UNIVERSIDADE FEDERAL DE SANTA CATARINA
PROGRAMA DE PÓS-GRADUAÇÃO EM
ENGENHARIA ELÉTRICA

Gilson de Moura Turchiello

PLATAFORMA VISUAL DE PROCESSAMENTO DIGITAL DE SINAIS BIOMÉDICOS

Florianópolis
2014

Gilson de Moura Turchiello

PLATAFORMA VISUAL DE PROCESSAMENTO DIGITAL DE SINAIS BIOMÉDICOS

Dissertação submetida ao Programa de Pós-Graduação em Engenharia Elétrica da Universidade Federal de Santa Catarina para a obtenção do Grau de Mestre em Engenharia Elétrica.
Orientador: Prof. Dr. José Marino Neto.
Coorientador: Prof. PhD. Jefferson Luiz Brum Marques.

Florianópolis
2014

Ficha de identificação da obra elaborada pelo autor,
através do Programa de Geração Automática da Biblioteca Universitária da UFSC.

Turchiello, Gilson de Moura
Plataforma Visual de Processamento Digital de Sinais Biomédicos / Gilson de Moura Turchiello ; orientador, José Marino Neto ; coorientador, Jefferson Luiz Brum Marques. - Florianópolis, SC, 2014.
81 p.

Dissertação (mestrado) - Universidade Federal de Santa Catarina, Centro Tecnológico. Programa de Pós-Graduação em Engenharia Elétrica.

Inclui referências

1. Engenharia Elétrica. 2. Engenharia Biomédica. 3. Processamento Digital de Sinais. 4. Sinais Biomédicos. 5. Software Educacional. I. Neto, José Marino. II. Marques, Jefferson Luiz Brum. III. Universidade Federal de Santa Catarina. Programa de Pós-Graduação em Engenharia Elétrica. IV. Título.

Gilson de Moura Turchiello

PLATAFORMA VISUAL DE PROCESSAMENTO DIGITAL DE SINAIS BIOMÉDICOS

Esta Dissertação foi julgada adequada para obtenção do Título de Mestre, e aprovada em sua forma final pelo Programa de Pós-Graduação em Engenharia Elétrica.

Florianópolis, 19 de dezembro de 2014.

_____
Prof. Carlos Galup Montoro, Dr.
Coordenador do Programa de Pós-Graduação em Engenharia Elétrica

_____
Prof. José Marino Neto, Dr.
Orientador
Universidade Federal de Santa Catarina

_____
Prof. Jefferson Luiz Brum Marques, ph.D.
Coorientador
Universidade Federal de Santa Catarina

Banca Examinadora:

_____
Prof. José Marino Neto, Dr.
Orientador
Universidade Federal de Santa Catarina

_____
Profa. Daniela Ota Hisayasu Suzuki, Dra.
Universidade Federal de Santa Catarina

_____
Prof. Fernando Mendes de Azevedo, Dr.
Universidade Federal de Santa Catarina

_____
Prof. Domitila Augusta Huber, Dra.
Universidade Federal de Santa Catarina

À família, professores e amigos.

AGRADECIMENTOS

À família, em especial aos meus pais e irmãos, pelos incentivos.

Aos professores José Marino Neto e Jefferson Luiz Brum Marques pelas orientações e apoio no desenvolvimento deste trabalho.

Ao Instituto de Engenharia Biomédica da UFSC, através de seus professores, funcionários e alunos, pela oportunidade de crescimento profissional e pessoal.

Aos amigos e colegas pelo companheirismo, momentos de descontração, ajudas e troca de experiências.

RESUMO

Os sinais biomédicos transportam informações que representam o estado dos sistemas vivos. Deste modo, a monitorização e interpretação destes sinais têm valor significativo para o diagnóstico médico e para a pesquisa clínica. Entretanto, para extrair as informações de interesse dos sinais é fundamental submetê-los a etapas de processamento pois, em sua maioria, estão contaminados por ruídos. Estes ruídos são oriundos das mais variadas fontes de interferência e impossibilitam uma análise imediata dos sinais. Este trabalho apresenta o desenvolvimento de uma plataforma computacional de processamento digital de sinais para uso em ensino e pesquisa em sinais biomédicos. Trata-se de uma ferramenta gráfica interativa, dinâmica e de simples utilização que faz uso de figuras geométricas, denominadas blocos funcionais. Através do ambiente gráfico da plataforma é possível criar diagramas em blocos, que irão descrever o fluxo de dados entre as funções, para realizar o processamento digital do(s) sinal(is) de interesse. Cada bloco realiza uma determinada função que pode ser modificada/controlada através de seus parâmetros, que estão acessíveis ao usuário para configuração. O processamento do projeto desenvolvido é realizado automaticamente a medida que informações são inseridas/excluídas do projeto e ferramentas permitem a visualização gráfica dos sinais em todos os pontos do projeto. Sinais estes que podem ser gerados pela própria plataforma ou exportados a partir de arquivos que contenham sinais anteriormente digitalizados. Através da plataforma é possível, por exemplo, projetar filtros digitais por meio de diferentes técnicas a fim de obter o melhor resultado. A plataforma surge também como aliada nas atividades de ensino, pois pode ser utilizada em disciplinas de processamento digital de sinais, auxiliando na apresentação dos conceitos fundamentais de processamento de sinais. Ao final do trabalho, foi realizado um questionário para avaliar a satisfação do usuário em relação a aspectos específicos do sistema desenvolvido. Onze usuários responderam ao questionário no qual obteve-se em 60,7% das perguntas notas consideradas ótimas, corroborando para uma boa aceitação da ferramenta.

Palavras-chave: Processamento digital de sinais. Sinais biomédicos. Software educacional.

ABSTRACT

The biomedical signals carry information that represent the state of living systems. Thus, monitoring and interpretation of these signals have significant value for the medical diagnosis and clinical research. However, to extract the information of interest signals is fundamental submitting them to processing steps because, in most cases, are contaminated by noise. These noises are coming from the most diverse sources of interference and make it impossible an immediate analysis of the signals. This paper presents the development of a computational platform for digital signal processing for use in teaching and research in biomedical signals. It is an interactive, dynamic and easy to use graphical tool that makes use of geometric figures, called functional blocks. Through the graphical environment of the platform is possible to create block diagrams that will describe the data flow between functions, to perform digital processing of the signals of interest. Each block performs a specific function that can be modified/controlled by its parameters that are accessible to the user for configuration. The processing of the project developed is automatically performed as the information is entered/deleted from the project and tools allow the graphical display of signs at all points of the project. These signals can be generated by the platform itself or exported from files that containing previously digitized signals. Through the platform is possible, for example, designing digital filters through of different techniques in order to get the best result. The platform also arises as an ally in teaching activities, since it can be used in courses in digital signal processing, assisting in the presentation of the fundamental concepts of signal processing. At the end of the work, a questionnaire was conducted to evaluate user satisfaction in relation to specific aspects of the developed system. Eleven users answered the questionnaire in which it was found in 60.7% of the questions notes considered optimal, corroborating a good acceptance of the tool.

Keywords: Digital signal processing. Biomedical signals. Educational software.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

A/D	–	Analógico para Digital
ADC	–	*Analog-to-Digital Converter* - Conversor Analógico para Digital
API	–	*Application Program Interface* - Interface de Programação de Aplicativos
bpm	–	Batimentos por Minuto
bB	–	Decibéis
DSP	–	*Digital Signal Processing* - Processamento Digital de Sinais
ECG	–	Eletrocardiograma
FC	–	Frequência Cardíaca
F_C	–	*Cutoff Frequency* - Frequência de Corte
FIR	–	*Finite Impulse Response* - Resposta Finita ao Impulso
F_S	–	*Sampling Frequency* - Frequência de Amostragem
GUI	–	*Graphic User Interface* – Interface Gráfica do Usuário
Hz	–	Hertz - Ciclos por Segundo
IBI	–	*Interbeat Interval* – Intervalo entre Batimentos
IDE	–	*Integrated Development Environment* - Ambiente de Desenvolvimento Integrado
IEB-UFSC	–	Instituto de Engenharia Biomédica da UFSC
IIR	–	*Infinite Impulse Response* - Resposta Infinita ao Impulso
IP	–	Intervalo de Pulso
JDK	–	*Java Development Kit*
JRE	–	*Java Runtime Environment*
JVM	–	*Java Virtual Machine* - Máquina Virtual Java
MCR	–	*MATLAB Compiler Runtime*
NA	–	Não Atende
PA	–	Filtro Passa-Altas
PB	–	Filtro Passa-Baixas
PF	–	Filtro Passa-Banda
POO	–	Programação Orientada a Objetos
QUIS	–	*Questionnaire for User Interaction Satisfaction* -
RF	–	Filtro Rejeita-Banda
SD	–	*Standard Deviation* – Desvio Padrão
SNR	–	*Signal-to-Noise Ratio* - Relação Sinal-Ruído
T_S	–	*Sampling Interval* - Intervalo de Amostragem
UFSC	–	Universidade Federal de Santa Catarina

SUMÁRIO

1 INTRODUÇÃO

O Processamento de Sinais é a análise ou manipulação de sinais a fim de extrair e/ou evidenciar informações neles contidas. Tal processamento pode ocorrer em sinais contínuos (analógicos) ou em sinais discretos (digitais), onde é conhecido como processamento digital de sinais (DSP – *Digital Signal Processing*). O DSP é uma das áreas mais importantes de processamento de sinais e suas técnicas evoluíram rapidamente devido ao desenvolvimento da microeletrônica, conquistando espaço em aplicações nas diversas áreas (ROSA, 2009). Uma destas áreas é a da Medicina e afins. Na Engenharia Biomédica o processamento de sinais é realizado em sinais oriundos de processos fisiológicos (no caso de aplicações em medicina) e/ou de fontes biológicas, chamados sinais biomédicos ou biosinais (MUTHUSWAMY, 2003).

Os sinais biomédicos (geralmente elétricos, mecânicos ou químicos) transportam informações relativas a um ou mais sistemas biológicos (RANGAYYAN, 2002; CERUTTI, 2011; SEMMLOW, 2012). Estas informações representam o estado dos sistemas vivos de modo que a monitorização e interpretação destes sinais têm valor significativo para o diagnóstico médico e pesquisadores para obterem informações relativas à saúde e doenças de seres vivos (BRUCE, 2001; HADDAD, 2009).

A aquisição dos sinais biomédicos é feita através de sistemas de medição que utilizam-se de transdutores para transformar a forma de energia do sinal fisiológico em um sinal de grandeza elétrica, normalmente analógica. O sinal analógico pode ainda passar por etapas de amplificação e filtragem, para adequação do sinal, antes da digitalização por conversores analógico/digital (RANGAYYAN, 2002; CERUTTI, 2011).

Porém, esta medição não é uma tarefa trivial. O ruído é inerente a qualquer sistema de medição, ou seja, onde há sinal também existirá ruído. Nos sinais biomédicos, frequentemente, o ruído limita sua utilidade. Eles estão sujeitos a várias fontes de ruído, principalmente devido à baixa amplitude dos sinais fisiológicos no(s) ponto(s) de medição. Deste modo, como ruído é informação indesejada presente no sinal, outros sinais biomédicos podem estar presentes em um determinado sinal e por isso serem considerados como ruídos (*e.g.*, a medida do sinal de eletrocardiograma (ECG) pode sofrer interferência de sinais musculares, EMG) (BRUCE, 2001; RANGAYYAN, 2002; ROCHA, 2008; SEMMLOW, 2012). Outra fonte de ruído bastante

comum é a rede de energia elétrica que causa ruído nos sinais com frequência igual a rede de energia (*e.g.*, 60 Hz).

As técnicas de processamento de sinais possibilitam minimizar estes ruídos, evidenciando a informação desejada do sinal. O DSP também pode ser utilizado para extrair e analisar as informações dos sinais.

Contudo, processamento de sinais é uma área complexa que demanda alto conhecimento de ferramentas matemáticas como, por exemplo, transformada Z, convolução, transformada de Fourier, transformada Wavelet e filtros digitais (ROSA, 2009). Alguns *softwares* auxiliam no desenvolvimento de ferramentas para o DSP, com destaque ao MATLAB®. Porém, uma grande variedade de algoritmos e técnicas podem ser encontradas, e que podem ter diferentes características conforme as particularidades do sinal a ser processado e da informação a ser extraída (SMITH, 1999).

O uso de programas que contenham *scripts* prontos de funções específicas (ferramentas matemáticas) pode simplificar e reduzir o tempo de projeto de um *software* para processamento digital de sinais biomédicos. Uma plataforma digital (*software*) pode, ainda, ser uma ferramenta auxiliar no ensino de disciplinas correlacionadas ao DSP e sinais biomédicos.

1.1 MOTIVAÇÃO

O desenvolvimento de algoritmos para o processamento digital de sinais não é uma atividade trivial, pois requerem conhecimento de alguma linguagem de programação específica, a qual, geralmente, requer o uso de licença para sua utilização. Estes fatores contribuem para o tempo de projeto e custos adicionais para o seu desenvolvimento, mesmo tratando-se de projetos acadêmicos. Além disso, estes algoritmos são desenvolvidos para satisfazerem as aplicações específicas, que podem sofrer modificações durante seu desenvolvimento. Isso leva à necessidade de alterações nas linhas de código para adequar-se as novas necessidades de projeto.

Além disso, o processamento de sinais precisa ser ensinado em uma gama cada vez mais ampla de cursos de graduação e pós-graduação (CAMPBELL, 2001; SPANIAS, 2001). A pós-graduação em Engenharia Biomédica, por ser uma área multidisciplinar, recebe alunos provenientes de várias formações distintas que podem nunca terem tido contato com DSP, para os quais é necessário o ensino de algumas técnicas específicas de DSP utilizadas na área. O ensino de DSP (tanto

em graduação ou pós-graduação) envolve conceitos teóricos matemáticos que podem causar dificuldades de aprendizado aos alunos (SMITH, 1999). O uso auxiliar de linguagens de programação em alto nível tem siso bastante utilizado, mas também pode surgir como uma barreira no desenvolvimento das atividades de ensino dada a complexidade destas ferramentas e não trazer benefícios adicionais à compreensão dos conceitos básicos de processamento digital de sinais (CAMPBELL, 2001).

1.2 OBJETIVOS

1.2.1 Objetivo Geral

O objetivo deste trabalho é o desenvolvimento de uma plataforma computacional visual para auxiliar no desenvolvimento de sistemas e procedimentos de processamento digital de sinais biomédicos de forma dinâmica e simples através de blocos funcionais configuráveis.

1.2.2 Objetivos Específicos

- Desenvolver uma plataforma computacional dinâmica que permita a inserção/manipulação de ferramentas de processamento digital de sinais;
- Desenvolver as ferramentas de DSP com parâmetros configuráveis pelo usuário;
- Implementar um conjunto de ferramentas de DSP, como filtros, interpolação e decimação de sinais, operações matemáticas e geradores de sinais;
- Implementar ferramentas que permitam importar sinais digitalizados salvos de arquivos;
- Implementar ferramentas de análise de sinais biomédicos;
- Implementar ferramentas de visualização gráfica dos sinais manipulados;
- Ilustrar a aplicação da ferramenta desenvolvida.

1.3 JUSTIFICATIVA

A matemática tradicionalmente utilizada no ensino de disciplinas de processamento digital de sinais é necessária e não pode ser dispensada inteiramente. No entanto, atualmente os alunos estão mais

propensos a ter habilidades em programação e se familiarizarem com ferramentas computacionais do que unicamente com a matemática subjacente destes processos (CAMPBELL, 2001; SMITH, 1999). Há várias metodologias e modelos para promover o aprendizado "eficiente". Embora discordantes entre si, grande parte dos autores destas metodologias e modelos concordam que a experimentação tem grande importância no processo de aprendizagem, o que pode ser facilmente alcançado com o uso de uma plataforma digital tal como a proposta neste trabalho.

Desta forma, uma plataforma visual apresenta-se como uma atraente ferramenta de ensino e pesquisa na área de processamento digital de sinais, onde alunos de graduação, pós-graduação e professores falam uma linguagem comum, simples e acessível a todos, sem a necessidade de conhecimentos específicos em alguma linguagem de programação de alto nível. Um *software* com blocos funcionais que podem ser interligados na forma de diagrama em blocos, e com seus parâmetros editáveis, pode tornar as atividades de ensino e pesquisa em DSP, aplicadas à engenharia biomédica, muito mais dinâmicas e interativas, proporcionando uma melhor compreensão dos princípios básicos de DSP (SPANIAS, 2001; CAMPBELL, 2001).

Outra vantagem do processamento digital de sinais é sua flexibilidade em relação aos sistemas analógicos, pois é fácil e conveniente realizar manipulações e transformações matemáticas em sinais digitais do que em analógicos. Além disso, algumas aplicações biomédicas não exigem processamento em tempo real e uma vez adquiridos, os sinais digitais, podem facilmente ser armazenados e/ou transmitidos de forma confiável, permitindo a manipulação através de diferentes técnicas a fim de obter o melhor resultado (RANGAYYAN, 2002).

Concomitantemente, a disponibilidade de blocos funcionais, orientados ao processamento de sinais biomédicos, permite ao usuário construir e simular uma grande variedade de sistemas em menor tempo, permitindo modificações rápidas. Além do mais, este tipo de plataforma pode vir a ser integrada com módulos eletrônicos (*hardware*) de aquisição de dados e processar sinais em tempo real através das portas de comunicação de um computador (microprocessador), transformando assim qualquer plataforma computacional em um instrumento biomédico.

1.4 ORGANIZAÇÃO DO TRABALHO

O Capítulo 2 apresenta a fundamentação teórica relacionada a sinais biomédicos. Este capítulo dá ênfase em sinais eletrocardiográficos pois estes foram utilizados em testes das ferramentas desenvolvidas na plataforma.

O Capítulo 3 apresenta a fundamentação teórica sobre processamento digital de sinais.

O Capítulo 4 descreve os materiais e métodos empregados na concepção da plataforma.

No Capítulo 5 são descritas a plataforma e as ferramentas concebidas.

O Capítulo 6 apresenta uma discussão sobre o desenvolvimento da plataforma.

No Capítulo 7 serão apresentadas as conclusões acerca da realização deste trabalho e as sugestões para trabalhos futuros.

2 SINAIS BIOMÉDICOS

Os sinais biomédicos transportam informações de variáveis fisiológicas e são relevantes no contexto clínico e de pesquisa. Diversas são as fontes geradoras destes sinais que podem ser classificados em (COHEN, 2000): bioelétricos, de bioimpedância, bioacústicos, biomagnéticos, biomecânicos, bioquímicos e bioópticos.

Destes, os sinais bioelétricos são os mais utilizados. As fontes geradoras de sinais bioelétricos são células nervosas e musculares que estão amplamente presentes nos sistemas fisiológicos dos seres vivos. O campo elétrico, resultante dos potenciais de ação gerados pelas células, se propagam pelo meio biológico e pode ser medido, por exemplo, através de eletrodos afixados na superfície do corpo sem a necessidade de métodos invasivos. Na Tabela 1 é possível verificar as características de frequência e amplitude dos sinais bioelétricos mais comuns.

Tabela 1 – Faixa de frequência e de amplitude dos principais sinais bioelétricos.

Classificação	**Faixa de Frequência**	**Faixa Dinâmica**
Potencial de ação	100 Hz - 2 kHz	10 μV - 100 mV
Eletroneurograma (ENG)	100 Hz - 1 kHz	5 μV - 10 mV
Eletrorretinograma (ERG)	0,2 - 200 Hz	0,5 μV - 1 mV
Eletrooculograma (EOG)	dc - 100 Hz	10 μV - 5 mV
Eletroencefalograma (EEG)		
Superfície	0,5 - 100 Hz	2 - 100 μV
Faixa delta	0,5 - 4 Hz	
Faixa teta	4 - 8 Hz	
Faixa alfa	8 - 13 Hz	
Faixa beta	13 - 22 Hz	
Fusos do sono	6 - 15 Hz	50 - 100 μV
Complexos K	12 - 14 Hz	100 - 200 μV
Eletromiograma (EMG)		
Fibra única (SFEMG)	500 Hz - 100 kHz	1 - 10 μV
Potencias de ação da unidade motora (MUAP)	5 Hz - 10 kHz	100 μV - 2 mV
EMG de superfície (SEMG)		
Músculo esquelético	2 - 500 Hz	50 μV - 5 mV
Músculo liso	0,01 - 100 Hz	10 μV - 100 mV
Eletrocardiograma (ECG)	0,05 - 100 Hz	1 - 10 mV
ECG de alta resolução	25 Hz - 1 kHz	10 μV - 2 mV

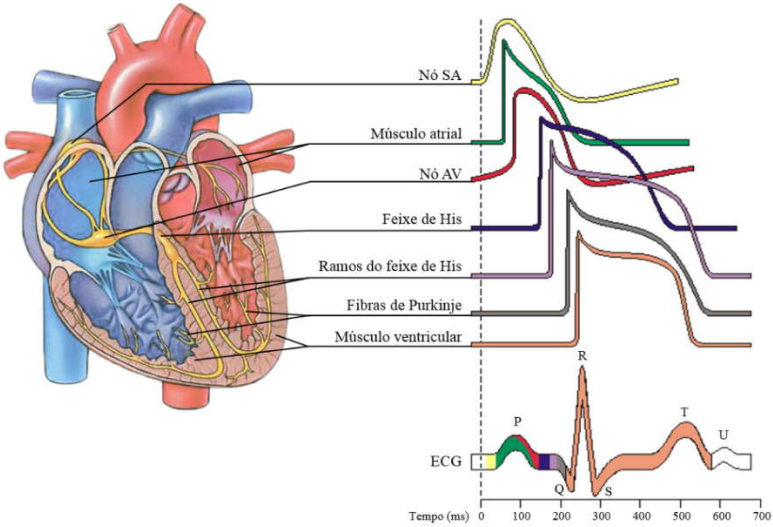
Fonte: Adaptado de COHEN, 2000.

Cada órgão apresenta sinais biomédicos distintos (morfologia) e a análise destes sinais pode ajudar no diagnóstico de anomalias. Um destes sinais é o eletrocardiograma (ECG) que reflete a atividade elétrica das células do coração, foco de aplicação deste trabalho.

2.1 ELETROCARDIOGRAMA

O eletrocardiograma é o registro temporal da atividade elétrica das células do coração obtido na superfície do corpo através de eletrodos. No ECG, cada ciclo cardíaco é constituído por um conjunto de ondas características que representam atividades em determinada parte do coração, como observa-se na Figura 1. O ECG normal possui três componentes principais: a onda P, o complexo QRS (combinação das ondas Q, R e S) e a onda T.

Figura 1 – Potenciais de ação ao longo do coração resultando no sinal de ECG padrão.



Fonte: Adaptado de MALMIVUO & PLONSEY, 1995.

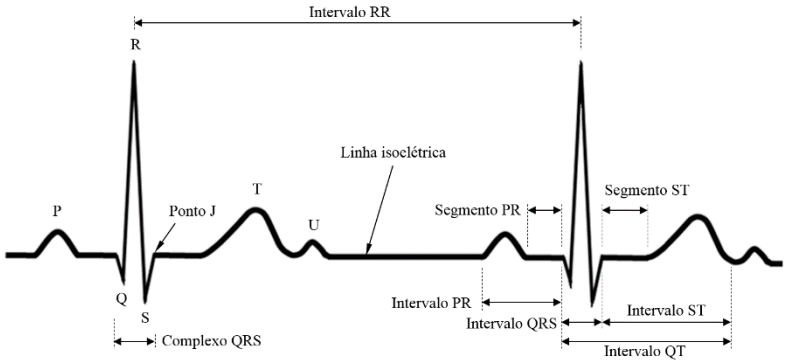
A onda P corresponde a atividade de despolarização dos átrios que antecede a contração dos mesmos. O complexo QRS é gerado pela onda de despolarização ventricular. Por fim, a onda T representa a repolarização (relaxamento) dos ventrículos. A onda de repolarização

atrial geralmente não é visível pois está incorporada ao complexo QRS. A onda U nem sempre está presente no sinal do ECG, sendo mais visível nas derivações unipolares precordiais V_3 e V_4. Ela é de baixa frequência e geralmente possui de 5% a 25% da amplitude onda T, com a mesma polaridade.

Uma informação importante obtida a partir do ECG é o intervalo entre os ciclos cardíacos consecutivos, denominado de intervalo RR ou intervalo de pulso (IP) (ver Figura 2). A frequência cardíaca (FC), calculada em batimentos por minuto (bpm), pode ser aproximada em função do intervalo de pulso projetado em um minuto: FC = 60 / IP, onde a unidade de IP é segundos.

Uma condição normal vai depender de muitas variáveis, e o que é considerado normal em uma pessoa pode não ser normal em outra. Entretanto, existem alguns intervalos de valores que podem informar uma normalidade válida para um conjunto significativo de pessoas. Estes valores são descritos a seguir para alguns dos parâmetros obtidos do sinal do ECG (Figura 2):

Figura 2 – Ondas, segmentos e intervalos do eletrocardiograma.



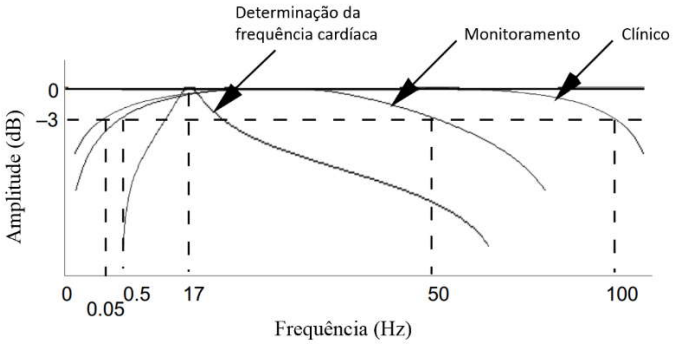
Fonte: Autor.

a) ***intervalo PR***: é medido do início da onda P até o início do complexo QRS e representa o tempo de condução atrioventricular. O valor normal varia entre 0,12 segundos (s) e 0,20 s (PETRY, 2006);
b) ***intervalo QRS***: é medido desde o início da onda Q até o final da onda S e corresponde ao tempo total da despolarização ventricular. Valor máximo de 0,10 s;

c) ***intervalo QT***: é medido desde o início da onda Q até o final da onda T e corresponde ao tempo total da despolarização e repolarização ventricular. Seu valor varia com a frequência cardíaca. O valor mínimo é 0,35 s e o valor máximo para homens é 0,44 s e para mulheres 0,46s (PETRY, 2006);
d) ***intervalo ST***: é medido desde o ponto J até o final da onda T;
e) ***ponto J***: final do complexo QRS;
f) ***segmento PR***: é medido do fina da onda P até o início do complexo QRS. Normalmente isoelétrico;
g) ***segmento ST***: é medido desde o ponto J até o início da onda T. Pode apresentar pequeno desvio da linha isoelétrica. O valor varia de 0,05 s a 0,15 s (PETRY, 2006).

O sinal do ECG adquirido na superfície do corpo possui amplitude de aproximadamente 1 milivolt (mV) no pico da onda R e a faixa de frequência adquirida depende da aplicação desejada, como pode ser observado na Figura 3 (TOMPKINS, 1995). Aplicações clínicas utilizam o ECG com frequências na faixa de 0,05 a 100 hertz (Hz ou ciclos por segundo) enquanto aplicações de monitoração utilizam uma faixa de 0,5 a 50 Hz. Medidores de frequência cardíaca utilizam uma faixa de frequência centrada em 17 Hz pois estão interessados somente na detecção do complexo QRS (atividade elétrica de maior frequência). Aplicações preocupadas em medir potenciais tardios, por exemplo, que são de alta frequência, utilizam frequências de até 1 kHz.

Figura 3 – Faixas de frequências utilizadas em diferentes aplicações na eletrocardiografia.

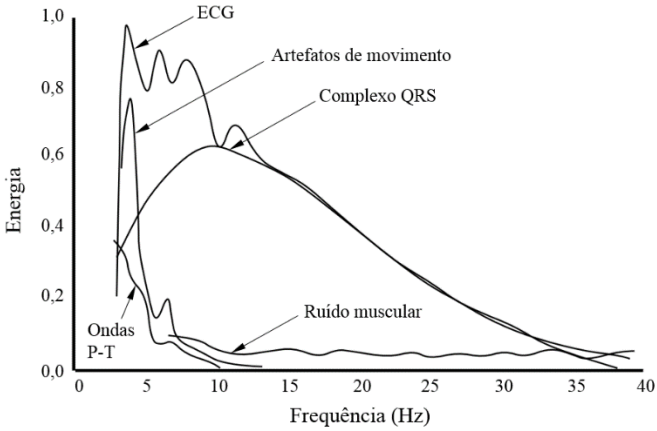


Fonte: Adaptado de TOMPKINS, 2000.

Esta delimitação de frequências na aquisição do ECG contribui para a atenuação de ruídos no sinal, causados principalmente pela movimentação dos eletrodos que estão em contato com a superfície da pele, contrações musculares e interferências da rede de energia elétrica.

A Figura 4 mostra o espectro de potência relativo do sinal de ECG, do complexo QRS, das ondas P e T, artefatos de movimento (ruído dos eletrodos em contato com a pele) e ruído muscular obtido com base numa média de 150 complexos (TOMPKINS, 2000). Através deste espectro é possível verificar que o complexo QRS apresenta maior energia em torno de 10 Hz.

Figura 4 – Espectro de potência do ECG, do complexo QRS, das ondas P e T, do artefato de movimento e do ruído muscular.



Fonte: Adaptado de TOMPKINS, 2000.

3 PROCESSAMENTO DIGITAL DE SINAIS

Os sinais biomédicos não podem ser diretamente analisados uma vez que frequentemente estão encobertos por outros sinais biomédicos ou contaminados por ruídos. Portanto, um pré-processamento geralmente faz-se necessário para melhorar a qualidade das informações de interesse antes da extração destas informações (MAINARDI, 2000).

Processamento de sinais é a manipulação ou transformação das informações contidas em um sinal. Este processamento pode ser realizado em sinais analógicos (contínuos) ou digitais (discretos). Neste último, os sinais são compostos por um conjunto de valores amostrados (digitalizados) a partir de sinais contínuos ou gerados por dispositivos digitais.

Uma grande parte dos sistemas baseados em circuitos analógicos passaram a ser implementados em sistemas digitais devido ao progresso ocorrido na tecnologia digital (tanto em *hardware* quanto em *software*). De fato, as técnicas digitais de processamento tornaram-se mais eficientes e flexíveis do que as analógicas e apresentam várias vantagens: maior desempenho, capacidade de implementação de algoritmos complexos, parâmetros de projeto podem ser facilmente alterado, não sofrem efeito do envelhecimento e tolerância dos componentes e da temperatura, podem facilmente ser reproduzidos, menos susceptível a ruídos, entre outras.

3.1 SINAIS

Um sinal é um fenômeno variável que pode ser medido. Ele define a variação de uma grandeza física, que contém a informação, como função de uma ou mais variáveis (ROSA, 2009).

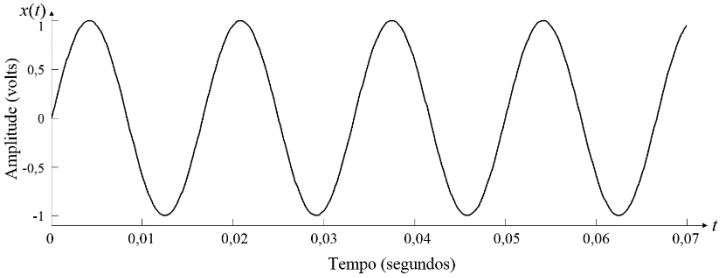
A maioria dos sinais presentes na natureza são contínuos, mas frequentemente necessitam ser representados, processados e armazenados em dispositivos digitais, tais como os computadores. Os sinais representados em dispositivos digitais são discretos e, geralmente, são uma aproximação de sinais contínuos que foram digitalizados (SEMMLOW, 2012; WEEKS, 2012).

3.1.1 Sinais Contínuos

Um sinal contínuo é um sinal que contém no mínimo uma variável independente que é contínua, geralmente o tempo. No sinal contínuo é possível obter um valor em qualquer instante arbitrário de

tempo, como ocorre no caso da leitura da temperatura em um termômetro de mercúrio, por exemplo. Por convenção, uma função no tempo é representada por $x(t)$. A Figura 5 mostra um exemplo de sinal contínuo que representa uma senóide com frequência de 60 Hz.

Figura 5 – Sinal de tempo contínuo (senóide de 60 Hz).



Fonte: Autor.

3.1.2 Sinais Discretos

Um sinal digital nada mais é do que uma lista de valores discretos. O sinal digital é obtido a partir do sinal contínuo através do processo de amostragem (*sampling*) ou digitalização.

Um sinal discreto é representado por $x(n)$. n é um inteiro que representa o número da amostra do sinal e se relaciona com o tempo através da Equação (1).

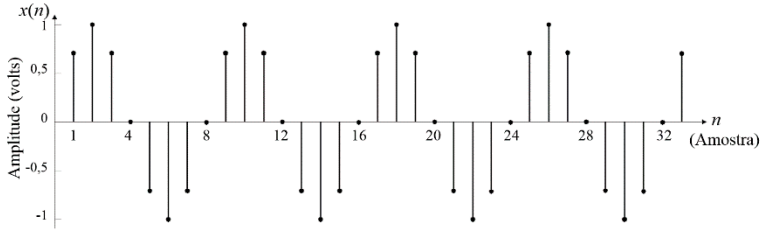
$$t = nT_S \tag{1}$$

onde T_S é um intervalo de tempo conhecido e constante chamado período de amostragem. O inverso do período de amostragem é a taxa de amostragem ou frequência de amostragem (F_S), dada em Hz.

Desta forma, a cada intervalo de tempo T um valor do sinal é amostrado e numerado sequencialmente com n. Consequentemente, os sinais discretos possuem valores conhecidos somente em instantes nTs de tempo e valores indefinidos para qualquer outro instante de tempo.

A Figura 6 mostra o sinal da Figura 5 amostrado a 480 Hz, ou 8 vezes a frequência do sinal. Isso quer dizer que são obtidas 480 amostras do sinal em um segundo, resultando em T = 2,083 milissegundos.

Figura 6 – Sinal de tempo discreto (senóide de 60 Hz amostrada em 480 Hz).

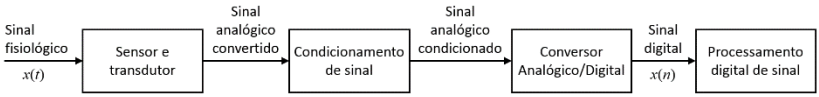


Fonte: Autor.

3.2 AQUISIÇÃO DE SINAIS (DIGITALIZAÇÃO)

A digitalização de um sinal analógico pode ser simplificada em três etapas, conforme pode ser observado na Figura 7: a captação/transdução, o condicionamento e a digitalização propriamente dita.

Figura 7 – Diagrama de blocos simplificado do processo de aquisição e digitalização de um sinal fisiológico. A etapa de condicionamento de sinal pode incluir etapas de amplificação, filtragem e limitação de banda do sinal (filtro *anti-aliasing*).



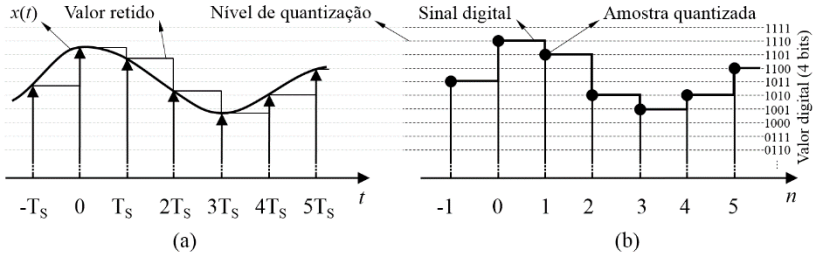
Fonte: Autor.

Os sinais fisiológicos são detectados por sensores (principalmente eletrodos na superfície da pele) e convertidos pelos transdutores em sinais elétricos. Após, normalmente, ocorre o condicionamento do sinal para então ser convertido pelo conversor analógico/digital (conversor A/D ou ADC – *Analog-to-Digital Converter*). Este condicionamento é realizado através de etapas de amplificação e filtragem analógica de tal forma a não comprometer as informações de interesse contidas no sinal. Nesta etapa também deve ser realizada a limitação da banda de frequência do sinal a fim de reduzir o efeito de *aliasing*. Este efeito é causado pelas altas frequências do sinal que são refletidas para uma

região de mais baixa frequência durante a digitalização, provocando distorção no sinal amostrado (WEEKS, 2012).

Por fim, o sinal é digitalizado pelo circuito ADC, resultando em uma sequência de amostras que podem ser processadas em tempo real e/ou armazenadas. Este processo é dividido em duas etapas (MAINARDI, 2000; ROCHA, 2008): a amostragem, onde o sinal contínuo é convertido em uma série de tempo discreto (amostras), e a quantização que atribui o valor discretizado da amplitude de cada amostra. A Figura 8 traz um exemplo que ilustra estas duas etapas do processo de conversão A/D de um sinal.

Figura 8 – Processo de conversão analógico para digital de um sinal. (a) Etapa de amostragem. (b) Etapa de quantização.



Fonte: Autor.

As informações contidas nos sinais biomédicos podem identificar ou indicar a presença de patologias, portanto o processo de conversão deve ser realizado com a mínima distorção possível no sinal para que ele venha a ser reconstruído e interpretado de forma apropriada (RATHKE, 2008). A amostragem e a quantização apresentam diferentes distorções (KUO, 2001): a amostragem introduz *aliasing* enquanto a quantização resulta ruído causado pelo erro de quantização. Para que o sinal possa ser reconstruído adequadamente as características de amostragem e quantização devem ser observadas.

3.2.1 Amostragem

O sinal contínuo é amostrado em instantes uniformemente espaçados de tempo T_S, conforme a Equação (1). No processo de amostragem o sinal contínuo é convertido para um sinal de tempo

discreto, porém com valor de amplitude ainda contínua, como observa-se na Figura 8(a).

Durante a digitalização, as informações existentes entre duas amostras consecutivas são perdidas. O aumento da frequência de amostragem é uma das formas de reduzir o efeito de *aliasing* e obter um sinal digital mais fiel ao sinal contínuo. Entretanto, há um limite máximo de frequência imposto pelo *hardware* (conversor A/D, memória e processador digital). Para cada aplicação existe uma frequência de amostragem mínima que é obtida respeitando o teorema de *Nyquist*, abordado em breve.

3.2.2 Quantização

Um quantizador é um sistema que transforma uma sequência de entrada $x(nTs)$, que possuem amplitude contínua mas tempo discreto (ver Figura 8(a)), em uma sequência na qual cada valor de entrada assume um número de uma gama finita de valores possíveis para a amplitude (ver Figura 8(b)) (HAYES, 1999). Esta gama de valores corresponde a 2^B níveis (normalmente igualmente espaçados), onde B é o número de *bits* que o ADC tem para representar cada amostra (KUO, 2001). Para o caso do exemplo mostrado na Figura 8 o ADC possui 4 *bits*, logo a amplitude de cada amostra será quantizada em um dos 16 valores possíveis ($2^4 = 16$).

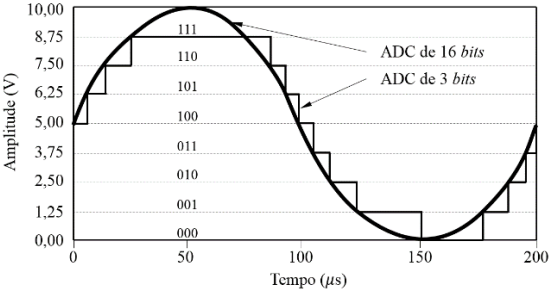
Consequentemente, cada amplitude discreta é codificada em uma palavra digital (binária) de tamanho B *bits* que é entendida pelo processador de sinais como uma sequência de números inteiros sequencialmente armazenados em uma memória (KUO, 2001; ROCHA, 2008).

O número de *bits* de um conversor está relacionado com sua resolução. Para um conversor de 8 *bits* e tensão de referência de 5 volts a resolução é a mínima variação de tensão detectada pelo conversor, produzindo variação no código digital, e será: $5/2^8 \approx 0{,}01953$ V. Para um conversor de 12 *bits*, nas mesmas condições, a resolução passa a ser $\approx 0{,}001221$ V. Assim, quanto maior o número de *bits* do ADC mais fiel será a digitalização e menos informação é perdida. A Figura 9 mostra o efeito da quantização de um sinal com dois conversores A/D de diferentes resoluções.

Todo valor amostrado é arredondado para o nível de quantização mais próximo (ver amostras na Figura 8), ocasionando o erro de

quantização. Este erro, que é a diferença entre o valor real amostrado e o valor quantizado, não pode ser removido e aparece como ruído na saída.

Figura 9 – Efeito da quantização de um sinal (senoide de 5 kHz) com ADCs de 16 e 3 *bits* de resolução.



Fonte: Adaptado de NATIONAL INSTRUMENTS, 2013.

O valor máximo do erro para cada amostra corresponde a meio intervalo de quantização (ou metade da resolução) e será menor quanto maior for o número de *bits* do ADC, pois maior será a quantidade de níveis disponíveis para a quantização do sinal e menor a diferença entre os níveis. Um valor aproximado da relação sinal-ruído (SNR - *Signal-to-Noise Ratio*), dado em decibéis (dB), é obtido em função do número de *bits* do conversor A/D (RATHKE, 2008; KUO, 2001): 6B bB. A Tabela 2 mostra a relação do número de *bits* adequado para a conversão de alguns dos sinais biomédicos.

Tabela 2 – Faixas de frequência, de amplitude e de quantização de alguns sinais biomédicos.

Sinal	**Faixa de frequência (Hz)**	**Amplitude**	**Quantização (*bits*)**
Eletroencefalograma	0,2 - 50	600 μV	4 - 6
Eletrooculograma	0,2 - 15	10 mV	4 - 6
Eletrocardiograma	0,15 - 150	10 mV	10 - 12
Eletromiograma	20 - 8000	10 mV	4 - 8
Pressão sanguínea	0 - 60	400 mmHg	8 - 10
Espirograma	0 - 40	10 L	8 - 10
Fonocardiograma	5 - 2000	80 dB	8 - 10

Fonte: MUSEN & VAN BEMMEL, 1997.

3.2.3 Teorema de Nyquist

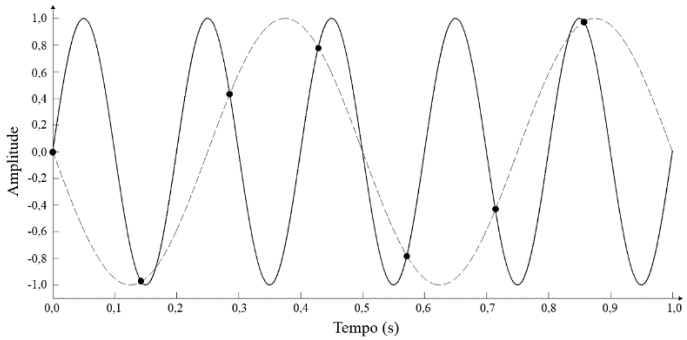
Partindo do princípio de que estamos interessados em todas as frequências a partir de 0 Hz até o componente de maior frequência (F_{max}), a largura de banda é igual ao componente de maior frequência. O sinal é limitado em F_{max} por meio de filtro analógico anti-*aliasing* inserido antes do ADC e, idealmente, não permite frequências fora da largura de banda definida.

O teorema de *Nyquist* diz que um sinal contínuo, limitado em banda, pode ser completamente reconstruído sem distorções se ele for amostrado com uma frequência de no mínimo duas vezes a frequência de maior interesse, conforme a Equação (2):

$$F_S > 2\ F_{max} \tag{2}$$

Se esta regra não for obedecida, e o sinal de entrada possuir frequências maiores que a metade da taxa de amostragem, o sinal será deformado de uma maneira irreversível (ROSA, 2009). A frequência mais alta permitida em um sinal amostrado, ou seja $F_S/2$, é conhecida como frequência de *Nyquist*. A Figura 10 mostra o efeito do *aliasing* causado pela baixa frequência de amostragem (SEMMLOW, 2012).

Figura 10 – Efeito do *aliasing* em sinais digitalizados. Onda senoidal de 5 Hz (linha contínua) amostrada a 7 Hz (pontos) sendo interpretada como uma senoide de 2 Hz (linha pontilhada) após a amostragem.



Fonte: Adaptado de SEMMLOW, 2012.

Neste exemplo, uma onda senoidal de 5 Hz (linha contínua) foi amostrada em 7 Hz (pontos), porém é possível verificar que uma onda

senoidal de 2 Hz (linha pontilhada) também passa pelos pontos amostrados. Esta onda se trata na realidade do fenômeno de *aliasing*, provocado pela reflexão das frequências acima da frequência de *Nyquist* para abaixo dela.

É comum a amostragem em 3 a 5 vezes a F_{max}, aumentando o espaçamento entre as frequências do sinal original e as geradas pelo processo de amostragem, entretanto nem sempre isso é necessário ou possível devido as limitações de *hardware*, já mencionadas (SEMMLOW, 2012).

3.3 FILTROS DIGITAIS

Os filtros são sistemas que possuem a capacidade de alterar seletivamente a forma de onda, removendo (e/ou atenuando) certos componentes de frequência de um sinal. Eles podem ser analógicos ou digitais. Filtros analógicos são implementados com base em componentes eletrônicos discretos, tais como amplificadores operacionais, resistores e capacitores. Os filtros digitais, por sua vez, usam um processador digital para realizar cálculos numéricos, implementados através de *software*, nos sinais discretos.

Os filtros digitais possuem algumas vantagens em relação aos analógicos (ROSA, 2009):

a) são mais fáceis de projetar, testar, modificar e implementar, pois são feitos em *software*;
b) não estão sujeitos a efeitos ambientais, como a variação de temperatura;
c) possuem certas características de resposta amplitude/frequência não possíveis em filtros analógicos;
d) tanto os dados filtrados quanto os não filtrados podem ser armazenados para utilização posterior;
e) vários canais de entrada podem se filtrados por um único filtro, sem necessidade de replicação de *hardware*;
f) filtros adaptativos podem auto ajustar sua resposta em frequência com o tempo;
g) permitem a implementação de filtros complexos e mais precisos;
h) podem ser implementados para frequências muito baixas;
i) favorecem a compactação de *hardware*.

Quanto a duração da resposta ao impulso, os filtros digitais podem ser divididos em duas classes: FIR (*Finite Impulse Response* ou

Resposta Finita ao Impulso) e IIR (*Infinite Impulse Response* ou Resposta Infinita ao Impulso). Os filtros FIR são não recursivos, onde a saída atual é uma função das amostras de entrada passadas e presente. Os filtros IIR, por sua vez, são recursivos e a saída depende das entradas passadas, presente e também das saídas passadas.

A escolha entre filtros FIR e IIR irá depender das características desejadas do filtro, tais como:

a) filtros FIR possuem resposta de fase linear e não provocam distorção de fase nos sinais enquanto filtros IIR possuem resposta de fase não linear;
b) filtros FIR são estáveis, o que nem sempre é possível nos filtros IIR;
c) filtros FIR demandam um número maior de coeficientes para uma atenuação mais acentuada, em comparação ao filtro IIR;
d) filtro FIR são mais fáceis de sintetizar com resposta de frequência arbitrária (não apenas filtros seletores);
e) filtros IIR possuem equivalência com filtros analógicos, e estes podem ser facilmente transformados em filtros IIR com especificações similares.

O cálculo dos coeficientes do filtro é realizado após a especificações das características desejadas do filtro.

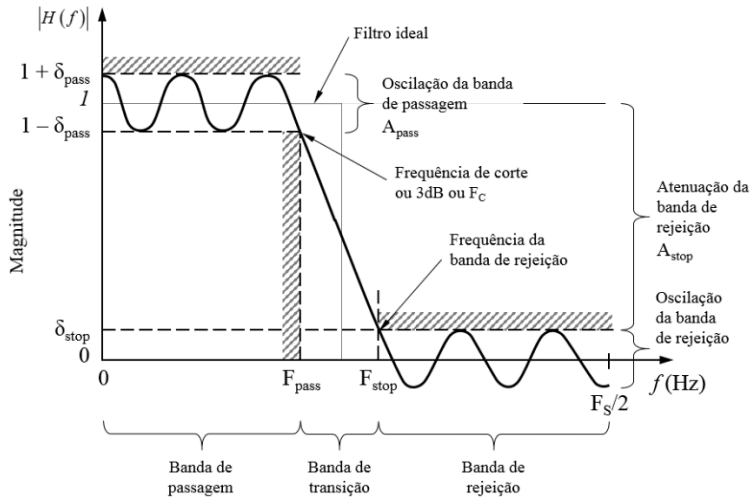
3.3.1 Especificações dos Filtros

Com relação a banda de passagem de frequência do filtro, há quatro tipos de filtros: passa-baixas (*lowpass* ou PB), passa-altas (*highpass* ou PA), passa-banda (*bandpass* ou PF) e rejeita-banda (*bandstop* ou RF). Como os próprios nomes indicam, os filtros passa-baixas permite a passagem de frequências abaixo de um determinado valor, denominado de frequência de corte (F_C). Os filtros passa-altas deixam passar frequências acima da frequência de corte, enquanto os filtros passa-banda e rejeita-banda permitem a passagem e rejeição, respectivamente, de uma determinada faixa de frequências, delimitada por duas frequências de corte.

Em filtros ideais, o ganho é fixo para as frequências de interesse (geralmente unitário) e zero para qualquer outra frequência na banda de rejeição. No entanto, é impossível obter um corte brusco entre a banda de passagem e a banda de rejeição. Há portanto uma banda de transição entre estas duas bandas, como pode ser observado na Figura 11.

As especificações de um filtro são feitas através em um modelo, desenhando suas características fundamentais e ajustando os valores que são requisitos para o filtro, conforme o tipo que está sendo projetado (ROSA, 2009). Um modelo real para o filtro passa-baixa é apresentado na Figura 11 juntamente com suas especificações de projeto, onde:

Figura 11 – Parâmetros de projeto do filtro real passa-baixas.



Fonte: Autor.

δ_{pass} : Amplitude do *ripple* da banda de passagem
δ_{stop} : Amplitude do *ripple* da banda de rejeição
F_{pass} : Frequência limite da banda de passagem
F_{stop} : Frequência limite da banda de rejeição
Fs : Frequência de amostragem

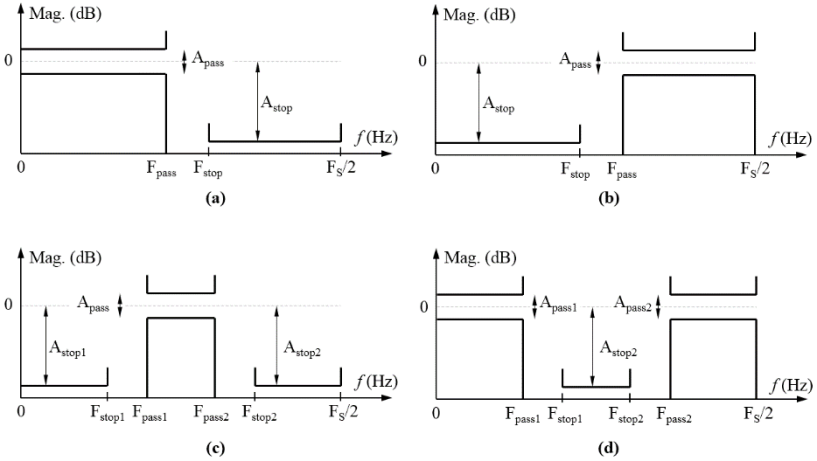
As amplitudes δ_{pass} e δ_{stop} são expressas em números ordinários. Para obter os parâmetros de amplitude em dB (A_{pass} e A_{stop}) deve-se utilizar as Equações (3) e (4). As frequências limites das bandas podem ser expressas em Hz ou normalizadas como uma fração de frequência de amostragem ($F_{pass/stop}$ / (F_S/2))

Os demais tipos de filtro seguem o mesmo padrão apresentado na Figura 11, modificando apenas o local(is) da(s) banda(s) de passagem e de rejeição, como mostra a Figura 12.

$$A_{pass} = 20\log_{10}\left(1+\delta_{pass}\right) \tag{3}$$

$$A_{stop} = -20\log_{10}\left(\delta_{stop}\right) \tag{4}$$

Figura 12 – Requerimentos para projeto dos filtros. (a) passa-baixas, (b) passa-altas, (c) passa-banda e (d) rejeita-banda.



Fonte: Autor.

O método usado para o cálculo dos coeficientes do filtro depende se ele é um filtro FIR ou IIR. Os métodos básicos são:

a) filtros FIR,

- janelamento (*window*): simples e flexível, porém não permite o controle adequado sobre os parâmetros do filtro,
- amostragem de frequência (*frequency sampling*): permite a realização recursiva do filtro, mas pouco eficiente na especificação e/ou controle dos parâmetros do filtro,
- otimização (*optimal*): é o mais adequado para projeto FIR;

b) filtros IIR,

- impulso invariante (*impulse invariant*): preserva a resposta ao impulso do filtro analógico mas não a resposta de amplitude-frequência, não sendo apropriado para filtros passa-altas e rejeita-banda,

- transformação bilinear (*bilinear transformation*): bastante adequado para o cálculo de coeficientes de filtros de frequência seletiva. Preserva a resposta de amplitude dos filtros analógicos, produzindo filtros digitais muito eficientes,
- posicionamento de polos e zeros (*pole-zero placement*): para filtros simples, baseia-se no posicionamento de polos e zeros por tentativa e erro.

A ordem dos filtros FIR é dada pelo número *n* de coeficientes. Já para os filtros IIR, a ordem é dada pelo número de saídas anteriores necessárias para calcular a próxima saída.

4 MATERIAIS E MÉTODOS

Neste capítulo serão descritos os *softwares* e ferramentas utilizadas para o desenvolvimento da ferramenta computacional proposta neste trabalho.

4.1 SOFTWARES E FERRAMENTAS

4.1.1 Desenvolvimento em Java

A plataforma Java, distribuída pela Oracle Corporation, é um ambiente computacional que permite desenvolver aplicativos utilizando qualquer uma das linguagens criadas para a plataforma Java, tal como a Linguagem Java. Java é uma linguagem de programação orientada a objetos (POO), desenvolvida na década de 90, que tem três plataformas principais: Java SE (*Standard Edition*), Java EE (*Enterprise Edition*) e Java ME (*Micro Edition*).

O Java é *open source* (código aberto), de programação simplificada, segura e com a grande vantagem de não estar presa a um sistema operacional ou *hardware*, permitindo a portabilidade. Os programas Java rodam através de uma máquina virtual (Máquina Virtual Java ou JVM - *Java Virtual Machine*) que traduz as instruções dos programas para o sistema operacional específico no qual está sendo emulada a JVM.

Tanto os usuários quanto os desenvolvedores precisam da JVM instalada. Em função disso, o Java é distribuído de duas formas: JRE (*Java Runtime Environment*), focado no usuário, traz o necessário para executar um aplicativo Java; e o JDK (*Java Development Kit*) que traz a máquina virtual, compilador e ferramentas e, portanto, voltado ao desenvolvedor.

4.1.2 NetBeans IDE

NetBeans IDE é um ambiente de desenvolvimento integrado (IDE) para desenvolvedores de *softwares*, que oferece as ferramentas de apoio necessárias, com o objetivo de agilizar o processo de desenvolvimento. Trata-se de uma ferramenta multiplataforma gratuita e de código aberto para desenvolvimento em linguagem Java, dentre outras.

O NetBeans, desenvolvido pela Oracle Corporation, possui um grande conjunto de bibliotecas, módulos e APIs (*Application Program*

Interface ou Interface de Programação de Aplicativos). As APIs constituem um conjunto de instruções e padrões estabelecidos por um *software* para a utilização de suas funcionalidades. Elas permitem, por exemplo, que os programadores criem janelas, desenhos geométricos, acessem arquivos, acessem *hardware*, etc.

4.1.3 JFreeChart

JFreeChart, desenvolvido pela Object Refinery Limited, é uma biblioteca Java de código aberto que permite aos desenvolvedores a criação de diversos tipos de gráficos, interativos ou não. Há vários tipos de gráficos, como os do tipo linha, de barra, pizza e gráficos combinados. Estes gráficos podem ser gerados em diversos tipos de saída como, por exemplo, componentes gráficos Java (*Swing*), arquivos de imagem (JPEG e PNG) e em arquivos de gráfico vetorial (PDF, EPS e SVG).

4.1.4 Desenvolvimento MATLAB

MATLAB® (*Matrix Laboratory* ou Laboratório de Matrizes) é uma linguagem de alto nível e um ambiente interativo para computação numérica, visualização de dados e programação, desenvolvido pela MathWorks, Inc.

Com o MATLAB é possível, por exemplo, construir gráficos, manipular funções matemáticas específicas e compilar funções. Ele possui uma variedade de bibliotecas auxiliares de ferramentas, chamadas de *toolbox*, com funções já definidas que permitem a utilização do MATLAB em uma variedade de aplicações, tais como processamento de sinais, de imagens, de vídeo, em sistemas de controle, medições, finanças computacionais e biologia computacional, entre outras.

Através de ferramentas disponíveis no MATLAB é possível desenvolver aplicações usando o MATLAB e utilizá-las em programas Java. Estas ferramentas (MATLAB Compiler e MATLAB Builder JA) permitem criar classes Java dos programas feitos em MATLAB e utilizá-las em computadores que não possuam o MATLAB instalado através do MATLAB Compiler Runtime (MCR).

O MCR é distribuído gratuitamente pela MathWorks, Inc. Trata-se de um conjunto completo e independente de bibliotecas que permitem a execução de aplicações MATLAB, fornecendo suporte a todos os recursos da linguagem e a maioria das suas *toolboxes*.

4.2 DESENVOLVIMENTO DA PLATAFORMA DE PROCESSAMENTO DIGITAL DE SINAIS

A interface gráfica da plataforma foi desenvolvida em linguagem Java, utilizando o programa NetBeans IDE na versão 7.3 com Java SE JDK na sua versão 6. As funções do MATLAB foram desenvolvidas e compiladas utilizando o *software* MATLAB na versão 7.14 (R2012a), o qual também possui a versão 6 do Java. Já a versão do MATLAB Compiler Runtime utilizada para desenvolvimento e testes foi a 7.17 (R2012a). A exibição dos gráficos dos sinais é realizada através da biblioteca JFreeChart, versão 1.0.14. Todos os *softwares* utilizados são para sistema operacional Windows® de 32 *bits*.

As ferramentas desenvolvidas para a plataforma foram avaliadas comparando-se as informações obtidas por elas na plataforma com as obtidas pelas funções no *software* MATLAB.

4.3 AVALIAÇÃO DE USABILIDADE DA PLATAFORMA

Para avaliar a satisfação do usuário na utilização da plataforma foi utilizado o questionário disponível no Apêndice A, adaptado do QUIS (*Questionnaire for User Interaction Satisfaction*). O QUIS é uma ferramenta desenvolvida pela Universidade de Maryland-USA para avaliar a satisfação subjetiva de usuários em relação a aspectos específicos da interface homem-máquina de sistemas computacionais. Originalmente o QUIS possui doze seções de perguntas, onde cada seção corresponde a um aspecto sobre usabilidade de sistemas.

O QUIS permite que apenas as seções/perguntas de interesse sejam feitas, sendo cada pergunta quantificada numa escala de 1 a 9. O questionário aplicado contém cinquenta e seis perguntas de sete seções do QUIS, que se mostraram pertinentes na avaliação do sistema desenvolvido. A primeira refere-se a experiências do usuário com computadores e as demais ao uso da plataforma. As sete seções avaliadas foram:

a) experiência anterior com computadores;
b) impressão como usuário;
c) telas;
d) terminologia e informação do sistema;
e) aprendizagem do sistema;
f) capacidade do sistema; e
g) multimídia.

Os resultados desta avaliação foram extraídos utilizando-se a escala de Likert (BOONE, 2012) nas respostas obtidas do questionário. A escala de Likert é uma escala psicométrica utilizada em pesquisa quantitativa que registra o nível de concordância ou discordância com uma dada declaração. Para análise através da escala de Likert, a escala das perguntas do questionário foi dividida em quatro grupos: ruim (1 a 3); médio (4 a 6); ótimo (7 a 9); e NA (Não Atende).

5 RESULTADOS

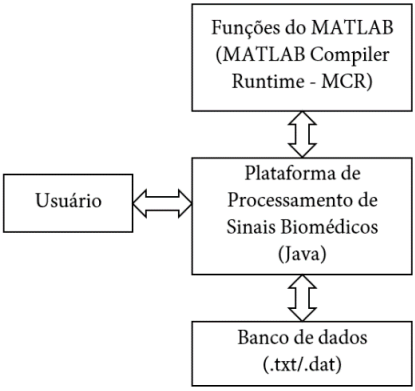
A plataforma de processamento digital de sinais biomédicos é formada por um conjunto de ferramentas desenvolvidas para auxiliar no ensino e pesquisa em projetos de DSP na área de Engenharia Biomédica. Estas ferramentas são visualmente representadas por blocos que podem ser interligados formando um diagrama de blocos, que é a representação gráfica de um processo ou modelo de um sistema mais complexo, já que cada bloco pode ser visto como um subsistema. Ligações entre blocos descrevem as relações entre eles e definem o fluxo de informação.

Este capítulo descreve o desenvolvimento da plataforma e suas ferramentas, bem como a utilização das mesmas.

5.1 ESTRUTURA GERAL

A Figura 13 mostra o fluxograma simplificado do fluxo de dados entre as partes constituintes da plataforma e com o usuário. O usuário interage com o ambiente gráfico da plataforma definindo/alterando os parâmetros das ferramentas matemáticas utilizadas no projeto de DSP. A plataforma, por sua vez, executa o processamento dos sinais (por meio do MCR) que são apresentados ao usuário e/ou utilizados nas etapas seguintes de processamento. O Apêndice B traz o diagrama de classes do projeto da plataforma.

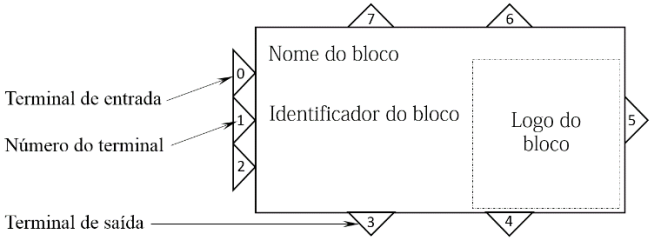
Figura 13 – Fluxograma da plataforma de processamento digital de sinais biomédicos, mostrando o fluxo de dados/informações entre as partes.



Fonte: Autor.

As ferramentas de DSP desenvolvidas são identificadas visualmente na plataforma como um retângulo, formando blocos, como mostra a Figura 14. Estes blocos são identificados por um nome, um símbolo (logo) e uma sequência numérica única para cada bloco. Os blocos possuem entradas e/ou saída, conforme a necessidade de cada função, para realizar ligações com outros blocos. O total de entradas e saídas que um bloco pode possuir é no máximo 8 (terminais 0 à 7). Um terminal de entrada é identificado por um triângulo cujo vértice está em contato com uma das arestas do retângulo do bloco (ver Figura 14). Já em um terminal de saída é a base do triângulo que está em contato com alguma das arestas do retângulo do bloco.

Figura 14 – Estrutura padrão de construção dos blocos funcionais. Esta estrutura permite a identificação do tipo do bloco e a ligação com outros blocos através dos terminais de entrada e/ou saída, disponíveis de acordo com a necessidade de cada bloco.



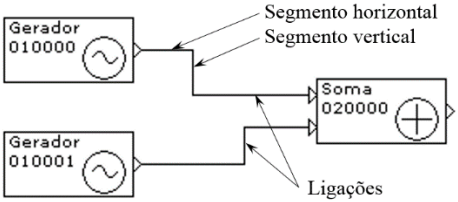
Fonte: Autor.

As ligações entre os blocos são compostas por segmentos de reta formando linhas, conforme observa-se na Figura 15. Uma ligação inicia com o clique do botão esquerdo do mouse no terminal do bloco ao qual se deseja realizar a ligação. A partir deste momento segmentos de reta podem ser inseridos na área de desenho, alternando entre segmentos horizontais e verticais a cada clique do botão esquerdo do mouse, formando linhas com curvas. O botão direito do mouse cancela o último segmento inserido e, para o caso deste ser o primeiro, cancela a ligação.

Os terminais 0, 1, 2 e 5 conectam-se a segmentos horizontais enquanto os terminais 3, 4, 6 e 7 a segmentos verticais. Uma ligação pode iniciar tanto por um terminal de entrada quanto de saída e é finalizada quando um segmento é inserido sobre um terminal de outro bloco e que este terminal não seja do mesmo tipo daquele que iniciou a

ligação. Ou seja, uma ligação só pode ser realizada entre terminais de dois blocos, sendo um terminal de saída e outro de entrada.

Figura 15 – Exemplo de ligação entre blocos. A ligação é identificada por uma linha composta por segmentos de reta, inserida com o auxílio do mouse. Ligações só são possíveis entre uma saída de um bloco e uma entrada de outro.



Fonte: Autor.

Cada tipo de bloco possui seus parâmetros que definem a função desempenhada por ele. Estes parâmetros estão acessíveis ao usuário para edição através da ação de dois cliques do mouse sobre o bloco, quando uma nova janela é mostrada com as opções do bloco. Em virtude da plataforma ser desenvolvida em linguagem de programação orientada a objetos os parâmetros definidos em dois ou mais blocos do mesmo tipo são tratados separadamente durante o processamento do projeto de DSP. Processamento este que é realizado automaticamente quando alterações são realizadas no mesmo, tais como: alterações de parâmetros das funções dos blocos; inserção/exclusão de blocos; e inserção/exclusão de ligações.

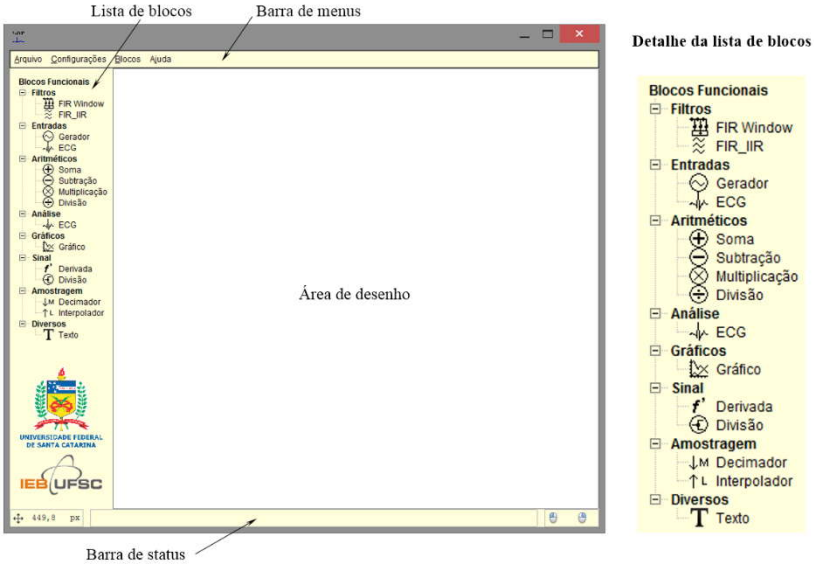
5.2 INTERFACE GRÁFICA

A Figura 16 mostra a interface gráfica da plataforma que pode ser dividida em quatro partes:

a) área de desenho: região onde os blocos são inseridos/manipulados formando um diagrama de blocos ou projeto de DSP;
b) lista de blocos: contém uma lista de todos os blocos disponíveis na plataforma para utilização em um projeto de DSP. Os blocos estão agrupados em categorias de acordo com a funcionalidade de cada um deles;
c) barra de menus: além da listagem de blocos, contém algumas configurações visuais da plataforma, tais como cores, fonte e

modo de visualização da lista de blocos. Esta barra também disponibiliza a opção de salvar e abrir projetos desenvolvidos na plataforma, como mostra a Figura 17; e,

Figura 16 – Tela principal da plataforma de processamento de sinais biomédicos desenvolvida em linguagem Java. No detalhe a lista de blocos disponíveis na plataforma.



Fonte: Autor.

Figura 17 – Detalhe das opções da barra de menus da plataforma.

Fonte: Autor.

d) barra de status: traz informações ao usuário, tais como posição do mouse na área de trabalho e informações sobre o broco/ligação sob o mouse. Ela também auxilia o usuário quanto as funções dos botões mouse durante as ligações e manipulações dos blocos, por exemplo.

Para facilitar a orientação dos blocos na área de desenho o *grid* da mesma pode ser ativado através da barra de menus ou por meio do botão direito do mouse em um local não ocupado da área de desenho.

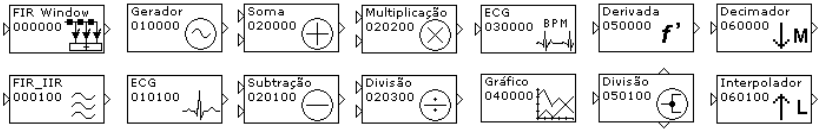
Tanto os blocos quanto os segmentos de reta das ligações podem ser facilmente movimentados dentro da área de desenho. Para realizar esta ação, é necessário posicionar o mouse sobre o componente de interesse (quando o cursor do mouse passa a ser representado por uma mão), pressionar o botão esquerdo do mouse e mantê-lo pressionado enquanto movimenta o componente. Para posicioná-lo basta liberar o botão do mouse.

Para a exclusão de blocos usa-se o botão direito do mouse quando o mesmo estiver sobre o bloco. Desta forma um menu *pop-up* irá surgir com as opções de editar os parâmetros do bloco ou excluí-lo. Se o bloco excluído possuir ligações com outros blocos elas também serão excluídas. Para a exclusão de ligações segue-se o mesmo processo, com a diferença de que deve-se clicar sobre algum segmento da ligação a ser excluída.

5.3 FERRAMENTAS (BLOCOS FUNCIONAIS)

A Figura 18 mostra a estrutura visual dos blocos desenvolvidos para a plataforma. Alguns destes blocos serão descritos com mais detalhes a seguir.

Figura 18 – Estrutura dos blocos da plataforma. Mostrando os nomes, identificadores, símbolos, entradas e saídas de cada bloco.

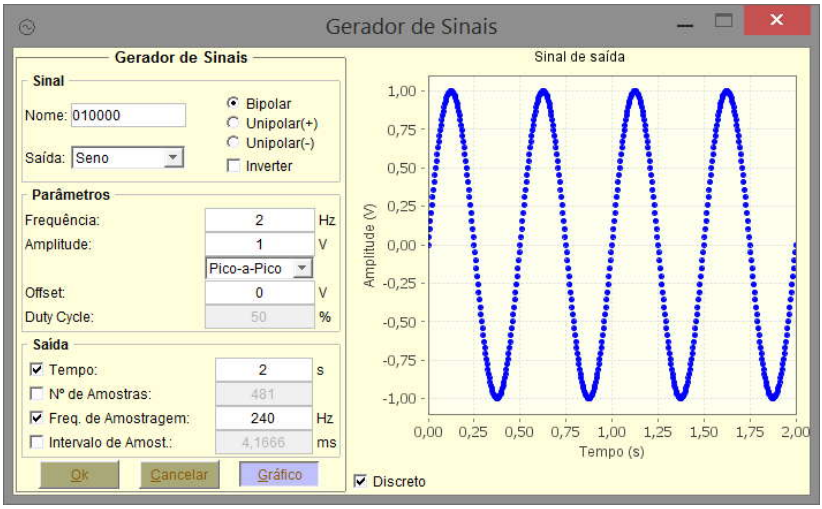


Fonte: Autor.

5.3.1 Gerador de Sinais

Este bloco gera um sinal conforme os parâmetros definidos através da interface como mostra a Figura 19. Estes parâmetros incluem, dentre ouros, o tipo do sinal (seno, cosseno, triângulo, quadrado, rampa e contínuo), frequência, amplitude, tempo e frequência de amostragem do sinal gerado.

Figura 19 – Parâmetros do bloco gerador de sinais.

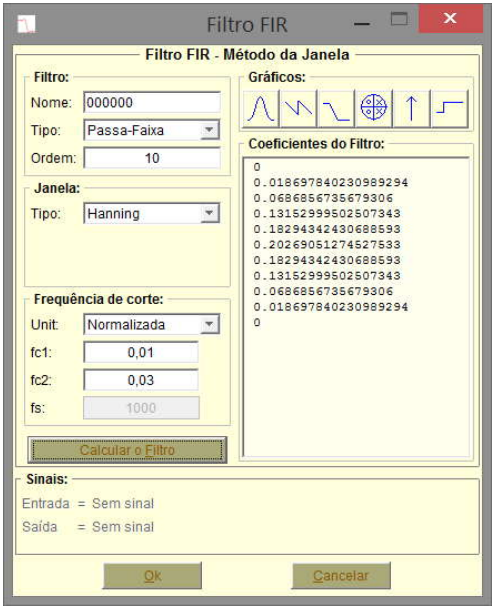


Fonte: Autor.

5.3.2 Filtros Digitais

Os filtros do tipo passa-baixas, passa-altas, passa-banda e rejeita-banda podem ser desenvolvidos a partir de dois blocos distintos. O primeiro, mostrado na Figura 20, define o projeto de filtros FIR através do método da janela (*window*). A ordem do filtro, tipo de janela e frequência(s) de corte são definidas pelo usuário, que pode ainda observar os gráficos de resposta do filtro projetado (resposta de fase, magnitude, ao impulso, ao degrau e gráfico de polos e zeros).

Figura 20 – Parâmetros do bloco para projeto de filtro FIR pelo método da janela.



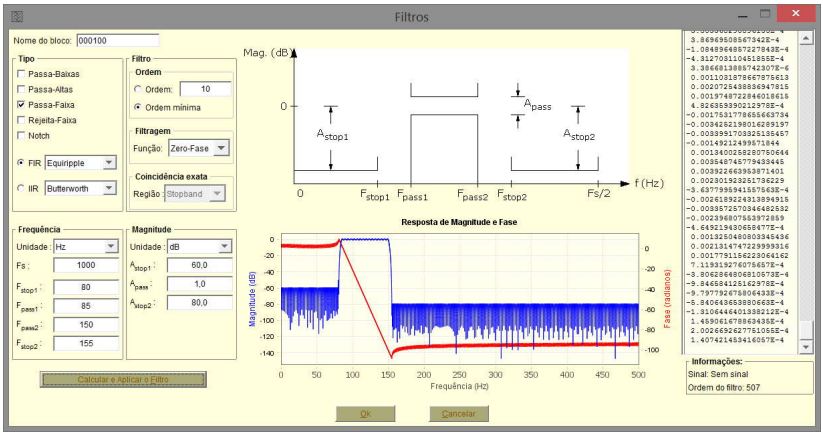
Fonte: Autor.

O segundo método de projeto (Figura 21) permite projetar filtros FIR (*equiripple*) e IIR (*butterworth, chebyshev, elliptic*) definindo os parâmetros de frequência e magnitude do filtro. Este bloco permite definir uma ordem fixa para o filtro ou que ela seja calculada conforme os parâmetros informados. O bloco traz também o desenho do filtro para referência e gráfico de resposta de magnitude e fase. Ambos os blocos retornam os coeficientes para o filtro calculado.

5.3.3 Funções Aritméticas

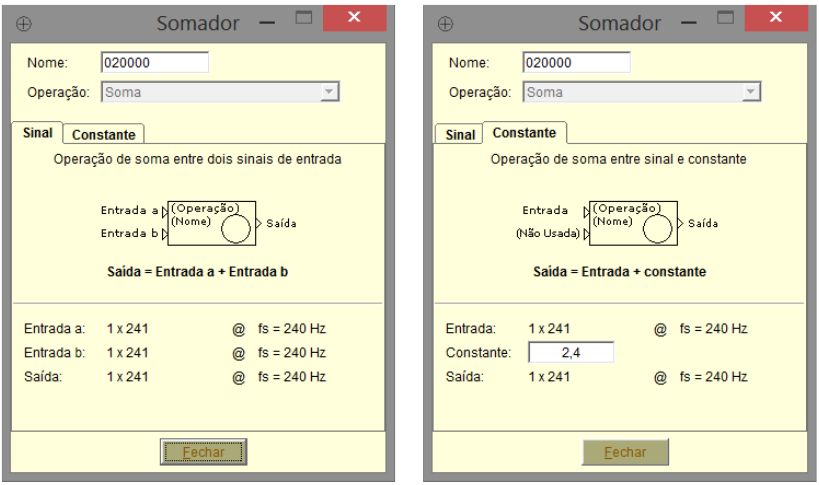
Os blocos de operações aritméticas (Figura 22) realizam operações de soma, subtração, multiplicação e divisão. Estas operações podem ser realizadas entre dois sinais (Figura 22(a) ou entre um sinal e uma constante (Figura 22(b).

Figura 21 – Parâmetros do bloco para projeto de filtros FIR e IIR.



Fonte: Autor.

Figura 22 – Parâmetros do bloco para funções aritméticas. (a) Operação soma entre dois sinais. (b) Operação soma ente um sinal e uma constante.



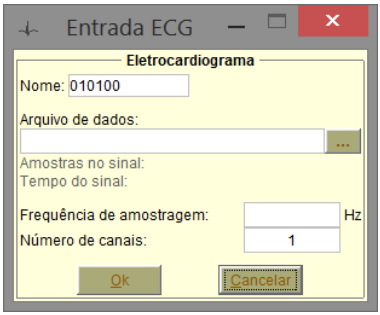
(a) (b)

Fonte: Autor.

5.3.4 Importador de Sinais

Este bloco, mostrado na Figura 23, é utilizado para importar sinais de eletrocardiograma salvos em arquivo de dados (.dat) ou de texto (.txt). Os valores das amostras presentes no arquivo selecionado devem estar separados por espaço ou nova linha para correta importação. A fim de reconstruir adequadamente o sinal deve-se informar a frequência com que ele foi amostrado durante a digitalização. Embora este bloco esteja configurado para sinais de ECG, ele pode ser utilizado para importar outros sinais que possuam a mesma formatação (valores separador por espaço em branco, tabulação ou nova linha).

Figura 23 – Bloco para importar sinais de eletrocardiograma.



Fonte: Autor.

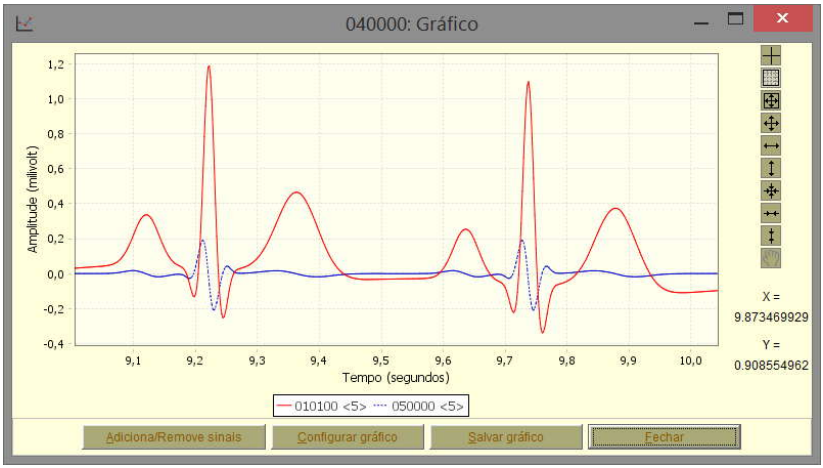
5.3.5 Visualizador de Sinais (Gerador de Gráficos)

Este bloco pode ser visualizado na Figura 24 e com ele é possível visualizar graficamente todos os sinais presentes no diagrama de blocos desenvolvido na plataforma. Ferramentas de *zoom* estão acessíveis para auxiliar na visualização dos gráficos, através da barra de ferramentas na lateral direita da tela do bloco. Através destas ferramentas o eixo X e/ou Y do gráfico pode ser expandido ou comprimido. O mouse também pode ser utilizado para aumentar ou diminuir o *zoom* aplicado na visualização.

Na maioria das situações não é necessária a visualização gráfica de todos os sinais presentes no projeto. Desta forma, através da opção "Adicionar/Remove sinais" na barra inferior da tela principal do bloco, é possível definir quais sinais serão visualizados, através da tela mostrada

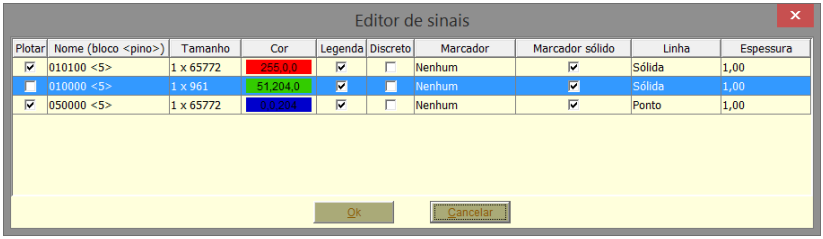
na Figura 25. Outras configurações do gráfico podem ser editadas nesta tela de forma independente para cada sinal, como por exemplo: marcações das amostras do sinal, cor, espessura e tipo da linha do gráfico.

Figura 24 – Bloco para visualização gráfica de sinais, mostrando um sinal de eletrocardiograma (vermelho) e a primeira derivada deste sinal (azul).



Fonte: Autor.

Figura 25 – Configurações dos sinais a serem visualizados no gráfico.



Fonte: Autor.

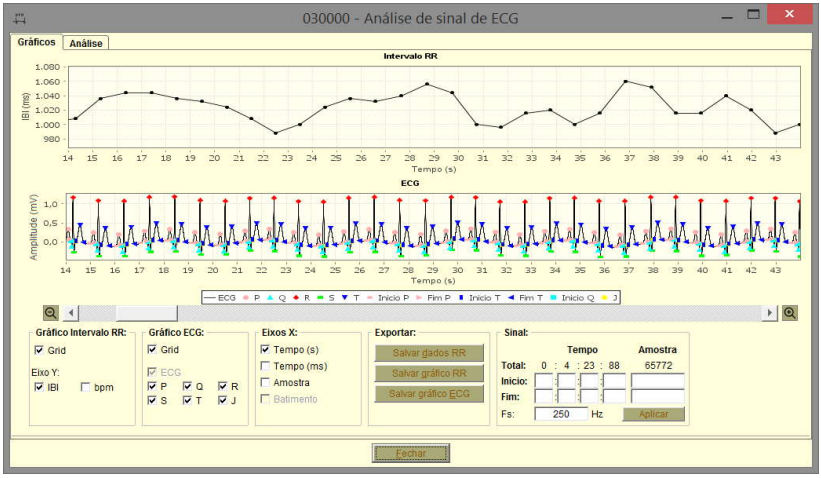
Outro conjunto de configurações está disponível através da opção "Configurar Gráfico". Estas configurações estão relacionadas o gráfico como um todo, por exemplo: legendas dos eixos e do gráfico, *grid* e escala dos eixos.

5.3.6 Análise de Sinais de Eletrocardiograma

O bloco de análise de sinais de eletrocardiograma realiza a marcação dos pontos característicos do ECG, como é mostrado na Figura 26, e computa os intervalos e segmentos obtidos a partir destes pontos (ver Figura 2). Tais pontos são: pico das ondas P, Q, R e S; início e fim das ondas P e T; início da onda Q; e ponto J. A análise aplicada pode ser limitada a um determinado intervalo de tempo do sinal de entrada, informando-se o início e o fim deste intervalo.

Através da identificação dos picos das ondas R o gráfico do intervalo RR é criado em sincronia temporal com o gráfico do sinal de ECG. A estes gráficos pode-se aplicar *zoom* (expandir ou contrair o eixo de tempo) e movimentá-los horizontalmente através da barra na parte inferior dos mesmos. As escalas dos eixos de amplitude (eixo Y) em ambos os gráficos são configurados automaticamente com base nos seus respectivos sinais e não permitem modificações.

Figura 26 – Bloco de análise de sinais de eletrocardiograma.



Fonte: Autor.

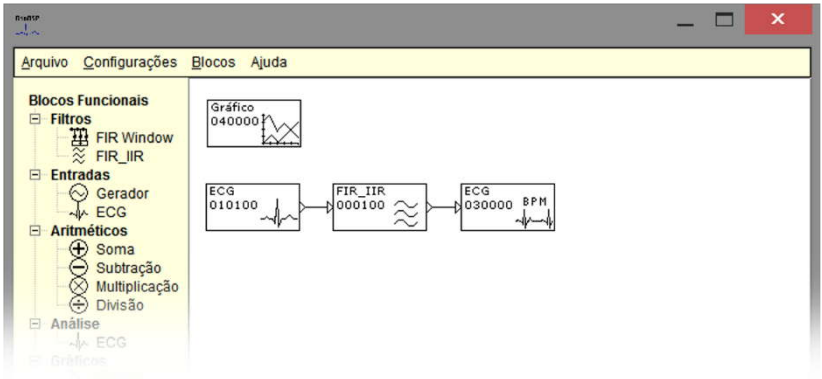
5.4 EXEMPLO DE APLICAÇÃO

5.4.1 Processamento de Sinais de ECG

A seguir é descrito o processo para criar um projeto de DSP na plataforma e realizar o processamento de um sinal de ECG (projeto mostrado na Figura 27):

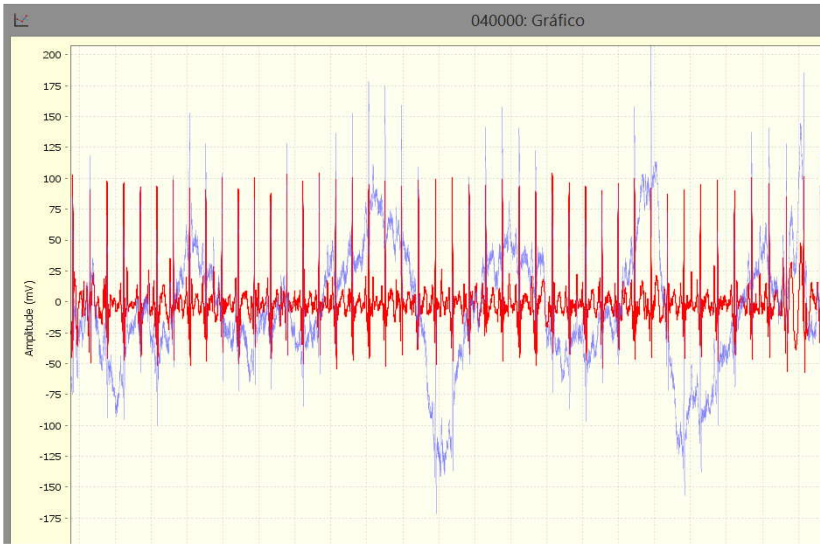
a) iniciar a plataforma;
b) inserir o bloco para importar o sinal de ECG de um arquivo (Grupo de blocos: Entradas; Bloco: ECG) e configurar os seus parâmetros (Figura 23),
   - informar o caminho para o arquivo do sinal. Foi utilizado 60 segundos de um sinal obtido da base de dados Physionet (GOLDBERGER, 2000) – *Database*: *Abdominal and Direct Fetal ECG Database* (adfecgdb); *Record*: r10.edf; *Signals*: Direct_1,
   - informar a frequência de amostragem na qual este sinal foi digitalizado. Para o sinal utilizado o valor é 1000 Hz;
c) inserir o bloco para projeto dos filtros digitais (Grupo de blocos: Filtros; Bloco: FIR_IIR),
   - realizar uma ligação entre o pino 1 deste bloco com o pino 5 do bloco anterior (importador de sinais de ECG);
d) configurar os parâmetros do bloco do filtro (Figura 21) para um filtro do tipo passa-banda,
   - selecionar o tipo do filtro: passa-faixa do tipo FIR Equirriple,
   - definir as frequência de corte para o filtro: frequência de corte inferior (F_{stop1}) em 0,5 Hz e superior (F_{stop2}) em 40 Hz; e banda passante (F_{pass}) entre 3 Hz e 35 Hz, pois é nesta faixa que os principais componentes da onda de ECG estão presentes (ver Figura 4) e evita-se o ruído proveniente da rede elétrica,
   - definir as magnitudes das banda de corte inferior, passante e superior em 60 dB, 1 dB e 80 dB, respectivamente,
   - calcular o filtro através do botão "Calcular e Aplicar o Filtro";
e) inserir o bloco para visualização gráfica dos sinais (Grupo de blocos: Gráficos; Bloco: Gráfico) e verificar o resultado da filtragem do sinal (Figura 28), modificando os parâmetros do filtro para obter o melhor resultado, se necessário (sempre fazendo uso o botão "Calcular e Aplicar o Filtro" quando algum parâmetro do filtro for alterado);

Figura 27 – Exemplo de projeto para processamento de sinal de ECG.



Fonte: Autor.

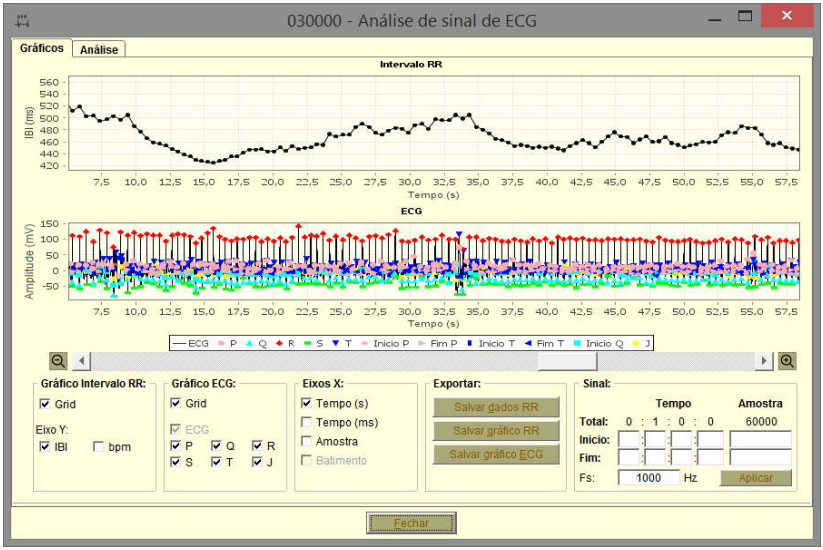
Figura 28 – Resultado do processamento de um sinal de ECG do exemplo da Figura 27. O gráfico em azul é o sinal de entrada (carregado do arquivo de dados) enquanto o gráfico vermelho é o sinal após passagem pelo filtro.



Fonte: Autor.

f) inserir o bloco para processamento e análise de sinais de ECG (Grupo de blocos: Análise; Bloco: ECG),
- realizar uma ligação entre o pino 1 deste bloco com o pino 5 do bloco do filtro,
- para o sinal utilizado no exemplo obteve-se o resultado gráfico mostrado na Figura 29,
- na aba "Análise" da Figura 29 é possível obter algumas informações do sinal aplicado ao bloco que estão expostas na Figura 30.

Figura 29 – Resultado do processamento de um sinal de ECG pelo bloco de análise de sinais de ECG para o exemplo da Figura 27.

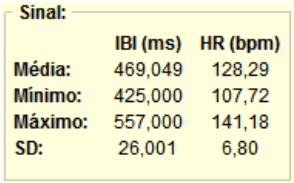


Fonte: Autor.

5.5 AVALIAÇÃO DE USABILIDADE

Os resultados da avaliação da satisfação de usabilidade da plataforma são mostrados na Tabela 3, para cada seção relacionada ao uso da plataforma.

Figura 30 – Informações do sinal de ECG processado no exemplo da Figura 27 pelo bloco de Análise de Sinais de ECG.



Fonte: Autor.

Através da Tabela 3 é possível observar que as seções 3, 4, 5 e 7 obtiveram as maiores porcentagens de notas consideradas ótimas, sendo a seção de avaliação de telas a que obteve melhor resultado com 80,68% das notas entre 7 e 9. Por meio da mesma análise, as seções 2 e 6 foram as que obtiveram maior percentual de notas consideradas médias. As menores notas obtidas estão relacionadas com a capacidade do sistema.

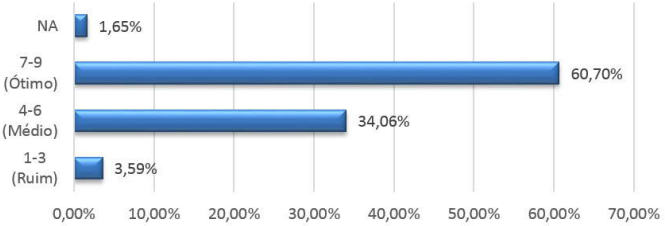
Tabela 3 – Resultados do questionário de avaliação de usabilidade da plataforma, por seção avaliada.

Seções	**1 a 3 Ruim**	**4 a 6 Médio**	**7 a 9 Ótimo**	**NA**
2 - Impressão como usuário	1,52 %	53,03 %	45,45 %	0,00 %
3 - Telas	0,00 %	19,32 %	80,68 %	0,00 %
4 - Terminologia e informação do sistema	6,29 %	36,36 %	57,34 %	0,00 %
5 - Aprendizagem do sistema	0,83 %	29,75 %	68,60 %	0,83 %
6 - Capacidade do sistema	8,33 %	47,73 %	43,94 %	0,00 %
7 - Multimídia	4,55 %	18,18 %	68,18 %	9,09 %
Médias gerais	**3,59 %**	**34,06 %**	**60,70 %**	**1,65 %**

Fonte: Autor.

A Figura 31 traz graficamente os índices gerais de satisfação da usabilidade da plataforma, adquiridos pelas médias dos resultados obtidos para cada seção. Analisando estes valores identifica-se que 60,7% das perguntas do questionário obtiveram notas entre 7 e 9, ou seja, os usuários consideram o sistema ótimo.

Figura 31 – Gráfico da satisfação global da usabilidade da plataforma.



Fonte: Autor.

6 DISCUSSÃO

Como resultado final do trabalho realizado, obteve-se um sistema computacional (*software*) voltado ao processamento digital de sinais biomédicos através de funções representadas na forma de blocos, e que pode servir como ferramenta auxiliar no ensino de DSP e da área da Engenharia Biomédica.

A plataforma apresentada neste trabalho possui uma interface interativa onde os blocos e as ligações entre os blocos são inseridas, excluídas e movimentadas na área de desenho com o auxílio do mouse. Os parâmetros das funções também são alterados facilmente com o mouse (duplo clique do mouse no bloco), abrindo a tela de ajuste dos parâmetros do bloco. O projeto de DSP desenvolvido na área de desenho da plataforma é automaticamente reprocessado quando alterações são realizadas no mesmo, refletindo estas alterações para o restante do projeto sem que o usuário tenha que realizar qualquer outra ação. As alterações compreendem inclusão/exclusão de blocos, inclusão/exclusão de ligações entre blocos e alterações de parâmetros de funções.

Trabalho semelhante é encontrado em SPANIAS (2001) onde uma ferramenta vem sendo desenvolvida na Universidade do Estado do Arizona (ASU) desde 1997. Esta ferramenta, chamada de J-DSP (*Java Digital Signal Processing*), é feita como um *applet* Java orientado a objetos. Ou seja, funciona na Web através de um navegador de *internet* (http://jdsp.asu.edu). O J-DSP também contém funções de processamento de sinais na forma de blocos configuráveis e se destina a ser usado como laboratório virtual no ensino de DSP principalmente em cursos de graduação em DSP e comunicação, no ensino presencial e a distância.

Diferentemente da ferramenta J-DSP, a plataforma apresentada neste trabalho é executada em ambiente *desktop*. Assim, é possível a sua utilização em qualquer momento e local já que não depende de uma conexão com a internet para funcionar. Uma limitação da plataforma J-DSP é que ela não possibilita salvar os projetos que lá são desenvolvidos. Esta funcionalidade está presente na plataforma desenvolvida e é importante para continuar o desenvolvimento e/ou realizar alterações nos projetos de DSP sem a necessidade de refazê-los novamente.

Ferramentas como o Simulink® (MathWorks, Inc.) também possuem um ambiente gráfico para criar diagramas de blocos. Simulink é uma ferramenta de modelagem, simulação e análise de sistemas dinâmicos integrada ao MATLAB, mas que não funciona

independentemente. Logo, para sua utilização, é necessária a instalação do MATLAB e adquirir licença para utilização desta ferramenta. Para utilizar a plataforma desenvolvida não há necessidade de compra de licença pois o MCR é distribuído gratuitamente pela MathWorks.

Os blocos executam funções desenvolvidas em MATLAB para desempenhar suas funcionalidades. Esta interação expande a capacidade da plataforma no desenvolvimento de ferramentas de DSP, visto que o MATLAB contém uma vasta coleção de funções para este fim. Entretanto, ocorrem atrasos no processamento de sinais aplicados a alguns blocos (tais como os filtros) em função da comunicação entre o aplicativo Java e o MCR bem como do tempo de execução da função. Quanto maior for o sinal aplicado maior o tempo de processamento. Este atraso é mais evidente na inserção de blocos ao projeto.

A incompatibilidade entre os tipos de dados utilizados nas duas linguagens (Java e MATLAB) também teve que ser observada e respeitada para o correto funcionamento. A principal diferença está nos índices de localização de elementos dentro de vetores e matrizes. Estes índices são números inteiros que iniciam em 1 (um) no MATLAB e em 0 (zero) no Java. Esta integração entre Java e MATLAB, juntamente com o inter-relacionamento necessário entre os blocos na transferência de dados durante um projeto de DSP desenvolvido, foram as principais complexidades enfrentadas no desenvolvimento da plataforma.

A validação das funções desenvolvidas foi realizada mediante a verificação e comparação dos resultados destas funções na plataforma com os resultados obtidos diretamente no *software* MATLAB. Como esperado, os resultados obtidos são os mesmos visto que as funções são executadas em Java através de uma aplicação em *background* do MATLAB via MCR.

Foram utilizados sinais de eletrocardiograma nos testes e exemplos realizados, obtidos da base de dados de sinais fisiológicos Physionet. Esta base de dados é internacionalmente difundida para a realização de pesquisas na área de sinais biológicos e contém uma variedade de sinais biomédicos digitalizados e disponibilizados gratuitamente, muitos deles com anotações de eventos ocorridos nos sinais. Nesta base é possível encontrar banco de sinais de diversas instituições, entre elas o Massachussets Institute of Technology (MIT), a European Society of Cardiology e a Creighton University.

O grupo de usuários que avaliou a plataforma em relação a sua usabilidade foi obtido entre os participantes do VII Minicurso de Engenharia Biomédica na Prática, realizado nos dias 6, 7 e 8 de agosto de 2014. Este minicurso é realizado anualmente pelo Instituto de

Engenharia Biomédica da UFSC (IEB-UFSC) e tem como público alvo alunos de graduação dos cursos de engenharia. Ele possui a finalidade de divulgar a Engenharia Biomédica através de atividades práticas e teóricas ministradas por professores e alunos de pós-graduação do IEB-UFSC. O total de participantes na avaliação foi de 11 pessoas entre 18 e 34 anos (média de 24 anos), sendo 10 do sexo masculino e uma do sexo feminino. Todos os participantes são alunos de cursos de graduação em engenharias, sendo a maioria dos cursos de Engenharia Elétrica e de Engenharia Eletrônica. A avaliação foi realizada com voluntários e aprovado pelo Comitê de Ética da Universidade sob o número CAEE 374.205.

7 CONCLUSÕES

O presente trabalho teve como objetivo principal o desenvolvimento de uma plataforma computacional para processamento digital de sinais biomédicos através de blocos funcionais configuráveis. A plataforma, embora ainda com um número limitado de funções (blocos), oferece possibilidades interessantes no ensino e pesquisa em processamento digital de sinais, especialmente os sinais biomédicos. Os blocos existentes permitem, por exemplo, o uso da plataforma em abordagens de filtragens e análise de sinais de eletrocardiograma digitalizados.

O usuário interage com a plataforma essencialmente através do mouse, permitindo que blocos sejam inseridos, excluídos ou movimentados na área de desenho. O mesmo pode ser feito com as ligações realizadas entre os blocos, que definem o fluxo de dados entre eles. Os parâmetros dos blocos, quando existentes, estão acessíveis para visualização/alteração através do mouse no próprio bloco. Outra funcionalidade importante da plataforma é a possiblidade de salvar o projeto desenvolvido para que possa ser aberto novamente quando necessário, sem ter que refazê-lo.

Embora o uso de funções do MATLAB nas ferramentas desenvolvidas acarrete em atrasos no processamento dos sinais, a usabilidade destas funções no desenvolvimento da plataforma se mostrou importante dada a facilidade de implementação, uma vez que as funções estão prontas e validadas. Concomitantemente, a plataforma não foi idealizada para processamento em tempo real.

O desenvolvimento do aplicativo em linguagem Java favorece a portabilidade do programa e permite que a plataforma seja executada em qualquer computador que contenha a JVM e o MCR instalados, ambos distribuídos gratuitamente por seus desenvolvedores.

7.1 TRABALHOS FUTUROS

Com base neste trabalho, segue algumas sugestões para trabalhos futuros, a fim de obter melhorias para o sistema apresentado neste trabalho:

a) melhorar o desempenho da plataforma, uma vez que a execução de funções do MATLAB através do MCR provoca atrasos no processamento das funções;

b) expandir a capacidade do sistema, integrando novas ferramentas de DSP e algoritmos de análise de sinais biomédicos;
c) possibilitar à plataforma o seu uso com sinais com múltiplos canais;
d) gerar relatórios com os resultados obtidos a partir do projeto desenvolvido na plataforma;
e) propor novas metodologias para representação gráfica dos sinais envolvidos nos projetos de DSP, objetivando maior desemprenho e menor consumo de memória do computador;
f) desenvolver novas ferramentas para exportar sinais digitalizados, possibilitando o carregamento de sinais com múltiplos canais e/ou seleção de canal e a leitura de arquivos com cabeçalho de informações, por exemplo;
g) propor e realizar atividades utilizando a plataforma e realizar avaliações com o intuito de identificar a eficácia da plataforma em relação ao ensino de DSP;
h) investigar a possibilidade de integrar *hardware* ao sistema, possibilitando a aquisição e processamento de sinais em tempo real.

REFERÊNCIAS

BRUCE, Eugene N. **Biomedical Signal Processing and Signal Modeling**. New York: John Wiley & Sons, 2001. ISBN: 0-471-34540-7.

CAMPBELL, Jonathan; MURTAGH, Fionn; KÖKÜER, Münevver. DataLab-J: A Signal and Image Processing Laboratory for Teaching and Research. **IEEE Transaction on Education**, v. 44, n. 4, p. 329-335, nov. 2001.

CERUTTI, Sergio; MARCHESI, Carlo. **Advanced Methods of Biomedical Signal Processing**. New Jersey: John Wiley & Sons, Piscataway: IEEE Press, 2011. ISBN: 978-0-470-42214-4.

COHEN, Arnon. Biomedical Signals: Origin and Dynamic Characteristics; Frequency-Domain Analysis. In: BRONZINO, Joseph D. **The Biomedical Engineering Handbook**. 2 ed. Boca Raton: CRC Press LLC, 2000. Cap. 52.

GOLDBERGER, Ary L; *et at*. PhysioBank, PhysioToolkit, and PhysioNet: Components of a New Research Resource for Complex Physiologic Signals. **Circulation 101 (23)**: e215-e220. 13 junho 2000. Disponível em: <http://circ.ahajournals.org/cgi/content/full/101/23/e215>. Acesso em: 24 agosto 2014. PMID: 10851218. DOI: 10.1161/01.CIR.101.23.e215

HADDAD, Sandro A. P., SERDIJN, Wouter A. **Ultra Low-Power Biomedical Signal Processing**: An Analog Wavelet Filter Approach for Pacemakers. Springer: 2009. ISBN: 978-1-4020-9072-1

HAYES, Monson H. **Schaum's Outline of Theory and Problems of Digital Signal Processing**. New York: McGraw-Hill, 1999. Schaum's Outline Series

KUO, Sen M; LEE, Bob H. **Real-Time Digital Signal Processing**: Implementations, Applications, and Experiments with the TMS320C55X. Chichester: John Wiley & Sons 2001.

MAINARDI, Lucas T; BIANCHI, Anna M; CERUTTI, Sergio. Digital Biomedical Signal Acquisition and Processing. In: BRONZINO, Joseph

D. **The Biomedical Engineering Handbook**. 2 ed. Boca Raton: CRC Press LLC, 2000. Cap. 53.

MALMIVUO, Jaakko; PLONSEY, Robert. **Bioelectromagnetism**: Principles and Applications of Bioelectric and Biomagnetic Fields. New York: Oxford University Press, 1995.

MUSEN, Mark A; VAN BEMMEL, Jan Hendrik. **Handbook of Medical Informatics**. 2 ed. [s.l.]: Springer, 1997.

MUTHUSWAMY, Jit. Biomedical Signal Analysis. In: KUTZ, Myer. **Standard Handbook of Biomedical Engineering and Design**. New York: McGraw-Hill, 2003. p. 18.1-18.30. ISBN: 0-07-135637-1.

NATIONAL INSTRUMENTS. How to Choose the Right DAQ Hardware for Your Measurement System. 02 jan. 2013. Disponível em: < http://www.ni.com/white-paper/13655/en/>. Acesso em: 24 agosto 2014.

PETRY, Daiana. **Sistema para Análise da Variabilidade de Sinais Fisiológicos**: Aplicação em Variabilidade da Frequência cardíaca e Intervalo QT. 2006. 146 f. Dissertação (Mestrado em Engenharia Elétrica) - Curso de Pós-Graduação em Engenharia Elétrica, Universidade Federal de Santa Catarina, Florianópolis, 2006.

RANGAYYAN, Rangaraj M. **Biomedical Signal Analysis**: A Case-Study Approach. New York: John Wiley & Sons, Piscataway: IEEE Press, 2004. ISBN: 0-471-20811-6.

RATHKE, Juliano Elesbão. **Sistema de Processamento de Sinais Biomédicos**: Módulo Didático de ECG, EMG, EOG e Conversão Analógico-Digital de Biosinais. 2008. 162 f. Dissertação (Mestrado em Engenharia Elétrica) - Curso de Pós-Graduação em Engenharia Elétrica, Universidade Federal de Santa Catarina, Florianópolis, 2008.

ROCHA, Adson Ferreira da; et al. Processamento de Sinais Biológicos. In: BRASIL, Lourdes Mattos. **Informática em Saúde**. Universa: Taguatinga; Londrina: Eduel, 2008. p. 381-416. ISBN: 978-85-60485-03-1.

ROSA, Diego Laucsen da. **Sistema de Processamento de Sinais Biomédicos**: Filtragem de Sinais de Eletroencefalograma. 2009. 135 f. Dissertação (Mestrado em Engenharia Elétrica) - Curso de Pós-Graduação em Engenharia Elétrica, Universidade Federal de Santa Catarina, Florianópolis, 2009.

SEMMLOW, John**. Signals and Systems for Bioengineers**: A Matlab-Based Introduction. 2. ed. Waltham: Elsevier, 2012.

SPANIAS, Andreas; BIZUNEH, Fikre. Development of New Functions and Scripting Capabilities in Java-DSP for Easy Creation and Seamless Integration of Animated DSP Simulations in Web Courses. In: IEEE INTERNATIONAL CONFERENCE ON ACOUSTICS, SPEECH, AND SIGNAL PROCESSING - CASSP, 07-11 maio 2001, Salt Lake. **Anais do CASSP 2001**. Salt Lake: 2001. p. 2717-2720.

SMITH, Steven W. **The Scientist and Engineer's Guide to Digital Signal Processing**. 2. ed. San Diego: California Technical Publishing, 1999.

TOMPKINS, Willis J. **Biomedical Digital Signal Processing**: C-Language Examples and Laboratory Experiments for the IBM PC. 2 ed. Upper Saddle River: Prentice Hall, 2000.

WEEKS, Michael. **Processamento Digital de Sinais Utilizando Matlab® e Wavelets**. 2ed. Rio de Janeiro: LTC, 2012. 436 p.

BOONE, Harry N. Jr.; BOONE, Deborah A. Analyzing Likert Data. **Journal of Extension**, v. 50, n. 2, Article 2TOT2, abril 2012. ISSN: 1077-5315. Disponível em: <http://www.joe.org/joe/2012april/tt2.php>. Acesso em: 12 novembro 2014.

APÊNDICE A – Questionário de Avaliação de Usabilidade

Questionário de Avaliação de Software
Questões Gerais

Qual sua formação (período e curso)?

Você utiliza o computador como ferramenta de estudo?
□ Sim □ Não

Você acredita que um sistema computacional pode auxiliá-lo no aprendizado?
□ Sim □ Não

Você conhece algum *software* para processamento digital de sinais? Qual?

O que seria importante em um *software* para processamento digital de sinais?

Instrumento de Avaliação da Satisfação do Usuário
(Traduzido do "Questionnaire for User Interaction Satisfaction" - QUIS)

Qual a sua idade?

Sexo: □ Feminino □ Masculino

Parte 2: Experiência Anterior com Computadores
2.1. Quantos sistemas operacionais você já utilizou (Windows, Linux, Mac, etc)?
□ Nenhum □ 1 □ 2 □ 3-4 □ 5-10 □ 5-10 □ Mais de 10

2.2. Avalie sua experiência com os computadores em geral:
□ Nenhuma □ Alguma □ Moderada □ Pouco alta □ Alta

Parte 3: Impressão como Usuário
Por favor, marque o número que reflete mais adequadamente a sua impressão sobre a utilização do sistema avaliado.

3.1. Em geral, o sistema, para você, é:	Péssimo ☐1 ☐2 ☐3 ☐4 ☐5 ☐6 ☐7 ☐8 ☐9	Excelente
3.2. Em relação à interação, o sistema é:	Frustrante ☐1 ☐2 ☐3 ☐4 ☐5 ☐6 ☐7 ☐8 ☐9	Satisfatório
3.3. Em relação ao uso geral o sistema é:	Tedioso ☐1 ☐2 ☐3 ☐4 ☐5 ☐6 ☐7 ☐8 ☐9	Estimulante
3.4.5. Em relação ao manuseio o sistema é:	Difícil ☐1 ☐2 ☐3 ☐4 ☐5 ☐6 ☐7 ☐8 ☐9	Fácil
3.5. Em relação ao seu aprendizado após usar o sistema, você teve um:	Domínio inadequado ☐1 ☐2 ☐3 ☐4 ☐5 ☐6 ☐7 ☐8 ☐9	Domínio adequado
3.6. A concepção geral do sistema é:	Rígida ☐1 ☐2 ☐3 ☐4 ☐5 ☐6 ☐7 ☐8 ☐9	Flexível

Parte 4: Telas

4.1. Letras na tela do computador:	Difícil de ler ☐1 ☐2 ☐3 ☐4 ☐5 ☐6 ☐7 ☐8 ☐9	Fácil de ler
4.1.1. Imagem das letras são:	Embaçada ☐1 ☐2 ☐3 ☐4 ☐5 ☐6 ☐7 ☐8 ☐9	Nítida
4.1.2. Forma da letra (fonte):	Pouco legível ☐1 ☐2 ☐3 ☐4 ☐5 ☐6 ☐7 ☐8 ☐9	Muito legível
4.3. A organização dos elementos na tela foram úteis?	Nunca ☐1 ☐2 ☐3 ☐4 ☐5 ☐6 ☐7 ☐8 ☐9	Sempre
4.3.1. A quantidade de informação que pode ser apresentada na tela é?	Inadequada ☐1 ☐2 ☐3 ☐4 ☐5 ☐6 ☐7 ☐8 ☐9	Adequada
4.3.2. A organização de informações na tela:	Ilógico ☐1 ☐2 ☐3 ☐4 ☐5 ☐6 ☐7 ☐8 ☐9	Lógico
4.4. A sequência de telas é:	Confusa ☐1 ☐2 ☐3 ☐4 ☐5 ☐6 ☐7 ☐8 ☐9	Clara

4.4.3. O desenrolar de tarefas relacionadas à atividade:	Confuso ☐1	☐2	☐3	☐4	☐5	☐6	☐7	☐8	Claramente definido ☐9

Parte 5: Terminologia e Informação do Sistema

5.1. O uso de terminologia em todo o sistema foi:	Inconsistente ☐1	☐2	☐3	☐4	☐5	☐6	☐7	☐8	Consistente ☐9
5.1.2. O uso de termos relacionados à atividade:	Inconsistente ☐1	☐2	☐3	☐4	☐5	☐6	☐7	☐8	Consistente ☐9
5.2. Os termos usados se relacionam com a tarefa que você está desempenhando?	Sempre ☐1	☐2	☐3	☐4	☐5	☐6	☐7	☐8	Nunca ☐9
5.2.2. Os termos apresentados na tela são:	Ambíguos ☐1	☐2	☐3	☐4	☐5	☐6	☐7	☐8	Precisos ☐9
5.3. Mensagens apresentadas na tela são:	Inconsistente ☐1	☐2	☐3	☐4	☐5	☐6	☐7	☐8	Consistente ☐9
5.4. Mensagens apresentadas na tela são:	Confusas ☐1	☐2	☐3	☐4	☐5	☐6	☐7	☐8	Claras ☐9
5.4.2. Instruções para correção de erros são:	Confusas ☐1	☐2	☐3	☐4	☐5	☐6	☐7	☐8	Claras ☐9
5.5. O sistema mantém você informado sobre o que ele está fazendo?	Nunca ☐1	☐2	☐3	☐4	☐5	☐6	☐7	☐8	Sempre ☐9
5.5.2. Realizar uma operação no sistema leva a um resultado previsível?	Nunca ☐1	☐2	☐3	☐4	☐5	☐6	☐7	☐8	Sempre ☐9

5.5.4. Duração da espera entre operações do sistema é:	Inaceitável ☐1 ☐2 ☐3 ☐4 ☐5 ☐6 ☐7 ☐8 ☐9	Aceitável
5.6. Mensagens de erro:	Inúteis ☐1 ☐2 ☐3 ☐4 ☐5 ☐6 ☐7 ☐8 ☐9	Úteis
5.6.1. Mensagens de erro esclarecem o problema:	Nunca ☐1 ☐2 ☐3 ☐4 ☐5 ☐6 ☐7 ☐8 ☐9	Sempre
5.6.2. Redação das mensagens de erro:	Desagradável ☐1 ☐2 ☐3 ☐4 ☐5 ☐6 ☐7 ☐8 ☐9	Agradável

Parte 6: Aprendizagem do Sistema

6.1. Aprender a operar o sistema é:	Difícil ☐1 ☐2 ☐3 ☐4 ☐5 ☐6 ☐7 ☐8 ☐9	Fácil
6.1.1. Iniciar o uso é:	Difícil ☐1 ☐2 ☐3 ☐4 ☐5 ☐6 ☐7 ☐8 ☐9	Fácil
6.1.2. Aprender funções avançadas é:	Difícil ☐1 ☐2 ☐3 ☐4 ☐5 ☐6 ☐7 ☐8 ☐9	Fácil
6.1.3. O tempo de aprendizado sobre o sistema é:	Longo ☐1 ☐2 ☐3 ☐4 ☐5 ☐6 ☐7 ☐8 ☐9	Curto
6.2. Explorar funções por tentativa e erro é:	Desencorajador ☐1 ☐2 ☐3 ☐4 ☐5 ☐6 ☐7 ☐8 ☐9	Encorajador
6.2.1. Explorar funções do sistema é:	Arriscado ☐1 ☐2 ☐3 ☐4 ☐5 ☐6 ☐7 ☐8 ☐9	Seguro
6.2.2. Descobrir novas funções é:	Difícil ☐1 ☐2 ☐3 ☐4 ☐5 ☐6 ☐7 ☐8 ☐9	Fácil
6.3. Relembrar nomes e uso de comandos é:	Difícil ☐1 ☐2 ☐3 ☐4 ☐5 ☐6 ☐7 ☐8 ☐9	Fácil
6.4.1. Número de etapas por tarefa é:	Excessivo ☐1 ☐2 ☐3 ☐4 ☐5 ☐6 ☐7 ☐8 ☐9	Adequado

6.4.2. As etapas para completar a tarefa seguem uma sequência lógica:	Nunca ☐1 ☐2 ☐3 ☐4 ☐5 ☐6 ☐7 ☐8 ☐9	Sempre
6.4.3. A resposta do sistema ao completar uma sequência de etapas é:	Confusa ☐1 ☐2 ☐3 ☐4 ☐5 ☐6 ☐7 ☐8 ☐9	Clara

Parte 7: Capacidade do Sistema

7.1. A velocidade do sistema é:	Muito baixa ☐1 ☐2 ☐3 ☐4 ☐5 ☐6 ☐7 ☐8 ☐9	Rápida o bastante
7.1.1. O tempo de resposta para a maioria das operações é:	Muito longo ☐1 ☐2 ☐3 ☐4 ☐5 ☐6 ☐7 ☐8 ☐9	Rápido o bastante
7.1.2. A velocidade com que a tela é atualizada com informações é:	Muito baixa ☐1 ☐2 ☐3 ☐4 ☐5 ☐6 ☐7 ☐8 ☐9	Rápida o bastante
7.2. O sistema é confiável?	Nunca ☐1 ☐2 ☐3 ☐4 ☐5 ☐6 ☐7 ☐8 ☐9	Sempre
7.2.2. Falhas no sistema ocorrem?	Frequentemente ☐1 ☐2 ☐3 ☐4 ☐5 ☐6 ☐7 ☐8 ☐9	Raramente
7.2.3. O sistema alerta sobre potenciais problemas?	Nunca ☐1 ☐2 ☐3 ☐4 ☐5 ☐6 ☐7 ☐8 ☐9	Sempre
7.4. Corrigir seus erros ao utilizar o sistema é:	Difícil ☐1 ☐2 ☐3 ☐4 ☐5 ☐6 ☐7 ☐8 ☐9	Fácil
7.4.1. Corrigir erros de digitação é:	Complexo ☐1 ☐2 ☐3 ☐4 ☐5 ☐6 ☐7 ☐8 ☐9	Simples
7.4.2. A capacidade de fazer operações é:	Inadequada ☐1 ☐2 ☐3 ☐4 ☐5 ☐6 ☐7 ☐8 ☐9	Adequada

7.5. A facilidade de operar o sistema depende do seu nível de experiência?	Nunca ☐1 ☐2 ☐3 ☐4 ☐5 ☐6 ☐7 ☐8 ☐9	Sempre
7.5.1. Você pode completar tarefas conhecendo somente poucos comandos?	Com dificuldade ☐1 ☐2 ☐3 ☐4 ☐5 ☐6 ☐7 ☐8 ☐9	Com facilidade
7.5.2. Você consegue usar os atalhos e as funções:	Com dificuldade ☐1 ☐2 ☐3 ☐4 ☐5 ☐6 ☐7 ☐8 ☐9	Com facilidade

Parte 10: Multimídia

10.1. A qualidade de figuras/fotografias é:	Ruim ☐1 ☐2 ☐3 ☐4 ☐5 ☐6 ☐7 ☐8 ☐9	Boa
10.1.1. As figuras/ fotografias são:	Pouco nítidas ☐1 ☐2 ☐3 ☐4 ☐5 ☐6 ☐7 ☐8 ☐9	Bem nítidas
10.4. As cores utilizadas são:	Pouco naturais ☐1 ☐2 ☐3 ☐4 ☐5 ☐6 ☐7 ☐8 ☐9	Naturais
10.4.1. A quantidade de cores disponíveis é:	Inadequada ☐1 ☐2 ☐3 ☐4 ☐5 ☐6 ☐7 ☐8 ☐9	Adequada

APÊNDICE B – Diagrama de Classes