e "Sistema universal de monitoramento da saúde com dispositivos conectados" (BHATIA et al., 2010).

O trabalho desenvolvido por Sampaio (2011) é divido em três partes. A primeira é composta por um emissor de sinais vitais que utiliza comunicação *Bluetooth* para envio de dados. A segunda tem como parte integrante um microcontrolador Freescale iMX53, que possui um processador *core* ARM Cortex-M3, sendo responsável pela coleta, processamento, armazenagem e análise dos dados obtidos. A última parte consiste na criação de um sistema Web para interação com o usuário (Figura 12). Caso haja a necessidade, o sistema permite o envio de mensagens de alerta usando *Short Message Service* (SMS) a um celular previamente cadastrado.

Figura 12 – Página Web gerada pelo sistema de monitoramento remoto de pacientes implementado em hardware de arquitetura ARM

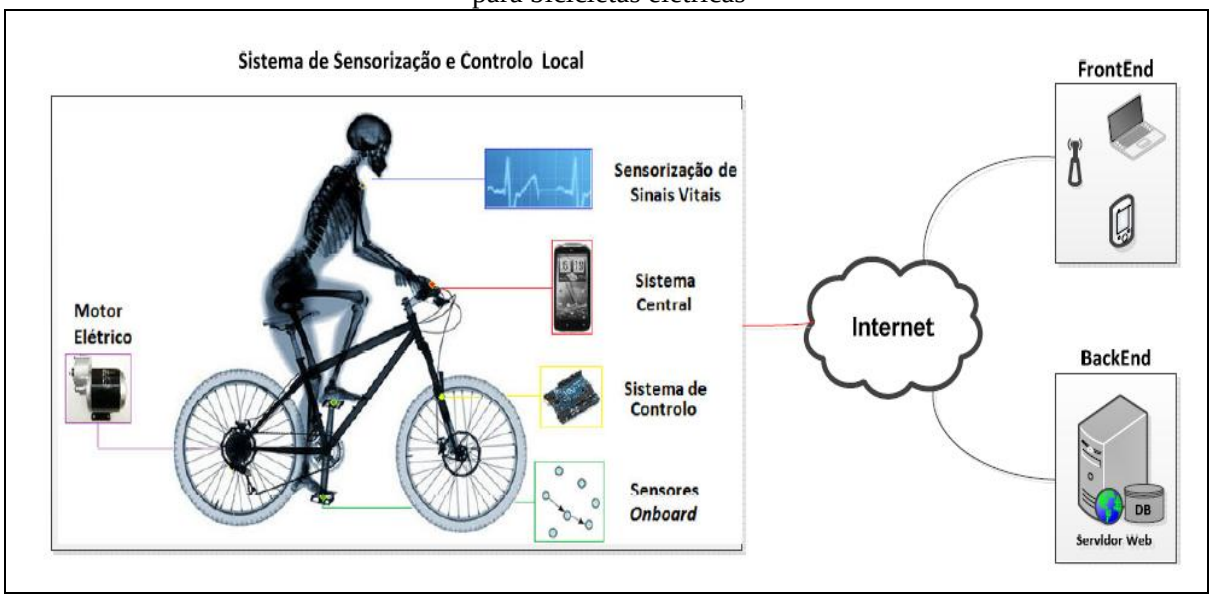


Fonte: Sampaio (2011, p. 48).

Como o tempo de atendimento a uma pessoa com parada cardíaca (sendo o cenário mais agravante para aplicação) é da grandeza de minutos, o resultado de menos de um segundo obtido pelo processamento, análise, geração de conteúdo Web e emissão de alerta via SMS foi considerado satisfatório (Sampaio, 2011).

Ferreira et al. (2012) desenvolveram um sistema para coletar dados vitais enquanto o usuário se exercita em uma bicicleta, fornecendo os batimentos cardíacos e o nível de saturação de oxigênio. Em conjunto a estes dados, os sensores espalhados pela bicicleta coletam os dados pertinentes à movimentação da mesma. Esta massa de informação é transmitida através de uma rede sem fios baseada na norma ZigBee, utilizando-se de um dispositivo Android como receptor, conforme mostra a Figura 13.

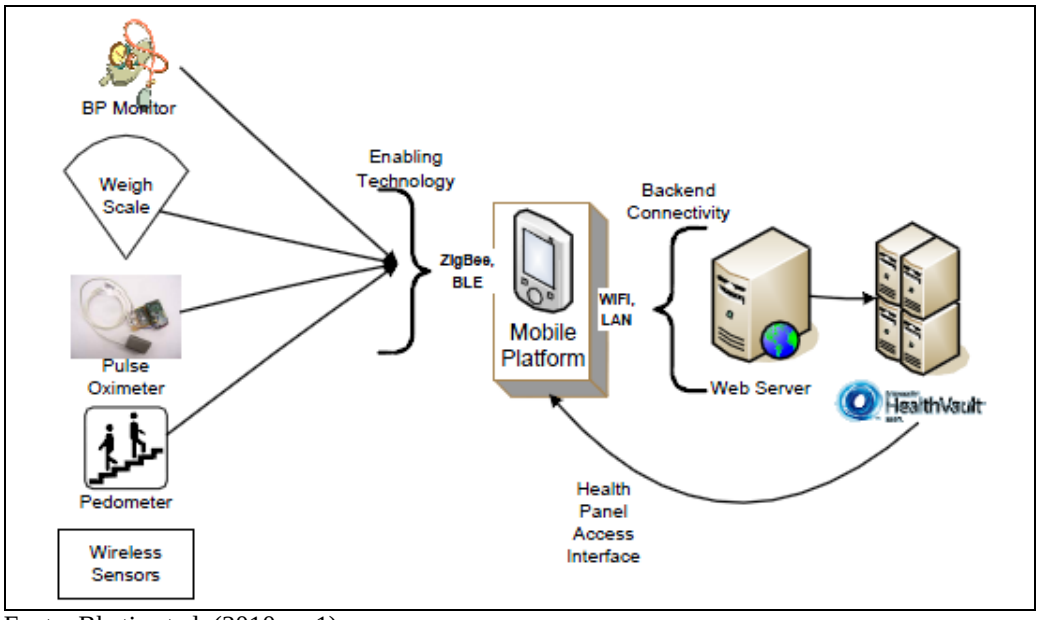
Figura 13 – Arquitetura conceitual do sistema de sensorização móvel e controle baseado em ZigBee para bicicletas elétricas



Fonte: Ferreira et al. (2012, p. 2).

O sistema idealizado por Bhatia et al. (2010) é composto por quatro partes (Figura 14). A primeira parte é composta por sensores vitais responsáveis pela coleta de dados médicos (pressão sanguínea e eletrocardiograma). Tais sensores usam comunicação *Bluetooth* e norma ZigBee para enviar as informações obtidas. A segunda parte consiste em um dispositivo *Open Multimedia Application Platform* (OMAP), representado por um dispositivo móvel. Sua função é armazenar toda a informação recebida localmente. A terceira parte deste trabalho consiste em um servidor Web responsável pela análise e processamento dos dados obtidos através de conexão WiFi/LAN efetuada pelo dispositivo OMAP. A quarta e última parte consiste em um banco de dados Microsoft HealthVault para armazenagem dos dados médicos obtidos.

Figura 14 – Arquitetura conceitual do sistema universal de monitoramento da saúde com dispositivos conectados



Fonte: Bhatia et al. (2010, p. 1).

O Quadro 1 apresenta de forma comparativa as características dos trabalhos apresentados nessa seção.

Quadro 1 - Características dos trabalhos relacionados

Características / trabalhos relacionados	Bhatia et al. (2010)	Sampaio (2011)	Ferreira et al. (2012)
comunicação utilizada	Bluetooth	Bluetooth	Bluetooth
possui interação com dispositivo móvel	Sim	Sim	Sim
possui interface Web	Sim	Sim	Sim
cenário utilizado para aplicação do trabalho	Genérico	Residência	Bicicleta

A partir do Quadro 1, é possível identificar que ambos os trabalhos correlatos trabalham com uma estrutura semelhante, sendo aplicada em cenários diferentes. Bhatia et al. (2010) propõe um sistema padrão de monitoramento de sinais vitais, sendo usado inclusive por Sampaio (2011). Ferreira et al. (2012) utiliza um cenário onde o usuário fornece informações de seus sinais vitais enquanto usa uma bicicleta. Sampaio (2011) utiliza um cenário residencial com o objetivo de fornecer uma melhor qualidade de vida durante o tratamento do usuário e diminuindo a superlotação dos hospitais (SAMPAIO, 2011, p. 26).

3 DESENVOLVIMENTO

Neste capítulo serão apresentadas as etapas do desenvolvimento do sistema de identificação de batimentos cardíacos. Na seção 3.1 são enumerados os principais requisitos do projeto a ser desenvolvido. A seção 3.2 apresenta sua especificação, a seção 3.3 apresenta detalhes da implementação e por fim, a seção 3.4 apresenta os testes efetuados no projeto bem como os resultados obtidos.

3.1 REQUISITOS PRINCIPAIS DO PROBLEMA A SER TRABALHADO

O sistema de identificação de batimentos cardíacos deverá:

a) permitir que usuário selecione o arquivo na base de dados MIT-BIH *Arrhytmia Database* (Physionet, 2012) (Requisito Funcional - RF);
b) filtrar o sinal eletrocardiográfico para eliminação de ruídos (RF);
c) calcular a quantidade de batimentos a partir do sinal filtrado (RF);
d) disponibilizar uma interface contendo informações pertinentes a saúde cardíaca do usuário (frequência cardíaca e data/hora em que foi calculado a esta informação) (RF);
e) disponibilizar comunicação *bluetooth* utilizando SPP (*Serial Port Profile*) (Requisito Não Funcional - RNF);
f) ser implementado utilizando a linguagem Java (RNF);
g) ser operado na plataforma Android (RNF);
h) ser implementado utilizando o ambiente de desenvolvimento Eclipse (RNF).

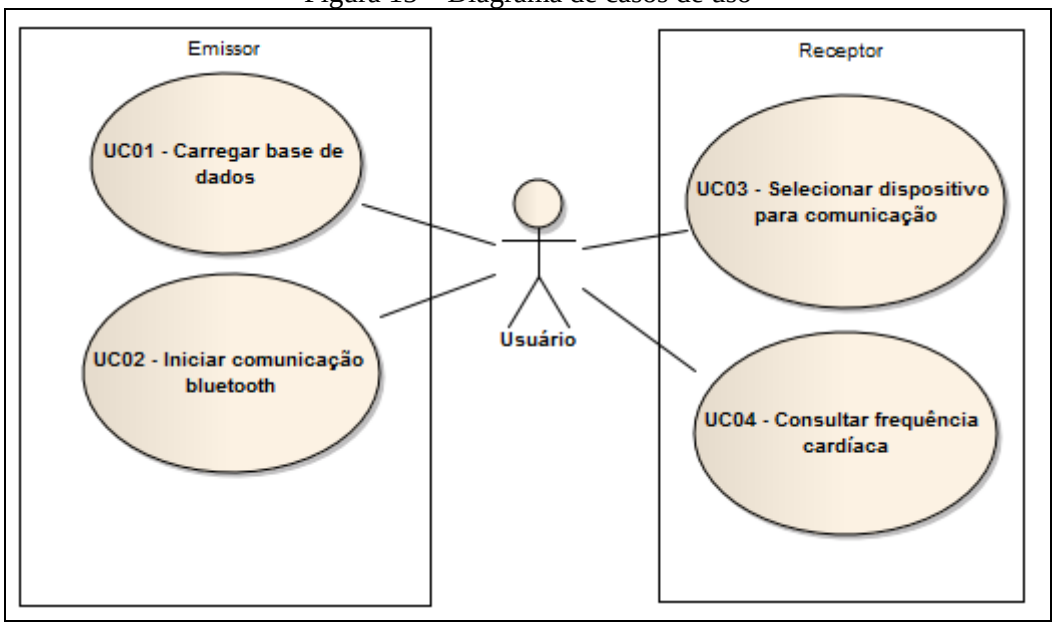
3.2 ESPECIFICAÇÃO

Nesta seção é apresentada a especificação do sistema de identificação de batimentos cardíacos utilizando diagramas da *Unified Modeling Language* (UML) criadas no ambiente Enterprise Architect 8.0. Foram utilizados os diagramas de classe, diagrama de casos de uso e diagrama de atividades para representar a metodologia utilizada no desenvolvimento do projeto.

3.2.1 Diagrama de casos de uso

A Figura 15 apresenta o diagrama de casos de uso com as ações necessárias para o funcionamento do sistema.

Figura 15 – Diagrama de casos de uso



Segue detalhamento dos casos de uso exibidos no diagrama da Figura 15:

a) `UC01 – Carregar base de dados`: permite que o usuário selecione as informações de um dos pacientes da base de dados *MIT-BIH Arrhythmia Database* (Physionet, 2012);

b) `UC02 – Iniciar comunicação bluetooth`: permite ao usuário estabelecer uma comunicação *bluetooth*;

c) `UC03 – Selecionar dispositivo para comunicação`: permite que o usuário selecione um dispositivo pareado para efetuar a leitura dos sinais vitais enviados pelo emissor;

d) `UC04 – Consultar frequência cardíaca`: permite que o usuário efetue a consulta da sua frequência cardíaca.

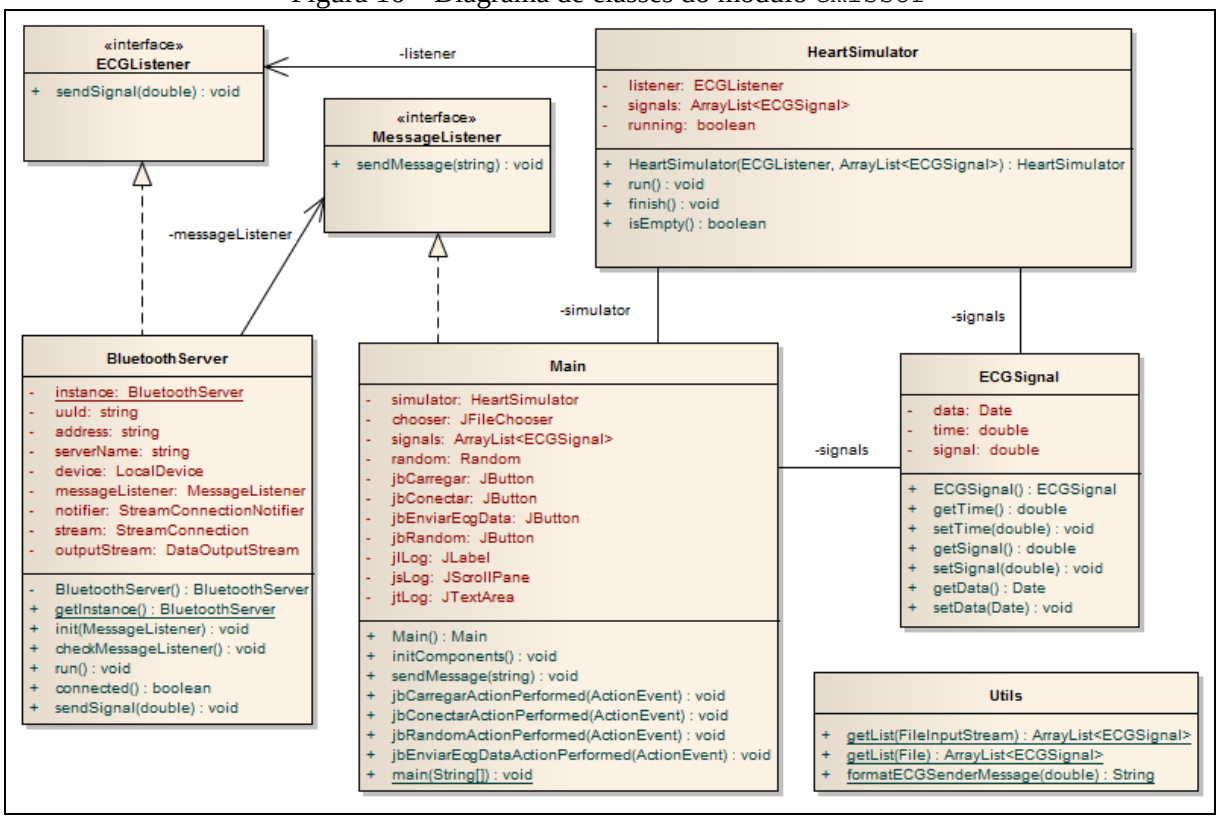
3.2.2 Diagrama de classes

Responsável por fornecer o panorama geral do relacionamento entre as classes do projeto, o diagrama de classes é apresentado em subseções para facilitar o entendimento da estrutura construída: diagrama de classes do módulo emissor e receptor

3.2.2.1 Diagrama de classes do módulo emissor

A Figura 16 apresenta o digrama de classes do módulo emissor.

Figura 16 – Diagrama de classes do módulo `emissor`

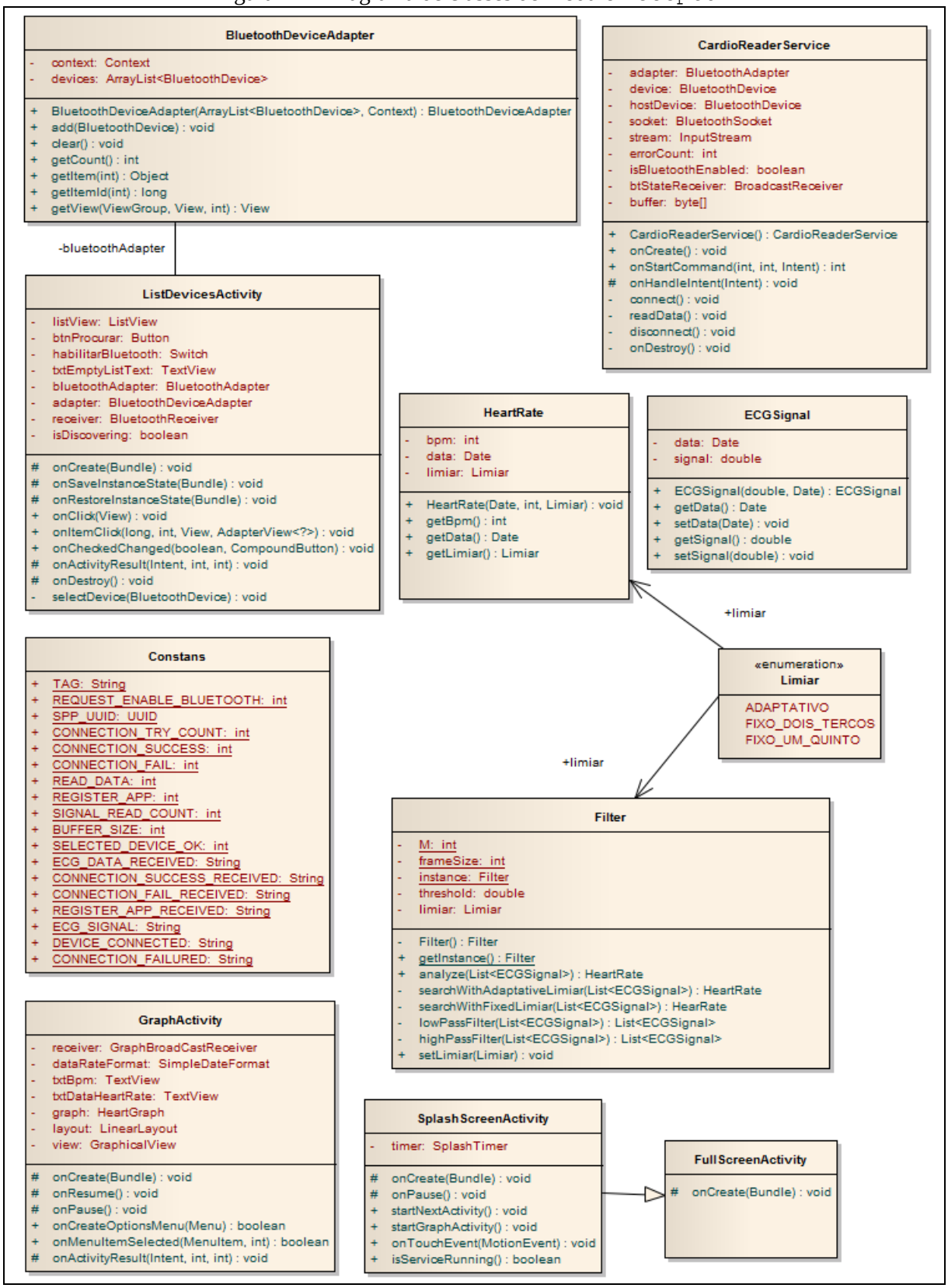


A classe `Main` é responsável pela interação com o usuário. É iniciada pelo método `main` e através dos métodos `jbCarregarActionPerformed`, `jbConectarActionPerformed` e `jbEnviarEcgDataActionPerformed` que carrega as informações da base de dados, efetua a conexão *bluetooth* e envia os dados carregados ao receptor respectivamente. Esta classe utiliza a classe `HeartSimulator` para a simulação do funcionamento de um coração humano através de uma `Thread` usando o método `run`. Para o envio dessas informações ao receptor, o módulo emissor utiliza a classe `BluetoothServer` que tem como função efetuar a conexão e enviar os dados usando comunicação *bluetooth*. A classe `ECGSignal` é responsável por armazenar o tempo e sinal dos batimentos cardíacos. As interfaces `MessageListener` e `ECGListener` são utilizadas pelo módulo emissor para promover a separação entre os pacotes `model` e `view` da aplicação. A classe `Utils` é utilizada no método `jbCarregarActionPerformed` da classe `Main` para carregar os dados da base de dados através do método `getList`.

3.2.2.2 Diagrama de classes do módulo receptor

A Figura 17 apresenta o diagrama de classes do módulo receptor.

Figura 17 – Diagrama de classes do módulo `receptor`

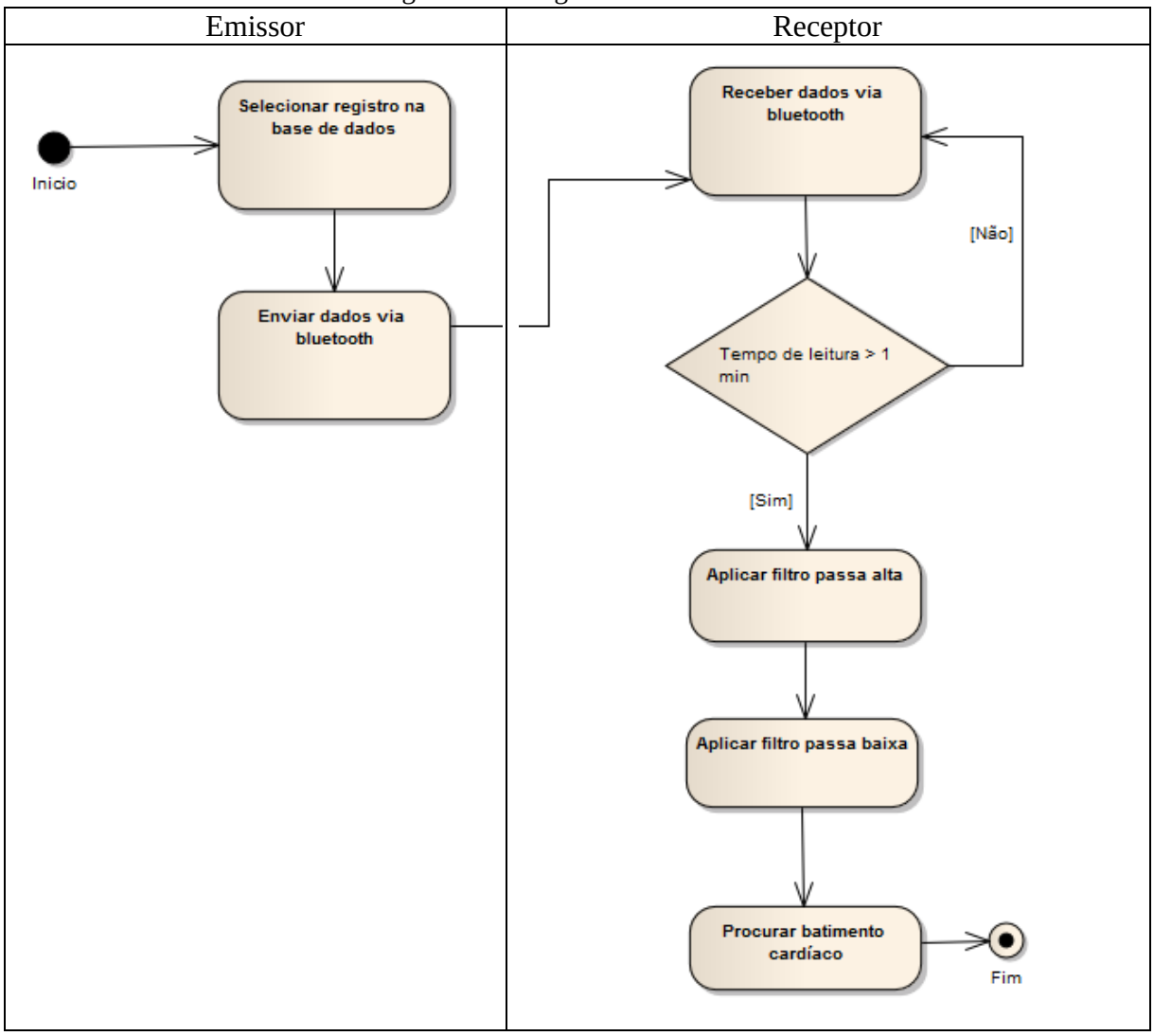


O módulo receptor é iniciado através da classe `SplashScreenActivity`. Esta classe herda a característica de uma janela *fullscreen* da classe `FullScreenActivity` através do método `onCreate`. Possui como atributo um *timer* de 5 segundos, sendo finalizado de duas maneiras: esgotando o tempo de vida estilpulado pelo *timer* ou quando o usuário toca na tela do *smartphone*, acionando o método `onTouchEvent`. Caso o usuário entre na aplicação pela segunda vez, a janela de *splash* não é visualizada, partindo assim para a próxima classe: `GraphActivity`. Esta classe é responsável pela visualização de toda a informação obtida pela aplicação: gráfico em tempo real, frequência cardíaca e a hora em que foi computado tal informação. É por ela que é chamado a classe `ListDevicesActivity` que tem como finalidade selecionar um dispositivo *bluetooth* pareado para conexão. Após efetuada a conexão, a classe `GraphActivity` se encarrega de iniciar a classe `CardioReaderService`, responsável pela recepção dos sinais vitais, filtragem e detecção de batimentos cardíacos. A classe `Constans` é reponsável por fornecer atributos estáticos usados por todas as classes do módulo receptor. A classe `Filter` é responsável por filtrar o sinal recebido pelo módulo através dos métodos `lowPassFilter` e `highPassFilter` e computar os batimentos cardíacos detectados através do método `searchWithAdaptativeLimiar` e `searchWithFixedLimiar`. A classe `ECGSignal` é responsável por armazenar o tempo e sinal dos batimentos cardíacos e a classe `HeartRate` é reponsável por armazenar a quantidade de batimentos lidos em um minuto e a data em que foi calculado essa informação.

3.2.3 Diagrama de atividades

O diagrama de atividades é responsável por representar os estados de uma computação. A Figura 18 apresenta os passos para funcionamento do sistema de identificação de batimentos cardíacos.

Figura 18 – Diagrama de atividades



Na Figura 18 pode-se observar que a primeira operação a ser efetuada é a seleção do registro na base de dados. Em seguida é efetuada a transmissão dos sinais vitais pelo módulo emissor. Depois desta etapa, o módulo receptor recebe os sinais vitais emitidos desta forma e, caso o tempo de leitura seja maior que um minuto, são efetuados os processos de filtragem, que compreendem a passagem de um filtro passa-alta e um filtro passa-baixa. Após essa etapa é possível procurar pelo batimentos cardíacos presentes no sinal filtrado.

3.3 IMPLEMENTAÇÃO

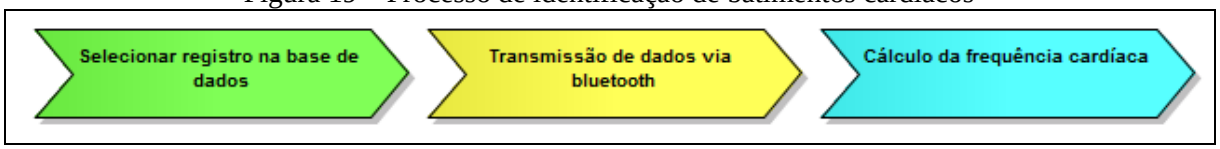
Esta seção descreve as técnicas e ferramentas utilizadas durante a implementação do sistema, assim como sua operacionalidade.

3.3.1 Técnicas e ferramentas utilizadas

O sistema de identificação de batimentos cardíacos foi desenvolvido usando a linguagem Java, comunicação *wireless bluetooth*, ambientes de desenvolvimento NetBeans 7.0.1, Eclipse ADT e sistemas operacionais Windows 7 e Android. Para efetuar a comunição *bluetooth* entre os sistemas operacionais foi utilizado a biblioteca BlueCove, versão 2.1.0. Esta biblioteca é escrita em Java e possui suporte para os sistemas operacionais Windows XP ou superior, Mac OS X e Linux.

O processo para identificação de batimentos cardíacos compreende em resumo, conforme exibido na Figura 19, a seleção de registro na base de dados, a transmissão desta informação através da comunicação *bluetooth*, e o cálculo da frequência cardíaca do usuário.

Figura 19 – Processo de identificação de batimentos cardíacos



Nas próximas seções são descritos as etapas da Figura 19, apresentando seus métodos e respectivos resultados.

3.3.1.1 Seleção de registro na base de dados

Conforme Figura 19, a primeira etapa do processo de identificação de batimentos cardíacos é a seleção de registro na base de dados. A base escolhida para uso no neste projeto foi a MIT-BIH *Arrhytmia Database* (Physionet, 2012). Segundo Moody e Mark (2001, p. 46), a base de dados é formada por 48 amostras de 30 minutos cada, coletadas de 47 pacientes (sendo que um dos paciente possui duas amostras na base).

O arquivo da base de dados é composto por três campos: tempo do batimento (medido em segundos) e dois campos de sinal captados de eletrodos diferentes (medido em milivolts). Segundo Moody e Mark (2001, p. 46) o primeiro campo de sinal captado é o MLII e o segundo campo varia entre V1, V2, V4 ou V5. A Figura 20 apresenta o formato do arquivo.

Figura 20 – Formato do arquivo da base de dados (paciente 100)

	Elapsed time	MLII	V5
	(seconds)	(mV)	(mV)
1	0.000	-0.145	-0.065
2	0.003	-0.145	-0.065
3	0.006	-0.145	-0.065
4	0.008	-0.145	-0.065
5	0.011	-0.145	-0.065
6	0.014	-0.145	-0.065
7	0.017	-0.145	-0.065
8	0.019	-0.145	-0.065
9	0.022	-0.120	-0.080
10	0.025	-0.135	-0.080
11	0.028	-0.145	-0.085
12	0.031	-0.150	-0.085
13	0.033	-0.160	-0.075
14	0.036	-0.155	-0.070
15	0.039	-0.160	-0.070

Optou-se por usar o primeiro campo como medida de tempo e o terceiro campo como medida de sinal para formar o batimento cardíaco. O terceiro campo foi escolhido como medida de sinal pelo motivo de localização, tendo em vista que os eletrodos possíveis nesse campo são posicionados próximos ao coração (conforme seção 2.2, Figura 2, item 3).

A Figura 2, item 3 da seção 2.2 demonstra o posicionamento dos 6 eletrodos utilizados em um eletrocardiograma. Os eletrodos são nomeados de V1 à V6 sendo posicionados no peito do paciente, próximos ao coração. Cada eletrodo resulta em um gráfico diferente no eletrocardiograma (conforme seção 2.2, Figura 4), sendo escolhidos os eletrodos V1, V2, V4 e V5 pela base de dados MIT-BIH *Arrhytmia Database* (Physionet, 2012) como fonte de dados para o terceiro campo de seus registros.

Após carregar as informações da base, o aplicativo emissor inicia a simulação do coração humano conforme Quadro 2.

Quadro 2 – Simulação do coração humano através de uma *`Thread`*

```
01. while (running && !isEmpty()) {
02.   double oldTime = 0;
03.   int milisecond = 0;
04.   for (ECGSignal s : signals) {
05.       milisecond = (int) (s.getTime() - oldTime) * 1000;
06.       oldTime = s.getTime();
07.       sleep(milisecond);
08.       listener.sendSignal(s.getSignal());
09.   }
10. }
```

Enquanto a `Thread` estiver ativa (linha 01 do Quadro 2) o envio do sinais vitais continua em execução. Há uma transfomação de tempo de segundos para milisegundos (linha 05 do Quadro 2) para que, em seguida, seja feita a pausa entre um batimento e outro (linha 07 do Quadro 2). Após a pausa, o sinal é enviado através do método `sendSingnal` do atributo listener (classe `ECGListener`).

3.3.1.2 Transmissão de dados via *bluetooth*

A segunda etapa da identificação de batimentos cardíacos, a transmissão de dados via *bluetooth*, é iniciada pela classe `BluetoothServer` no módulo emissor. Esta classe implementa a interface `ECGListener` (atributo da classe `HeartSimulator`) e utiliza o padrão de projeto *Singleton* para que apenas uma instância dessa classe seja usada na aplicação. A classe possui três métodos principais: `init`, `run` e `sendSingnal`. Os Quadros 3, 4 e 5 demonstram o código desses métodos.

Quadro 3 – Código do método `init` da classe `BluetoothServer`

```
01.  public void init(MessageListener listener)
02.  throws BluetoothStateException, Exception  {
03.      messageListener = listener;
04.      checkMessageListener();
05.      device = LocalDevice.getLocalDevice();
06.      device.setDiscoverable(DiscoveryAgent.GIAC);
07.
08.      messageListener.showMessage(
09.        "Device name: " + device.getFriendlyName());
10.
11.      messageListener.showMessage(
12.          "Bluetooth Address: " + device.getBluetoothAddress());
13.  }
```

O método `init` recebe como parâmetro uma instância da interface `MessageListener`, responsável pela comunicação entre as camadas `model` e `view` do aplicativo emissor (linha 01 do Quadro 3). A finalidade deste método é tornar a aplicação visível para outros dispositivos bluetooth (linha 06 do Quando 3) e prover informações do *hardware* emissor, como por exemplo nome e endereço *bluetooth* (linhas 08 e 11 do Quadro 3, respectivamente).

Quadro 4 – Código do método `run` da classe `BluetoothServer`

```
01.  public void run() throws Exception {
02.      checkMessageListener();
03.      if (notifier == null) {
04.          notifier =
05.               (StreamConnectionNotifier) Connector.open(address);
06.
07.          stream = notifier.acceptAndOpen();
08.          outputStream = stream.openDataOutputStream();
09.          RemoteDevice remoteDevice =
10.              RemoteDevice.getRemoteDevice(stream);
11.          messageListener.showMessage("Conected device name: " +
12.              remoteDevice.getFriendlyName(false));
13.          messageListener.showMessage("Conected device address: " +
14.              remoteDevice.getBluetoothAddress());
15.          messageListener.showMessage("Data Atual: " +
16.               new SimpleDateFormat("dd/MM/yyyy HH:mm:ss.SSS").
17.               format(new Date()));
18.      }
19.  }
```

O método `run` tem como finalidade estabelecer uma conexão *bluetooth* (linha 07 do Quadro 4), mostrando ao usuário informações sobre o dispositivo ao qual efetuou-se a conexão (linhas 11, 13 e 15 do Quadro 4).

Quadro 5 – Código do método `sendSignal` da classe `BluetoothServer`

```
01.  public void sendSignal(double signal) throws Exception {
02.      checkMessageListener();
03.      if (outputStream != null) {
04.          outputStream.writeDouble(signal);
05.          outputStream.flush();
06.          messageListener.showMessage(
07.              Utils.formatECGSenderMessage(signal));
08.      }
09.  }
```

O método `sendSignal` tem como finalidade de enviar o sinal vital via comunicação *bluetooth* (linha 04 do Quadro 5) e de atualizar a camada *view* da aplicação emissora (linha 06 do Quadro 5), finalizando assim o papel do emissor cardíaco no processo de identificação de batimentos cardíacos.

Após a emissão do sinais vitais, entra em funcionamento o receptor cardíaco, um aplicativo desenvolvido na plataforma Android. A classe responsável por receber esses dados é a `CardioReaderService`. Essa classe herda características da classe `IntentService`, cuja classe pai é a classe `Service`. Segundo Lecheta (2013, p. 348) a classe `Service` é utilizada para executar um serviço em segundo plano, geralmente vinculado a algum processo que deve executar por tempo indeterminado e tem um alto consumo de recursos, memória e CPU. A classe `CardioReaderService` possui três métodos principais: `onHandleIntent`, `connect` e `readData`. Os Quadros 6, 7 e 8 demonstram os códigos dos métodos.

Quadro 6 – Código do método `onHandleIntent` da classe `CardioReaderService`

```
01.  protected void onHandleIntent(Intent intent) {
02.   /* Enquanto o Bluetooth estiver ativo, o serviço é executado. */
03.   while (isBluetoothEnabled &&
04.       errorCount < Constants.CONNECTION_TRY_COUNT) {
05.       try {
06.           connect();
07.           readData();
08.       } catch (Exception e) {
09.           Log.e(Constants.TAG, e.getMessage());
10.           errorCount++;
11.           try {
12.               Intent intentConnFailured =
13.                     new Intent(Constants.CONNECTION_FAIL_RECEIVED);
14.
15.               intentConnFailured.putExtra(
16.                     Constants.CONNECTION_FAILURED, e);
17.
18.               sendBroadcast(intentConnFailured);
19.               disconnect();
20.         } catch (Exception ex) {
21.         }
22.       }
23.     }
24.  }
```

O método `onHandleItent` é o principal método da classe `IntentService`. Segundo Lecheta (2013, p. 374) esse método é executado em uma `Thread` separada, criada automaticamente pela própria classe. Ao finalizar o método, o serviço criado é finalizado. Esse método é executado enquanto a comunicação *bluetooth* estiver ativa no dispositivo móvel e o número de falhas na comunicação for menor que o valor da constante `Constants.CONNECTION_TRY_COUNT`. Caso o número de falhas exceda tal valor, uma mensagem de erro é emitida ao sistema operacional (linha 18 do Quadro 6) e as aplicações que tirevem interesse em processar tal mensagem o fazem. O código dos métodos `connect` e `readData` são demonstrados nos quadros a seguir.

Quadro 7 – Código do método `connect` da classe `CardioReaderService`

```
01.  private void connect() throws IOException {
02.     if (socket == null && device != null) {
03.         socket =
04.           device.createRfcommSocketToServiceRecord(Constants.SPP_UUID);
05.
06.         adapter.cancelDiscovery();
07.         socket.connect();
08.         stream = socket.getInputStream();
09.         hostDevice = socket.getRemoteDevice();
11.         Intent intent =
12.           new Intent(Constants.CONNECTION_SUCCESS_RECEIVED);
13.         intent.putExtra(Constants.DEVICE_CONNECTED, hostDevice);
14.         sendBroadcast(intent);
15.     }
16.  }
```

O método `connect` tem como finalidade efetuar a conexão com o dispositivo *bluetooth* pareado (linha 07 do Quadro 7). Antes disso (linha 06 do Quadro 7) é chamado o método `cancelDiscovery` da classe `BluetoothAdapter`. Esse método cancela qualquer busca por dispositivos pendentes, liberando recurso para que o método de conexão seja executado. Após efetuada a conexão, é enviado uma mensagem de conexão bem sucedida, para que qualquer aplicação interessada em processar tal mensagem o faça.

Quadro 8 – Código do método `readData` da classe `CardioReaderService`

```
01.  private void readData() throws IOException {
02.      if (stream != null) {
03.        int avaiable = stream.available();
04.        if (avaiable > 0) {
05.          stream.read(buffer);
06.          ByteBuffer byteBuffer = ByteBuffer.wrap(buffer);
07.          double valor = byteBuffer.getDouble();
08.          Date data = new Date();
09.          ECGSignal signal = new ECGSignal(data, valor);
10.
11.          Intent intent = new Intent(Constants.ECG_DATA_RECEIVED);
12.          intent.putExtra(Constants.ECG_SIGNAL, signal);
13.          sendBroadcast(intent);
14.
15.          signals.add(signal);
16.          if(signals.size() == 1){
17.              initialTime = signal.getData().getTime();
18.          }
19.          else{
20.              long currentTime = signal.getData().getTime();
21.              if(currentTime - initialTime >= 60000){
22.                  initialTime = currentTime;
23.                List<ECGSignal> analyzeList =
24.                          new ArrayList<ECGSignal>(signals);
25.
26.                task = (FilterTask) task.execute(analyzeList);
27.                signals.clear();
28.              }
29.           }
30.        }
31.     }
32.  }
```

O método `readData` é responsável por ler e armazenar a informação lida em um *buffer* (linha 05 do Quadro 8), transformar essa informação em uma variável primitiva *double* (linha 06 do Quadro 8), criar uma instância da classe `ECGSignal` (linha 09 do Quadro 8) e enviar essa informação para o sistema operacional. Feito isso, a informação obtida na leitura é armazenada em uma lista (linha 15 do Quadro 8) e ao completar um minuto de leitura (linha 21 do Quadro 8) é feito a análise para detectar os batimentos cardíacos existentes nesse periodo. O método `readData` conclui a segunda etapa da identificação de batimentos cardíacos (transmissão de dados via *bluetooth*) e inicia a terceira etapa do processamento, o cálculo da frequência cardíaca.

3.3.1.3 Cálculo da frequência cardíaca

O cálculo da frequência cardíaca tem as seguintes etapas: filtro passa-alta, filtro passa-baixa e tomada de decisão (ver figura 9 da seção 2.4). A primeira etapa do cálculo da frequência cardíaca (filtro passa-alta) tem como finalidade realçar o complexo QRS, suprimindo ondas indesejáveis ao cálculo, como as ondas P e T e normalizando a movimentação das ondas em torno da linha base. A segunda etapa (filtro passa-baixa) consiste em preservar o complexo QRS suavisando artefatos de baixa amplitude. Finalizando, a última etapa (tomada de decisão) consiste na aplicação de um algoritmo para identificação de batimentos cardíacos presentes no sinal filtrado.

A classe `Filter` é responsável por implementar todos os passos do cálculo da frequência cardíaca. Esta classe implementa o padrão de projeto *Singleton* (para que a aplicação use apenas uma instância da classe).

O método `analyze` implementa o funcionamento geral descrito na Figura 9 da seção 2.4 conforme Quadro 9.

Quadro 9 – Código do método `analyze` da classe `Filter`

```
01.  public HeartRate analyze(List<ECGSignal> signals) {
02.    List<ECGSignal> highFiltered = null, lowFiltered = null;
03.    try {
04.      treshold = 0;
05.      highFiltered = highPassFilter(signals);
06.      lowFiltered = lowPassFilter(highFiltered);
07.      switch (limiar) {
08.        case FIXO_DOIS_TERCOS:
09.        case FIXO_UM_QUINTO:
10.          return searchWithFixedLimiar(lowFiltered);
11.      default:
12.        return searchWithAdaptativeLimiar(lowFiltered);
13.      }
14.    } finally {
15.      highFiltered.clear();
16.      lowFiltered.clear();
17.    }
18.  }
```

O método `analyze` é iniciado recebendo uma lista de sinais vitais armazenados no tempo de um minuto (linha 01 do Quadro 9). Em seguida é inicializado o atributo `treshold` (linha 04 do Quadro 9) e são executados os métodos de filtro, conforme Figura 9 da seção 2.4 (linhas 05 e 06 do Quadro 9). Tendo o sinal filtrado, executa-se o método de procura por batimento (ou método de decisão, conforme Figura 9 da seção 2.4) identificado nas linhas 10 e 12 do Quadro 9.

3.3.1.3.1 Filtro passa-alta

Conforme linha 05 do Quadro 9, o primeiro filtro a ser aplicado no sinal lido é o filtro passa-alta, implementado pelo método `highPassFilter`, conforme Quadro 10.

Quadro 10 – Código do método `highPassFilter` da classe `Filter`

```
01.  private List<ECGSignal> highPassFilter(List<ECGSignal> signals) {
02.      List<ECGSignal> result = new ArrayList<ECGSignal>();
03.
04.      double constant = (double) 1 / M;
05.      double y1, y2;
06.      int size = signals.size();
07.
08.      for (int i = 0; i < size; i++) {
09.          int y2_index = i - ((M + 1) / 2);
10.          if (y2_index < 0) {
11.              y2_index = size + y2_index;
12.          }
13.          y2 = signals.get(y2_index).getSignal();
14.
15.          double y1_sum = 0;
16.          for (int j = i; j > i - M; j--) {
17.              int x_index = i - (i - j);
18.              if (x_index < 0) {
19.                  x_index = size + x_index;
20.              }
21.              y1_sum += signals.get(x_index).getSignal();
22.          }
23.
24.          y1 = constant * y1_sum;
25.          result.add(new ECGSignal(signals.get(i).getData(), y2 - y1));
26.      }
27.      return result;
28.  }
```

Fonte: adaptado de Cigan (2011).

Inicialmente, o método `highPassFilter` recebe como parâmetro o sinal não filtrado, compreendido no intervalo de um minuto (linha 02 do Quadro 10). Em seguida a variável `constant` é inicializada (linha 04 do Quadro 10). Segundo Chen e Chen (2003, p. 586) o valor do atributo `M` (linha 04 do Quadro 10) deve ser entre 5 e 7, sendo escolhido 5, de acordo com Cigan (2011). Para uma explicação mais detalhada do algoritmo, a Figura 21 demonstra matematicamente o resultado do algoritmo.

Figura 21 – Expressão matemática do resultado do método `highPassFilter`

$$y[n] = y_2[n] - y_1[n]$$

Fonte: Chen e Chen (2003, p. 586).

Conforme visto na Figura 21, cada sinal filtrado pelo método `highPassFilter` consiste na subtração de dois valores: `y2` e `y1` (linha 25 do Quadro 10). A Figura 22 demonstra a expressão para o cálculo da variável `y2`.

Figura 22 – Expressão matemática para cálculo da variável `y2`

$$y_2[n] = x[n - \frac{M + 1}{2}]$$

Fonte: Chen e Chen (2003, p. 586).

O cálculo demonstrado na Figura 22 pode ser visualizado no Quadro 10, linhas 09 à 13. A Figura 23 demonstra o cálculo da última variável, `y1`.

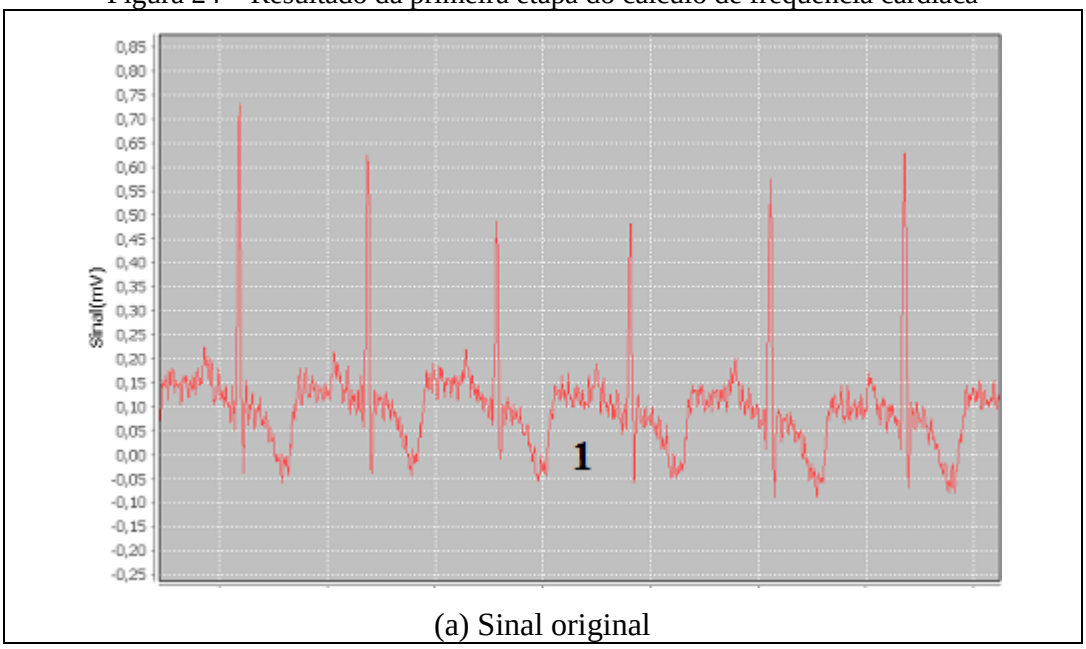
Figura 23 – Expressão matemática para cálculo da variável `y1`

$$y_1[n] = \frac{1}{M} \sum_{m=0}^{M-1} x[n - m]$$

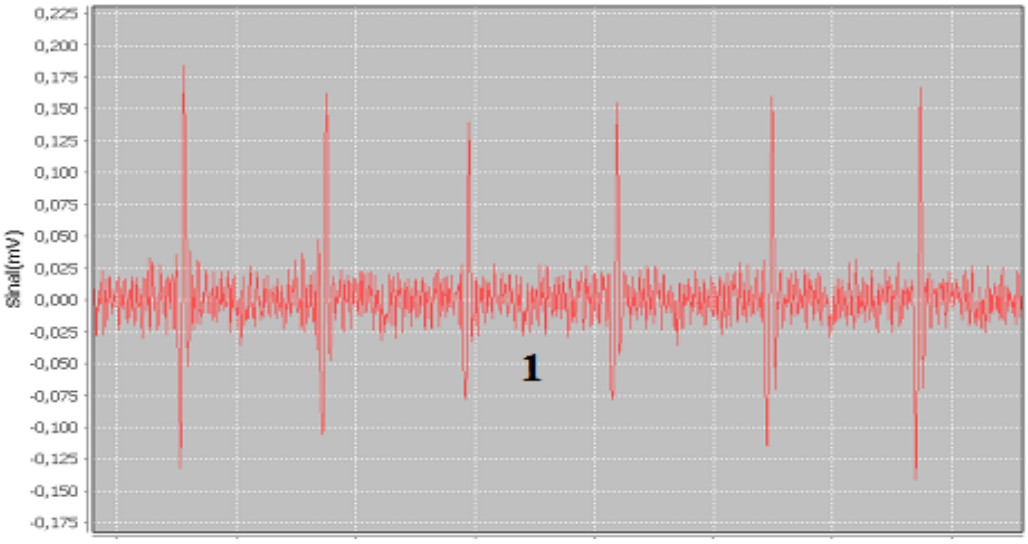
Fonte: Chen e Chen (2003, p. 586).

O cálculo demonstrado na Figura 23 pode ser visualizado no Quadro 10, linhas 17 à 24. A Figura 24a demonstra uma amostra original da base de dados MIT-BIH *Arrhytmia Database* (Physionet, 2012) e a Figura 24b demonstra o mesmo sinal depois de passar pelo método `highPassFilter`.

Figura 24 – Resultado da primeira etapa do cálculo de frequência cardíaca



(a) Sinal original



(b) Sinal filtrado pelo método highPassFilter

Conforme visto na Figura 24a, o item 1 apresenta a oscilação da linha base no sinal. Esta oscilação é suprimida após a aplicação do filtro passa-alta, podendo ser visto na Figura 24b (item 1).

3.3.1.3.2 Filtro passa-baixa

Com o sinal filtrado pelo método `highPassFilter`, o segundo filtro (passa-baixa) é aplicado, conforme linha 06 do Quadro 9. O código do segundo filtro pode ser visualizado no Quadro 11.

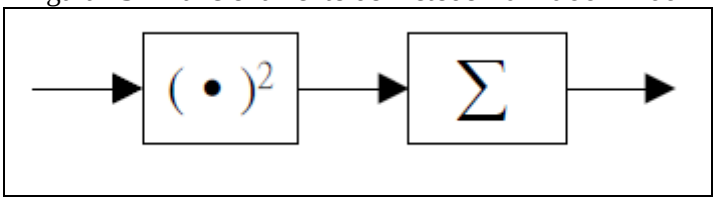
Quadro 11 – Código do método `lowPassFilter` da classe `Filter`

```
01.  private List<ECGSignal> lowPassFilter(List<ECGSignal> highFiltered) {
02.      List<ECGSignal> result = new ArrayList<ECGSignal>();
03.      int size = highFiltered.size();
04.      for (int i = 0; i < size; i++) {
05.          float sum = 0;
06.          if (i + 30 < size) {
07.            for (int j = i; j < i + 30; j++) {
08.                sum += Math.pow(highFiltered.get(j).getSignal(), 2);
09.            }
10.          } else if (i + 30 >= size) {
11.            int over = i + 30 - size;
12.            for (int j = i; j < size; j++) {
13.                sum += Math.pow(highFiltered.get(j).getSignal(), 2);
14.            }
15.            for (int j = 0; j < over; j++) {
16.                sum += Math.pow(highFiltered.get(j).getSignal(), 2);;
17.            }
18.         }
20.         result.add(new ECGSignal(highFiltered.get(i).getData(), sum));
21.      }
22.      return result;
23.  }
```

Fonte: adaptado de Cigan (2011).

O método `lowPassFilter` recebe um parâmetro: o sinal filtrado pelo método `highPassFilter` conforme linha 01 do Quadro 11. O funcionamento do método em resumo, pode ser visualizado na Figura 25.

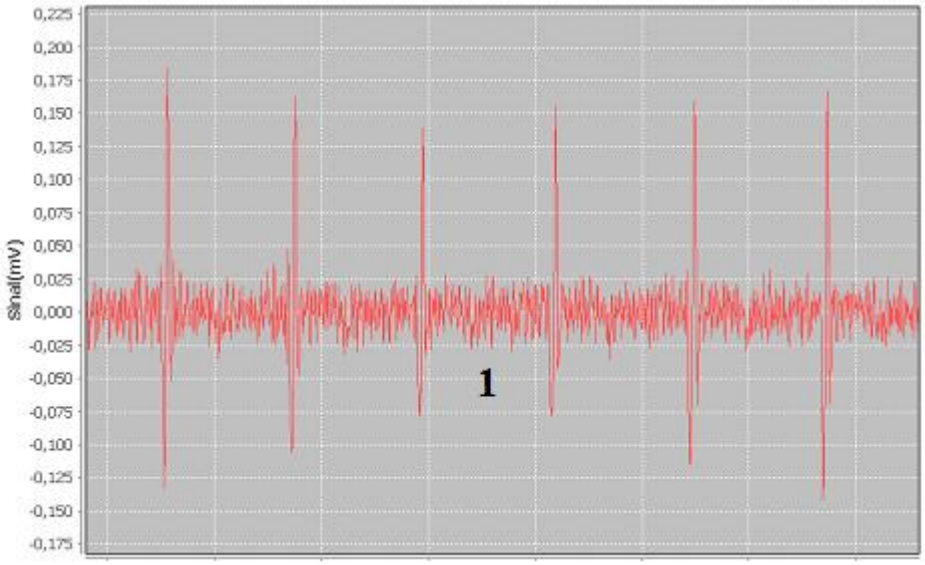
Figura 25 – Funcionamento do método `lowPassFilter`



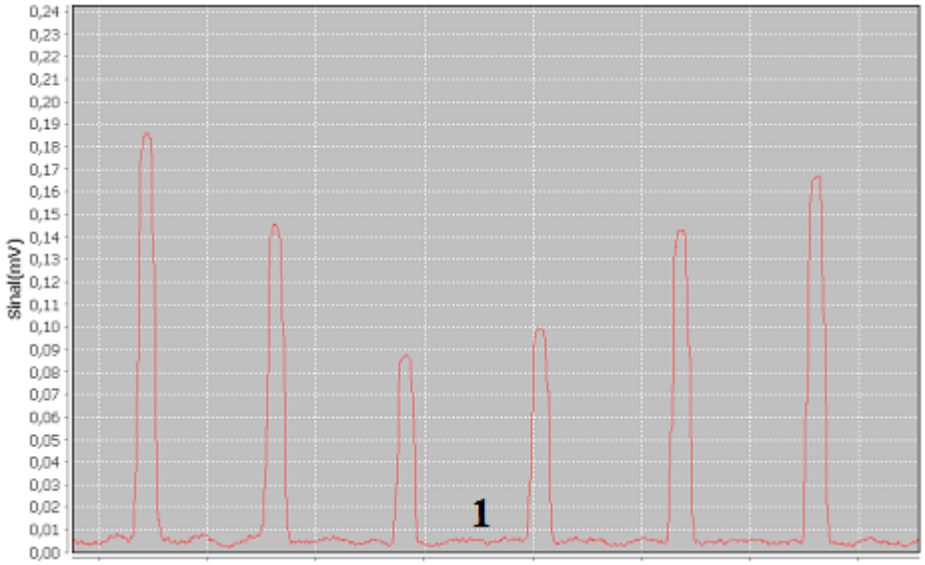
Fonte: Chen e Chen (2003, p. 586).

Conforme Figura 25, o algoritmo tem como finalidade a aplicação da raiz quadrada de cada ponto dentro de um intervalo de 30 pontos, atribuindo o resultado a uma variável de soma. No final, usa-se a variável de soma para criar um novo ponto. Esse algoritmo serve para realçar os picos e suavisar os demais componentes da onda com baixa amplitude. De acordo com Chen e Chen (2003, p. 587) o intervalo de 30 pontos foi usando com base em estudos anteriores. Esse intervalo corresponde, de acordo com esses estudos, a 150 milisegundos. A comparação entre o resultado dos dois filtros pode ser visualizado na Figura 26.

Figura 26 – Resultado dos algoritmos de filtro



(a) Sinal filtrado pelo método `highPassFilter`



(b) Sinal filtrado pelo método `lowPassFilter`

Conforme visto na Figura 26a, o item 1 apresenta ruído oriundo de atividade elétrica no sinal. Este ruído foi removido após a aplicação do filtro passa-baixa, podendo ser visto na Figura 26b (item 1).

3.3.1.3.3 Reconhecimento dos batimentos cardíacos

Após a execução dos métodos de filtro, faz-se necessário a procura por batimentos cardíacos no sinal filtrado. Neste projeto foi implementado dois métodos para esta etapa: usando limiares adaptativo e fixo. A necessidade de implementar dois métodos com limiares diferentes veio da comparação da melhor forma de identificação de batimentos cardíacos.

3.3.1.3.3.1 Reconhecimento de batimentos cardíacos usando limiar adaptativo

O Quadro 12 demonstra o código do método reponsável pela procura dos batimentos cardíacos usando limiar adaptativo.

Quadro 12 – Código do método `searchWithAdaptativeLimiar` da classe `Filter`

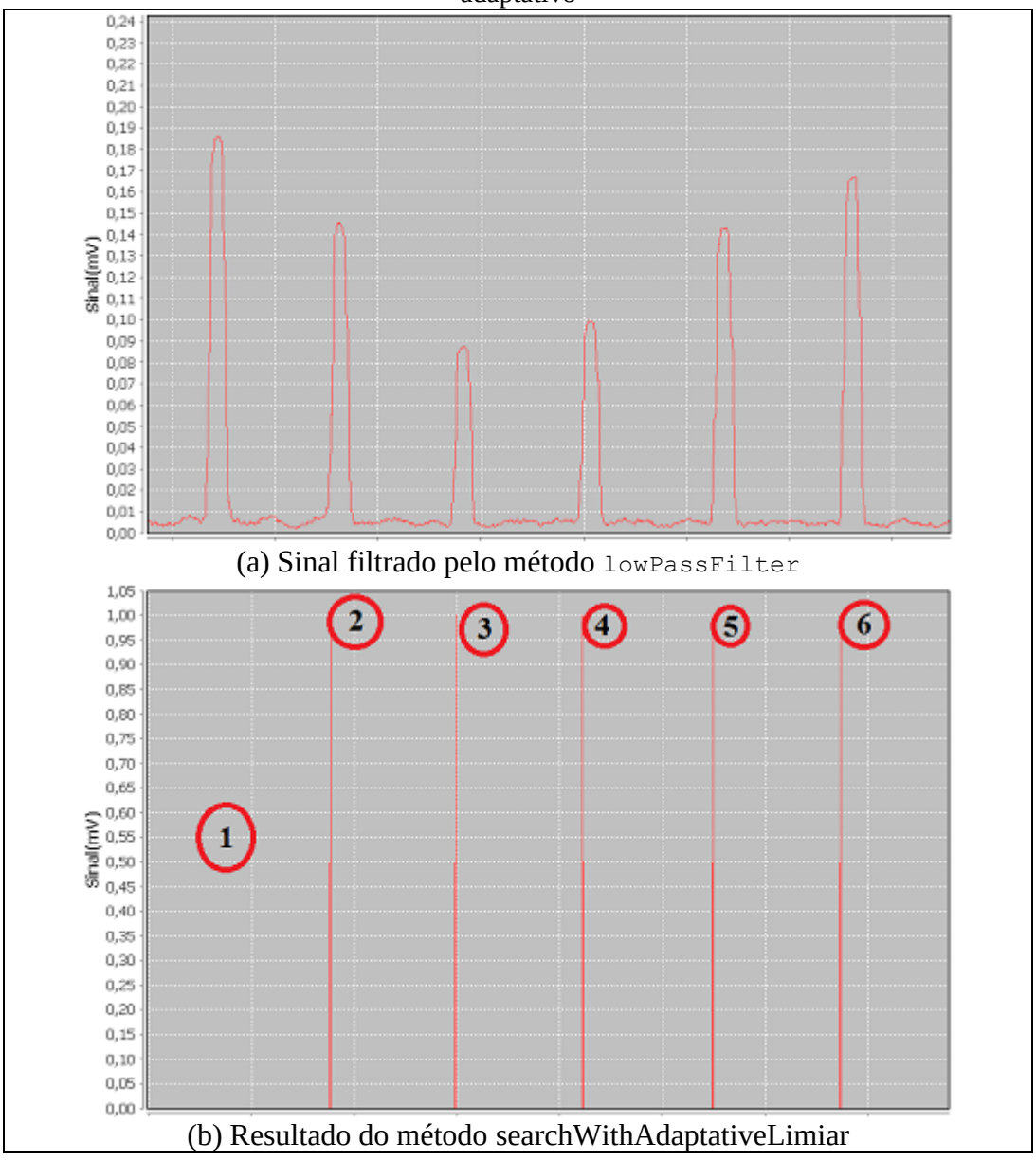
```
01.    private HeartRate searchQRS(List<ECGSignal> lowFiltered) {
02.        int beatCount = 0;
03.        int size = lowFiltered.size();
04.        for (int i = 0; i < 200; i++) {
05.            if (lowFiltered.get(i).getSignal() > treshold) {
06.                 treshold = lowFiltered.get(i).getSignal();
07.            }
08.        }
09.
10.
11.        for (int i = 0; i < size; i += frameSize) {
12.            double max = 0;
13.            int index = 0;
14.            if (i + frameSize > size) {
15.                index = size;
16.            } else {
17.                index = i + frameSize;
18.            }
19.            for (int j = i; j < index; j++) {
20.                if (lowFiltered.get(j).getSignal() > max) {
21.                    max = lowFiltered.get(j).getSignal();
22.                }
23.            }
24.            boolean added = false;
25.            for (int j = i; j < index; j++) {
26.                double ecgSignal = lowFiltered.get(j).getSignal();
27.                if (ecgSignal > treshold && !added) {
28.                    beatCount++;
29.                    added = true;
30.                }
31.            }
32.
33.            double gama = (Math.random() > 0.5) ? 0.15 : 0.20;
34.            double alpha = 0.01 + (Math.random() * ((0.1 - 0.01)));
35.
36.            treshold = alpha * gama * max + (1 - alpha) * treshold;
37.        }
38.        return new HeartRate(
39.            beatCount, lowFiltered.get(size - 1).getData());
40.    }
```

Fonte: adaptado de Cigan (2011).

O método `searchQRSWithAdaptativeLimiar` da classe `Filter` inicia recebendo como parâmetro o resultado dos sinais filtrados pelo método `lowPassFilter`. Em seguida há a calibragem do atributo `treshold`, sendo o valor do maior ponto encontrando dentro de um conjunto de 200 pontos (linha 06 do Quadro 12). Em seguida, dentro de um intervalo de 250 em 250 pontos, é efetuado a procura do maior sinal (linha 21 do Quadro 12) para ser usado mais a frente (linha 36 do Quadro 12) e caso o sinal corrente da iteração seja maior que o

valor da variável `treshold`, faz-se então o incremento da quantidade de batimentos cardíacos encontrados (linha 28 do Quadro 12). Após essa iteração é feita a atualização da variável `treshold` (linhas 33, 34 e 36 do Quadro 12). No final do método é retornado uma instância da classe `HeartRate`, responsável pela informação de quantidade de batimentos lidos e data e hora em que foi computado tal informação. A Figura 27 demonstra o resultado obtido com o algoritmo de detecção de batimentos com limiar adaptativo, bem como uma comparação com o resultado do algoritmo passa-baixa implementado no método `lowPassFilter`.

Figura 27 – Resultado do algoritmo de reconhecimento de batimentos cardíacos usando limiar adaptativo



(a) Sinal filtrado pelo método `lowPassFilter`

(b) Resultado do método searchWithAdaptativeLimiar

A Figura 27b demonstra o resultado do algoritmo de detecção de batimentos cardíacos com limiar adaptativo. O item 1 representa a inicialização do algoritmo, atribuindo valor à

variável `treshold`. Em seguida, os itens 2, 3, 4, 5 e 6 representam o momento em que o algoritmo encontra um batimento cardíaco, totalizando no exemplo, 5 batimentos.

3.3.1.3.3.2 Reconhecimento de batimentos cardíacos usando limiar fixo

O Quadro 13 demonstra o código do método reponsável pela procura dos batimentos cardíacos usando limiar fixo.

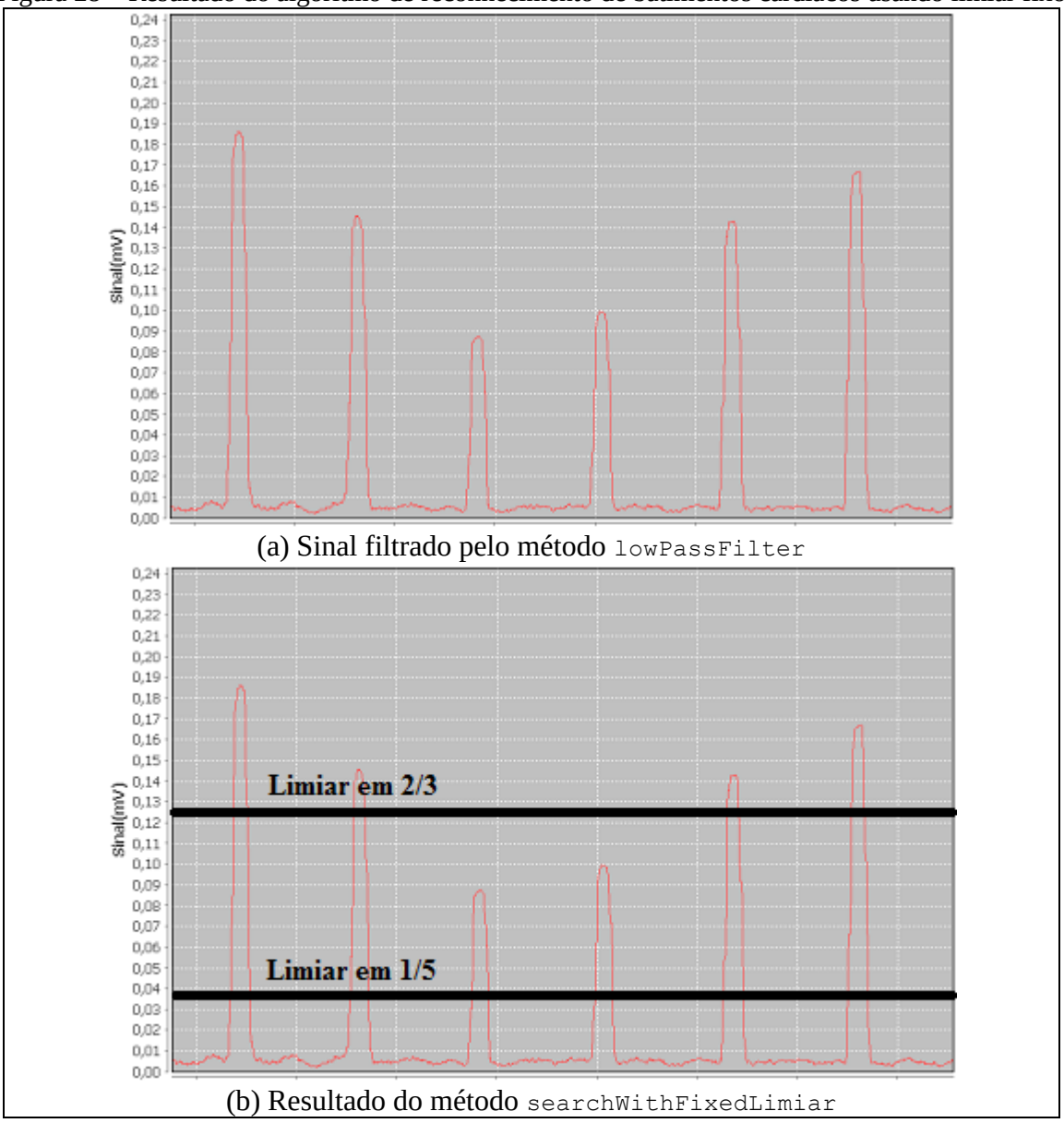
Quadro 13 – Código do método `searchWithFixedLimiar` da classe `Filter`

```
01.  private HeartRate searchWithFixedLimiar(List<ECGSignal> lowFiltered){
02.    // maior valor encontrado.
03.    double maxValue = 0;
04.
05.    int beatCount = 0;
06.
07.    // procura pelo maior valor dentro do intervalo
08.    for(ECGSignal s : lowFiltered){
09.      if(s.getSignal() > maxValue){
10.        maxValue = s.getSignal();
11.      }
12.    }
13.
14.    // limiar : 2/3 ou 1/5
15.    treshold = maxValue * limiar.getValue();
16.
17.    // análise dos batimentos
18.    boolean isIncrease = false;
19.    for(ECGSignal s : lowFiltered){
20.      if(s.getSignal() > treshold){
21.        isIncrease = true;
22.      }
23.      else{
24.        if(isIncrease){
25.          beatCount++;
26.          isIncrease = false;
27.        }
28.      }
29.    }
30.    return new HeartRate(beatCount,
31.      lowFiltered.get(lowFiltered.size() - 1).getData(), limiar);
32.  }
```

O método `searchWithFixedLimiar` da classe `Filter` inicia recebendo como parâmetro a lista de sinais vitais filtrados. Em seguida, o método procura pelo maior sinal dentro da lista de sinais filtrados, armazenando esse valor na variável `maxValue` (linha 10 do Quadro 13). Após essa etapa, inicializa-se a varíável `treshold` (limiar) de acordo com o opção escolhida (limiar em 2/3 ou 1/5) (linha 15 do Quadro 13). Na etapa final do processo, procura-se por valores maiores que o limiar. Caso seja encontrado um valor desta forma, ao encontrar um valor menor que o limiar, incrementa-se o valor de batimentos cardíacos identificados (linha 25 do Quadro 13). Esse processo visa percorrer a onda por completo

(precisa subir o valor, iniciando a onda e descer o valor, onde a onda termina). A Figura 28 demonstra o resultado obtido com o algoritmo de detecção de batimentos com limiar fixo, bem como uma comparação com o resultado do algoritmo passa-baixa implementado no método `lowPassFilter`.

Figura 28 – Resultado do algoritmo de reconhecimento de batimentos cardíacos usando limiar fixo



(a) Sinal filtrado pelo método `lowPassFilter`

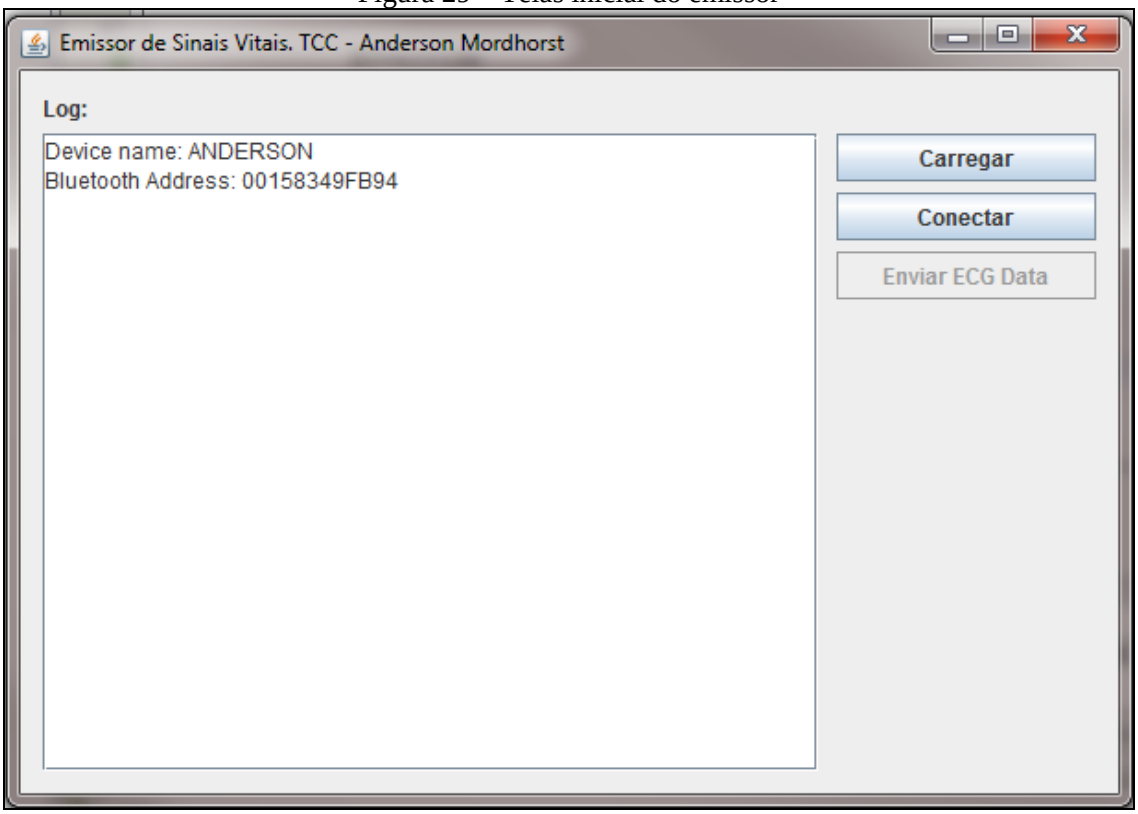
(b) Resultado do método `searchWithFixedLimiar`

A Figura 28b demonstra o resultado do algoritmo de detecção de batimentos cardíacos com limiar fixo, usando dois valores: 2/3 e 1/5 do maior valor encontrado no intervalo. Para o limiar de 2/3 obteve-se como resultado 4 batimentos identificados contra 6 batimentos identificados pelo limiar em 1/5.

3.3.2 Operacionalidade da implementação

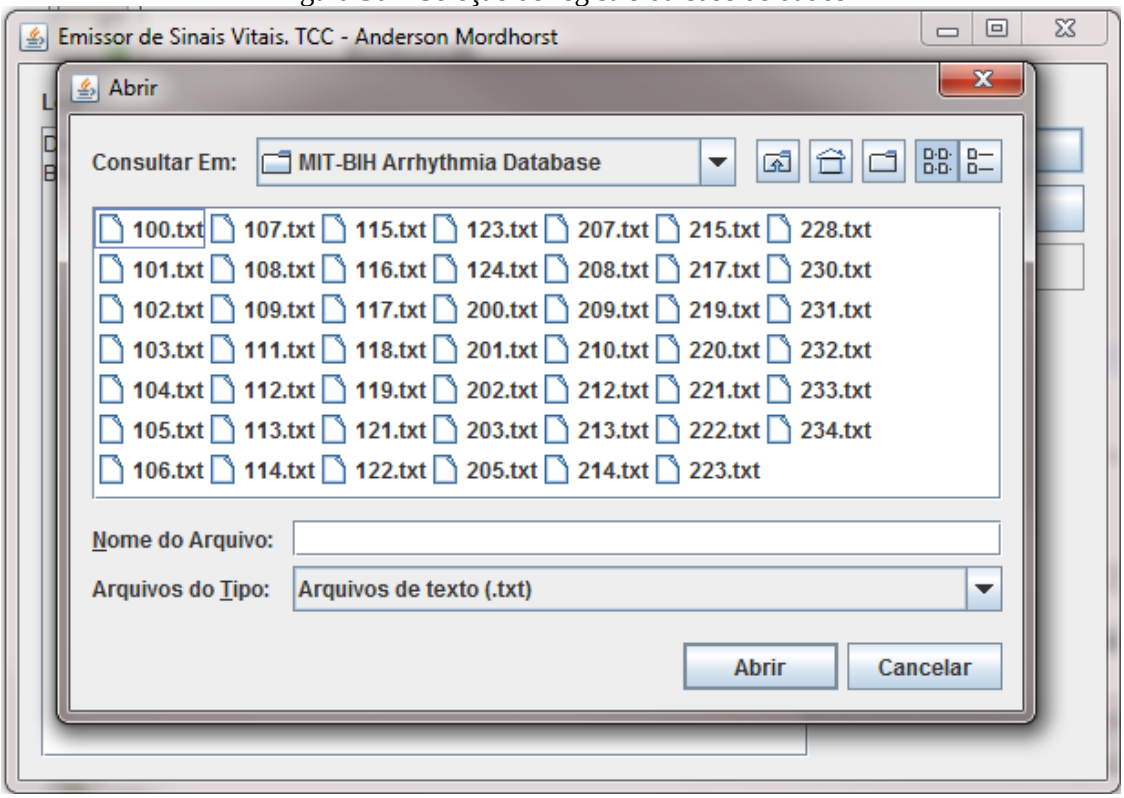
Para utilização do sistema de identificação de batimentos cardíacos o usuário deve iniciar o módulo receptor, conforme visto na Figura 29.

Figura 29 – Telas inicial do emissor



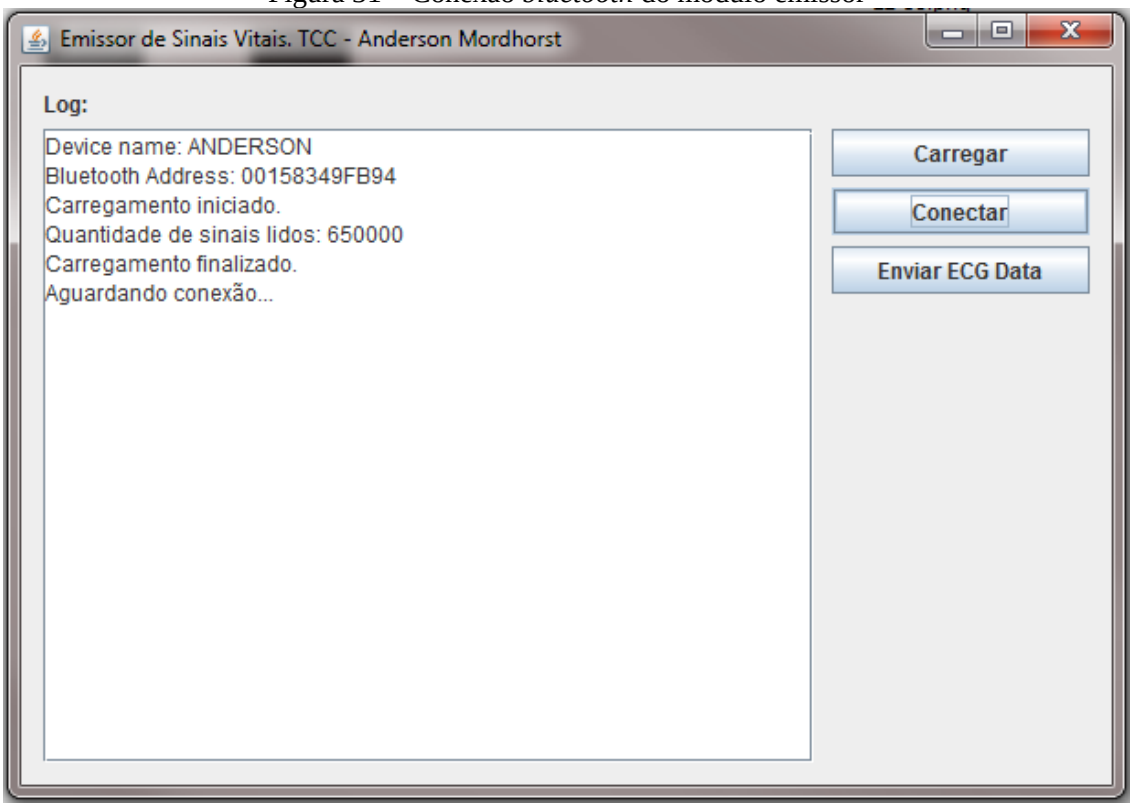
O próximo passo é a seleção de registro na base de dados, conforme mostra a Figura 30.

Figura 30 – Seleção de registro da base de dados



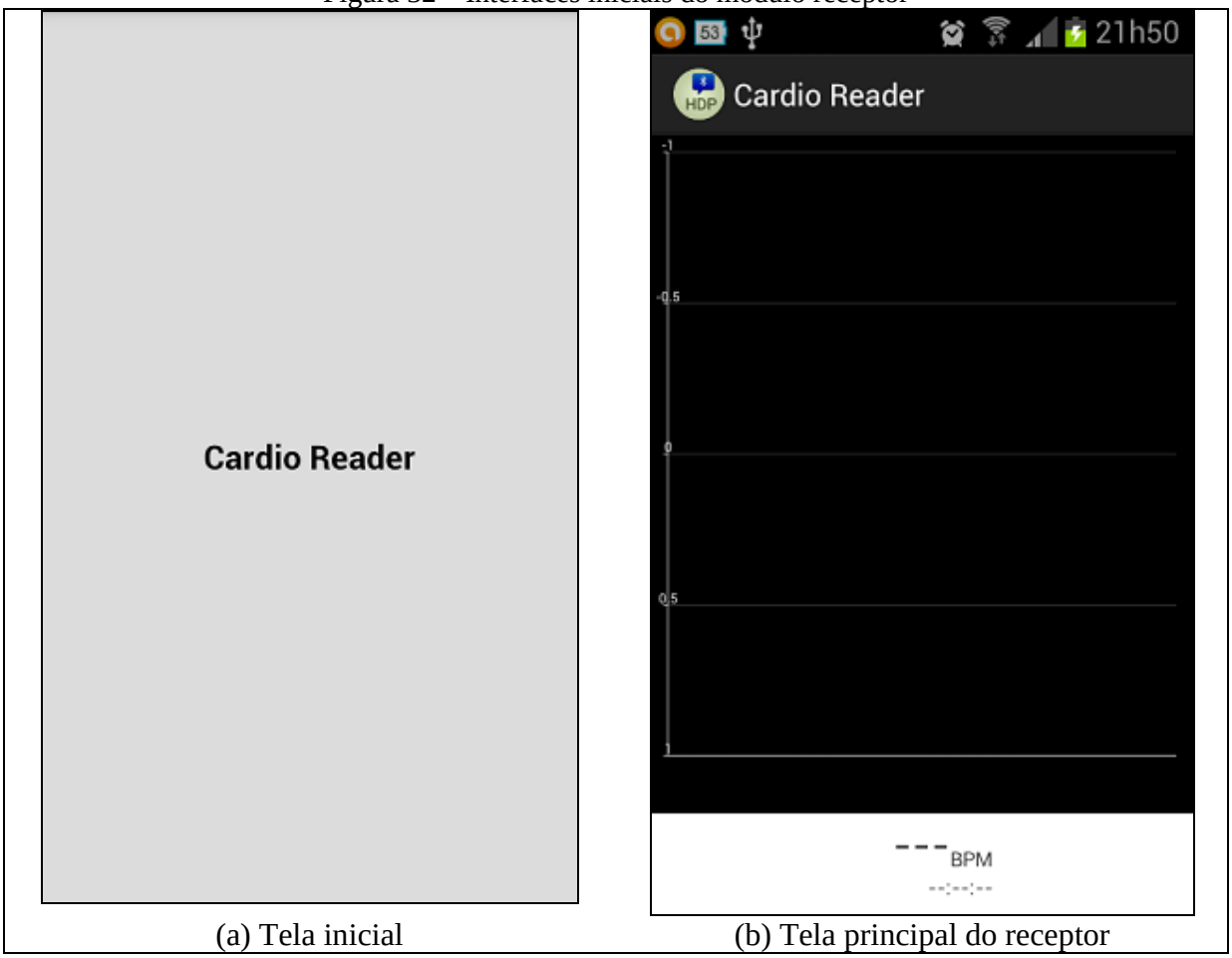
Depois de selecionado o registro, o módulo emissor totaliza a quantidade de sinais lidos e ao clicar no botão `Conectar`, inicia-se a conexão, aguardando a resposta do receptor, conforme Figura 31.

Figura 31 – Conexão *bluetooth* do módulo emissor



O próximo passo é iniciar a aplicação receptora, conforme Figura 32a. Em seguida é visualizado a tela principal da aplicação, onde são mostrados o gráfico em tempo real e as informações dos sinais vitais do usuário conforme Figura 32b.

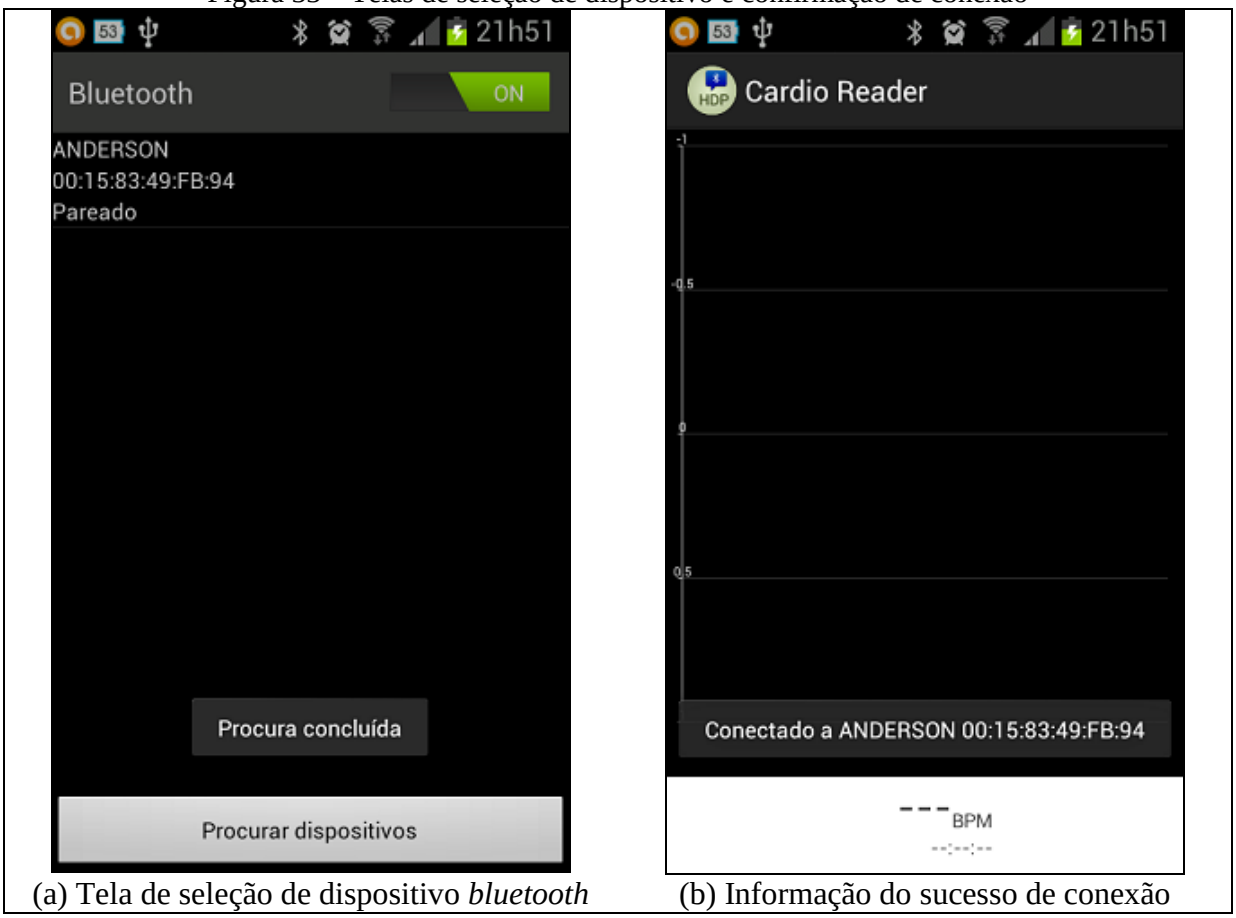
Figura 32 – Interfaces iniciais do módulo receptor



(a) Tela inicial (b) Tela principal do receptor

O próximo passo é ativar a conexão *bluetooth* e selecionar um dispositivo pareado para efetuar a conexão, conforme Figura 33a. Após essa etapa, retorna-se a tela principal da aplicação receptora onde é informado ao usuário o sucesso da conexão conforme Figura 33b.

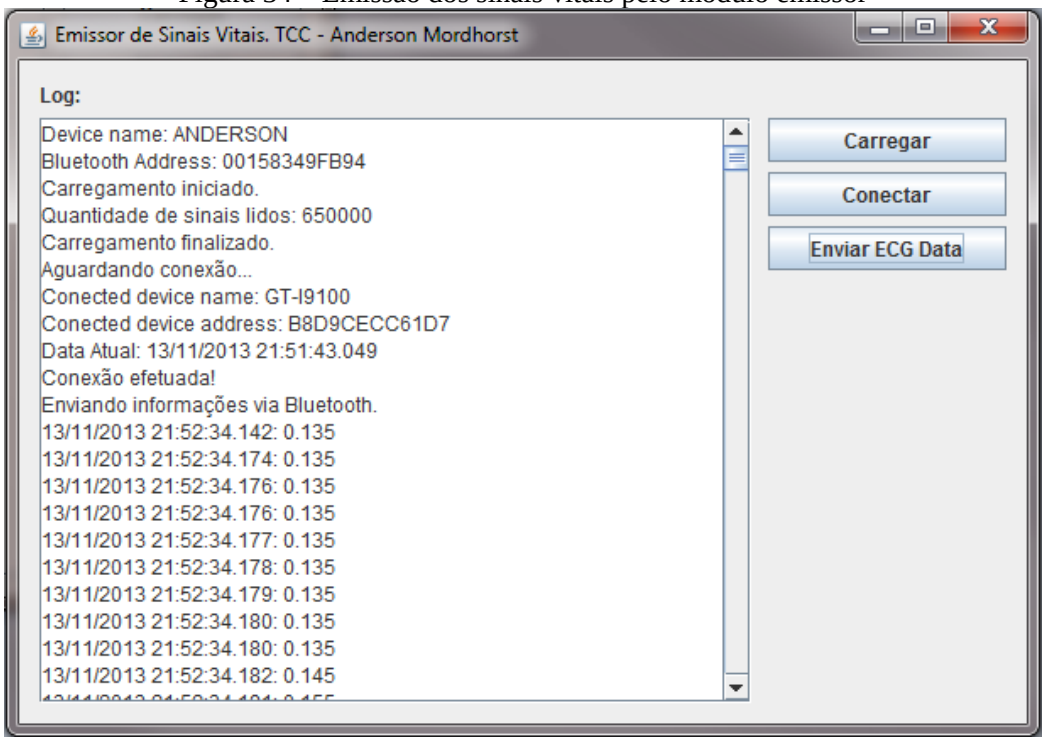
Figura 33 – Telas de seleção de dispositivo e confirmação de conexão



(a) Tela de seleção de dispositivo *bluetooth* (b) Informação do sucesso de conexão

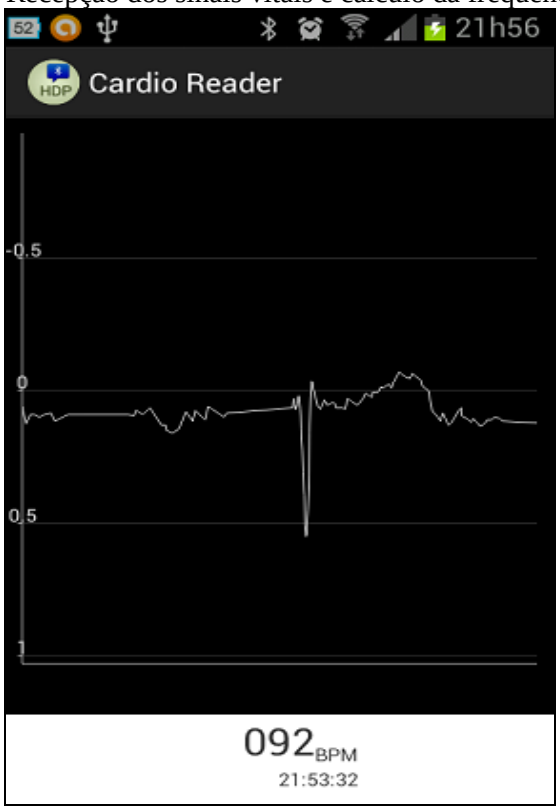
Em seguida, no módulo emissor, inicia-se o processo de emissão dos sinais vitais, conforme Figura 34.

Figura 34 – Emissão dos sinais vitais pelo módulo emissor



Finalizando o processo, o módulo receptor capta as informações emitidas pelo módulo emissor e calcula a frequência cardíaca, conforme Figura 35.

Figura 35 – Recepção dos sinais vitais e cálculo da frequência cardíaca



3.4 RESULTADOS E DISCUSSÃO

Para validação deste projeto foi efetuado testes em duas partes do projeto: comunicação *bluetooth* e identificação de batimentos cardíacos. Para os testes na comunicação *bluetooth* foi utilizado a versão 2.1 desta comunicação. Nos testes de identificação de batimentos cardíacos foi utilizado um *smartphone* Samsung *Galaxy* SII com processador *dual-core* 1.2 GHz Cortex-A9 e Android versão 4.0.4.

3.4.1 Comunicação *bluetooth*

Para o teste de comunicação *bluetooth* foram levados em consideração dois fatores: distância de transmissão e tempo de resposta.

Para o teste de distância foram utilizados dois cenários: um ambiente fechado e um ambiente aberto. O ambiente fechado consiste em um quarto, de aproximadamente 12m² (metro quadrado). Para acessar o ambiente é utilizado um corredor de aproximadamente 1,5m². O ambiente aberto consiste em uma sala comercial, cercada por divisórias com vidro. Nos dois cenários foi utilizado o envio de um sinal de 4 *bytes* para verificar a distância em que é possível transmitir dados via *bluetooth*. A Tabela 1 demonstra os resultados obtidos.

Tabela 1 – Distância máxima em ambiente aberto e fechado

Ambiente	**Distância máxima de conexão (metros)**
Aberto	Aprox. 15 metros
Fechado	Aprox. 5 metros

Um detalhe a ser observado no ambiente fechado e a posição da perda de conexão. Dentro deste ambiente a conexão permaneceu estável em todo o local. A perda de conexão ocorreu no corredor de acesso, onde haviam obstáculos entre o emissor do sinal e o receptor, como por exemplo paredes e mobílias. O tempo de resposta médio registrado nos testes é de 58 ms (milisegundos).

Em relação aos trabalhos correlatos, nenhum dos autores dispõe de informações de tempo em relação a comunicação *bluetooth*, impossibilitando qualquer comparação. Contudo, os cenários descritos nestes trabalhos utilizam uma comunicação bem próxima a seu usuário, tornando os resultados obtidos neste projeto satistfatórios.

3.4.2 Identificação de batimentos cardíacos

Para efetuar os testes sobre a identificação dos batimentos cardíacos foi utilizado a MIT-BIH *Arrhytmia Database* (Physionet, 2012). De acordo com Moody e Mark (2001, p. 46), a base de dados é formada por 48 amostras de 30 minutos cada, coletadas de 47 pacientes (sendo que um dos paciente possui duas amostras na base). Porém a base utilizada contém 48 amostras de aproximandamente 24 minutos cada.

A execução dos testes foi efetuada da seguinte forma: para cada registro na base é aplicado o processo de identificação de batimentos cardíacos, registrado a quantidade de batimentos reconhecidos em cada minuto e somado todos os batimentos lidos desta maneira. Os testes foram separados por registro e algoritmo (limiar adaptativo, limiar fixo em 2/3 e limiar fixo em 1/5), tendo seus resultados exibidos na Tabela 2.

Tabela 2 – Resultado do processamento dos algoritmos de identificação de batimentos

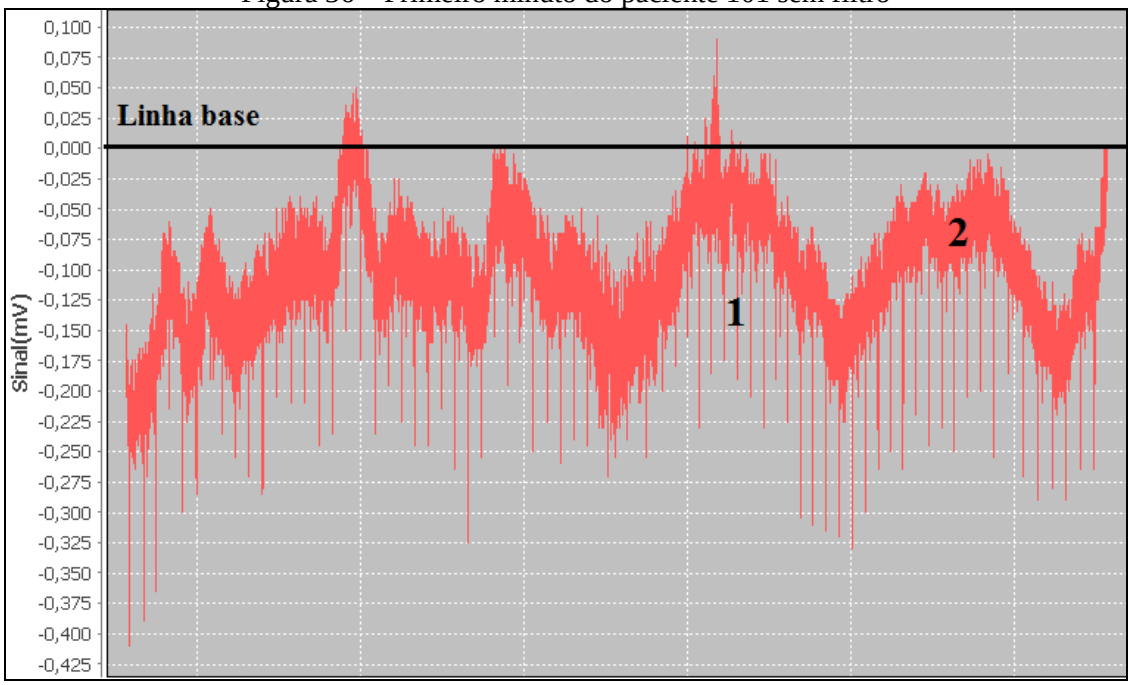
Paciente	Qtd Batim. base	Limiar					
		Adaptativo		Fixo 2/3		Fixo 1/5	
		Batim. recon.	Dif. (%)	Batim. recon.	Dif. (%)	Batim. recon.	Dif. (%)
100	2273	2477	36,21	1038	-42,94	2264	24,51
101	1865	2465	65,20	889	-40,43	12632	746,65
102	2187	2328	33,06	909	-48,06	3062	74,99
103	2084	2210	32,53	1145	-31,33	1826	9,50
104	2229	2439	36,74	359	-79,90	2312	29,65
105	2572	2351	14,23	120	-94,17	1337	-35,03
106	2027	2095	29,16	180	-88,90	969	-40,26
107	2137	2432	42,26	1143	-33,18	3648	113,38
108	1774	1818	28,07	111	-92,22	1188	-16,29
109	2532	2534	25,08	1743	-13,98	2506	23,70
111	2124	2303	35,50	539	-68,31	1973	16,10
112	2539	2457	20,95	569	-72,00	2510	23,55
113	1795	2002	39,39	1055	-26,57	1796	25,07
114	1879	2142	42,47	1293	-14,00	1879	24,96
115	1953	2105	34,72	814	-47,93	1603	2,56
116	2412	2429	25,87	356	-81,55	2303	19,32
117	1535	1748	42,35	1343	9,32	1535	24,95
118	2288	2494	36,23	1343	-26,66	2283	24,69
119	1987	2018	26,93	601	-62,20	1916	20,53
121	1863	2106	41,28	1030	-30,92	1901	27,54
122	2476	2561	29,28	2068	4,40	12459	528,96
123	1518	1766	45,39	1305	7,44	1515	24,70
124	1619	1880	45,15	1412	9,02	1612	24,46
200	2601	1978	-4,96	521	-74,97	2909	39,79
201	2000	1935	20,90	750	-53,15	1723	7,65
202	2136	2104	23,13	749	-56,18	2113	23,64
203	2980	2391	0,27	178	-92,55	2487	4,30
205	2656	2490	17,17	607	-71,46	2635	23,98
207	2332	2068	10,85	809	-56,65	1894	1,50
208	2955	2256	-4,57	197	-91,68	1721	-27,21
209	3005	2453	2,03	169	-92,98	2347	-2,40
210	2650	2328	9,81	122	-94,26	2207	4,08
212	2748	2438	10,88	832	-62,15	2179	-0,91
213	3251	2457	-5,54	599	-76,99	3299	26,82
214	2262	2206	21,88	597	-67,02	2206	21,88
215	3363	2427	-9,81	152	-94,35	2755	2,38
217	2208	2193	24,14	773	-56,25	1960	10,96
219	2287	2220	21,34	1364	-25,45	2061	12,64
220	2048	2246	37,06	1218	-25,68	2061	25,78
221	2427	2188	12,69	128	-93,41	1440	-25,83
222	2483	2361	18,85	1178	-40,72	2477	24,69
223	2605	2357	13,09	1025	-50,83	2330	11,79
228	2053	2148	30,78	1207	-26,55	2000	21,77
230	2256	2428	34,53	1139	-36,92	2216	22,78
231	1573	1871	48,63	1202	-4,51	1439	14,30
232	1780	1918	34,66	434	-69,55	1889	32,64
233	3079	2407	-2,31	1570	-36,28	2836	15,13
234	2753	2526	14,67	1392	-36,80	2769	25,72
	110159	107554		40277		122982	

Conforme Tabela 2, para cada algoritmo utilizado há dois campos: `Batim. Recon.` (batimentos reconhecidos) e `Dif. (%)` (diferença de percentual). O campo `Batim. Recon.` possui a quantidade de batimentos reconhecidos em um intervalo de 24 minutos, diferente da quantidade de batimentos computados da base de dados (30 minutos). Para nomalizar essas quantidades, fez-se o seguinte cálculo: quantidade de batimentos estimado = quantidade de batimentos lidos * 30 / 24. Com o valor da quantidade de batimentos estimados já conhecido, pode-se então calcular o percentual de diferença entre o algoritmo e a base de dados. Para isso, faz-se o seguinte cálculo: percentual de diferença = (quantidade de batimentos estimado * 100 / quantidade de batimentos da base de dados) – 100.

Analisando a coluna `Dif. (%)` de cada algoritmo, nota-se que, apesar de apresentar uma quantidade de falsos positivos elevada (maior valor identificado no paciente 101, sendo 65,20% maior que a base de dados) o algoritmo com limiar adaptativo mostra-se o mais estável dos dois algoritmos testados. O algoritmo com limiar fixo em 2/3 mostra-se o mais ineficiente, tendo seus resultados com menor índice de acerto em identificar os batimentos cardíacos (a maioria dos pacientes nesse algoritmo tem diferença negativa, ou seja, a quantidade de batimentos identificados é muito inferior ao presente na base de dados). O algoritmo com limiar em 1/5 apresenta a maior variação de resultandos em relação aos demais testados, reconhecendo 746,65% mais batimentos que a própria base de dados (paciente 101) e reconhecendo 40,26% menos batimentos cardíacos que a base de dados utilizado (paciente 106).

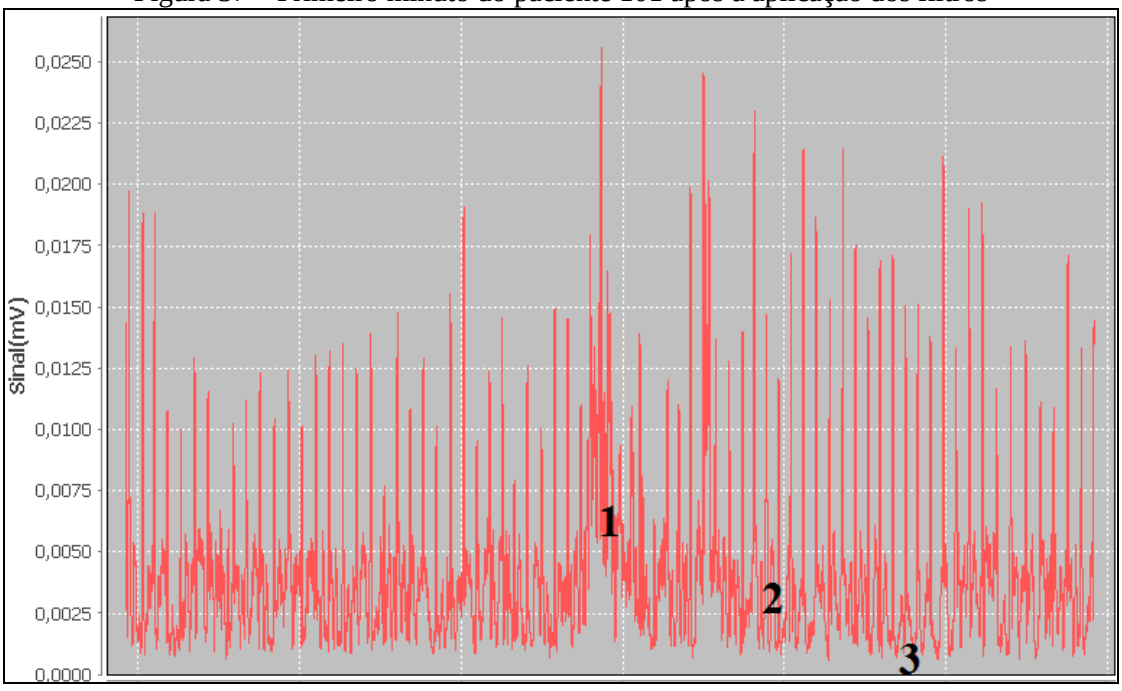
Para uma análise mais detalhada dos resultados obtidos, optou-se por escolher os registros da Tabela 2 que mais se destacaram nos testes efetuados (destacados com a cor verde). O primeiro registro a ser analisado foi o paciente 101 que se destaca pela alta taxa de batimentos identificados (746,45% maior que a base de dados), o paciente 203 foi o segundo registro escolhido devido a sua proximidade em relação a base de dados (com limiar adaptativo 0,27% maior e com limiar fixo em 1/5 com 4,30% maior que a base de dados. A Figura 36 mostra o primeiro minuto do paciente 101 sem a passagem de filtro.

Figura 36 – Primeiro minuto do paciente 101 sem filtro



A Figura 36 demonstra a existência de ruídos oriundos de movimentação muscular (item 1) e de interferência elétrica (item 2) no primeiro minuto de leitura do paciente 101. Em seguida, a Figura 37 demonstra esse mesmo paciente e período após a aplicação dos filtros passa-alta e passa-baixa.

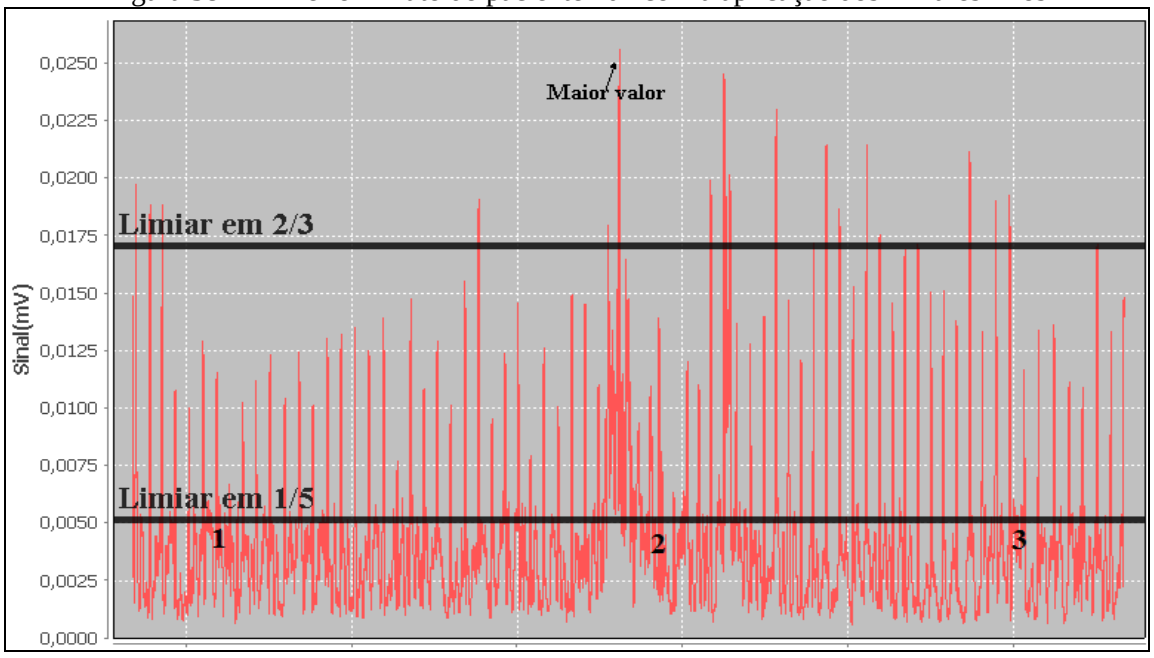
Figura 37 – Primeiro minuto do paciente 101 após a aplicação dos filtros



Conforme visto na Figura 37, mesmo após a aplicação dos filtros passa-alta e passa-baixa no registro do paciente 101, ainda é possível visualizar a existência de pequenos trechos

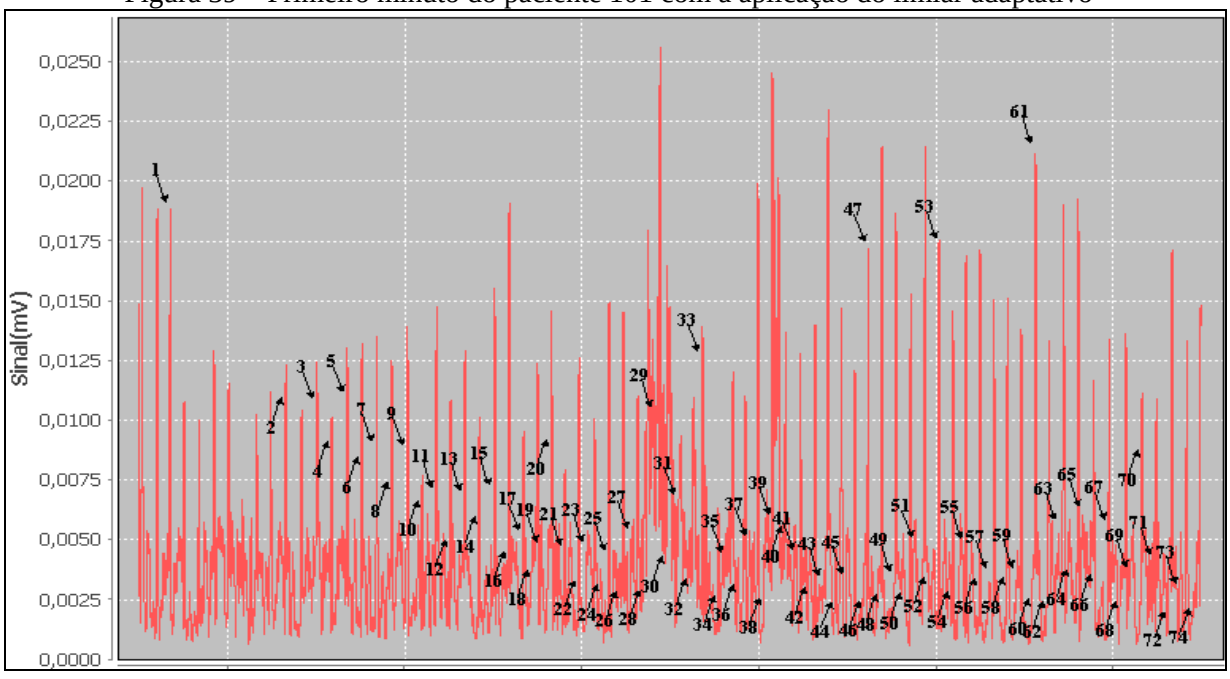
de ruído (item 1 e 2) e a elevação da linha base do sinal (item 3). A Figura 38 mostra o próximo passo do algoritmo, a aplicação dos limiares fixos.

Figura 38 – Primeiro minuto do paciente 101 com a aplicação dos limiares fixos



A Figura 38 demonstra as duas faixas de aplicação dos limiares do algoritmo com limiar fixo. Nota-se que o limiar fixo em 2/3 está muito alto e que o limiar fixo em 1/5 acaba computando áreas de ruído (itens 1, 2 e 3). A Figura 39 demonstra o último passo para esse registro, a aplicação do limiar adaptativo.

Figura 39 – Primeiro minuto do paciente 101 com a aplicação do limiar adaptativo

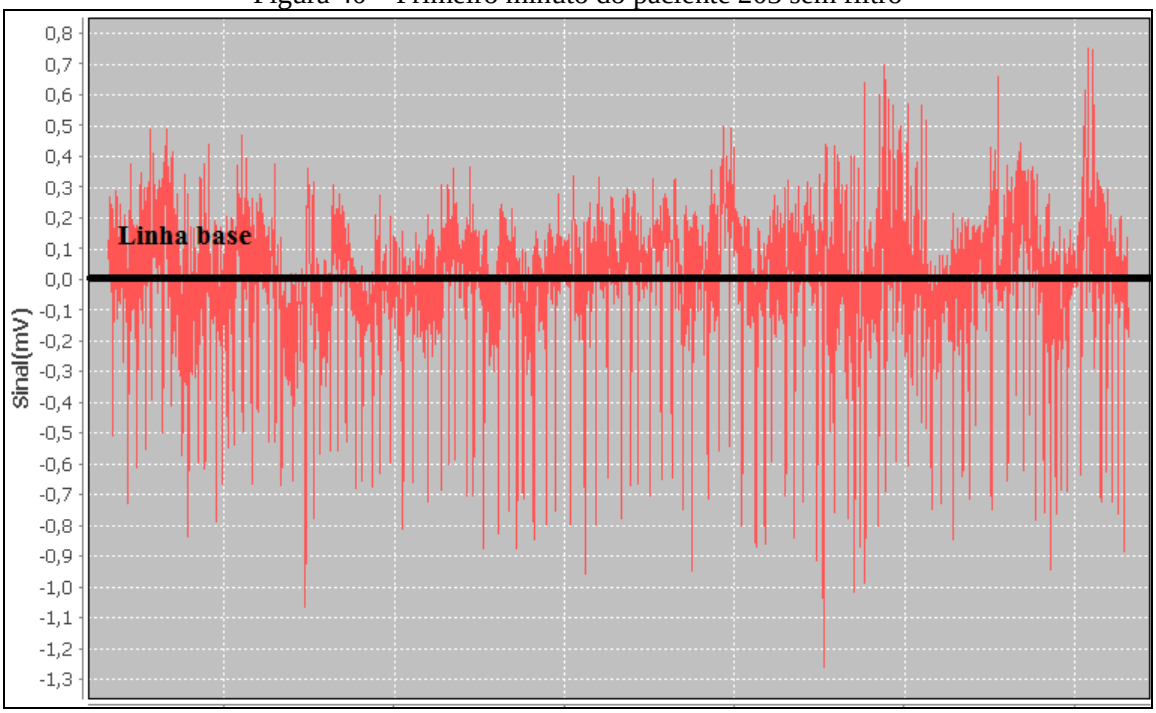


A Figura 39 demonstra a aplicação do algoritmo de limiar adaptativo no registro do paciente 101. Nota-se que há muitos batimentos cardíacos identificados em meio ao sinal ruidoso (como por exemplo os batimentos 12, 16, 18, 24, 26, 28, 29 e 32).

Com base no primeiro minuto de leitura do paciente 101 pode-se observar que os filtros passa-alta e passa-baixa possuem um desempenho ruim em registros com muito ruído. O limiar em 2/3 para esse registro é muito alto, não computando a maioria dos batimentos cardíacos nesse período (ocasionando a diferença de 40,43% menor que a base de dados) e o limiar em 1/5 acaba computando falsos batimentos pelo fato de estar sendo empregado em uma área com ruído (ocasionando a diferença de 746,65% a mais de batimentos cardíacos identificados que a base de dados). O limiar adaptativo não se mostra muito diferente, computando muitos batimentos cardíacos em sinal ruidoso (ocasionando a diferença de 65,20% maior que a base de dados).

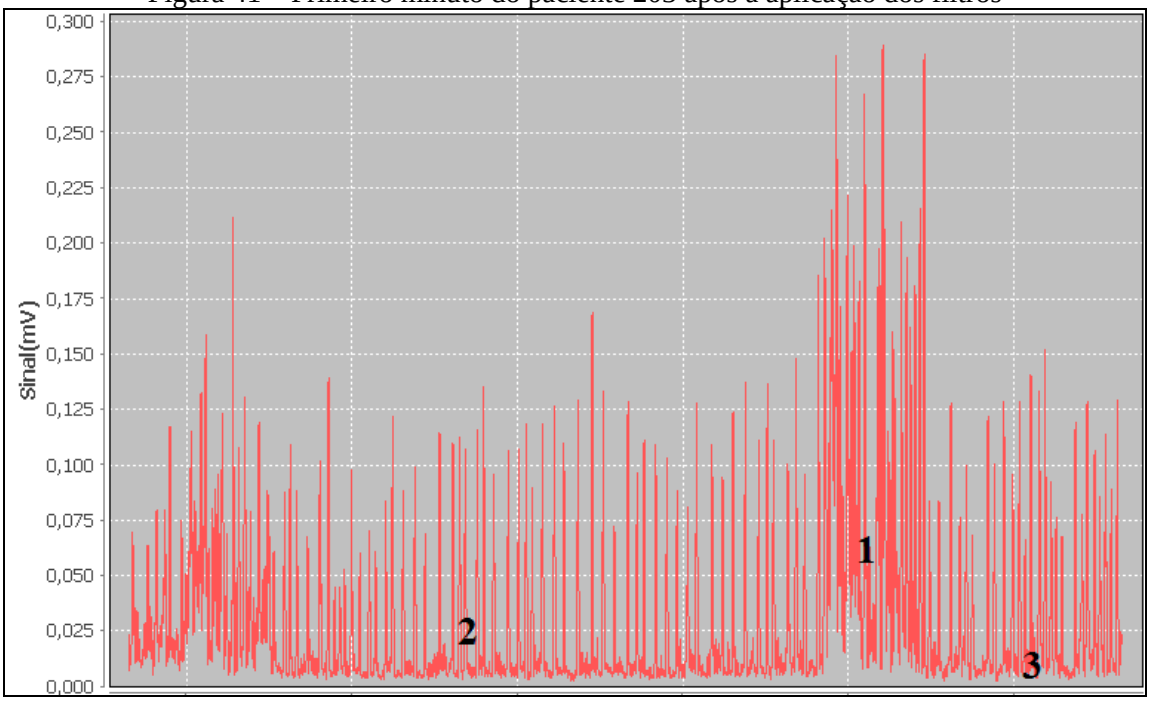
O segundo registro a ser analisado pertence ao paciente 203. A Figura 40 demonstra o primeiro minuto lido desse paciente sem a aplicação de filtros.

Figura 40 – Primeiro minuto do paciente 203 sem filtro



A Figura 40 mostra o intevalo de um minuto de leitura do paciente 203. Comparado ao mesmo intevalo de leitura do paciente 101 (ver figura 36), nota-se que incidência de ruídos é bem menor. Em seguida, a Figura 41 demonstra o mesmo período de leitura do paciente 203 após a aplicação dos filtros para eliminação de ruído.

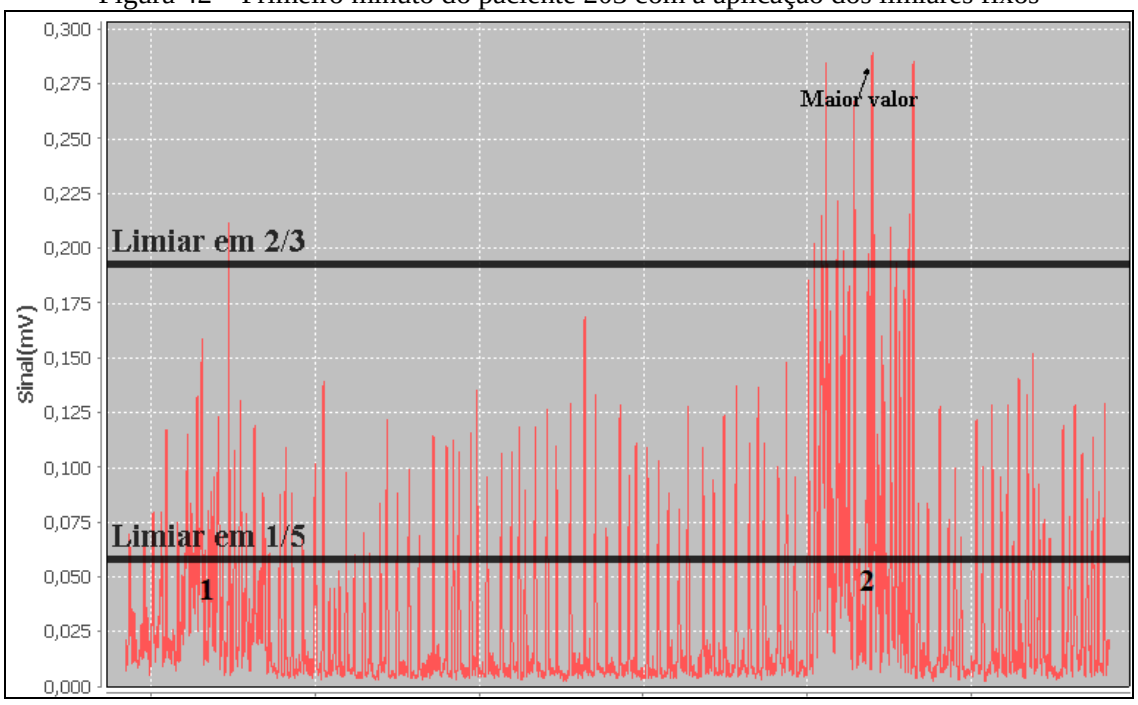
Figura 41 – Primeiro minuto do paciente 203 após a aplicação dos filtros



Conforme visto na Figura 41, o primeiro minuto de leitura do paciente 203 apresenta pequenas ocorrências de ruído (itens 1 e 2). Porém, em comparação ao mesmo período do paciente 101 (ver figura 37) as incidências de ruído são menores, bem como a elevação da linha base (item 3).

A Figura 42 mostra o último passo do algoritmo, a aplicação dos limiares fixos.

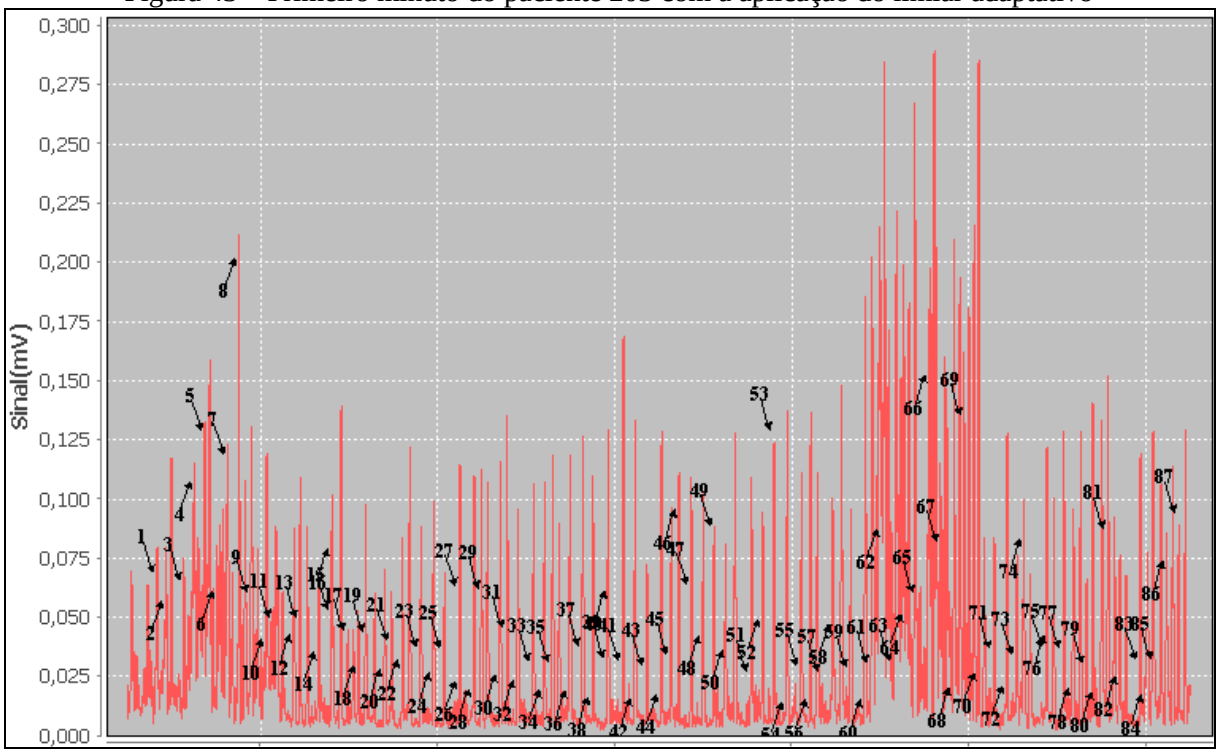
Figura 42 – Primeiro minuto do paciente 203 com a aplicação dos limiares fixos



A Figura 42 demonstra a aplicação dos dois limiares fixos no período de um minuto do paciente 203. Pode-se visualizar na figura que o limiar em 2/3 está muito alto, ocasionando a diferença de 92,55% inferior a base de dados. Apesar de o limiar em 1/5 ser aplicado em uma área com ruído (itens 1 e 2), a linha base desse paciente está mais próximo de zero do que o paciente 101 (devido a pouco ocorrência de ruído). O fator da linha base inferior ao paciente 101 influencia na quantidade de batimentos reconhecidos no paciente 203, resultando na diferença de apenas 4,30% a mais que a base de dados. Outro ponto que deve-se observar é que nesse período de um minuto do paciente 203, há uma quantidade de interferência elétrica bem menor que o mesmo período de leitura do paciente 101.

A Figura 43 demonstra o último passo a ser aplicado no registro 203, a aplicação do limiar adaptativo.

Figura 43 – Primeiro minuto do paciente 203 com a aplicação do limiar adaptativo



A Figura 43 demonstra a aplicação do algoritmo com limiar adaptativo no registro do paciente 203. Em comparação ao mesmo período do paciente 101, nota-se uma redução de batimentos cardíacos reconhecidos em sinal ruidoso (como por exemplo os batimentos 6, 10, 64, 65, 67 e 70).

Analisando ambos os registros, nota-se a importância dos filtros passa-alta e passa-baixa. Sua eficácia influencia diretamente na etapa seguinte, a tomada de decisão, onde é feito o reconhecimento dos batimentos cardíacos. Nos dois casos analisados, os filtros obtiveram resultados fracos (devido a alta taxa de ruído) repercutindo em uma análise incorreta pelos

métodos de limiar adaptativo e fixo. Em relação ao limiar fixo, o péssimo resultado do limiar em 2/3 se deve ao fato de ser um limiar muito alto, não contabilizando a maioria dos batimentos cardíacos existentes nos registros testados. O limiar fixo em 1/5 possui resultados melhores comparados ao limiar de 2/3 pelo seu valor, porém, a existência de ruídos nos registros mesmo após a aplicação dos filtros acaba influenciando nos resultados obtidos através da aplicação desse limiar. O limiar adaptativo é o que mais se adequa aos registros, porém, pode-se notar que há a identificação de batimentos cardíacos em sinal ruidoso nos dois pacientes analisados, resultando em uma análise incorreta do sinal lido.

4 CONCLUSÕES

Este trabalho propôs o desenvolvimento de um sistema de identificação de batimentos cardíacos utilizando sistemas operacionais Windows (módulo emissor) e Android (módulo receptor), utilizando comunicação *bluetooth* e desenvolvido em linguagem Java.

Como fonte de dados de sinais vitais, foi utilizado a MIT-BIH *Arrhytmia Database* (Physionet, 2012), sendo composta de 48 amostras de 24 minutos cada, coletadas de 47 pacientes (sendo que um dos paciente possui duas amostras na base).

Para identificação de batimentos cardíacos na base de dados, foi utilizado algoritmos de filtragem de sinais (passa-alta e passa-baixa) para eliminação a atenuação de sinais indesejados para que se pudesse computar a quantidade de batimentos cardíacos existentes em cada registro da base, utilizando algoritmos de limiar fixo e adaptativo.

Os resultados obtidos pelo sistema de identificação de batimentos cardíacos foram considerados razoáveis. Os algoritmos de filtragem obtiveram resultados ruins na maioria dos registros, não removendo os sinais ruidosos em boa parte da base de dados. Em relação aos algoritmos de identificação propostos, ambos (limiar fixo e limiar adaptativo) tiverem seus resultados prejudicados pela existência de ruídos no sinal filtrado, sendo o algoritmo de limiar adaptativo o que mais se aproxima dos resultados da base de dados, seguido pelo algoritmo de limiar fixo em 1/5 e por último o algoritmo de limiar fixo em 2/3.

Os trabalhos correlatos de Sampaio (2011), Ferreira et al. (2012) e Bhatia et al. (2010) não fornecem dados comparáveis com o trabalho proposto. No entanto, os resultados adquiridos em testes demonstram que o sistema de identificação de batimentos cardíacos proposto neste trabalho pode ser aplicado facilmente nos cenários idealisados pelos trabalhos correlatos. A combinação de um emissor de sinal cardíaco comunicando com um *smartphone* via *bluetooth* torna o projeto versátil e de fácil utilização.

Destaca-se como limitação do projeto o funcionamento do módulo emissor em máquinas virtuais Java em 32 bits (devido a limitação da biblioteca utilizada BlueCove). No módulo receptor, somente é possível efetuar a conexão *bluetooth* em dispositivos já pareados (a aplicação não efetua pareamento automatizado). A análise de sinais vitais é feita no período de um em um minuto.

As dificuldades encontradas no decorrer do desenvolvimento do projeto foram a ineficiênca do emulador Android para *desktop* (não possui suporte ao *bluetooth*, dificultando o processo de depuração do módulo receptor) e a ausência de um especialista na área de

cardiologia (que responderia as muitas perguntas levantadas no desenvolimento desse projeto).

Por fim, este trabalho apresenta um conjunto de funcionalidades que podem ser empregados em muitos cenários, inclusive nos cenários descritos pelos trabalhos correlatos. Este trabalho pode servir de base para futuros trabalhos na área de identificação de arritmias cardíacas, item muito importante para o diagnóstico preciso de um cardiologista.

4.1 EXTENSÕES

Como sugestão de extensão para o sistema de identificação de batimentos cardíacos propõem-se:

a) melhorar a técnica de filtragem de sinais vitais;
b) melhorar a técnica de identificação de batimentos cardíacos;
c) calcular a frequência cardíaca em um intervalo menor de tempo (sugestão de 3 em 3 segundos);
d) desenvolver a integração do sistema de identificação de batimentos cardíacos com *hardwares* específicos de emissão de sinais vitais via *bluetooth*;
e) desenvolver um gráfico diário das frequências cardíacas computadas pelo sistema;
f) desenvolver uma comunicação direta entre o aplicativo emissor e um outro aplicativo utilizado por um cardiologista.

REFERÊNCIAS BIBLIOGRÁFICAS

AGUIAR, Rogério O. **Classificação não supervisionada de sinais de eletrocardiograma**. 2006. 84 f. Trabalho de Conclusão de Curso (Graduação em Engenharia Elétrica) – Departamento de Engenharia Elétrica, Universidade Federal do Espírito Santo, Espírito Santo.

AMARAL, Nilcéia N. et al. Assistência domiciliar à saúde (*home health care*): sua história e sua relevância para o sistema de saúde atual. **Revista Neurociências,** São Paulo, v. 9, n. 3, p. 111-117, set. 2001.

AZEVEDO, Décio F. de. **Iniciação à eletrocardiografia**. Porto Alegre: Artes Médicas Sul Ltda, 1999.

BHATIA, Dinesh et al. **A pervasive health monitoring system for connected health**. [S.l.], [2010]. Disponível em: <http://ama-ieee.embs.org/2010conf/wp-content/themes/ieee/papers/March%2023%20-%20AM/Bhatia%20Abstract%2063.pdf>. Acesso em: 27 mar. 2013.

BORN, Ricardo S. **Filtros adaptativos aplicados a sinais biomédicos**. 2000. 52 f. Trabalho de Conclusão de Curso (Bacharelado em Informática) – Instituto de Física e Matemática, Universidade Federal de Pelotas, Pelotas.

CARVALHO, Marco A. S. **Um sistema de monitoramento remoto de pacientes usando rede sem fio**. 2005. 180 f. Dissertação (Pós-Graduação em Ciência da Computação) - Curso de Pós-Graduação em Ciências da Computação, Universidade de Minas Gerais, Belo Horizonte.

CHEN, H. C; CHEN. S. W. A moving average based filtering system with its application to real-time QRS detection. In: COMPUTERS IN CARDIOLOGY, 30th, 2003, Greece. **Proceedings…** Greece: Computers in Cardiology, 2003. p. 585-588.

CIGAN, Bostjan. **QRS detector implementation in Java using the WFDB Swig library**. [S.l.], [2011]. Disponível em: <http://zerocool.is-a-geek.net/?p=62>. Acesso em: 09 out. 2013.

CORREIA, Rafael J. P. **Monografia do seminário computação móvel na saúde**. São Paulo, 2008. Disponível em: <http://grenoble.ime.usp.br/~gold/cursos/2008/movel/monoSemCorrecao/RafaelCorreia.pdf>. Acesso em: 19 abr. 2013.

FERREIRA, Paulo et al. **Sistema de sensorização móvel e controlo baseado em ZigBee para bicicletas elétricas**. [S.l.], [2012]. Disponível em: <http://repositorium.sdum.uminho.pt/bitstream/1822/21626/1/SAAEI2012-PF.pdf>. Acesso em: 06 mar. 2013.

INSTITUTO DE ARRITMIAS CARDÍACAS. **Arritmias cardíacas.** [S.l.], 2009. Disponível em: <http://www.arritmiascardiacas.com.br/arritmia.html>. Acesso em: 20 nov. 2013.

KLABUNDE, Richard E. **Electrocardiogram chest leads (unipolar)**. [S.l.], [2008]. Disponível em: <http://www.cvphysiology.com/Arrhythmias/A013c.htm>. Acesso em: 16 nov. 2013.

LECHETA, Ricardo R. **Google Android: aprenda a criar aplicações para dispositivos móveis com o Android SDK**. 3. ed. São Paulo: Novatec, 2013.

MOODY, George R.; MARK Roger R. The impact of the MIT-BIH arrhytmia database. **Engineering in Medicine and Biology Magazine**, [S.l.], n. 20, p. 45-50, May/June 2001.

MURAD, Deniset L. et al. **Sistema de monitoramento em grupo**. São Paulo, 2009. Disponível em: <http://www.inicepg.univap.br/cd/INIC_2009/anais/arquivos/RE_0292_1239_03.pdf>. Acesso em: 20 abr. 2013.

PHYSIONET. **MIT-BIH arrhytmia database**, [S.l.], [2012?]. Disponível em <http://physionet.org/physiobank/database/mitdb/>. Acesso em: 12 jun. 2013

ROCHA, Adson F. Processamento de sinais biológicos. In: Carvalho, J. L. A.; Berger, P. A.; Nascimento, F. A. O. Informática e Saúde. Paraná: Eduel, 2008. p. 381-416.

SAMPAIO, Ítalo C. **Sistema de monitoramento remoto de pacientes implementado em hardware de arquitetura ARM**. 2011. 56 f. Trabalho de Conclusão de Curso (Graduação em Engenharia de Teleinformática) – Departamento de Engenharia de Teleinformática, Universidade Federal do Ceará, Ceará.

VIER, Ana P. **Sistema de eletrocardiograma (ECG) com reconhecimento de arritmias**. 2008. 49 f. Trabalho de Conclusão de Curso (Graduação em Engenharia de Computação) – Núcleo de Ciências Exatas e Tecnológicas, Universidade Positivo, Curitiba.

VOLPATO, Edgar E. **Processamento digital de eletrocardiograma:** estudo e implementação de um detector de arritmias cardíacas. 2005. 73 f. Trabalho de Conclusão de Curso (Bacharelado em Ciência da Computação) – Centro de Tecnologia, Universidade Federal de Santa Maria, Rio Grande do Sul.