UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ
DEPARTAMENTO ACADÊMICO DE INFORMÁTICA
CURSO DE ENGENHARIA DE COMPUTAÇÃO

ANDRÉ BIFFE DI RENZO

SISTEMA DE PRÉ-PROCESSAMENTO PARA DADOS OBTIDOS DE PROCESSOS MASTIGATÓRIOS DE RUMINANTES UTILIZANDO SENSORES A FIBRA ÓTICA

TRABALHO DE CONCLUSÃO DE CURSO

PATO BRANCO
2015

ANDRÉ BIFFE DI RENZO

SISTEMA DE PRÉ-PROCESSAMENTO PARA DADOS OBTIDOS DE PROCESSOS MASTIGATÓRIOS DE RUMINANTES UTILIZANDO SENSORES A FIBRA ÓTICA

Trabalho de Conclusão de Curso de graduação, apresentado à disciplina de Trabalho de Diplomação 2, do Curso Superior de Engenharia de Computação, da Universidade Tecnológica Federal do Paraná, Câmpus Pato Branco, como requisito parcial para obtenção do título de bacharel.

Orientador: Prof. Vinicius Pegorini
Coorientador: Prof. Jean Carlos Cardozo da Silva, D. Sc.

PATO BRANCO
2015

MINISTÉRIO DA EDUCAÇÃO
Universidade Tecnológica Federal do Paraná
Câmpus Pato Branco
Departamento Acadêmico de Informática
Curso de Engenharia de Computação

TERMO DE APROVAÇÃO

Às 10 horas e 30 minutos do dia 24 de junho 2015, na sala V108, da Universidade Tecnológica Federal do Paraná, Câmpus Pato Branco, reuniu-se a banca examinadora composta pelos professores Vinícius Pegorini (Orientador), Rafael Cardoso e Gilda Aparecida de Assis para avaliar o trabalho de conclusão de curso com o título **Sistema de pré-processamento para dados obtidos de processos mastigatórios de ruminantes utilizando sensores a fibra ótica**, do aluno **Andre Biffe Di Renzo**, matrícula 1210122, do curso de Engenharia de Computação. Após a apresentação o candidato foi arguido pela banca examinadora. Em seguida foi realizada a deliberação pela banca examinadora que considerou o trabalho aprovado.

Vinícius Pegorini
Orientador (UTFPR)

Rafael Cardoso
(UTFPR)

Gilda Aparecida de Assis
(UTFPR)

Beatriz Terezinha Borsoi
Coordenador de TCC

Marco Antonio de Castro Barbosa
Coordenador do Curso de
Engenharia de Computação

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

Albert Einstein

RESUMO

DI RENZO, André Biffe. Sistema de pré-processamento para dados de processos mastigatórios de ruminantes utilizando sensores a fibra ótica. 2015. 67 f. Monografia (Trabalho de Conclusão de Curso) - Curso de Engenharia de Computação, Universidade Tecnológica Federal do Paraná, Câmpus Pato Branco. Pato Branco, 2015.

Neste trabalho é proposto um sistema para efetuar o pré-processamento de sinais provenientes de sensores a fibra ótica, utilizado para identificar eventos de mastigação em ruminantes. Os sensores utilizados são baseados em redes de Bragg. A tecnologia utilizada é uma alternativa para os principais métodos utilizados atualmente pela agropecuária, sendo estes o método acústico e direto. O pré-processamento é necessário, pois permite a segmentação dos dados coletados pelo sensor a cada evento de mastigação dos animais. Durante o pré-processamento também é possível extrair algumas caraterísticas do sinal segmentado, que auxiliam no processo de identificação de eventos de mastigação de ruminantes. Os algoritmos para efetuar a segmentação dos dados foram desenvolvidos em linguagem G, utilizando a IDE LabVIEW™. Após o desenvolvimento dos algoritmos, foram efetuados testes de segmentação de dados obtidos de um sensor a fibra ótica, posicionado na mandíbula de um bovino. Para se obter os dados do sensor, foi utilizado um interrogador óptico, com taxa de aquisição de 1000 amostras por segundo. Com resultados positivos da segmentação dos dados obtidos do ensaio *In Vivo*, os algoritmos para pré-processamento foram integrados ao sistema de aquisição do interrogador óptico para se segmentar os eventos em tempo real.

Palavras-chave: pré-processamento, sensores a fibra ótica, mastigação, ruminante

ABSTRACT

DI RENZO, André Biffe. Preprocessing system for jaw movements of ruminants using fiber optical sensors. 2015. 67 f. Monografia (Trabalho de Conclusão de Curso) - Curso de Engenharia de Computação, Universidade Tecnológica Federal do Paraná, Câmpus Pato Branco. Pato Branco, 2015.

In this work a system that executes the preprocessing of signals from fiber optic sensors is proposed, used to identify chewing events in ruminants. The sensors used are based on the fiber Bragg Gratings theory. The proposed technology is an alternative to the current available methods used by the livestock industry. The current technology uses acoustic and direct methods. The preprocessing is necessary, because allow the segmentation in every chewing movement of the acquired data by the sensor. In the preprocessing is possible to extract some characteristics of the segmented signal, which assist the process of identification of chewing events in ruminants. These algorithms to execute the segmentation of the data were developed in G language, using IDE LabVIEW™. After developing the algorithms, tests were executed to segment the acquired data from the fiber-optic sensor, positioned in the jaw of the veal cow. To get the data from the sensor, an optical interrogator was utilized, with an acquiring rate of 1000 samples per second. With positive results from the segmentation-acquired data in the In Vivo experiment, the preprocessing algorithms were integrated with the acquisition system of the optical interrogator to segment real time events.

Keywords: preprocessing, optical fiber sensors, chewing, ruminant

LISTA DE FIGURAS

LISTA DE QUADROS

LISTA DE LISTAGENS

LISTA DE SIGLAS

DTFS	Série de Fourier de Tempo Discreto (*Discrete Time Fourier Series*)
FBG	Redes de Bragg em fibras óticas (*Fiber Bragg Grating*)
FFT	Transformada Rápida de Fourier (*Fast Fourier Transform*)
HBM	*Hottinger Baldwin Messtechnik GmbH*
IDE	Ambiente integrado de desenvolvimento (*Integrated Development Environment*)
RNA	Rede Neural Artificial
VI	Instrumento Virtual (*Virtual Instrument*)

LISTA DE ACRÔNIMOS

LabVIEW	Laboratório de engenharia com instrumentos de bancada virtuais (*Laboratory Virtual Instrument Engineering Workbench)*
IGER	Instituto de pastagem e pesquisa ambiental (*Institute of Grassland and Environmental Research)*

SUMÁRIO

1 INTRODUÇÃO

Neste capítulo será apresentada a introdução do trabalho, assim como suas considerações iniciais, objetivos, justificativa e estrutura.

1.1 CONSIDERAÇÕES INICIAIS

No ano de 2015, o Brasil é considerado o líder em exportações de carne bovina, ocupando esta posição desde 2004, sendo este um dos principais destaques do agronegócio brasileiro (MINISTÉRIO..., 2015). Diante desse quadro, agricultores estão investindo no que é chamado de pecuária de precisão, que consiste em utilizar sistemas eletrônicos e de decisão para integrar os conhecimentos do comportamento animal (LACA, 2008). Com o auxílio desses sistemas, é possível verificar a maneira como o animal está se alimentado, bem como se observar dados relacionados a sua saúde. Também é possível analisar o impacto da atividade de pastejo nos ambientes pastoris, pois a administração errada das áreas de pastagem pode causar impacto negativo ao meio ambiente (CARVALHO, 2009), (UGAR, 1996).

Durante o processo de ingestão de alimentos pelos ruminantes alguns aspectos devem ser observados, como a mastigação, o alimento que está sendo manipulado e a ruminação. Esses dados são utilizados pelos pesquisadores de nutrição animal para avaliar a saúde animal e outros índices zootécnicos (JOCHIMS *et al.,* 2010).

Na pecuária de precisão, várias técnicas podem ser utilizadas para avaliar o comportamento ingestivo de ruminantes. Normalmente, é empregada a técnica de visualização direta, que consiste em observar o animal e coletar informações relacionadas ao pastejo de forma manual, podendo ser uma técnica imprecisa e exaustiva (SANTOS, 2009). Também existem técnicas que aplicam sistemas eletrônicos. Uma das técnicas que tem o seu uso mais praticado é o método acústico para obter os dados de mastigação dos animais em ambientes de pastejo. Essa técnica consiste em utilizar sensores de áudio para obter as respostas dos movimentos mandibulares que o animal executa durante o pastejo. Mesmo essa técnica possuindo a vantagem de não ser invasiva, os dados coletados na maioria dos casos têm resposta ruidosa, dificultando assim a análise desses dados (LACA *et al.*,

1994). Outra técnica acústica utilizada é o *IGER Behavior Recorder*, a qual utiliza o *software Graze* para processar os sinais de áudio coletados (RUTTER *et al.*, 1997).

Uma nova técnica introduzida por Wosniak *et al.* (2012), utiliza sensores a fibra ótica para obter os sinais relativos ao movimento que o ruminante executa durante o pastejo. Como esse tipo de sensor possui dimensões reduzidas, imunidade a forte campo eletromagnético e respostas com alta sensibilidade (OTHONOS, 1997). A utilização desse tipo de sensor possibilita uma análise mais precisa dos dados obtidos nos eventos que ocorreram em um determinado instante de tempo. Essa técnica obtém os dados através da deformação mecânica da mandíbula do animal (WOSNIAK *et al.,* 2012).

Mesmo existindo sensores elétricos para efetuar a medição dessas grandezas físicas, esses sensores possuem resposta prejudicada quando aplicado em locais com forte campo eletromagnético ou alta tensão e locais em que se precise de respostas com alta sensibilidade, como exemplo, sensores biomédicos. Outras vantagens dos sensores ópticos é a biocompatibilidade, ser inerte quimicamente, pois a sílica não é toxica e, por isso, não é rejeitada pelo organismo e ter dimensões reduzidas (KALINOWSKI *et al.*, 2010), o que permite ensaios *In Vivo.* Devido a isto, este tipo de sensor possui uma vasta gama de aplicações.

Os sensores a fibra ótica mais utilizados são os baseados em redes de Bragg em fibra ótica (*Fiber Bragg Grating* - FBG*)* (OTHONOS, 1997). Para o caso estudado, será dada ênfase no uso desses sensores para medir deformação mecânica. Normalmente, sistemas onde deve ser analisado a sua deformação mecânica são considerados dinâmicos, ou seja, o seu estado não é constante, sempre estará ocorrendo variações. Devido a isso, é necessário adquirir mais de uma amostra por segundo para se ter uma análise completa da deformação que está ocorrendo. Um exemplo de sistema dinâmico pode ser a análise da mastigação de um animal ruminante, onde pode ser analisada a deformação mecânica de sua mandíbula. A partir dos dados da deformação mecânica, os mesmos podem ser submetidos a algoritmos para classificação de padrões durante o processo de ingestão de alimento pelo animal.

1.2 PROBLEMA

Efetuar a análise de sistemas dinâmicos pode gerar uma quantidade considerável de dados, devido a necessidade de se obter mais de uma amostra em um segundo. Nesse trabalho, será efetuada a aquisição de forças biomecânicas da mandíbula de ruminantes em processos mastigatórios utilizando sensores ópticos. Esses dados serão pré-processados para poderem ser utilizados posteriormente em sistemas que reconheçam padrões nos dados pré-processados, a classificação e reconhecimento de padrões não faz parte do escopo deste trabalho.

Nos trabalhos desenvolvidos por Wosniak *et al.* (2012) e Karam *et el.* (2014) em que foram utilizados sensores ópticos para identificar eventos de mastigação em ruminantes, foi utilizada uma taxa de aquisição de 1000 amostras por segundo, pois nestes trabalhos, verificou-se que um movimento de mastigação pode acontecer em menos de um segundo. Sendo assim, se for utilizada uma taxa de aquisição menor, haverá perda de informações prejudicando a qualidade dos resultados.

Como os movimentos de mastigação dos animais podem ter duração de menos de um segundo e, também não possuem um intervalo fixo de duração, torna-se necessário o desenvolvimento de um algoritmo que faça a segmentação de cada amostra de mastigação. Após a aquisição de uma quantidade de amostras, essas amostras serão pré-processadas para ser efetuada a segmentação do sinal da FBG, identificando-se os eventos de mastigação que o animal executou. Além disso, outras informações podem ser extraídas do sinal como, por exemplo, realizando o cálculo da Transformada Rápida de Fourier (*Fast Fourier Transform* – FFT) (OPPENHEIM; SCHAFER, 2010), sendo que através desse cálculo é possível ter uma análise do espectro da frequência do sinal. A FFT pode ser utilizada para adicionar informações que podem melhorar o desempenho no processo de classificação dos dados.

1.3 OBJETIVOS

Diante do problema apresentado, é necessário propor uma solução para o mesmo. Dessa forma, é necessário definir alguns objetivos que deverão ser atingidos com esse trabalho de pesquisa.

1.3.1 Objetivo Geral

Efetuar a aquisição e pré-processamento dos dados oriundos de sensores a fibra ótica baseados em redes de Bragg, aplicados na medição de forças biomecânicas envolvidas no processo de classificação de padrões mastigatórios de ruminantes, visando a segmentação dos mesmos.

1.3.2 Objetivos Específicos

Realizar a aquisição dos dados de forma automática;

Segmentar os movimentos de mastigação a cada conjunto de amostras coletados;

Extrair as características relacionadas aos sinais de mastigação de diferentes alimentos.

1.4 JUSTIFICATIVA

Agrônomos e veterinários analisam o comportamento ingestivo de ruminantes de forma visual, o que demanda um grande tempo de execução. Por exemplo, o trabalho executado por Santos (2009) utiliza da técnica de visualização direta e de pesagem para verificar a quantidade de alimento que o animal ingere. Outros trabalhos utilizam técnicas acústicas para verificar os eventos mastigatórios e de ruminação desses animais (LACA *et al.,* 1994), (MILONE *et al.*, 2009), (RUTTER *et al.,* 1997), (TRINDADE, 2011). Mesmo nos trabalhos com técnica acústica, o processamento das respostas adquiridas do processo mastigatório dos ruminantes geralmente é efetuado de maneira manual, o que demanda um grande tempo para executar a tarefa. Além disso, efetuar esse processo de forma manual pode gerar imprecisão na resposta obtida.

O uso de sensores FBG com técnicas de aprendizagem de máquina para classificação de padrões pode gerar maior precisão na coleta e classificação dos dados referentes ao comportamento ingestivo de ruminantes (WOSNIAK *et al,* 2012), (PEGORINI *et al.*, 2014), (KARAM *et al.*, 2014). A etapa de pré-processamento é de grande importância para o sucesso dessa técnica, pois é necessário um sistema que

efetue a segmentação da resposta do sensor a cada conjunto de amostras coletado. Os movimentos de mastigação ocorrem em intervalos menores que um segundo, dessa maneira, separar cada atividade ingestiva que o animal esteja efetuando facilita a classificação da mesma. Além disso, a segmentação realizada em conjunto com o sistema de aquisição dos dados permitirá a análise em tempo real dos dados coletados. Com isso, têm-se um conjunto de dados com maior qualidade, ou seja, sem ruídos e com eventos de mastigação completos, ao invés de se ter um conjunto de amostras com um sinal incompleto para ser processado pelo algoritmo de aprendizagem de máquina e, também, automatiza-se a segmentação dos dados obtidos. Além das vantagens que foram citadas, o pré-processamento da resposta dos sensores será empregado com o objetivo de melhorar a resposta dos classificadores, adicionando novas características ao sinal que podem ser utilizadas durante a classificação dos dados.

1.5 ESTRUTURA DO TRABALHO

O trabalho apresentando está estruturado em seis capítulos, sendo o capítulo 2 a revisão teórica, o capítulo 3 apresenta os materiais e os métodos utilizados para desenvolver o trabalho, o capítulo 4 apresenta os resultados que foram obtidos através do trabalho de pesquisa, o capítulo 5 que apresenta as conclusões obtidas com o trabalho desenvolvido.

2 REFERENCIAL TEÓRICO

Neste capítulo, será apresentado o referencial teórico utilizado como base para o desenvolvimento do trabalho.

2.1 MÉTODOS PARA AVALIAÇÃO DO COMPORTAMENTO INGESTIVO DE RUMINANTES

Para se avaliar o comportamento ingestivo de ruminantes, pesquisadores empregam algumas técnicas para se efetuar essa análise. Na literatura é amplamente abordada a utilização do método de observação direta. Além desse método, existem os métodos acústicos que utilizam sensores de áudio para analisar o comportamento ingestivo do animal. Como alternativa ao método acústico, que possui o problema de sinal ruidoso, Wosniak *et al.* (2012) apresentam um método que utiliza sensores ópticos para obter as informações da atividade mastigatório que o animal está desempenhando.

2.1.1 Métodos Diretos

O método direto é uma técnica amplamente utilizada pelos pesquisadores devido a sua praticidade. Esse método possui como vantagem o fato de não ser invasivo, e possui baixo custo. Entretanto, suas principais desvantagens são a imprecisão dos dados, pois como toda observação é efetuada de forma visual, pode ocorrer distrações do observador, ou falhas nas anotações que causarão erros na análise. Outra desvantagem é o tempo necessário para verificar o comportamento ingestivo dos animais, em que usualmente, os pesquisadores precisam revezar na observação, pois normalmente existe a necessidade de observar os animais durante o dia todo, inclusive à noite.

Santos (2009) empregou o método direto para avaliar o comportamento alimentar de ruminantes. Nesse estudo, o autor efetuou a observação direta através do ganho de peso para analisar o comportamento ingestivo dos animais. Assim, os animais foram equipados com fraudas para reter as fezes e a urina. Antes de os animais serem encaminhados para o pasto, esses foram pesados e após o pastejo

foram pesados novamente. Dessa forma, de acordo com o peso ganho durante o pastejo, a observação da quantidade de material excretado, o pesquisador pode verificar como o comportamento ingestivo do animal aconteceu. A Figura 1 mostra os animais equipados com as fraldas, a pesagem e os animais no pasto.

Figura 1 - Animais equipados com fraldas; pesagem dos animais.
Fonte: Santos (2009, p. 22).

Silva *et al.* (2009), também utilizou do método direto para avaliar o comportamento de caprinos em diferentes forragens com diferentes alturas. Nesse trabalho, o principal objetivo era analisar a preferência dos caprinos de acordo com a altura da forragem oferecida. Para efetuar a coleta das informações, os pesquisadores se revezavam para analisar o comportamento do animal, efetuando anotações da preferência do alimento e velocidade de colheita da forragem ou taxa de bocados, utilizando para isso contadores e cronômetros em cada animal. Para análise das informações obtidas, foi utilizado o programa estatístico SAS (SAS, 1993), para avaliar a variância dos dados obtidos.

2.1.2 Método Acústico

Laca *et al.* (1994) empregou o método acústico como uma boa alternativa para verificar movimentos de mastigação. A grande vantagem desse método é de não ser um método invasivo.

Dessa forma, diversos trabalhos utilizam o método acústico para avaliar o comportamento ingestivo do animal. Esses métodos consistem em utilizar um sensor de áudio posicionado na cabeça do animal, sendo que esse sensor adquire os dados

de mastigação que o animal executa (RUTTER *et al.*, 1997), (MILONE *et al.,* 2009), (CLAPHAM *et al.*, 2011), (TRINDADE, 2011).

Além de gravar o áudio da resposta de processos mastigatórios, é necessário se empregar, simultaneamente, o método de contagem visual para, assim, verificar com o sinal obtido, quando ocorreu um movimento da mandíbula do animal. A aplicação de sistemas com método acústico é de difícil segmentação, pois o sinal obtido do sensor geralmente possuiu resposta ruidosa como, por exemplo, máquinas operando próximo ao animal ou até mesmo outros animais próximos ao animal em estudo.

Diante desse problema, Rutter *et al.* (1997) desenvolveu um *software* nomeado *Graze*, para processar os dados provenientes do gravador de áudio digital nomeado IGER *Behavior Recorder*. Com esse *software* é possível visualizar os movimentos mandibulares que o animal executou, pois, o *software* identifica automaticamente os movimentos mandibulares executados a partir dos dados obtidos e pelas configurações que o usuário efetua. Esses eventos podem ser identificados de acordo com a amplitude do sinal em conjunto com o tempo de duração desse evento de mastigação. Com essas informações, o *software* analisa o sinal adquirido e informa a atividade que o animal estava executando naquele momento. Outra maneira para se identificar movimentos de mastigação por esse *software*, é informar um intervalo de tempo para se procurar nesse intervalo de tempo, um certo número de eventos ocorridos, para identificar se isso é uma sequência de sinais de alimentação ou de ruminação. Por fim, o usuário também pode manipular os dados e utilizar regras especificas para discriminar as atividades que o animal executou (RUTTER, 2000).

De forma demonstrativa, na Figura 2, é apresentado o local onde o sensor e o dispositivo para adquirir os dados pode ser posicionado em uma ovelha e na Figura 3 é apresentado o *software Graze*. Os testes efetuados com esse dispositivo foram comparados com o método de visualização direta e os resultados foram satisfatórios (RUTTER, 1997). Para verificar a eficiência entre o método *IGER* e o método acústico, Ungar e Rutter (2006) efetuaram uma comparação entre os dois métodos, e verificou-se que o método acústico teve um desempenho superior ao método *IGER*, mas a diferença é que o método acústico necessita de processamento manual.

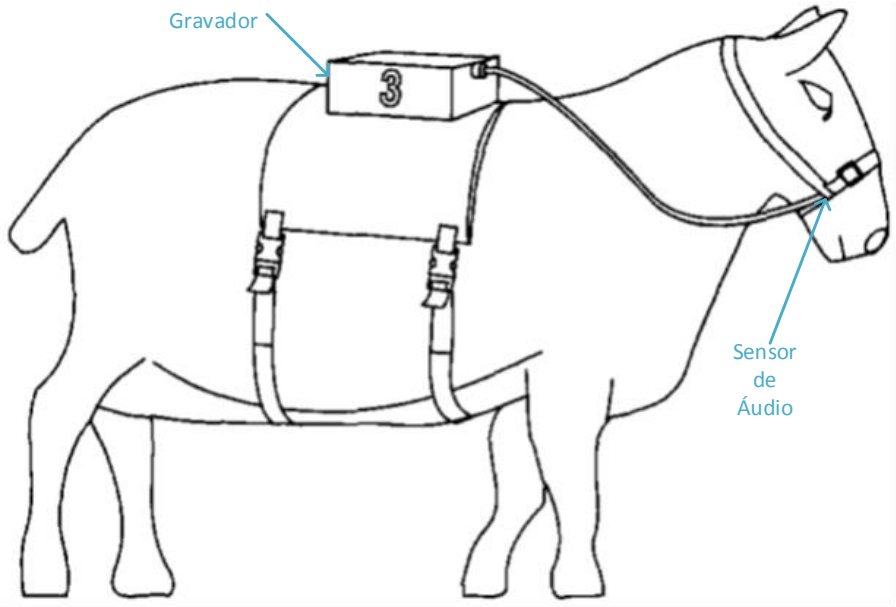

Figura 2 - Local de posicionamento do *IGER* em uma ovelha.
Fonte: Adaptado de Rutter (1997, p. 189).

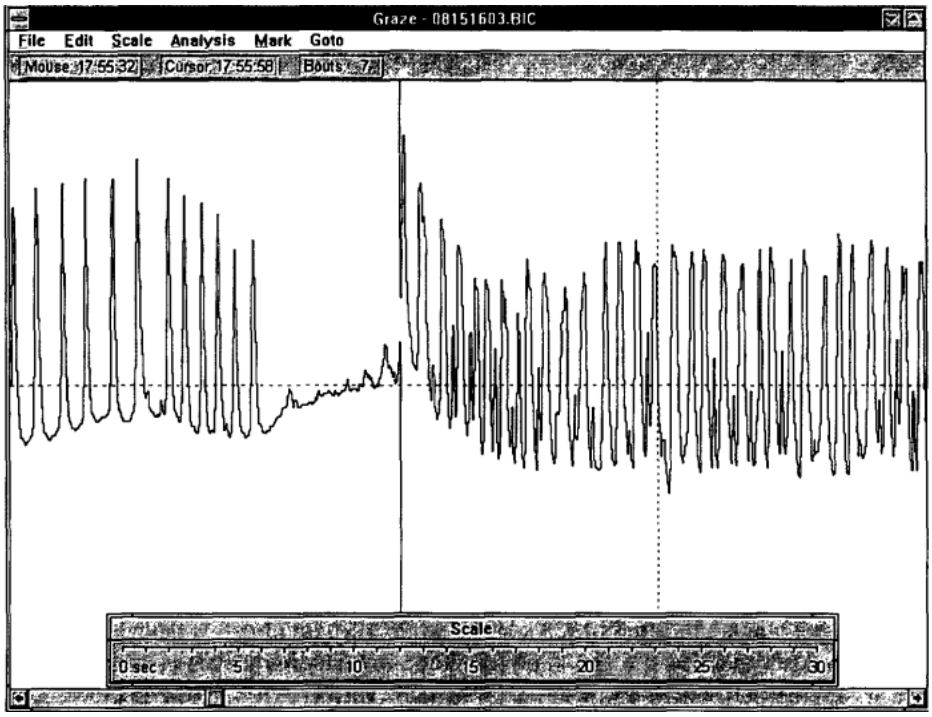

Figura 3 - *Software Graze* de Análise de dados obtidos do *IGER*.
Fonte: Rutter (1997, p. 190).

Trindade (2011) utilizou o método acústico para avaliar o comportamento ingestivo de ruminantes, utilizando microfones para gravação em alta definição e a análise da resposta obtida foi efetuada com *software* para processamento de áudio, sendo o pré-processamento dos dados efetuado manualmente. Mesmo com essas ferramentas, houve divergência entre os dados adquiridos manualmente e os dados

obtidos pelos microfones. Em observações diretas, verificou-se que o animal efetuou atividade de pastejo por 371 minutos com erro de ±61,7 minutos e o método acústico obteve 365 minutos de pastejo com erro de ±55,8 minutos. Já para a ruminação a diferença é maior, sendo que pelo método direto se observou 102 minutos com erro de ±23,5 minutos, e pelo método acústico foi obtido 146 minutos de ruminação com erro de ±28,7 minutos. Na Figura 4 é apresentado o animal com os dispositivos e imagens do *software* com os dados obtidos, (TRINDADE, 2011).

Para melhorar o método acústico (MILONE *et al.,* 2009), desenvolveu um método que utiliza os modelos ocultos de *Markov,* para segmentar os dados obtidos dos sensores de áudio e classifica-los. Com esse método, foi possível reconhecer os dois alimentos que foram fornecidos ao animal, obtendo uma média de 84% de acerto.

Com o mesmo intuito de melhorar o método acústico, Clapham *et al.* (2011), utilizou o *software* Audacity (AUDACITY, 2015) para o pré-processamento dos sinais de áudio obtido em conjunto com o *software* SIGNAL (ENGINEERING..., 2015) para se efetuar a identificação, enumeração e medição dos eventos de mastigação efetuados pelo animal. Os dados foram comparados com o método manual e em 95% dos dados, foram apresentados os mesmos resultados. Nesse trabalho não foi efetuado a classificação das diferentes forrageiras utilizadas durante a alimentação, sendo avaliado apenas o processo de pastejo do animal.

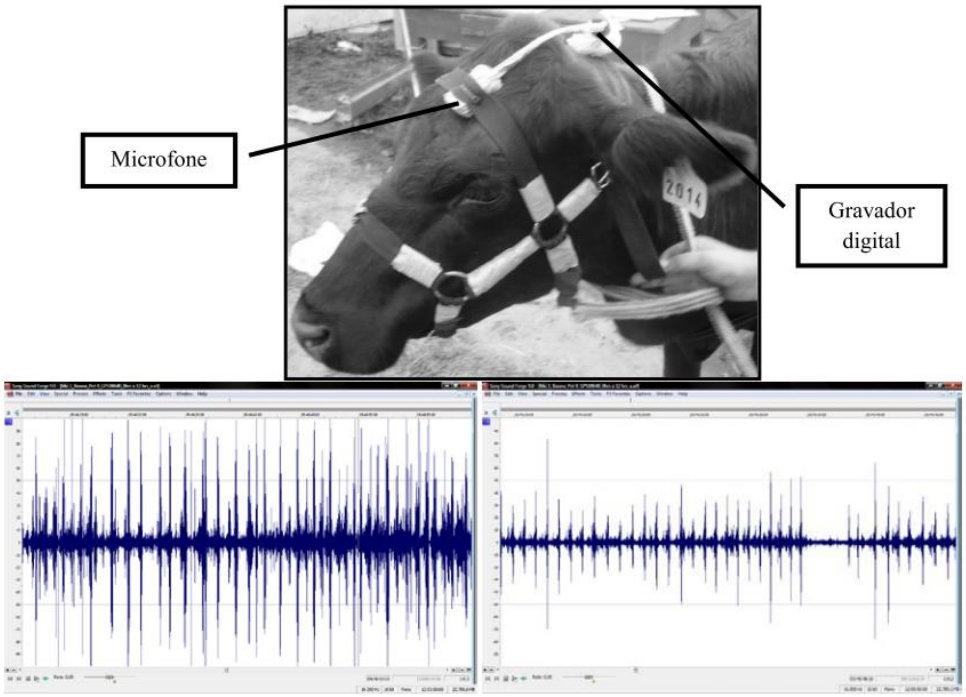

Figura 4 - Bovino equipado com microfone e gravador de áudio e imagens do *software* de processamento de áudio com os dados obtidos.
Fonte: Adaptado de Trindade (2011, p. 32).

2.1.3 Sensor óptico

Uma alternativa para o método acústico, além do método de visualização direta, é a utilização de sensores ópticos. Este método foi proposto por Wosniak *et al.* (2012) e apresenta como vantagem a ausência de interferências externas nos dados coletados, podendo gerar maior precisão nos resultados. A desvantagem desse método é o fato dele ser invasivo. Entretanto, como o sensor tem dimensões reduzidas, é necessária apenas uma pequena cirurgia para posicionar o sensor na mandíbula do animal.

No método empregando esse tipo de sensor, é medida a deformação mecânica da mandíbula do animal conforme a atividade ingestiva que o mesmo executa.

Wosniak *et al.* (2012) e Silva (2014), posicionaram um sensor a fibra ótica em uma caveira de um caprino, cedida pelo Instituto Agronômico do Paraná, para avaliar o tipo de alimento que o mesmo estaria ingerindo conforme a deformação de sua mandíbula. Na Figura 5 é apresentado o crânio com o sensor posicionado.

Com esse crânio, foram realizadas simulações de mastigação de duas plasticinas, aveia, feno e ausência de alimento entre a mandíbula. Dessa forma, os dados obtidos eram classificados utilizando redes neurais artificiais (RNA) treinadas com os dados obtidos. O desempenho geral da rede apresentou taxa de acerto de 100% para as plasticinas, mas ao adicionar novos alimentos, como o feno e a aveia, houve perda de desempenho do classificador. Com os dados que foram obtidos dessa simulação de mastigação, Pegorini *et al.* (2014) utilizaram a técnica de aprendizagem de máquina por árvores de decisão e obteve-se resposta média com 78,75% de acerto para cinco classes.

Além de utilizar um crânio de caprino, Karam *et al.* (2014) posicionaram uma FBG na mandíbula de um bovino, sendo esse um ensaio *Ex vivo*, diferente do ensaio *In Vitro* executado com a caveira. Para esse caso, foram utilizadas as mesmas plasticinas que Wosniak utilizou, e como terceiro padrão, foram executados ensaios sem material na arcada dentária. Com os dados obtidos, foi treinada uma RNA que obteve, em média, 95% de acerto.

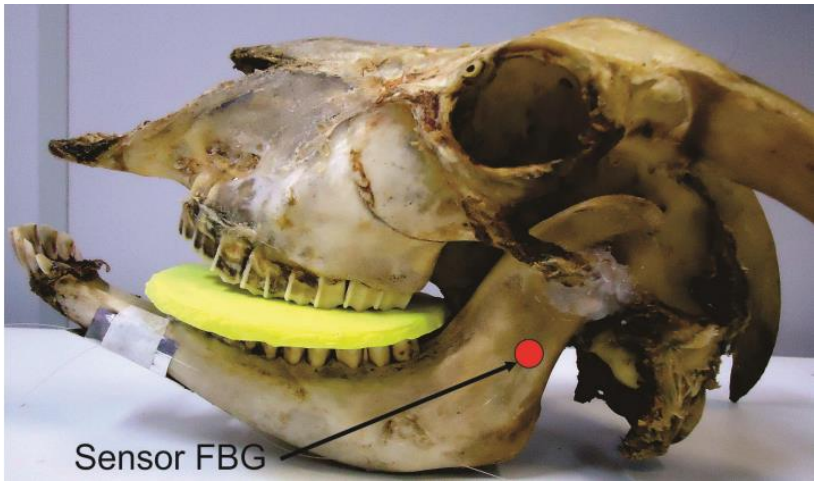

Figura 5 - Caveira de caprino com sensor a fibra ótica posicionado na mandíbula da caveira.
Fonte: Wosniak (2012, p. 2).

2.2 REDES DE BRAGG EM SENSORES A FIBRA ÓTICA

Dentre as técnicas mais utilizadas para o desenvolvimento dos sensores estudados, destaca-se a de Redes de Bragg em fibras óticas. Essas redes são constituídas por uma modulação periódica no índice de refração da fibra através da direção longitudinal. Essa modulação provoca uma reflexão seletiva de uma pequena parte do espectro de um sinal de banda larga, que é transmitido pela fibra. Esta reflexão é baseada na difração do feixe, esse formado pela estrutura de modulação do índice de refração, centrada em um comprimento de onda específico, denominado comprimento de onda de Bragg (OTHONOS, 1997). A Figura 6 mostra a rede de Bragg em uma fibra ótica.

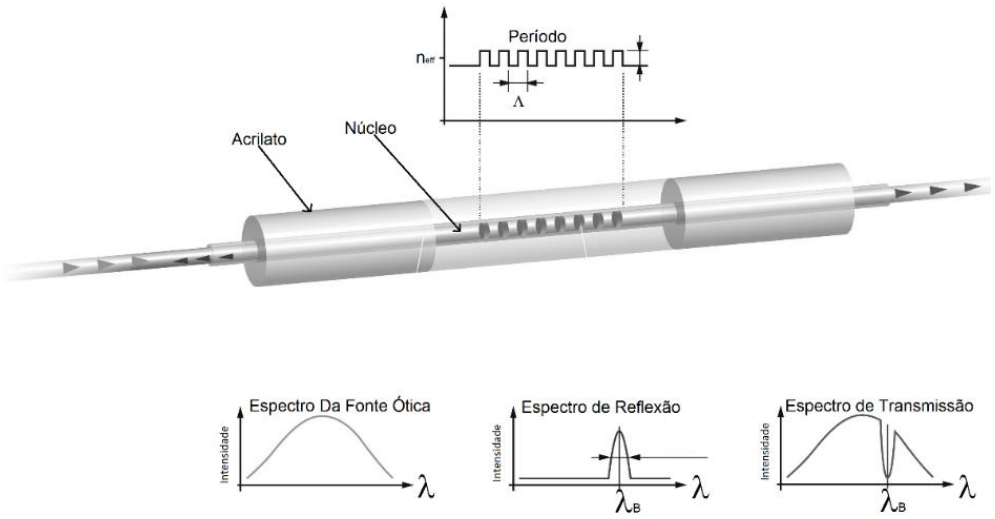

Figura 6 - Rede de Bragg e seus princípios de funcionamento.
Fonte: Traduzido de HBM (2014a, p.10).

Esse tipo de sensor é sensível a grandezas físicas como temperatura e deformação, isto devido ao efeito termo ótico e fotoelástico da fibra. Havendo mudanças na periodicidade espacial, "Λ", ou no índice de refração efetivo, "n_{eff}", da rede, implica na mudança do comprimento de onda de Bragg, "λ_B". Portanto, qualquer alteração nas grandezas mencionadas, como temperatura ou deformação mecânica, provocará mudança relativa na posição do espectro de reflexão de Bragg. Esse deslocamento é representado pela equação (1) (OTHONOS, 1997).

$$\Delta\lambda_B = 2\left(\Lambda\frac{\partial n_{eff}}{\partial l} + n_{eff}\frac{\partial\Lambda}{\partial l}\right)\Delta l + 2\left(\Lambda\frac{\partial n_{eff}}{\partial T} + n_{eff}\frac{\partial\Lambda}{\partial T}\right)\Delta T \quad (1)$$

onde, "l" é o comprimento da rede e "T" a temperatura. A primeira parcela da equação (1) é relacionada ao efeito da deformação mecânica, enquanto a segunda parcela é referente a variação da temperatura (OTHONOS, 1997).

Este trabalho dará ênfase na utilização de Redes de Bragg apenas para monitorar deformação mecânica. Quando o sensor for utilizado para medir deformações mecânicas, apenas a primeira parcela da equação (1) é considerada, pois considera-se que o segundo termo é constante, em que, para se garantir isso, o sensor deve ser aplicado em um local onde a temperatura seja constante. Como este sensor será aplicado em um animal que possui temperatura corporal constante, considera-se que não haverá variações de temperatura, e desta forma, a segunda parcela da equação (1) é desconsiderada. Com o aumento ou decréscimo da deformação, tem-se o deslocamento no comprimento de onda de Bragg, devido à propriedade fotoelástica que resulta da modificação no espaçamento da rede e consecutivamente na mudança do índice de refração. Este deslocamento, também pode ser representado pela equação (2):

$$\Delta\lambda_B = \lambda_B(1 + p_{\mathrm{e}})\varepsilon_{\mathrm{Z}} \quad (2)$$

em que a componente de deformação por unidade de comprimento é representada por "ε_Z" e a constante fotoelástica é representada por "p_e", que pode ser definida pela equação (3):

$$p_{\mathrm{e}} = \frac{n_{eff}^2}{2}\left[p_{12} - v(p_{11} + p_{12})\right] \quad (3)$$

sendo que "p_{11}" e "p_{12}" são componentes do tensor foto-elástico. A partir do coeficiente de Poisson para a sílica no núcleo da fibra, representado por "v", $p_{11} = 0{,}113$, $p_{12} = 0{,}252$ e $v = 0{,}16$. Com esses parâmetros e $n_{eff} = 1{,}482$ a equação (2) apresenta sensibilidade de 1,2 pm para 1 μ strain de deformação da rede de Bragg centrada no comprimento de onda de 1550 nm (OTHONOS; KALLI, 1999).

2.3 PRÉ-PROCESSAMENTO

Nos trabalhos anteriores, o pré-processamento dos dados era efetuado de acordo com um tempo de aquisição ou de acordo com a quantidade de amostras que foram adquiridas por segundo (WOSNIAK *et al,* 2012), (PEGORINI *et al.*, 2014) e (SILVA, 2014). Outra possibilidade era o sinal ser segmentado de acordo com a quantidade fixa de amostras que se estima que seja o tamanho do movimento de mastigação (KARAM *et al.*, 2014). A execução do pré-processamento por essas técnicas pode gerar imprecisão, pois para os dois casos utilizados anteriormente, existe a possibilidade de ocorrer mais de um evento de mastigação na quantidade de amostras que foram coletadas. Além disso, no final ou início da amostragem, o sinal pode não ter sido totalmente transmitido. Outro problema com a técnica em que se estima o tamanho (em número de amostras) do movimento de mastigação é de o tamanho escolhido para o movimento de mastigação, ser menor ou maior que um movimento completo, ou seja, o sinal será segmentado de forma errônea. Assim, com essas técnicas de pré-processamento, o sinal poderá conter falhas, inserindo erros no conjunto de dados para classificação.

Diante desses problemas, um algoritmo que pré-processe os dados de forma automática, pode auxiliar de forma positiva na etapa de classificação dos dados, em que, com o pré-processamento, se ganha maior precisão no conjunto de dados utilizado nos classificadores. Como o algoritmo de pré-processamento fará a segmentação do sinal, de forma que cada atividade mastigatória que o animal esteja executando seja analisada individualmente. Isso possibilita a análise da atividade do animal em tempo real. Assim, os dados pré-processados podem ser submetidos para classificação logo após o seu pré-processamento. Como mencionado anteriormente, a classificação, não faz parte do escopo deste trabalho.

2.3.1 A Transformada Rápida de Fourier

A análise de Fourier normalmente é utilizada para efetuar uma análise do sinal obtido no domínio frequência, ou seja, fornece o espectro de frequência do sinal.

Para se analisar sinais amostrados, normalmente, se utiliza a série de Fourier de tempo discreto (HAYKIN; VAN VEEN, 2001). A representação pela série de Fourier de tempo discreto (*Discrete Time Fourier Series* - DTFS) de um sinal x[n], é dada por (HAYKIN; VEEN, 1999):

$$x[n] = \sum_{k=\langle N \rangle} X[k]\, e^{jk\Omega_0 n}, \quad (4)$$

$$X[k] = \frac{1}{N} \sum_{n=\langle N \rangle} x[n]\, e^{-jk\Omega_0 n}, \quad (5)$$

sendo que $\Omega_0 = 2\pi/N$, em que "N" é o período fundamental de x[n]. Os sinais obtidos pelas equações (4) e (5) são um par DTFS, e esses sinais fornecem uma descrição completa do sinal (HAYKIN; VAN VEEN, 2001).

Mesmo utilizando a série de Fourier para obter a sua transformada, sua execução em sistemas digitais tem alto custo computacional. Diante desse problema, foram desenvolvidos algoritmos para efetuar a transformada de Fourier de forma mais rápida. Por exemplo, Cooley e Tukey, propuseram um algoritmo que reduziu a ordem de complexidade para o cálculo da transformada de Fourier de N^2 para $N \log_2 (N)$, sendo $N = 2^n$ (DUHAMEL; VETTERLI, 1990). Esse algoritmo se denomina transformada rápida de Fourier (Fast Fourier Transform - FFT).

Como a DTFS é periódica, a FFT utiliza dessas propriedades para efetuar um número menor de cálculos. Esse algoritmo utiliza a técnica de dividir as sequencias da DTFS em sequências menores, o que diminui expressivamente a ordem do algoritmo para o cálculo da transformada de Fourier (CHASSAING, REAY, 2008).

3 MATERIAIS E MÉTODO

Como o objetivo do trabalho desenvolvido é efetuar o pré-processamento de sinais proveniente de sensores a fibra ótica, utilizados para medir a deformação mecânica na mandíbula de um ruminante, foi necessário definir alguns materiais que tornassem possível o desenvolvimento desse trabalho. Além dos materiais, foi necessário definir um método para a execução da solução proposta pelo trabalho. Esses tópicos são apresentados neste capítulo.

3.1 MATERIAIS

3.1.1 Ferramenta de desenvolvimento LabVIEW™

Ambiente integrado de desenvolvimento (*Integrated Development Environment* – IDE), em linguagem G, uma linguagem de programação gráfica, que proporciona maior facilidade de desenvolvimento do código e da interface que o usuário final utilizará. Semelhante a ferramenta de simulação matemática Simulink® da MathWorks®, o LabVIEW™ utiliza o conceito de diagrama de blocos para se construir um código ao invés de linhas de programação. Cada programa gerado é chamado de Instrumento Virtual (*Virtual Instrument* – VI).

No LabVIEW™, a execução do código é efetuada com o conceito de fluxo de dados (*Data flow*), que consiste em executar uma parte do código e após essa execução, enviar os dados para a próxima sequência de códigos, através de conexões que o programador efetua entre as estruturas ou objetos. Essa IDE proporciona uma maior agilidade para construir uma interface gráfica e também efetuar conexão com dispositivos, além de proporcionar maior facilidade para salvar dados em arquivos, gerar um arquivo executável e um instalador.

Esse IDE, em seu núcleo, possui parte dos códigos desenvolvidos em linguagem C/C++ proporcionando assim, maior eficiência na execução do código. Além dos blocos e conexões, é possível inserir outras linguagens de programação dentro do código, como C e scripts Matlab (NATIONAL INSTRUMENTS, 2015). Na Figura 7 é apresentada a interface que o desenvolvedor utiliza para desenvolver seus VI’s.

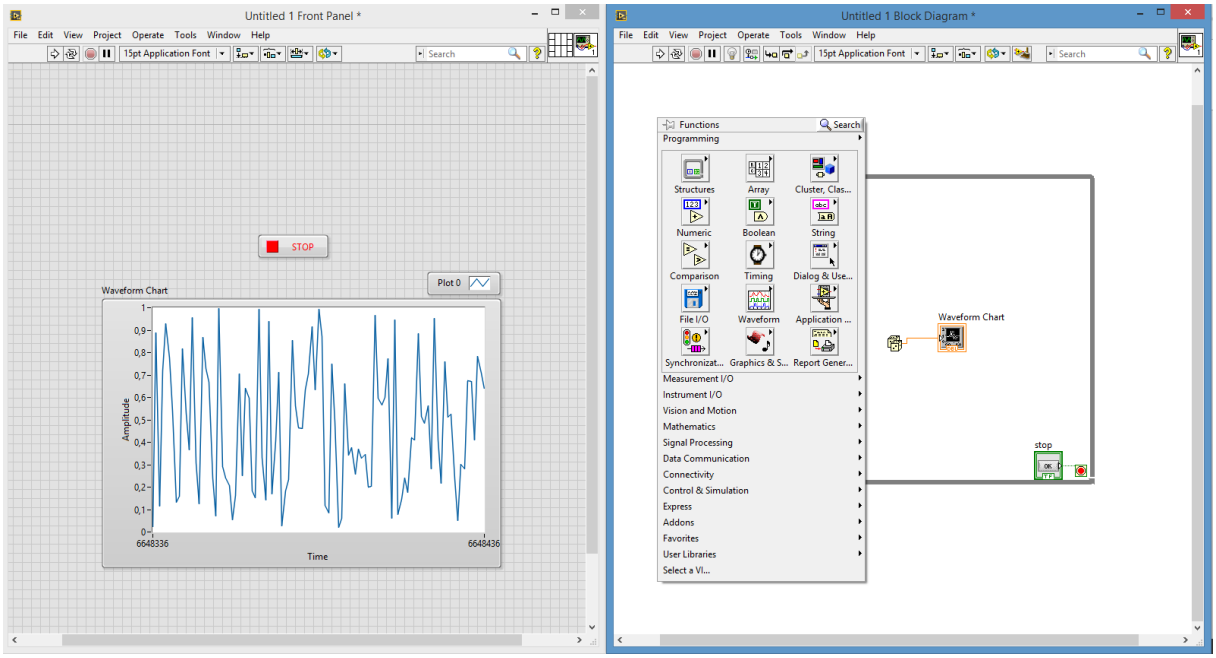

Figura 7 **- Interface da IDE LabVIEW™.**

A utilização desse IDE se torna necessária neste trabalho, pelo fato de os fabricantes dos dispositivos que forão utilizados para adquirir as amostras dos sensores, disponibilizarem *drivers* para conexão com os mesmos. Assim, utiliza-se esses *drivers* para adquirir os sinais e configurar os dispositivos.

3.2.2 Sistema para monitoramento de sensores baseados em Redes de Bragg

Neste trabalho foram utilizados dois interrogadores ópticos do fabricante HBM (*Hottinger Baldwin Messtechnik GmbH*), sendo seus respectivos modelos nomeados SI101 e DI410. No Quadro 1 é apresentada uma comparação entre cada modelo.

Especificação	**SI101**	**DI410**
Número de Conectores	1	4
Faixa de Abrangência de medição de comprimento de onda	1520 – 1570nm	1510 – 1590nm
Variação no comprimento de onda	<2pm	<2pm
Repetitividade	0,5 com uma amostra por segundo	0,05pm em uma média de 1000 valores medidos
Taxa de Aquisição máxima	1 amostra por segundo	1000 amostras por segundo

Quadro 1 - Especificações técnicas dos interrogadores ópticos.
Fonte: HBM(2014a, p.46) e HBM(2014b, p.45)

Em relação a qual modelo de interrogador utilizar, deve-se analisar o ambiente ao qual os sensores a serem monitorados estão submetidos. Comparando-se os modelos, o modelo SI101 tem seu uso mais propício para sistemas estáticos, onde

haja pouca variação abrupta de valores. Já o Modelo DI410, por ter uma taxa de aquisição maior, é utilizado em sistemas onde exista variações rápidas, em que seja necessário adquirir vários pontos por segundo, ou seja, sistemas dinâmicos. Os dois modelos utilizam a técnica de filtro ajustável de Fabry-Perot para detectar os sensores ópticos (HBM, 2014a) (HBM, 2014b).

Como nesse trabalho de pesquisa, um evento de mastigação pode ter uma duração menor que um segundo, e devido a isto é considerado um sistema dinâmico, para aquisição dos dados foi utilizado o interrogador de modelo DI410. O Modelo SI101 foi utilizado para verificar o espectro da fibra ótica que contém o sensor, pois devido a um problema de *firmware*, o modelo DI410 não está apresentando o espectro da fibra conectada.

3.2 MÉTODO

Para este trabalho, inicialmente foram utilizados dados de um ensaio *In Vivo*, onde uma microcirurgia foi efetuada na mandíbula de um bovino para o posicionamento da FBG. A Figura 8 apresenta o local de posicionamento do sensor no bovino. Como não é o escopo deste trabalho, a instrumentação do sensor foi desenvolvida em um projeto paralelo ao projeto de pré-processamento e classificação dos dados. O sensor foi desenvolvido de forma que o corpo do animal não tivesse rejeição a estrutura construída. Foi utilizada uma placa e parafusos de titânio para fixação da mesma na mandíbula, bem como, um cateter intravenoso para conduzir a fibra no interior do animal.

Após a microcirurgia, foram executados ensaios para se obter os dados do sensor. Foram coletados dados referentes a três tipos de alimentos: ração, feno e azevém. Também foi monitorado o processo de ruminação do animal, e os momentos que o animal não possuía alimento na arcada dentária. Assim empregou-se de visualização direta, anotação dos momentos em que o animal executava alguma atividade e também foi efetuada uma gravação em vídeo para após o processo de aquisição, caso necessário, recorrer a esta gravação para se verificar que atividade o animal estava executando em um devido momento

Após a etapa de aquisição dos dados, um algoritmo foi projetado para efetuar a segmentação dos mesmos. Na segmentação, será possível o usuário escolher se deseja utilizar a FFT e quantos elementos da FFT do sinal serão utilizados. O

algoritmo de segmentação será utilizado posteriormente em conjunto com o sistema de aquisição de dados do interrogador óptico. Por isso, o mesmo foi desenvolvido de forma a minimizar o tempo de execução, pois, como se tem 1000 amostras a cada 1 segundo, é de grande importância que o algoritmo execute em um tempo inferior a este.

Figura 8 - Local demonstrativo de posicionamento da FBG.

Com o algoritmo de segmentação desenvolvido, o mesmo foi integrado ao sistema de aquisição, para o sinal proveniente do sensor ser processado em tempo real. Para exemplificar melhor as etapas deste trabalho, a Figura 9 apresenta um diagrama que indica os passos que serão seguidos.

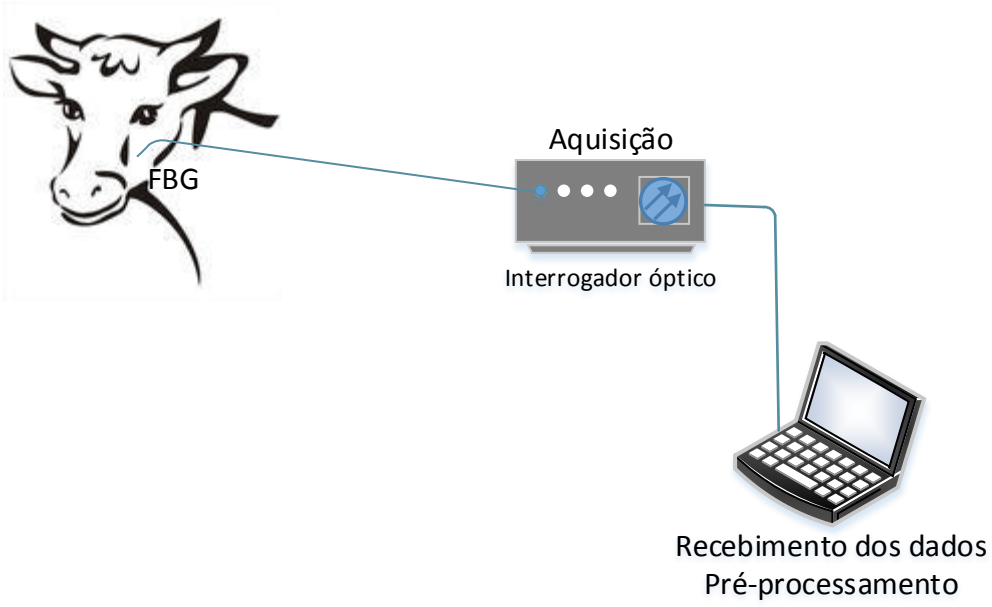

Figura 9 - Diagrama de passos a serem seguidos para pré-processar o sinal.

Após a segmentação dos dados obtidos, os mesmos podem ser submetidos a técnicas de aprendizagem de máquina, para treinamento e classificação. Com isso, foi desenvolvido uma função para salvar os dados obtidos para serem usados posteriormente. A classificação dos dados segmentados não faz parte do escopo desse trabalho.

4 ALGORITMOS PARA SEGMENTAÇÃO

O presente capítulo tem o propósito de apresentar e analisar os algoritmos desenvolvidos neste trabalho. Serão apresentados os algoritmos em execução para segmentação dos dados.

4.1 ALGORITMO DE SEGMENTAÇÃO

A partir dos dados de movimentos mastigatórios de um bovino, obtidos no ensaio *In Vivo*, é possível visualizar quando um evento de mastigação ocorre, em que o sinal proveniente da FBG varia de acordo com a atividade mastigatória que o animal esteja executando. Isso pode ser verificado na Figura 10.

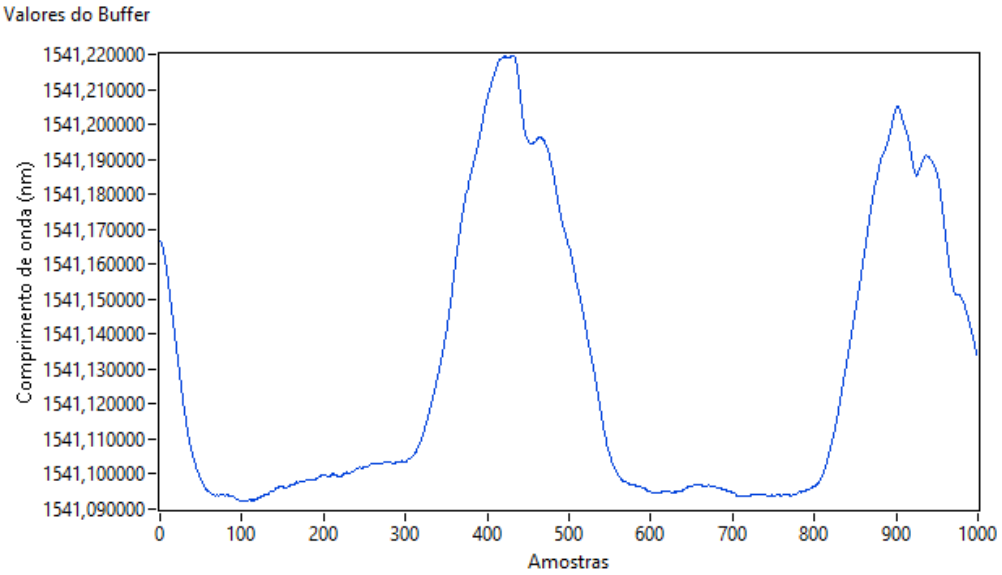

Figura 10 – Alteração do sinal obtido da FBG quando o animal efetua evento de mastigação.

Essa alteração, terá sua amplitude e o tempo de duração do evento de mastigação modificado, conforme o alimento ou atividade que o animal estiver executando em determinado momento. No ensaio *In Vivo* executado, foi fornecido ao animal feno, ração e azevém para ele se alimentar. Além dos alimentos, foram adquiridos dados de quando o animal estava com ausência de alimento em sua mandíbula e quando ele estava ruminando.

Para fazer a análise desse sinal, é necessário centralizá-lo. Apenas com as medidas em comprimento de onda, a identificação da ocorrência de um movimento

de mastigação ou não, pode dificultar processamento do sinal, pois cada sensor óptico pode apresentar valores de comprimento de onda específicos, devido a gravação do comprimento de onda do sensor (OTHONOS, 1997). Após o seu posicionamento na mandíbula do animal, esse valor de comprimento de onda pode alterar. Dessa forma, o valor da base do sinal não possui um centro definido, necessitando, centralizar o sinal proveniente do sensor a cada amostragem.

Para efetuar a centralização, são executados os seguintes passos. Inicialmente é adquirida uma quantidade de amostras do sensor. Como a aquisição dos dados de mastigação do animal para este trabalho foi executada na taxa de 1000 amostras por segundo, a quantidade de amostras a serem adquiridas foi definida como 1000. Essas amostras foram armazenadas em um vetor temporário, no caso um *buffer*. Com os valores armazenados no *buffer*, executa-se o cálculo da média desse conjunto de valores. Com a média obtida, é realizada a subtração de cada amostra do sinal presente no *buffer* pela média. Dessa maneira, o sinal estará centralizado e será armazenado em outro vetor. Essa centralização facilita a identificação das alterações que ocorrem no sinal de acordo com a atividade executada pelo animal, em que, com os valores centralizados, é possível identificar com maior exatidão o início e o final do evento de mastigação, pois com o sinal centralizado, é possível identificar os pontos em que o sinal cruza o valor de zero.
Utilizando o sinal que é apresentado na Figura 10, o mesmo foi centralizado e está apresentado na Figura 11. Nesta figura, é dado destaque na origem, nos pontos que cruzam a origem e nos pontos de mínimos encontrados nesse sinal centralizado.

Comparando-se os sinais da Figura 10 e Figura 11, verifica-se que efetuando a centralização, as características do sinal não são perdidas, sendo que a principal diferença é a escala do sinal, em que, o sinal centralizado tem uma escala onde é possível identificar com maior precisão onde se inicia o evento de mastigação e onde ela foi finalizada.

Esse sinal centralizado é utilizado para verificar se ocorreu um movimento de mastigação. O sinal que será segmentado é o sinal original (sinal contido no *buffer)*, ou seja, o sinal com valores em comprimento de onda (nm). A centralização do sinal foi implementada em conjunto com a aquisição das amostras enviadas pelo interrogador. Assim, a Listagem 1 apresenta o pseudocódigo para adquirir os dados e centralizar o vetor. O pseudocódigo não apresenta a parte da aquisição das amostras do interrogador, pois as amostras assim que são recebidas, são inseridas em uma fila

e são retiradas da fila na centralização do sinal. Desta forma, evita-se que alguma amostra seja perdida durante o pré-processamento.

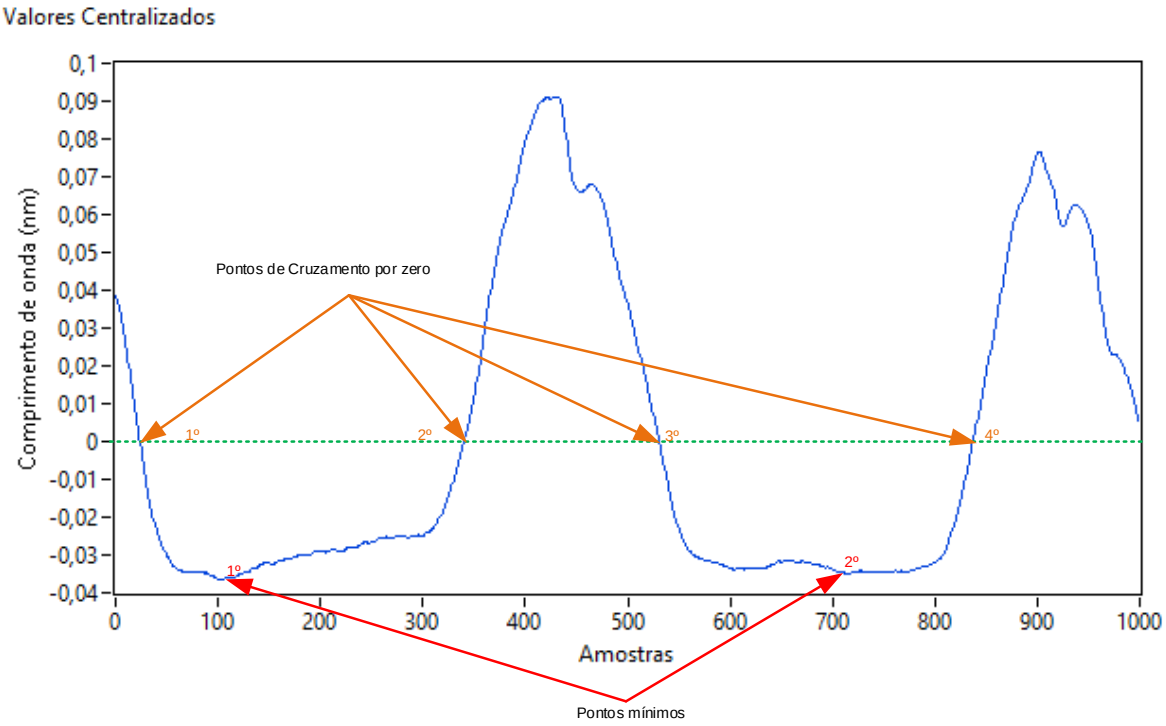

Figura 11 - Sinal centralizado.

```
inteiro valor_valido = 0;//valor válido recebido do interrogador
inteiro valor_recebido = 0;//valor recebido do interrogador
Real vetor_buffer[1000];//vetor com os valores do buffer
inteiro i_buffer = 0;//quantidade de amostras no buffer, índice do vetor de buffer
Real media = 0;//média do sinal no buffer
Real vetor_centralizado[1000];//vetor com os valores do buffer centralizados
inteiro qtd_elem_obtidos = 0;//quantidade de elementos obtidos no sinal segmentado
Real sinal_seg[1000];//sinal segmentado pelo algoritmo;
inteiro i_cent = 0;//índice do sinal centralizado
inteiro sem sinal algoritmo;//algoritmo selecionado
Booleano[1] sinal_novo = FALSE;//indica se um sinal novo foi inserido no buffer
Booleano minimo = FALSE;//indica se um mínimo foi encontrado

/*A aquisição das amostras do interrogador é efetuada em um laço paralelo a este. Essas
amostras são inseridas em uma fila, pois assim, durante o processamento do sinal, não se
perderá as amostras que foram adquiridas durante o processamento*/
enquanto existir amostras para serem analisadas, então faça{
	//retira da fila neste laço
	valor_recebido = retira_fila();//retira amostra da fila
	se(i_buffer < 1000) então{
		/*primeiro verifica se o valor recebido está na faixa de leitura do
		interrogador. Isto é efetuado para evitar ruídos, pois caso o valor seja
		menor que 1510 ou maior que 1590, estará fora da faixa de comprimento de
		onda de interrogador identificará sensores, apresentando assim, valores
		inválidos como 0, valores negativos ou valores na casa dos milhares*/
		se((valor_recebido > 1510) && (valor_recebido < 1590)) então{
```

[1] Booleano: Tipo de Variável que aceita como valor Falso ou Verdadeiro, (0 ou 1), ou seja, variável de lógica binária, referente a Álgebra Booleana (DAGHLIAN, 1995).

```
                        /*se estiver na faixa de valores que o interrogador efetua a
                        leitura*/
                        vetor_buffer[i_buffer] = valor_recebido;
                        /*valor_recebido recebe o ultimo valor válido recebido. Isto é
                        efetuado pois quando se encontrar um valor inválido, este valor
                        será utilizado ao invés do valor errôneo*/
                        valor_valido = valor_recebido;
                }
                Senão {
                        //recebe o ultimo valor válido
                        vetor_buffer[i_buffer] = valor_valido;
                }
                i_buffer++;//incrementa valor de buffer
                /*Caso o sinal que está sendo processado pelo algoritmo de mínimos seja
                atualizando enquanto procura o próximo mínimo para finalizar o
                processamento do sinal então*/
                Se (minimo == TRUE) então
                {
                        //indique que um novo sinal foi inserido
                        sinal_novo = TRUE;
                }
                Senão
                {
                        //reinicie a variável que indica novo sinal
                        sinal_novo = FALSE;
                }
        }
        Senão {//se o valor de i_buffer for maior que 1000
                i_buffer = 0;//zere o índice do buffer
                media = media(vetor_buffer);//obtém a média das amostras no buffer
                /* É utilizada uma função para normalizar o
                vetor.centraliza_vetor(vetor[],valor): Esta função retorna o vetor
                centralizado e para fazer isso deve-se informar o vetor a ser centralizado,
                e o valor para a centralização. Na execução dessa função, analisa-se todas
                as posições do vetor informador, efetuado a subtração do valor contido no
                vetor com o valor informado na variável valor.*/
                vetor_centralizado = centraliza_vetor(vetor_buffer,media);
                Se(qtd_elem_obtidos == 0) então{
                        /*utiliza uma função para inicializar um vetor.
                        Inicializa_vetor(vetor[],valor,tamanho): Esta função percorre todos
                        os índices do vetor, em que cada índice se insere o valor que o
                        usuário desejar, é necessário informar a quantidade de índices que
                        se deseja analisar. Neste caso o sinal segmentado será inicializado
                        com a média*/
                        inicializa_vetor(sinal_seg,media,1000);
                }
                i_cent = 0;
                Enquanto(i_cent < 1000) faça
                {
                        //pré-processa o sinal que está no buffer
                        Se(algoritmo == 'Z') então
                        {
                                /*utiliza o algoritmo de
                                Zeros para analisar o sinal*/
                        }
                        Senão (algoritmo == 'M') então
                        {
                                /*utiliza o algoritmo de
                                mínimos para analisar o sinal*/
                        }
                }
        }
}
```

Listagem 1 - Pseudocódigo para aquisição e centralização das amostras do interrogador.

Com o sinal centralizado, é necessário verificar se houve um movimento de mastigação. Para isso, foram utilizadas duas abordagens que resultaram em dois algoritmos de pré-processamento. Uma das abordagens analisa o sinal em busca de pontos que tenham cruzado por zero, que significa encontrar um zero nos dados coletados. Diante disso, esse algoritmo foi nomeado algoritmo de cruzamento por zeros. Já a outra abordagem analisa o sinal em busca do ponto mínimo naquele movimento de mastigação. Como essa é sua principal função, o algoritmo foi nomeado de algoritmo de mínimos.

4.1.1 Algoritmo de Cruzamento por Zeros

Após os dados serem obtidos e centralizados, o algoritmo de cruzamento por zeros pode ser utilizado para identificar os eventos de mastigação que o animal executou. Esse algoritmo é iniciado após os dados serem centralizados, pois como o *buffer* e os dados centralizados tem a mesma quantidade de elementos, o índice para analisar esses vetores pode ser o mesmo. O conjunto centralizado é utilizado para encontrar em que índice o sinal cruzou por zero, para, assim, o sinal que está no *buffer* ser segmentado.

Sendo o princípio de funcionamento desse algoritmo a procura por cruzamento por zero, ou seja, os zeros do sinal centralizado, será utilizada a Figura 11 para explicar o funcionamento deste algoritmo.

Com o sinal centralizado, o algoritmo irá percorrer o sinal até encontrar o primeiro zero. Analisando a figura com o sinal centralizado, e supondo-se que se deseja segmentar o sinal que está no centro da figura, o primeiro zero é encontrado quando o valor anterior for negativo e o valor atual for positivo, ou o valor atual for igual a zero, na figura este ponto é representado pelo segundo ponto onde se cruza por zero. Mas como pode ocorrer de se ter ruído no local onde se identificou o zero, é necessário analisar uma certa quantidade a mais de amostras para verificar se o zero encontrado é válido, pois caso seja somente um ruído que contém vários zeros, o sinal será segmentado de forma errada se esta condição não for analisada. Neste trabalho, a quantidade de amostras a mais que serão analisadas para verificar a validade desse zero foi considerada como sendo 75 amostras. Esse número foi definido de forma empírica após testes efetuados com os movimentos adquiridos no ensaio *In Vivo*, e a partir desses testes, verificou-se que em alguns eventos, existe uma variação do sinal

em torno do zero, e caso essa verificação seja desconsiderada, o algoritmo efetuará a segmentação de um sinal que pode conter somente uma amostra ou segmentar o sinal no índice errado. Assim, com esses testes, obteve-se o valor de quantas amostras devem ser analisadas, de forma que esse valor não prejudique o sinal, e o valor encontrado foi de 75 amostras.

Essa verificação é necessária para evitar que o conjunto de dados contenha dados errôneos, o que pode prejudicar a análise das atividades que o animal estava executando. Com essas condições satisfeitas, o índice utilizado para analisar as amostras dos vetores, é subtraído das 75 amostras que foram verificadas. Com o índice no ponto de zero do sinal, este índice é subtraído de mais 50 amostras. Esta ação é necessária pelo fato de o sinal centralizado ter seu zero localizado algumas amostras anteriores ao cruzamento por zero, em que este número foi definido a partir de ensaios empíricos, verificando-se assim que este foi o valor que apresentou melhores resultados. Isto pode ser visualizado na Figura 11, em que, o cruzamento por zero está algumas amostras a frente do ponto do início do evento de mastigação. O próximo passo é encontrar o próximo índice onde ocorreu o cruzamento por zero, ou seja, quando o valor atual for positivo e o próximo for negativo. Esse ponto é encontrado no terceiro ponto de cruzamento por zero, demarcado na Figura 11. Neste caso, também é efetuada a verificação se este é um “falso” cruzamento por zero. Finalmente, para se encontrar o final do movimento de mastigação, é efetuada a procura do terceiro cruzamento por zero. Quando este ponto é encontrado, além de se verificar a “validade” desse cruzamento, é verificado se esta análise ocorrerá ao final das amostras contidas no *buffer*. Caso isso ocorra, o algoritmo armazena essas amostras para que as mesmas não sejam perdidas na análise. O final do sinal pode ver visualizado pelo ponto 4 apresentado na Figura 11. Assim, o final do sinal segmentado é preenchido com a média dos valores contidos no *buffer* em comprimento de onda, pois a análise dos zeros adiciona amostras que não serão utilizadas. A Listagem 2 apresenta o algoritmo em pseudocódigo.

```
Real valor_atual = 0;//valor atual do sinal centralizado
Real prox_valor = 0;//próximo valor do sinal centralizado
inteiro cruzou_zero = 0;//contador que indica quantas vezes cruzou por zero
Booleano Safe_zero2 = FALSE;//verificação de zero "verdadeiro". Zero encontrado aceito
Booleano Safe_zero = FALSE;//verificação de zero "verdadeiro". Ativa contador
inteiro c_safe_zero = 0;//contador para verificação de zero "verdadeiro"
Booleano zero = FALSE;//variável que indica se um zero foi aceito
Real temp[1000];//variável temporária
Booleano FFT;//variável que indica se será ou não utilizado a FFT
```

```
inteiro qtd_sinais = 0;//quantidade de sinais segmentados
inteiro qtd_parametros;//quantidade de parâmetros utilizados da FFT
Real sinal_seg_fft[1000+qtd_parametros];/*sinal segmentado com a quantidade de parâmetros
                                        selecionados da FFT*/
Booleano Zero_Border;//variável que indica se houve um zero no final do buffer
Real zero_border_temp[1000];/*variável temporária que salva os valores antes do final do
                            buffer*/


valor_atual = vetor_centralizado[i_cent];
i_cent++;//incrementa índice do vetor centralizado
prox_valor = vetor_centralizado[i_cent];
/*a função ver_sinal(valor) verifica se o valor passado é positivo, negativo ou zero.
Essa função retornará 1 se for positivo, 0 se for zero e -1 se for negativo. Se o valor
atual tiver sinal diferente do próximo valor ou se o zero encontrado é verdadeiro*/
Se (((ver_sinal(prox_valor)) != (ver_sinal(valor_atual))) || (Safe_zero2 == TRUE)) então
{
	/*faça as seguintes verificações. Se já encontrou algum cruzamento pela origem*/
	Se (cruzou_zero != 0) então
	{
		//Caso o teste de zero verdadeiro ainda não tenha sido efetuado
		Se (Safe_zero2 == FALSE) então
		{
			//ative a verificação
			Safe_zero = TRUE;
		}
	}
	/*Se ainda não foi encontrado um ponto em que as amostras cruzem a origem ou a
	verificação de zero verdadeiro já foi efetuada*/
	Se (cruzou_zero == 0 || Safe_zero2 == TRUE) então
	{
		//reinicie as variáveis de verificação zero falso
		Safe_zero = Safe_zero2 = zero_border = FALSE;
		//incrementa o contador de cruzamento de origem
		cruzou_zero++;
		//reinicia variável de contagem de verificação de zero "verdadeiro"
		c_safe_zero = 0;
		Se ((cruzou_zero >= 1) && (cruzou_zero < 3) então
		{
			zero = TRUE;
		}
		//se já foram contados três cruzamentos pela origem
		Se (cruzou_zero == 3) então
		{
			/*primeiro verifica se esse cruzamento não foi entre uma aquisição
			e outra*/
			Se (i_cent >= 75) então
			{
				//Senão for, inicializa vetor temp com 0
				inicializa_vetor(temp, 0, 1000);
				inicializa_vetor(zero_border_temp, 0, 1000);
			}
			Senão
			{
				//se for entre a as aquisições
				Enquanto (inteiro i = 0; i <= i_cent; i++) então faça
				{
					//salve o sinal restante em uma variável temporária
					temp[i] = sinal_seg[i];
				}
			}
			//limpe o final do sinal devido a verificação das 75 amostras
			//mais 50 amostras que compõem o próximo sinal
			Enquanto (inteiro i = 0; i <= 125; i++) então faça
			{
				//sinal segmentado recebe a média armazenada no sinal
```

```
                sinal_seg[qtd_elem_obtidos-i] = sinal_seg[999];
        }
        /*subtrai a quantidade de elementos obtidos da quantidade analisada
        a mais*/
         qtd_elem_obtidos = qtd_elem_obtidos – 125;
        //se for selecionado efetuar a FFT do sinal
        Se (FFT == TRUE) então
        {
                //faça FFT e salve esse sinal
                /*fft(vetor[],tamanho,qtd_parametros): esta função utiliza o
                algoritmo contido no LabVIEW para efetuar a FFT do sinal.
                Nessa função é informado o vetor que contém os valores que
                se deseja efetuar a FFT, a quantidade de elementos que esse
                vetor possui e quantos parâmetros da FFT se deseja
                retornar. A quantidade de elementos que o vetor possui é
                importante pois assim se efetua a FFT somente do sinal
                segmentado, sem o final do sinal que foi preenchido
                com a média*/
                sinal_seg_fft =
                        fft(sinal_seg,qtd_elem_obtidos,qtd_parametros);
                /*Adciona-se a FFT do sinal ao final do sinal segmentado*/
                sinal_seg = sinal_seg + sinal_seg_fft;
                /*salva_sinal(vetor[],qtd_sinais,qtd_elem_obidos): esta
                função salva o sinal segmentado em um arquivo para uso
                posterior. A variável qtd_sinais é utilizada para se dar o
                nome ao arquivo que indica o valor do sinal segmentado e a
                qtd_elem_obitidos indica quantos elementos foram obtidos da
                segmentação nesse sinal*/
                salva_sinal(sinal_seg, qtd_sinais, qtd_elem_obtidos);
        }
        Senão
        {
                //Senão
                salva_sinal(sinal_seg, qtd_sinais, qtd_elem_obtidos);
        }
        //incrementa a quantidade de sinais segmentados
        qtd_sinais++;
        //reinicia segmentado com a média
        inicializa_vetor(sinal_seg, media, 1000);
        //zera o contador de cruzamento pela origem
        cruzou_zero = 0;
        /*verifica se para segmentar o sinal
        Analisou-se mais de 100 amostras do sinal
        centralizado*/
        Se (i_cent >= 75) então
        {
                //reinicie a quantidade de elementos obtidos
                qtd_elem_obtidos = 0;
                /*subtrai o índice do sinal centralizado devido a
                verificação do zero*/
                i_cent = i_cent-1;
                //reinicie a contagem de cruzamento pela origem
                cruzou_zero = 0;
                //reinicia a flag de zero para FALSE
                zero = FALSE;
        }
        Senão
        {
                /*Se foram analisadas mais de 75 amostras, Reinicia a
                quantidade de elementos obtidos com o índice do sinal
                centralizado mais a quantidade de elementos que foram
                analisados na borda do sinal. A função
                tamanho_vetor(vetor[]) retorna a quantidade de elementos que
                esse vetor possui*/
                qtd_elem_obtidos = i_cent + tamanho_vetor(zero_border_temp)
                -1;
```

```
                                /*sinal segmentado recebe a parte do sinal que que foi
                                armazenada nas variáveis temporárias*/
                                sinal_seg = zero_border_temp + temp;
                                //como um zero já foi encontrado
                                cruzou_zero = 1;
                                //reinicia a flag de zero para TRUE
                                zero = TRUE;
                        }
                }
        }
}
//Zero encontrado
Se (zero == TRUE) então
{
        /*faça as seguintes verificações. Se ainda não foi obtido nenhum elemento*/
        Se (qtd_elem_obtidos == 0) então
        {
                //verifica que o índice do sinal centralizado é maior que 50
                Se ((i_cent-1) >= 50) então
                {
                        /*se for, retorne 50 elementos do sinal para obter
                        amostras antes do zero, devido a características do sinal*/
                        Enquanto (inteiro i = 0; i <= 50; i++) então faça
                        {
                                sinal_seg[qtd_elem_obtidos] = vetor_buffer[(i_cent - 51) +
i];
                                qtd_elem_obtidos++;
                        }
                }
                Senão
                {
                        /*Senão, retorna à quantidade de elementos que já foram
                        percorridos*/
                        Enquanto (inteiro i = 0; i <= i_cent; i++) então faça
                        {
                                sinal_seg[qtd_elem_obtidos] = vetor_buffer[i];
                                qtd_elem_obtidos++;
                        }
                }
        }
        Senão
        {
                /*Se já foram obtidas algumas amostras Verifique inicialmente se é
                necessário verificar se o zero encontrado é verdadeiro.*/
                Se (safe_zero == TRUE) então
                {
                        /*se for necessário verificar incremente o contador a cada entrada
                        nesta condição*/
                        c_safe_zero++;
                        /*se foram analisadas mais de 100 amostras*/
                        Se (c_safe_zero >= 75) então
                        {
                                //zero encontrado é verdadeiro
                                Safe_zero2 = TRUE;
                        }
                        /*verifica se o zero encontrado será verificado na borda do sinal*/
                        Se (((999-i_cent) < 75) && ~zero_border) então
                        {
                                //usa a função parte_vetor(vetor[],qtd,indice) que retorna
                                uma parte do vetor sendo seus parâmetros o vetor que se
                                deseja copiar uma parte (vetor[]), a quantidade de elementos
                                que se deseja obter (qtd) e em que índice iniciar a cópia
                                (índice). Assim, é obtido a parte do sinal que poderia ser
                                perdida mais 50 amostras que é a quantidade de amostras do
                                início do sinal.
```

```
                                zero_border_temp = parte_vetor(vetor_buffer,(i_cent-
                                                                        50),(999-(i_cent-50)));
                                zero_border = TRUE;
                        }

                }
                //salve a amostra no vetor de sinal segmentado
                sinal_seg[qtd_elem_obtidos] = vetor_buffer[i_cent - 1];
        }
}
```

Listagem 2 - Algoritmo de Cruzamento por Zeros

A partir dos dados obtidos no ensaio *In Vivo*, foram realizados testes para verificar se todas as funcionalidades da ferramenta desenvolvida estavam sendo corretamente executadas. As figuras Figura **12**, Figura **13**, Figura **14**, Figura **15** e Figura **16**, apresentam a segmentação das cinco atividades executadas pelo animal juntamente com o sinal original que foi segmentado e sua respectiva FFT, em que são apresentados os primeiros 30 elementos da FFT do sinal segmentado. O sinal centralizado, apresentado na Figura 11, utilizado para exemplificar o funcionamento do algoritmo, corresponde ao sinal segmentado que está apresentado na Figura 13.

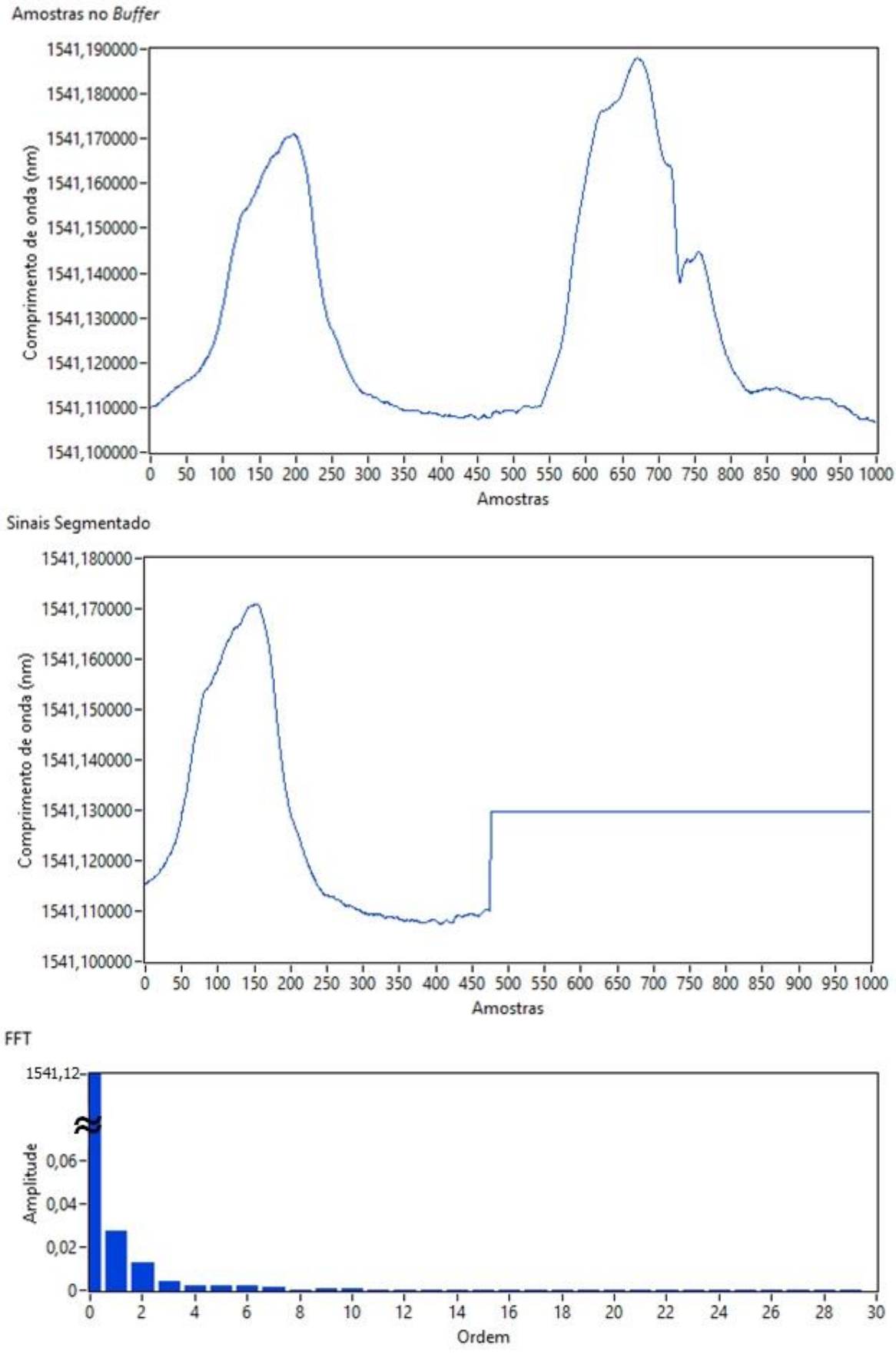

Figura 12 - Sinal obtido do animal se alimentando com Azevém, com o sinal segmentado e sua FFT utilizando o algoritmo de zeros.

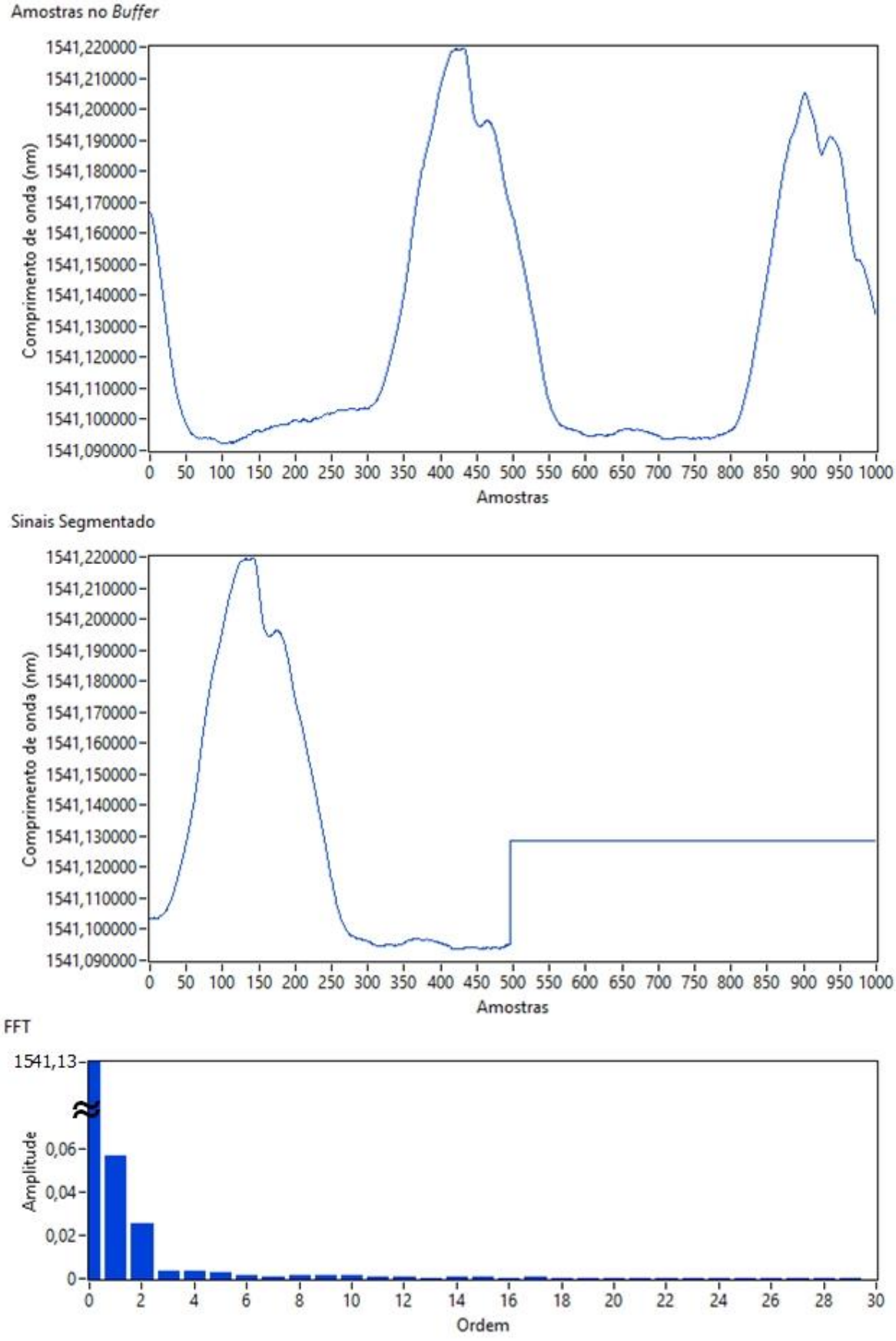

Figura 13 - Sinal obtido do animal se alimentando com Feno, com o sinal segmentado e sua FFT utilizando o algoritmo de zeros.

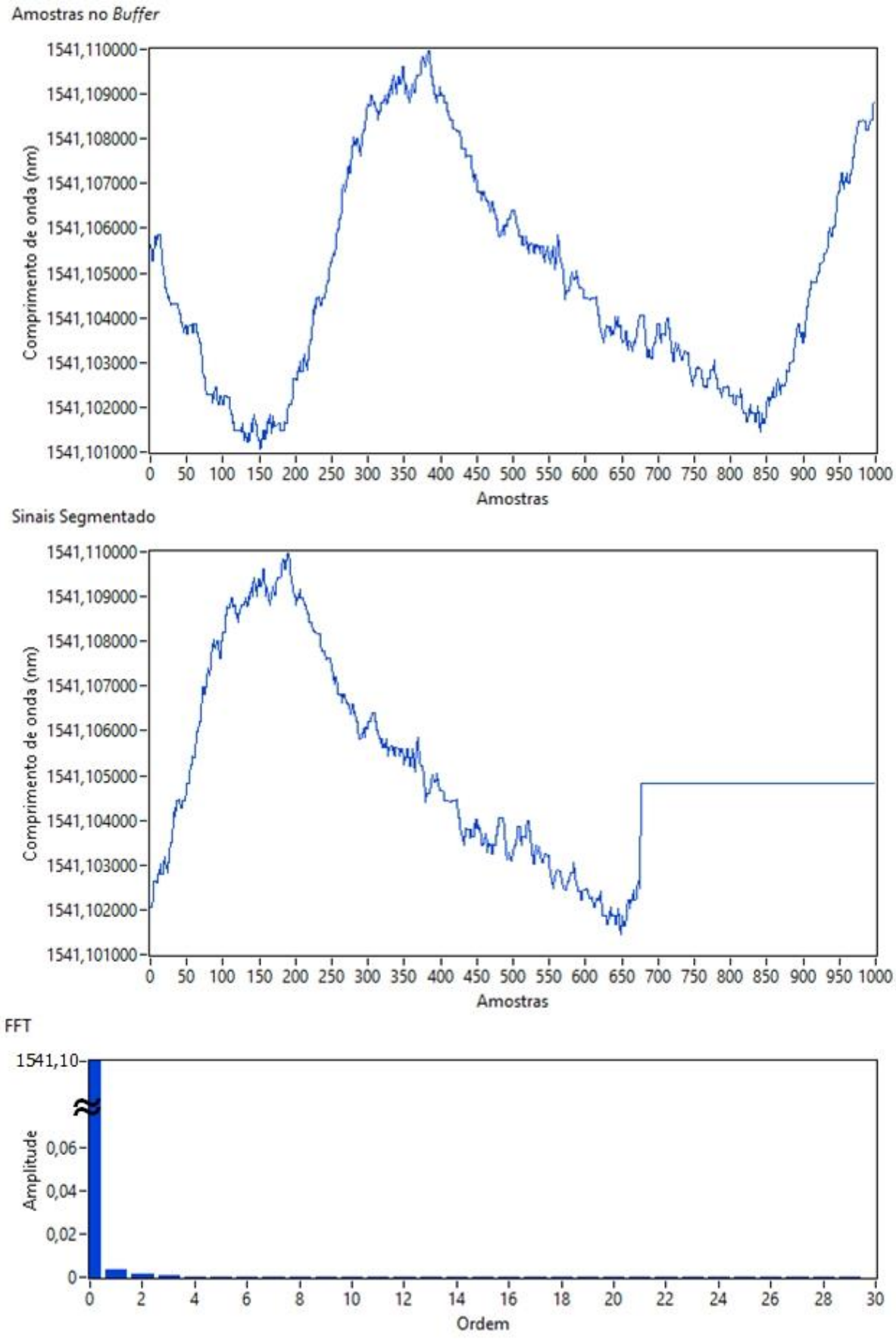

Figura 14 - Sinal obtido do animal com ausência de alimento em sua mandíbula, com o sinal segmentado e sua FFT utilizando o algoritmo de zeros.

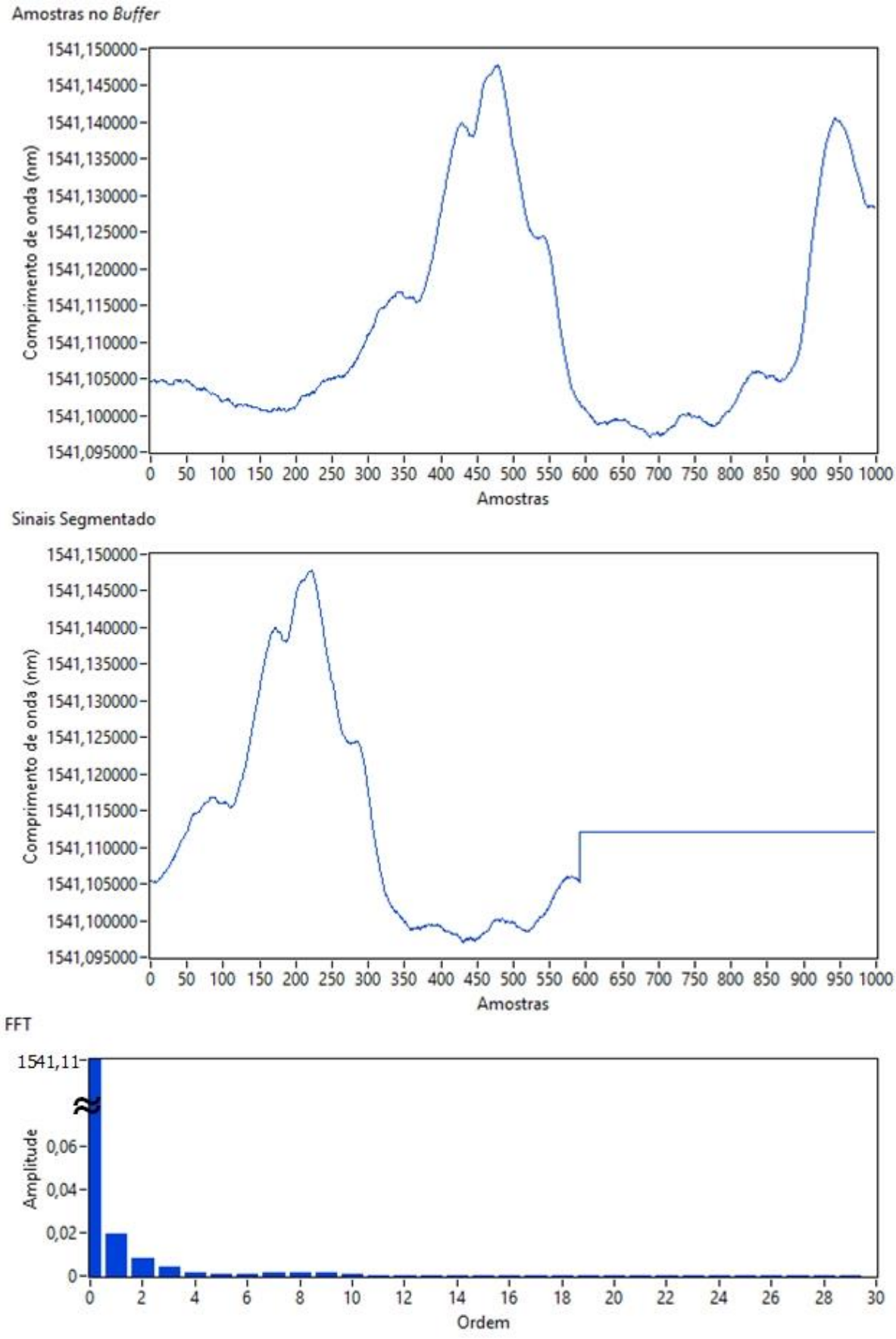

Figura 15 - Sinal obtido do animal se alimentando com Ração, com o sinal segmentado e sua FFT utilizando o algoritmo de zeros.

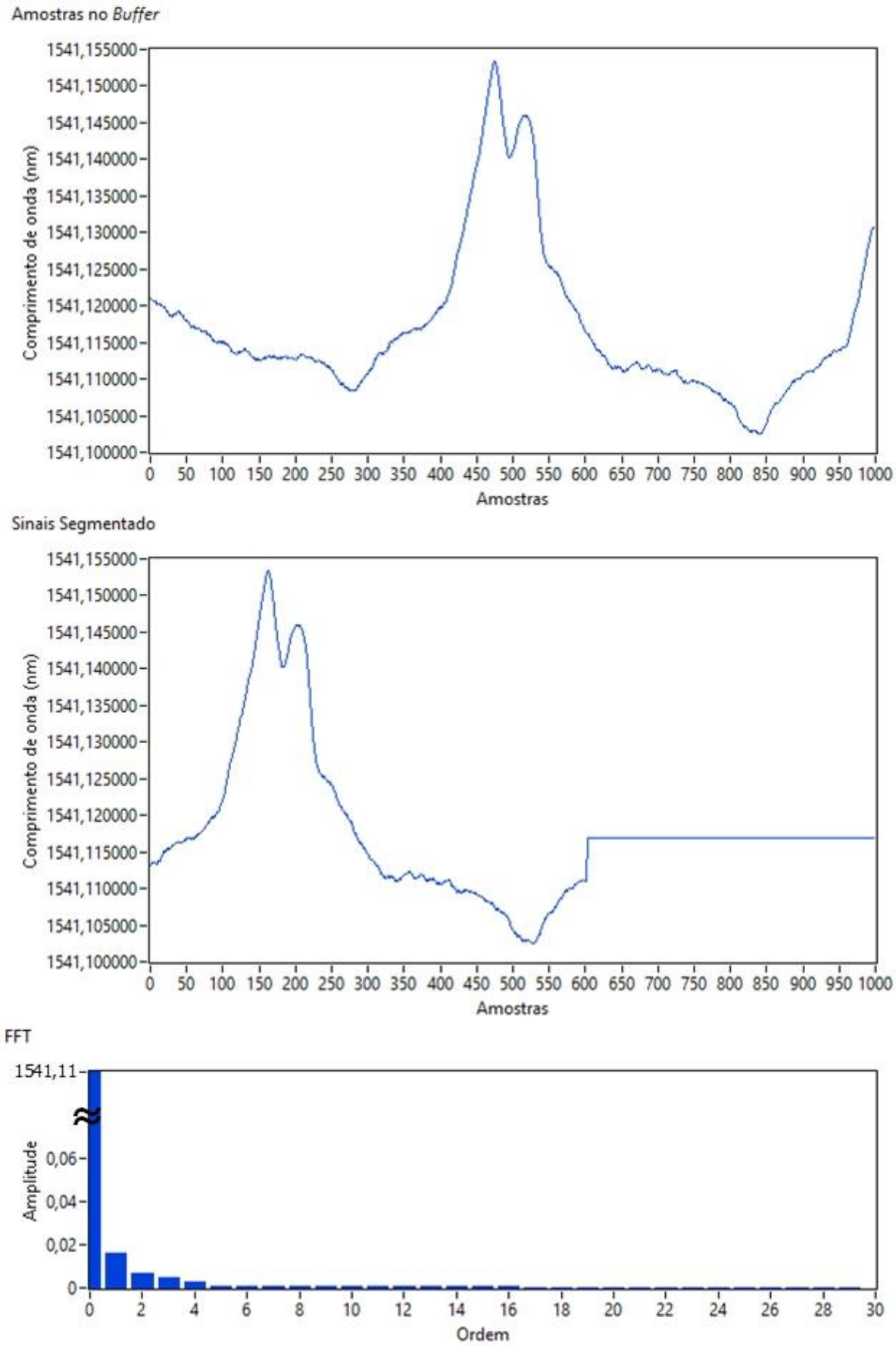

Figura 16 - Sinal obtido do animal ruminando, com o sinal segmentado e sua FFT utilizando o algoritmo de zeros.

Com a análise das figuras que apresentam o sinal pré-processado para cada alimento, é possível verificar que o algoritmo segmentou de forma correta.

4.1.2 Algoritmo de Mínimos

Com o sinal do *buffer* centralizado, o algoritmo de mínimos irá analisar cada amostra para se identificar as atividades que o animal esteja executando. Para exemplificar os pontos de mínimo de um sinal centralizado, a Figura 11 apresenta um sinal centralizado com seus pontos de mínimo destacados.

Como o algoritmo de mínimos procura pelo valor mínimo no sinal centralizado, quando um valor de mínimo é encontrado, são analisadas 150 amostras a mais para verificar se o mínimo encontrado não é um falso mínimo, ou seja, somente ruído. O valor de 150 amostras a serem analisadas, foi definido a partir de testes efetuados com os dados obtidos do ensaio *In Vivo*. Nestes testes, verificou-se qual seria a quantidade de amostras que devem ser analisadas para verificar se o mínimo encontrado é verdadeiro e não somente um ruído, ou se não foi encontrado um novo mínimo nesse intervalo de 150 amostras. Assim, se o mínimo for verdadeiro, o sinal segmentado recebe as 150 amostras analisadas e quando a análise do sinal encontrar o cruzamento por zero, o valor de mínimo é reiniciado para se encontrar o próximo mínimo, ou seja, é necessário encontrar dois mínimos para se identificar um evento de mastigação, um que indique o início do movimento e outro que indique o seu término. Os pontos de mínimo de um sinal podem ser visualizados na Figura 11, em que os dois pontos de mínimo encontrados indicam o término e o início de um evento de mastigação. Quando o próximo mínimo for encontrado, é efetuado também a análise se o mínimo é verdadeiro, e se for, o sinal é segmentado. Como é analisado 150 amostras a mais, essas 150 amostras são substituídas pela média do sinal segmentado. A partir disso, o algoritmo foi desenvolvido e a Listagem 3 apresenta o pseudocódigo do algoritmo de mínimos.

```
/*Essas variáveis não são declaradas necessariamente nesta parte. Foram declaradas nesta
parte para melhor entendimento do pseudocódigo.*/
Booleano atualiza_media = FALSE;//indica se deve ou não atualizar a média
Real media_anterior = 0;
Real valor_atual;//valor atual do sinal centralizado
Real prox_valor;//próximo valor do sinal centralizado
Booleano Aux_min = FALSE;//flag para verificar se o mínimo encontrado é "verdadeiro"
Real valor_minimo = 0;//valor mínimo encontrado
```

```
Booleano zera_minimo = FALSE;//flag usada para verificar se o mínimo teve seu valor
                             configurado em zero
inteiro c_aux_min = 0;//contador de amostras analisadas para verificar se um mínimo é
                      "verdadeiro
inteiro qtd_min = 0;//quantidade de mínimos encontrados

/*a função first_call verifica se é a primeira vez que entrou no laço que pré-processa o
sinal. Se for solicitado que atualize a média ou é a primeira vez que o laço é chamado*/
Se (atualiza_media == TRUE || first_call())então
{
       //atualize a média
       media_anterior = media;
       //e não permita que seja atualizada até que seja solicitado novamente
       atualiza_media = FALSE;
}
valor_atual = vetor_centralizado[i_cent];
i_cent++;//incrementa índice do vetor centralizado
prox_valor = vetor_centralizado[i_cent];
/*Verifica se o valor atual e o próximo valor possuem sinal diferente, e se o prox_valor
é menor que zero e o valor atual é positivo e se a flag de verificação de mínimo não está
ativa e se o índice do sinal centralizado não é igual a zero*/
Se ((ver_sinal(prox_valor) != ver_sinal(valor_atual)) &&
       (ver_sinal(prox_valor) <= 0) && (ver_sinal(valor_atual) >= 0)
       && (!Aux_min) && (i_cent != 0)) então
{
       //zere o valor do mínimo;
       valor_minimo = 0;
       //indica que o mínimo foi reiniciado em zero
       zera_minimo = TRUE;
}
/*Verifica se o próximo valor é menor que o atual e menor que o valor mínimo e o índice
do vetor centralizado é diferente de zero*/
Se ((prox_valor <= valor_atual) && (prox_valor <= valor_minimo) && (i_cent != 0)) então
{
       /*Então um mínimo foi encontrado. Ative a verificação de mínimo verdadeiro. O
       valor de mínimo recebe o próximo valor, zera-se o valor do contador da verificação
       de mínimo e reinicia o vetor temporário de verificação de mínimos*/
       Aux_min = TRUE;
       valor_minimo = prox_valor;
       c_aux_min = 0;
       inicializa_vetor(temp, 0, 150);
}
/*Se a verificação de mínimo for ativada*/
Se (Aux_min == TRUE) então
{
       //se o mínimo não for zerado
       Se (zera_minimo == FALSE) então
       {
              //então reinicia a flag de mínimo
              minimo = FALSE;
       }
       /*O vetor temp recebe os elementos do buffer enquanto se efetua uma contagem para
       verificar se em um intervalo de valores não encontra-se um novo mínimo*/
       temp[i] = vetor_buffer[i_cent - 1];
       c_aux_min++;
}
/*Senão for encontrado nenhum mínimo nas próximas 150 amostras*/
Se (c_aux_min >= 150) então
{
       /*Então o mínimo é verdadeiro. Zera-se o contador de verificação de mínimo e
       incrementa a quantidade de mínimos encontrados*/
       c_aux_min = 0;
       qtd_min++;
       zera_minimo = aux_min = FALSE;
       //Se foram encontrados dois mínimos
       Se (qtd_min == 2) então
```

```
	{
		/*Como foram analisados 150 elementos a mais, desconte esses elementos e
		reescreva o vetor de sinal segmentado nesses índices com a média*/
		enquanto (inteiro i = 0; i < 150; i++) então faça
		{
			/*se o sinal do buffer foi atualizado enquanto se procurava o
			próximo mínimo*/
			Se (sinal_novo == TRUE) então
			{
				//então utilize a média anterior
				sinal_seg[(qtd_elem_obtidos - 150) + i] = media_anterior;
			}
			Senão
			{
				//Senão utilize a média atual
				sinal_seg[(qtd_elem_obtidos - 150) + i] = media;
			}
		}
		//se for selecionado efetuar a FFT do sinal
		Se (FFT == TRUE) então
		{
			//faça a FFT e salve esse sinal
			sinal_seg_fft = fft(sinal_seg,qtd_elem_obitidos,qtd_parametros);
			//Adicione ao final do sinal segmentado a FFT desse sinal
			sinal_seg = sinal_seg + sinal_seg_fft;
			//Salve esse sinal
			salva_sinal(sinal_seg, qtd_sinais, qtd_elem_obtidos);
		}
		Senão
		{
			salva_sinal(sinal_seg, qtd_sinais, qtd_elem_obtidos);
		}
		//incrementa a quantidade de sinais segmentados
		qtd_sinais++;
		//reinicia o sinal segmentado com a media
		inicializa_vetor(sinal_seg, media, 1000);
		/*reinicia a quantidade de elementos obtidos e a quantidade de mínimos*/
		qtd_elem_obtidos = qtd_min = 0;
		//permite a atualização da média
		atualiza_media = TRUE;
		//reinicia a flag de mínimo
		minimo = FALSE;
		//reinicia valor do mínimo
		valor_minimo = 0;
		//se o índice do sinal centralizado for maior que 150
		Se (i_cent >= 150) então
		{
			/*subtraia as 150 amostras a mais analisadas para verificar que o
			mínimo encontrado não era "falso"*/
			i_cent = i_cent - 150;
		}
		Senão
		{
			//Senão  volta para a primeira amostra
			i_cent = 0;
			//se foi inserido algum elemento no vetor temporário
			Se (tamanho_vetor(temp) != 0) então
			{
				/*então um mínimo já foi encontrado*/
				qtd_min = 1;
				minimo = TRUE;
				//sinal segmentado recebe o vetor temp
				sinal_seg = temp;
				/*se a quantidade de amostras em temp for maior ou igual que
				150*/
				Se (tamanho_vetor(temp) >= 150) então
```

```
                                {
                                        qtd_elem_obtidos = 150;
                                }
                                Senão
                                {
                                        qtd_elem_obtidos = tamanho_vetor(temp);
                                }
                                //reinicia vetor temp
                                inicializa_vetor(temp, 0, 150);
                        }
                }
        }
        Senão
        {
                //um mínimo foi encontrado
                minimo == TRUE;
                /*sinal segmentado recebe as amostras analisadas para verificar se não
                continha mais nenhum mínimo nessas amostras para evitar segmentação de
                ruído*/
                sinal_seg = temp;
                Se (tamanho_vetor(temp) >= 150) então
                {
                        qtd_elem_obitdos = 150;
                }
                Senão
                {
                        qtd_elem_obtidos = tamanho_vetor(temp);
                }
                inicializa_vetor(temp, 150, 0);
        }
}
//se um mínimo foi encontrado
Se (minimo == TRUE) então
{
        //armazene as amostras do buffer no sinal segmentado
        sinal_seg[qtd_elem_obtidos] = vetor_buffer[i_cent-1];
}
```

Listagem 3 - Algoritmo de mínimos.

Tendo-se as amostras do ensaio *In Vivo* executado, essas amostras foram submetidas a este algoritmo. Desta forma, as figuras Figura 17, 18, 19, 20 e Figura 21, apresentam o sinal presente no buffer, o sinal segmentado e a FFT do sinal segmentado, em que são apresentados os 30 primeiros elementos da FFT desse sinal. O sinal utilizado como exemplo apresentado na Figura 11, foi processado por este algoritmo e o resultado do seu processamento está apresentado na Figura 18.

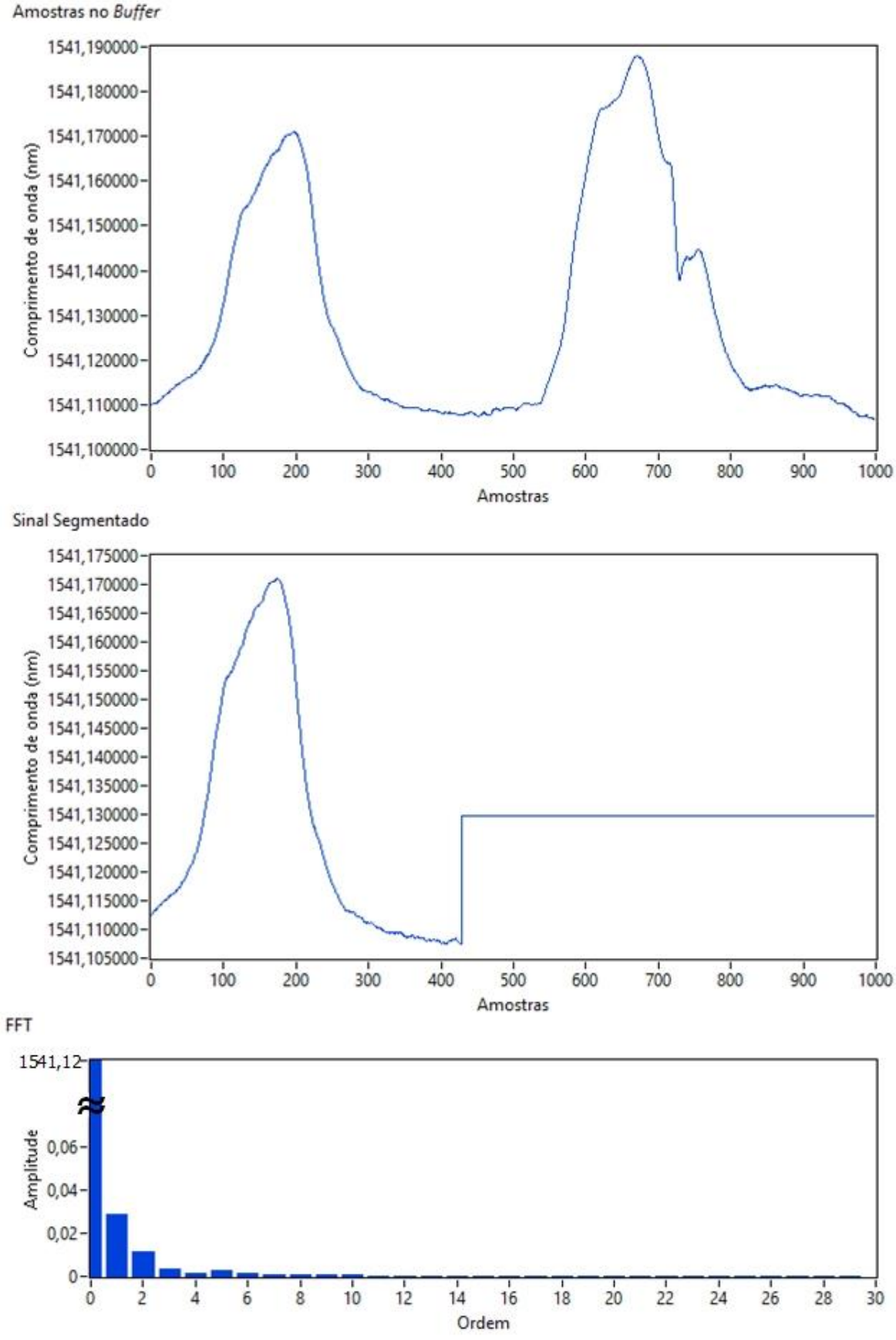

Figura 17 - Sinal obtido do animal se alimentando com azevém, com o sinal segmentado e sua FFT utilizando o algoritmo de mínimos.

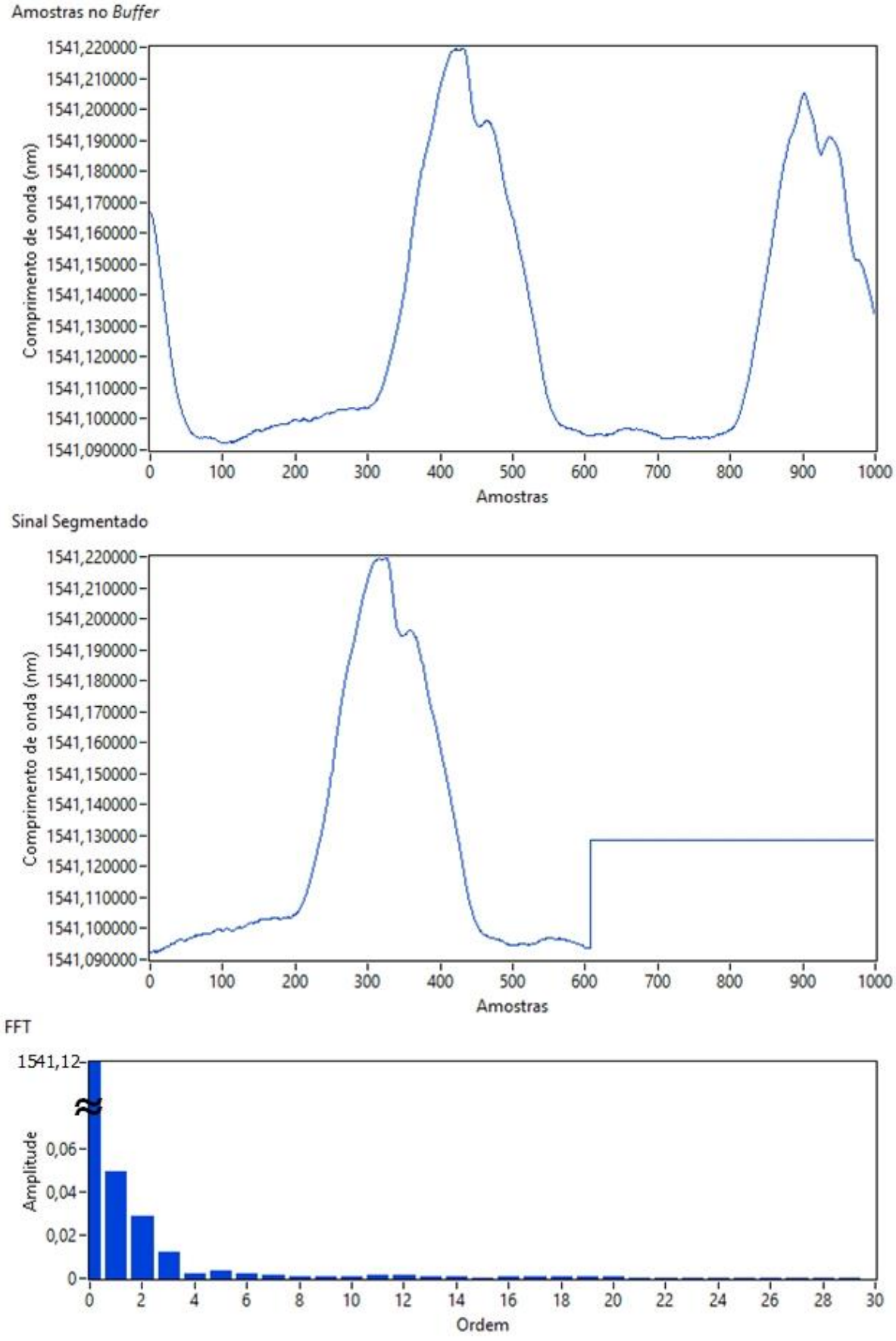

Figura 18 - Sinal obtido do animal se alimentando com feno, com o sinal segmentado e sua FFT utilizando o algoritmo de mínimos.

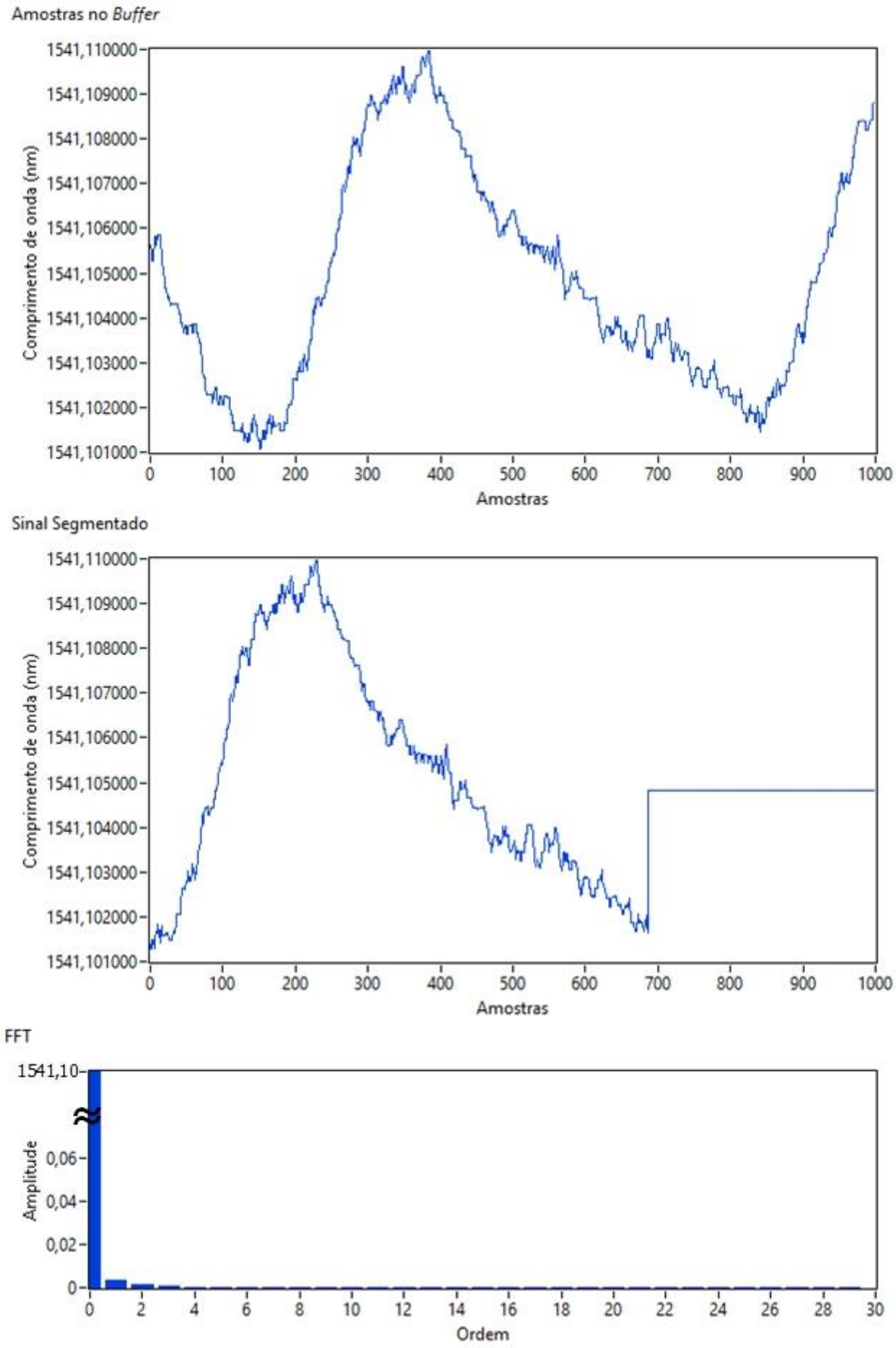

Figura 19 - Sinal obtido do animal com ausência de alimento em sua mandíbula, com o sinal segmentado e sua FFT utilizando o algoritmo de mínimos.

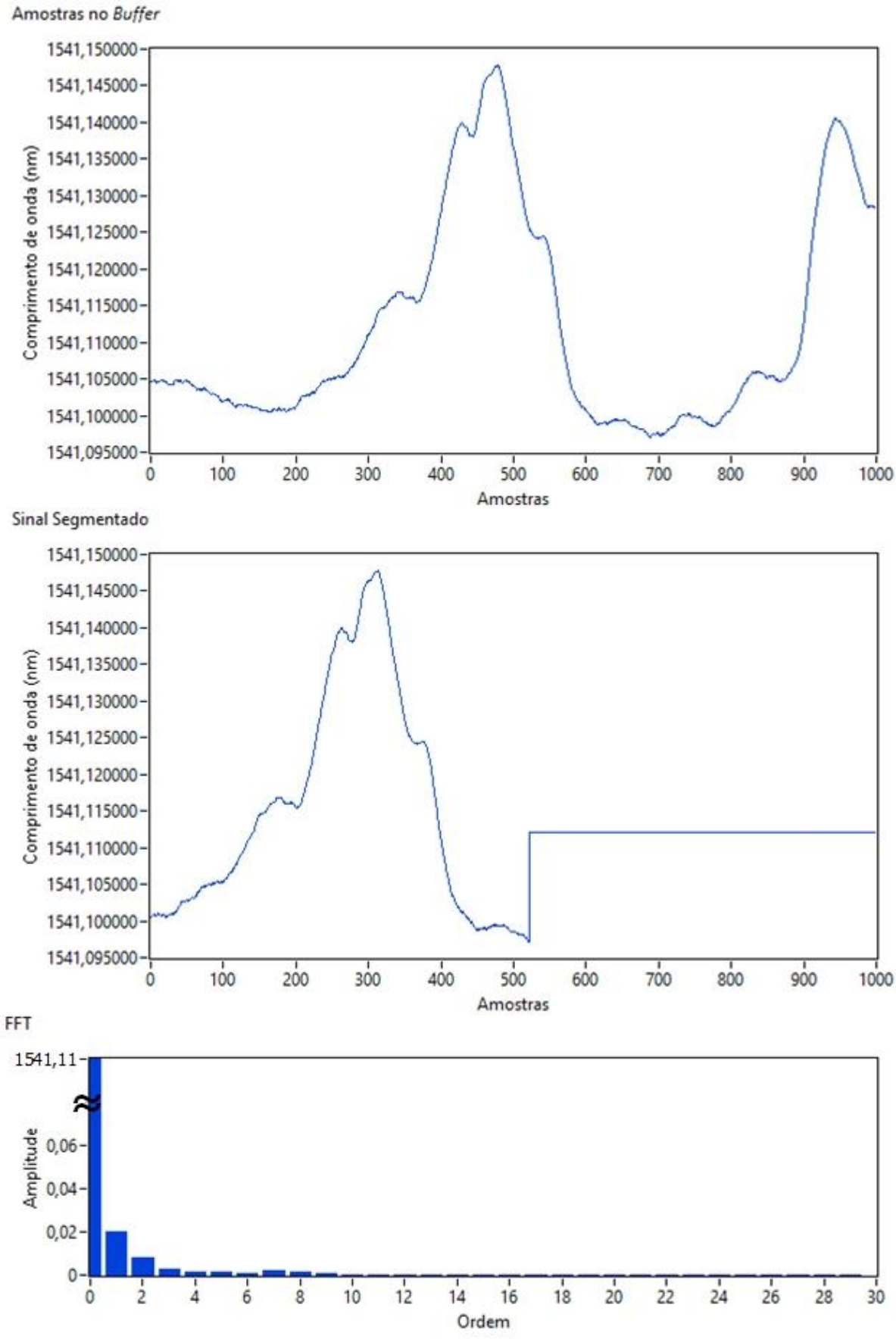

Figura 20 - Sinal obtido do animal se alimentando com ração, com o sinal segmentado e sua FFT utilizando o algoritmo de mínimos.

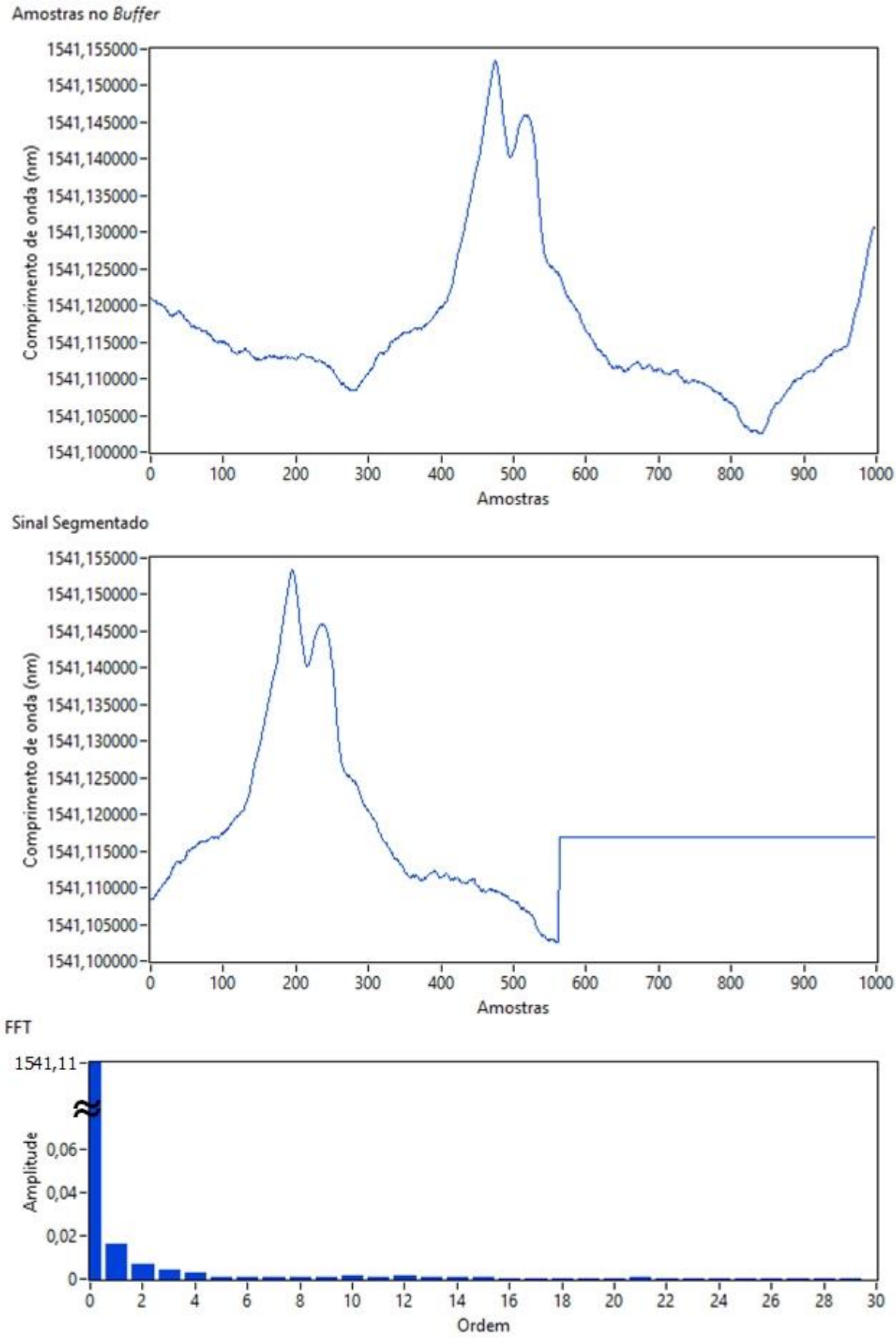

Figura 21 - Sinal obtido do animal ruminando, com o sinal segmentado e sua FFT utilizando o algoritmo de mínimos.

Analisando-se as figuras que apresentam o sinal segmentado, verifica-se que o algoritmo de mínimos efetuou o pré-processamento da forma desejada. Nas figuras é possível observar que foram preservadas as características de cada movimento de mastigação.

4.2 SEGMENTAÇÃO EM TEMPO REAL

Após o desenvolvimento dos algoritmos, os mesmos foram integrados ao sistema de aquisição do interrogador óptico. Com isso, o sistema de aquisição foi modificado para receber as configurações do pré-processador. O usuário poderá selecionar o sensor que deseja monitorar, o algoritmo de segmentação a ser utilizado, o diretório onde os arquivos serão salvos, se deseja utilizar a FFT e quantos elementos da FFT deseja utilizar e o modo de operação. Para se armazenar os dados obtidos, após a segmentação dos dados, o vetor com a segmentação do sinal, complementado com a média, será salvo em um arquivo do tipo “arff” para ser utilizado em softwares de aprendizado de máquina. Caso o usuário tenha selecionado efetuar a FFT do sinal, ao final do vetor de dados, os valores da FFT são adicionados, conforme a quantidade de componentes da FFT que o usuário tenha selecionado. Outra opção que o usuário possui é se deseja efetuar a coleta para treinar algum algoritmo de aprendizagem de máquina, em que essa função é utilizada para a partir de um valor base informar se o animal está se alimentando ou não. Ou se deseja somente adquirir os dados e após a aquisição e o pré-processamento reconhecer o sinal pré-processado. Além dessas configurações, o usuário tem disponível as configurações do interrogador para identificar os sensores ópticos. A Figura 22 apresenta a interface de configuração.

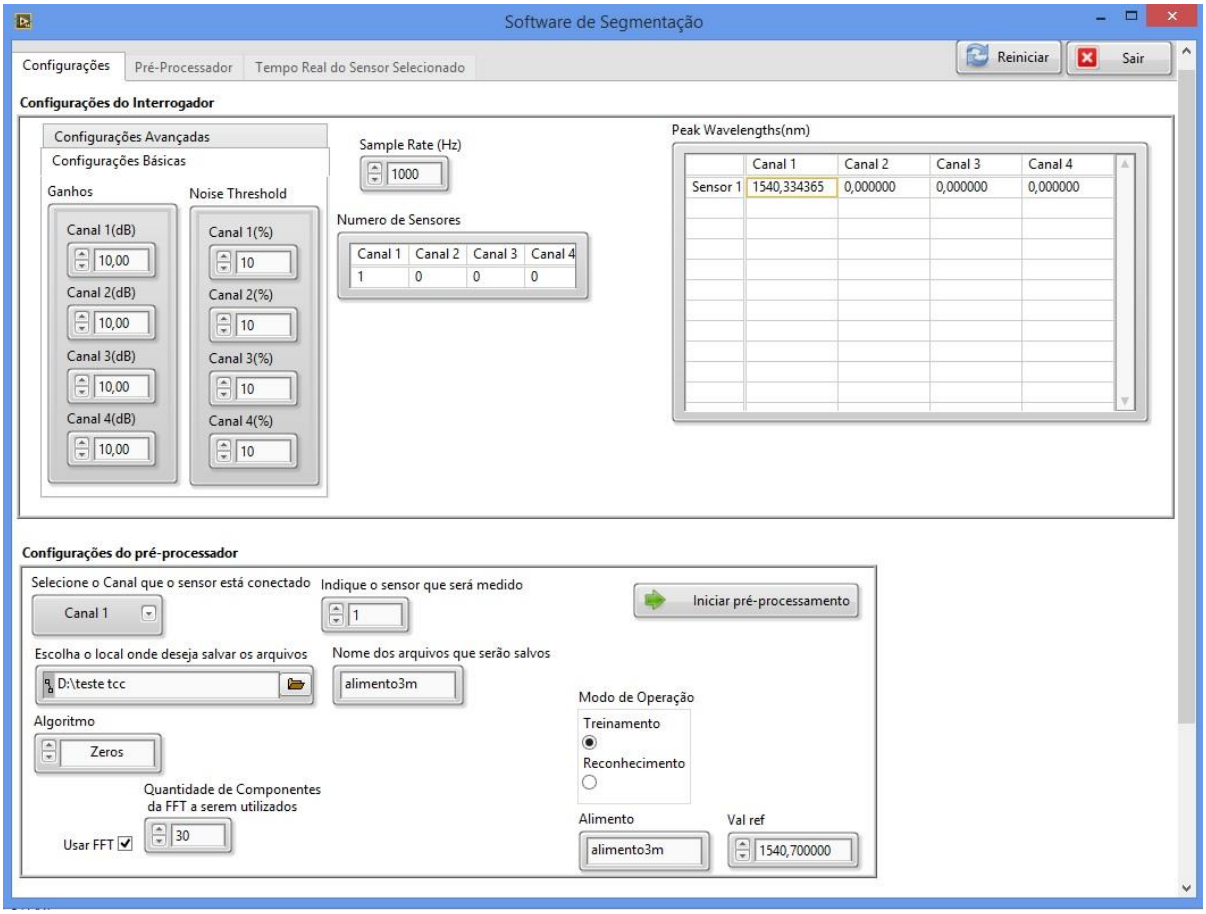

Figura 22 - Tela de configuração do sistema de aquisição e do pré-processador.

Estando configurado o interrogador e o sistema de pré-processamento, no momento que o usuário clicar no botão “Iniciar pré-processamento”, o mesmo será redirecionado para a aba do “Pré-Processador” que contém os gráficos que apresentam o sinal no *buffer*, o sinal centralizado, o sinal segmentado e o sinal armazenado. A partir do momento que o usuário iniciar o pré-processamento, as amostras obtidas do interrogador são armazenadas em um fila. Os dados começam a ser processados quando o usuário clicar no botão “Iniciar”, que está localizado na aba do “Pré-Processador”. Quando iniciado a segmentação, o *software* retira as amostras que foram inseridas na fila. Além disso, apresenta quantos elementos foram obtidos no sinal que foi segmentado e quantos sinais já foram segmentados. Desta forma, a Figura 23 apresenta a interface do pré-processador.

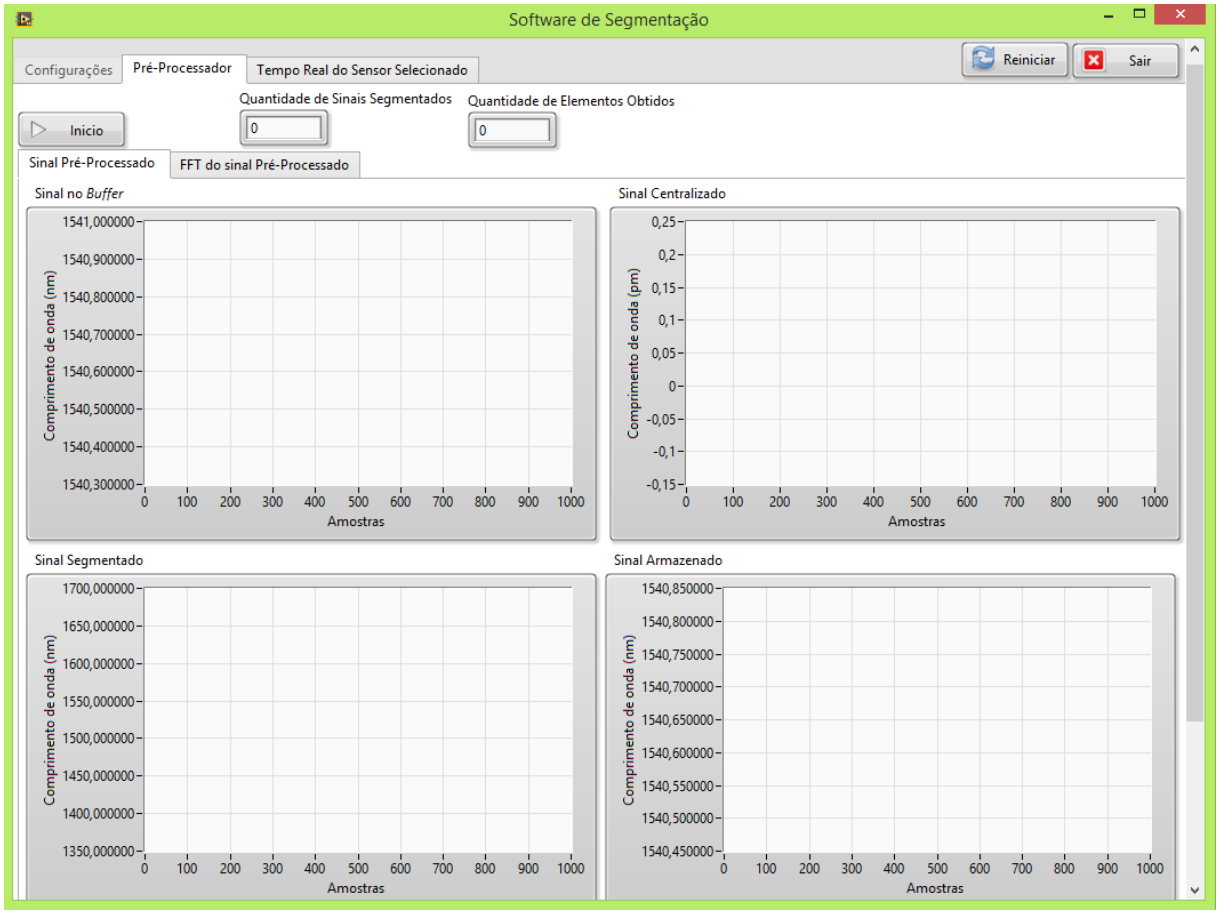

Figura 23 - Interface do pré-processador.

A Figura 24 mostra a aba que apresenta a FFT do sinal. Essa aba contém dois gráficos da FFT, um com a quantidade de elementos selecionados pelo usuário da FFT e outro gráfico que apresenta a mesma FFT, mas sem o primeiro elemento, pois esse elemento contém o valor médio do sinal, dificultando a visualização do gráfico. Assim, a Figura 24 apresenta o conteúdo dessa aba descrita.

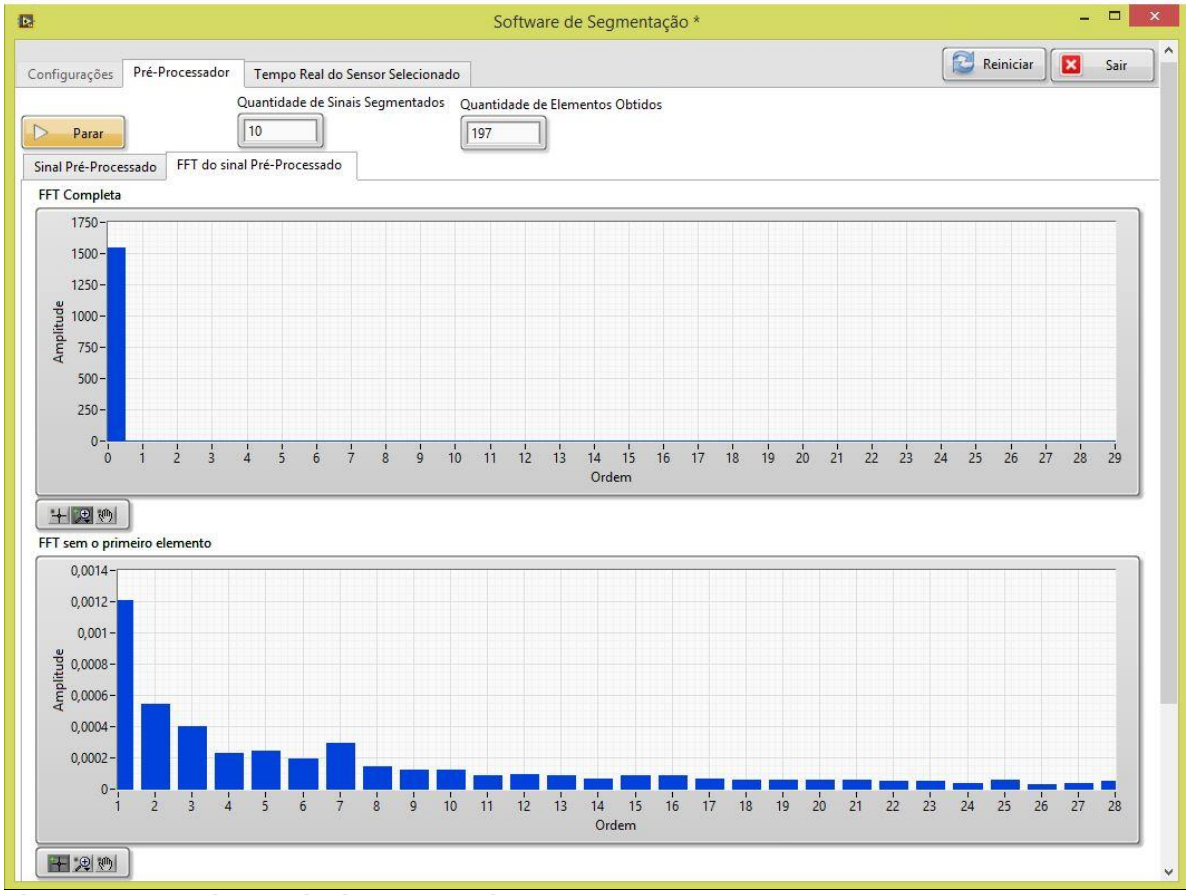

Figura 24 - FFT de um sinal segmentado.

Por fim, ainda é possível durante a aquisição e pré-processamento do sinal, o usuário visualizar em tempo real o sinal recebido da FBG. Esse sinal pode ser visualizado na aba “Tempo Real do Sensor Selecionado”. A Figura 25 apresenta o gráfico que o usuário pode visualizar em tempo real do sensor selecionado.

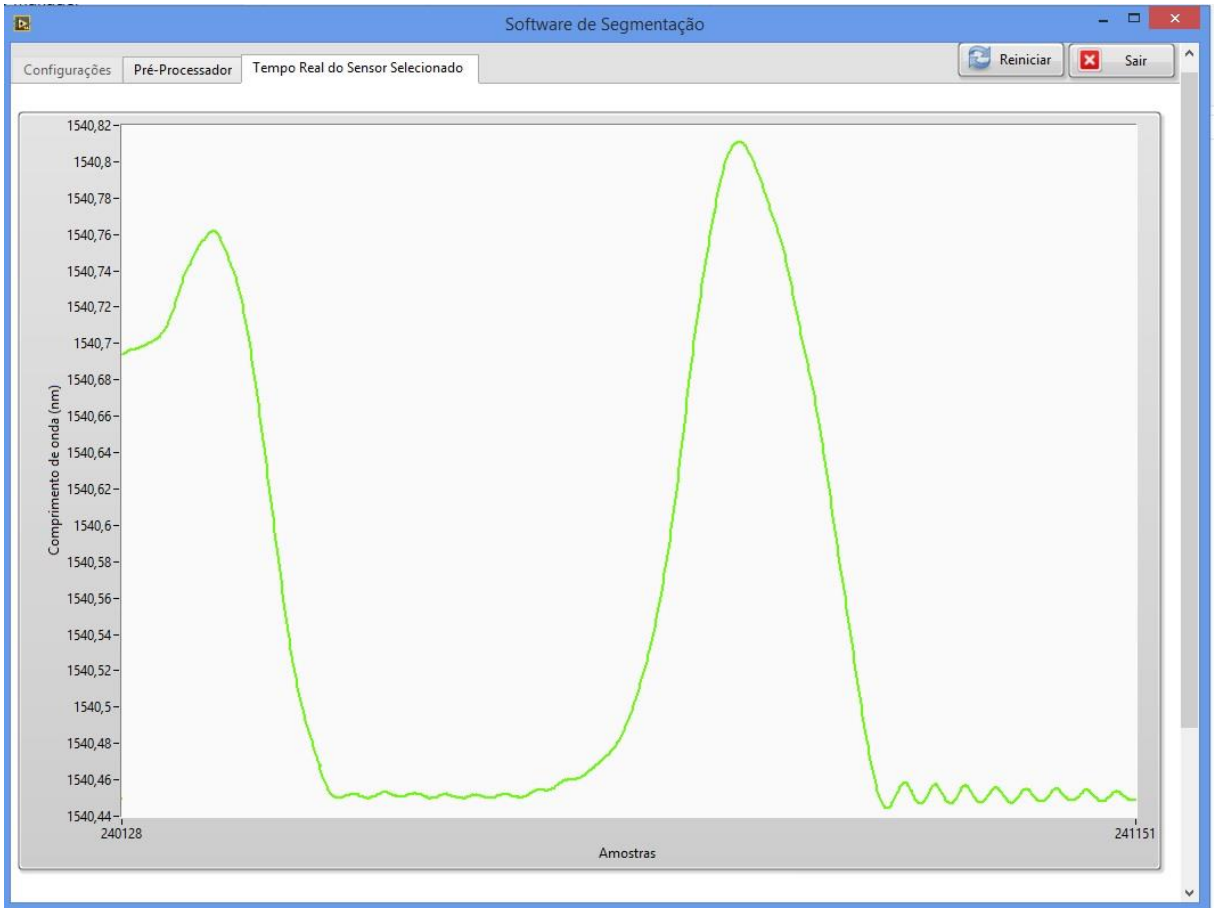

Figura 25 - Visualização em tempo real do sensor selecionado para adquirir e segmentar o sinal.

Para verificar o funcionamento dos algoritmos em conjunto com o sistema de aquisição, utilizou-se uma FBG que foi colada em um retângulo de acrílico. Esse sensor está apresentado na Figura 26. Conforme se aplica uma força nesse sensor, procurou-se reproduzir um sinal semelhante a um evento de mastigação. A partir disso, foi possível efetuar uma simulação do sistema adquirindo e pré-processando o sinal em tempo real. A Figura 27 apresenta o sensor conectado ao interrogador e um computador pessoal recebendo as amostras adquiridas e pré-processando essas amostras.

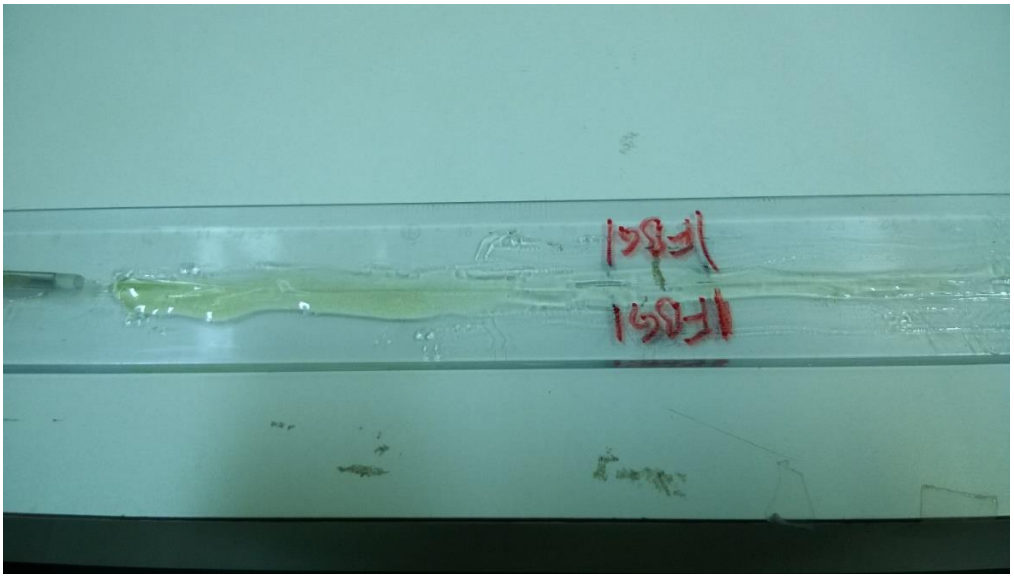

Figura 26 - FBG utilizada para simular um evento de mastigação.

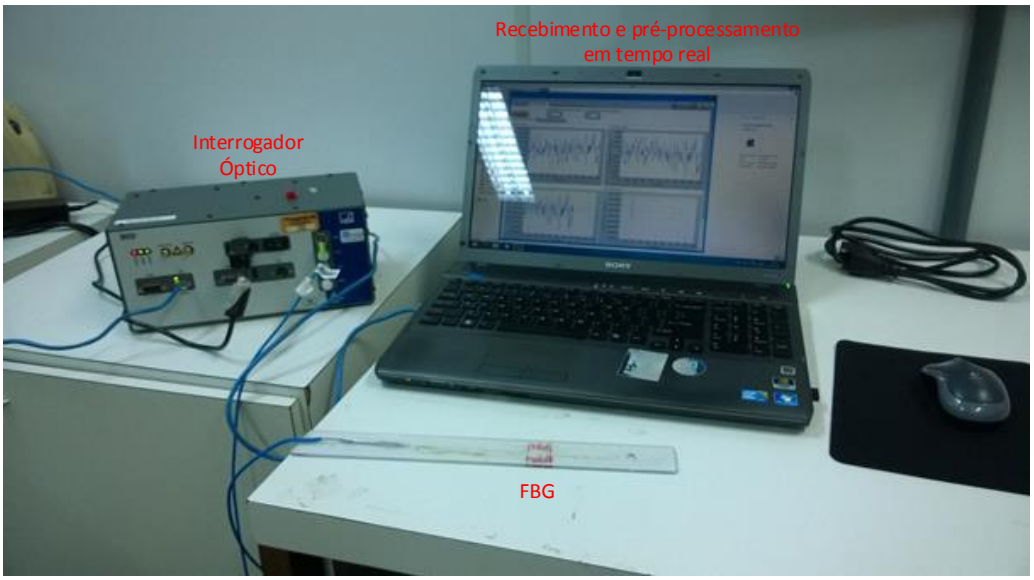

Figura 27 - Sistema em funcionamento.

Tendo-se um sensor para criar picos no sinal, simulando o comportamento semelhante a um evento de mastigação, o *software* foi configurado para utilizar um dos dois algoritmos para segmentar, em tempo real, o sinal proveniente do sensor. Assim, a Figura 28 apresenta o *software* em funcionamento executando o algoritmo de zeros e a Figura 29 apresenta o *software* executando o algoritmo de mínimos.

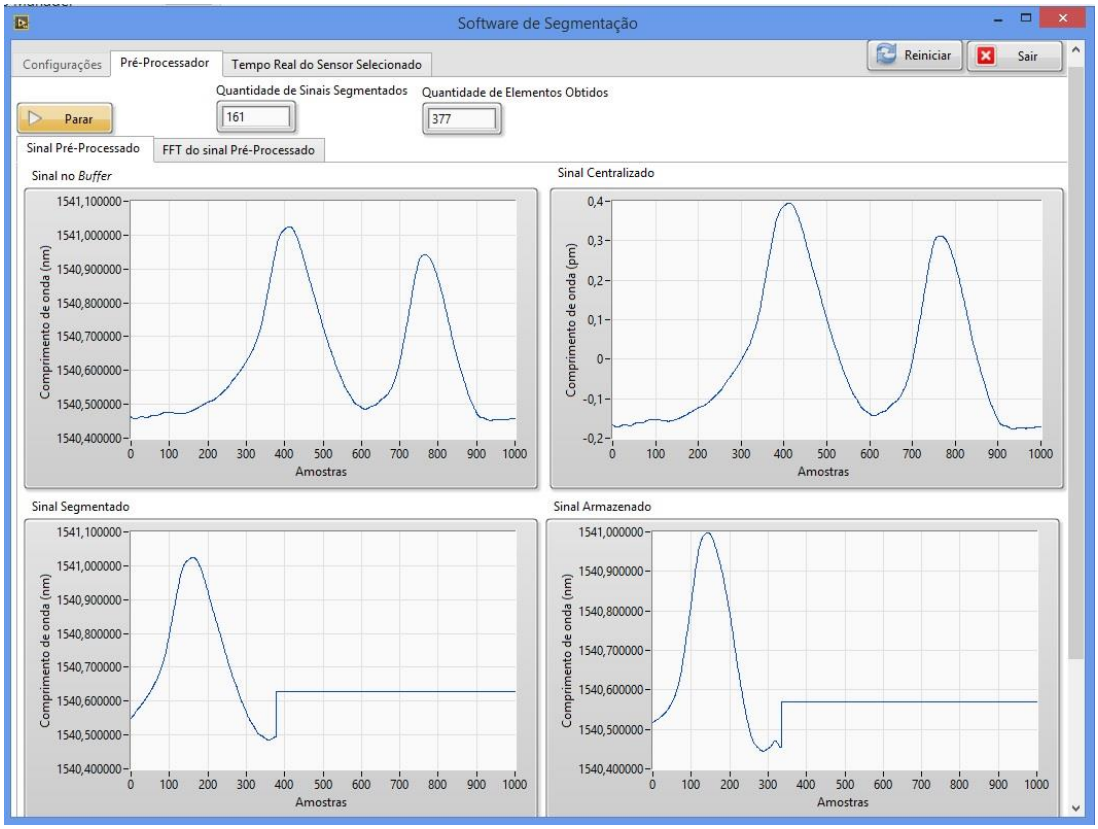

Figura 28 - Sistema executando a segmentação em tempo real de um sinal com algoritmo de cruzamento por zeros.

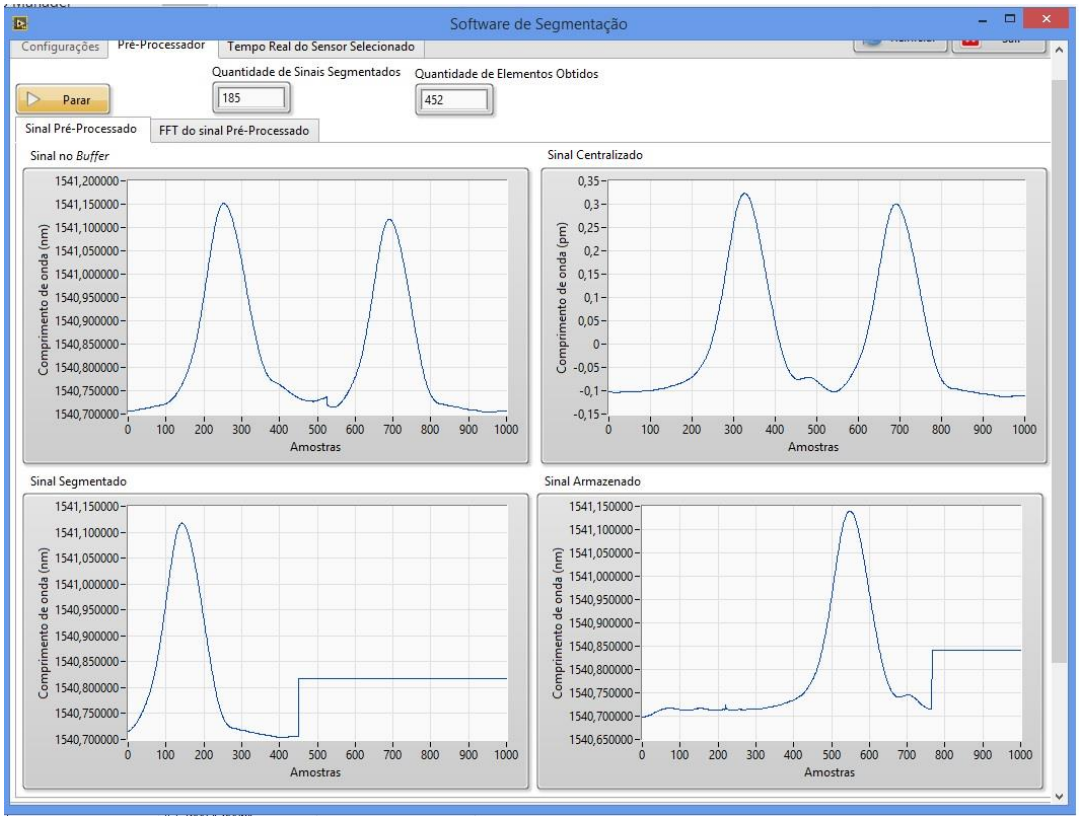

Figura 29 - Sistema executando em tempo real a segmentação de um sinal utilizando o algoritmo de mínimos.

Com estes resultados, foi possível verificar que o pré-processador executou a segmentação do sinal obtido do sensor óptico de forma correta com os dois algoritmos, em que cada algoritmo deixa o seu sinal característico. O algoritmo de mínimos normalmente adquire mais amostras, pois o ponto de mínimo normalmente está mais distante em número de amostras que o cruzamento por zero. Já o algoritmo de cruzamento por zeros deixa mais expressivo a elevação que ocorreu no sinal, ou seja, o valor de pico do sinal gerado pelo evento de mastigação.

Além de segmentar o conjunto de amostras obtidos, o sistema permite a geração de arquivos ".arff" para que as amostras segmentadas sejam utilizadas por outros *software*s, que efetuam a classificação do sinal segmentado.

Outra função que o sistema de pré-processamento possui, é a possibilidade de rotular os dados que foram segmentados, em que o usuário poderá selecionar um comprimento de onda base. Esse valor escolhido será utilizado para indicar quando o evento desejado ocorreu ou se o animal não estava executando nenhuma atividade. Por exemplo, considerando que o valor de um sensor sem atividade seja 1530 nm, se o usuário selecionar o valor de 1530,55 nm para efetuar a rotulação, quando o sinal for segmentado e seu valor máximo for maior que 1530,55 nm será rotulado com o nome que o usuário informou, caso contrário será rotulado como "Parado", ou seja, o animal não estava executando nenhuma atividade. No sistema, essa opção foi nomeada como "Treinamento", pois a rotulação é essencial para se efetuar o treinamento de um algoritmo de reconhecimento de padrões. Essa opção pode ser verificada na interface do sistema que está apresentada na Figura 22.

Caso o usuário já tenha treinado o seu algoritmo de aprendizagem de máquina através dos dados rotulados, ele poderá utilizar a opção "Reconhecimento", na qual é realizado o processo de segmentação dos dados e cálculo da FFT, porém os dados não são rotulados de acordo com o tipo de alimento ou atividade que o animal está executando. A tarefa de reconhecer o evento ou alimento que o animal está ingerindo deve ser realizada por um *software* que efetue a classificação de padrões. Para efetuar o processo de treinamento e reconhecimento de padrões, o usuário deverá utilizar outro *software* para efetuar este procedimento, pois o sistema de pré-processamento não executa essas ações, somente deixa os dados prontos para serem analisados.

5 CONCLUSÃO

O aumento do uso da pecuária de precisão exige sistemas que forneçam respostas precisas. Existem várias técnicas para se efetuar o monitoramento de ruminantes. Neste trabalho foi dada ênfase no uso de sensores ópticos para se efetuar o monitoramento ingestivo de ruminantes.

Verificou-se que monitorar a ingestão de alimentos pelo animal é gerada uma quantidade de informações da ordem de 1000 amostras por segundo. Como essas informações são obtidas em tempo real, caso não haja uma segmentação do sinal obtido, é difícil reconhecer os padrões de cada atividade realizada pelo animal.

Diante deste problema, foram propostos dois algoritmos para segmentar o sinal proveniente do sensor. No desenvolvimento desses algoritmos, utilizou-se a linguagem G, com IDE LabVIEW™, facilitando a integração entre o algoritmo desenvolvido e o sistema de aquisição de dados do interrogador óptico. O interrogador possui *drivers* para se efetuar a conexão ao sistema e aquisição dos dados. Esses *drivers* são disponibilizados pelo fabricante do interrogador, e foram desenvolvidos em linguagem G utilizando o LabVIEW™.

Uma das principais vantagens do método utilizado é que, mesmo possuindo variações no sinal, que podem dificultar a identificação do início ou final de um evento de mastigação, a resposta do sensor é precisa, com sensibilidade de 1μ strain por 1,2 pm de variação do sensor e imune a ruídos causados por campos eletromagnéticos, facilitando a identificação de padrões e a segmentação do mesmo.

A maior dificuldade encontrada foi efetuar o tratamento do sinal quando o mesmo cruza várias vezes pela origem, ou quando o sinal possuiu vários pontos de mínimo local após a centralização. Nessa etapa, foi necessário efetuar vários testes para verificar qual valor de amostras seria mais recomendado para que não sejam contabilizados vários cruzamentos por zero ou vários mínimos, segmentando-se assim somente ruído, ou segmentando o sinal no local errado, inserindo assim, dados imprecisos.

Como o algoritmo desenvolvido neste trabalho tem a função de segmentar sinais obtidos de um sensor óptico aplicado na monitoração de eventos de mastigação de bovinos, o mesmo algoritmo pode ser utilizado para segmentar outros tipos de sinais como. Um exemplo de sinais que podem ser segmentadores seria sinais de batimentos cardíacos. Para utilizar outros sinais, é necessário primeiramente

identificar os padrões do sinal, para assim configurar a centralização, caso seja necessário e a característica do sinal, pois sinais biomédicos possuem características especificas, ou seja, seria necessário ter conhecimento de quantos pontos de mínimo normalmente um sinal desse possui ou quantas vezes o mesmo cruza por zero, para assim se efetuar a sua segmentação, além de fazer o ajuste para rejeitar valores ruidosos.

5.1 TRABALHOS FUTUROS

Diante dos resultados positivos que foram obtidos dos testes do algoritmo com dados de um ensaio *In Vivo* e de testes com pré-processamento em tempo real, em trabalhos futuros, o sistema de aquisição e o pré-processador podem ser utilizados em conjunto com um sistema para reconhecer, em tempo real, a atividade que o animal esteja efetuando naquele instante.

Além disso, podem ser efetuadas melhorias nos dois algoritmos propostos. No algoritmo de cruzamento por zero, ao invés de no ponto de zero se subtrair um valor fixo de amostras, em que neste trabalho foi definido como 50, pode-se analisar a derivada do sinal para verificar onde se tem uma maior variação da derivada, iniciando a segmentação do sinal neste ponto. Já para o algoritmo de mínimo, a melhoria fica por parte de analisar mínimos locais, que podem ser classificados como ruído, e com a análise desses mínimos locais, pode-se ter um conjunto de dados mais preciso.

REFERÊNCIAS

AUDACITY. **Audacity is free, open source, cross-platform software for recording and editing sounds.** Disponível em: <http://audacityteam.org/>. Acesso em 27 jun. 2015.

CARVALHO, Paulo C. de F.; TRINDADE, Julio K. da.; MEZZALIRA, Jean C.; POLI, Cesar H. E. C.; NABINGER, Carlos;. GENRO, Teresa C. M.; GONDA, Horacio L. Do bocado ao pastoreio de precisão: Compreendendo a interface planta-animal para explorar a multi-funcionalidade das pastagens. 46ª Reunião Anual da Sociedade Brasileira de Zootecnia, 2009, Maringá. **Anais...** Maringá, 2009. 1 CD-ROM.

CHASSING, Rulph; REAY, Donald. **Digital Signal Processing and Applications with the TMS320C6713 and TMS320C6416 DSK.** 1. ed. Wiley-IEEE Press: 2008. 608 p.

CLAPHAM, William M.; FEDDERS, James M.; BEEMAN, Kim;. NEEL, James P.S. Acoustic monitoring system to quantify ingestive behavior of free-grazing cattle. **Computers and Electronics in Agriculture.** v. 76, p. 96-104, 2011.

DAGHLIAN, Jacob. **Lógica e álgebra de Boole.** 4. ed. São Paulo, SP: Atlas, c1995. 167 p.

DUHAMEL, Pierre; VETTERLI, Martin. Fast fourier transforms: A tutorial review and state of the art. **Signal Processing.** v. 19, n. 4, p. 259-299, 1990.

ENGINEERING DESIGN. **Software for sound analysis.** Disponível em: <http://www.engdes.com>. Acesso em: 27 jun. 2015.

HAYKIN, Simon; VAN VEEN, Barry. **Sinais e sistemas.** Porto Alegre: Bookman, 2001. xvii, 668 p

HBM. **Interrogator for optical strain gages:** DI. Darmstadt: Alemanha. Disponível em: <http://www.hbm.com/fileadmin/mediapool/hbmdoc/technical/a2301.pdf>. Acesso em: 15 out. 2014.

HBM. **Interrogator for optical strain gages:** SI. Darmstadt: Alemanha. Disponível em: <http://www.hbm.com/fileadmin/mediapool/products/images/a2220.pdf>. Acesso em: 15 out. 2014.

JOCHIMS, Felipe; PIRES, Cleber C.; GRIEBLER, Letieri; BOLZAN, Anderson M. S.; DIAS, Felipe D.; GALVANI, Diego B. Comportamento ingestivo e consumo de forragem por cordeiras em pastagem de milheto recebendo ou não suplemento. Revista Brasileira de Zootecnia, v. 39, p. 572–581, 2010.

KALINOWSKI, H. J.; ABE, I.; SIMÕES, J. A.; RAMOS, A. **Trends in Photonics - Application of fibre Bragg grating sensors in biomechanics**. Transworld Research Network, 315-343, 2010.

KARAM, Leandro. Z.; PEGORINI, Vinicius; PITTA, Cristiano S. R.; ASSMANN, Tangriani, S.; CARDOSO, Rafael; KALINOWSKI, Hypolito J.; SILVA, Jean C. C. Ex Vivo determination of chewing patterns using FBG and Artificial Neural Networks. In: INTERNATIONAL CONFERENCE ON OPTICAL FIBRE SENSORS, 23, 2014, Santander. **Proc. SPIE 9157…** Santander, 2 jul. 2014, 91573Z-1 – 91573Z-4.

LACA, Emilio A. GANADERÍA DE PRECISIÓN. In: REUNIÓN DEL GRUPO TÉCNICO EM FORRAJERAS DEL CONO SUR GRUPO CAMPOS, XXII, 2008, Minas. **Anais...** p. 75-80.

LACA, Emilio A.; UNGAR, Eugene D.; DEMMENT, Montague W. Mechanisms of handling time and intake rate of a large mammalian grazer. **Applied Animal Behaviour Science.** v. 39, n. 1, p. 3-19, 1994.

MILONE, Diego H.; RUFINER, Hugo L.; GALLI, Julio R.; LACA, Emilio A., CANGIANO, Carlos A. Computational method for segmentation and classification of ingestive sounds in sheep. **Computers and Electronics in Agriculture.** v. 65, p. 228-237, 2009.

MINISTÉRIO DA AGRICULTURA. **Bovinos e Bulbalinos.** Disponível em: < http://www.agricultura.gov.br/animal/especies/bovinos-e-bubalinos>. Acesso em: 23 de abr. 2015.

NATIONAL INSTRUMENTS. **LabVIEW.** Disponível em: < http://www.ni.com/labview>. Acesso em: 03 ago. 2014.

OPPENHEIM, Alan V.; SCHAFER, Ronald W. **Discrete-time signal processing.** 3rd ed. Upper Saddle River: Prentice-Hall, 2010. xviii, 1108 p.

OTHONOS, Andreas. Fiber Bragg gratings. **Review of Scientific Instruments**, vol. 68, p. 4309-4341, 1997.

OTHONOS, Andreas; KALLI, Kyriacos. **Fiber Bragg Grating: Fundamentals and Applications in Telecommunications and Sensing**, London, Artech House, 1999.

PEGORINI, Vinicius; RIBEIRO, Richardson; SILVA, Jean. C. C.; ASSMANN, Tangriani; S. PITTA, Cristiano. S. R.; CARDOSO, Rafael. Aprendizagem de Máquina Aplicada na Classificação de Padrões de Mastigação de Ruminantes. In: CONGRESSO BRASILEIRO DE AUTOMÁTICA, XX, 2014, Belo Horizonte. **Anais...** Belo Horizonte: 2014, p. 1034-1041.

RUTTER, Steven M.; CHAMPION, R. A.; PENNING, P. D. An automatic system to record foraging behaviour in free-ranging ruminants. **Applied Animal Behaviour Science.** v. 54, n. 2-3, p. 185-195, 1997.

RUTTER, Steven. M. Graze: A program to analyze recordings of the jaw movements of ruminants. **Behavior Research Methods, Instruments & Computers.** v. 32, p. 86-92, 2000.

SANTOS, Nailson L. **Avaliação do Capim-tanzânia manejado com diferentes IAF residuais sob lotação rotacionada por cabras Boer X Saanen.** 2009. Dissertação (mestrado) – Universidade Estadual Paulista, Faculdade de Ciências Agrárias e Veterinárias, 2009. Disponível em: <http://www.fcav.unesp.br/download/pgtrabs/zoo/m/3699.pdf>. Acesso em: 09 out. 2014, 09:00.

SAS. Institute Inc. **SAS Language reference**. Version 6. Cary, NC: SAS Institute, 1993. 1042 p.

SILVA, Cláudio J. A. da; DITTRICH, João R.; MONTEIRO, Alda L. G.; MORAES, Aníbal de; BARROS, Carina S. de; OLIVEIRA, Edílson B de. PREFERÊNCIA DE CAPRINOS EM PASTEJO: EFEITO DA ALTURA DOS DOSSÉIS DAS FORRAGEIRAS ARUANA E HEMÁRTRIA. **Ciência Animal Brasileira.** v. 10, n. 3, p. 698-710, 2009.

SILVA, Wesley J. da. **ESTUDO DO EMPREGO DE REDES NEURAIS ARTIFICIAIS PARA A CLASSIFICAÇÃO DE PADRÕES MASTIGATÓRIOS DE CAPRINOS.** 2014. 73 f. Trabalho de Conclusão de Curso (graduação) – Universidade Tecnológica Federal do Paraná, Curso Bacharelado em Engenharia Elétrica, Pato Branco, PR, 2014.

TRINDADE, Júlio K. da. **COMPORTAMENTO E CONSUMO DE FORRAGEM DE BOVINOS DE CORTE EM PASTAGEM NATURAL COMPLEXA.** 2011. 193 f. Tese (Doutorado em Zootecnia) – Faculdade de Agronomia, Universidade Federal do Rio Grande do Sul, Porto Alegre, 2011.

UNGAR, Eugene D. Ingestive behavior. Int: Hodgson, J; J. Illius, A. W. (Ed.) **The Ecology of management of grazing systems.**, p. 185–218, 1996.

UNGAR, Eugene David; RUTTER, Steven Mark. Classifying cattle jaw movements: Comparing IGER Behaviour Recorder and acoustic techniques. **Applied Animal Behaviour Science.** v. 98, p. 11-27, 2006.

WOSNIAK, Carla; SILVA, Wesley J; CARDOSO, Rafael; ASSMANN, Tangriani S.; HILL, João A. G.; SILVEIRA, André L. F.; SOUSA, Kleiton M.; KALINOWSKI, Hypolito J.; SILVA, Jean C. C. Determination of Chewing Patterns in Goats using Fiber Bragg Gratings. In: INTERNATIONAL CONFERENCE ON OPTICAL FIBER SENSORS**,** 22, 2012, Beijing. **Proc. SPIE 8424...** Beijing: 4 out. 2012 p. 84214F-1 – 842414F-4.