UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL

ESCOLA DE ENGENHARIA

DEPARTAMENTO DE ENGENHARIA ELÉTRICA

Sandro Buzacchi

AVALIAÇÃO DE UM SISTEMA DE ACELEROMETRIA DE BAIXO CUSTO EM ANÁLISE DE MOVIMENTOS DOS MEMBROS SUPERIORES

Porto Alegre, 2015

Sandro Buzacchi

AVALIAÇÃO DE UM SISTEMA DE ACELEROMETRIA DE BAIXO CUSTO EM ANÁLISE DE MOVIMENTOS DOS MEMBROS SUPERIORES

Trabalho de conclusão de curso apresentado como requisito parcial para a obtenção do grau de Bacharel em Engenharia Elétrica na Universidade Federal do Rio Grande do Sul.

Orientadora: Prof.ª Dra. Leia Bernardi Bagesteiro

Porto Alegre, 2015

CIP - Catalogação na Publicação

Buzacchi, Sandro
AVALIAÇÃO DE UM SISTEMA DE ACELEROMETRIA DE BAIXO CUSTO EM ANÁLISE DE MOVIMENTOS DOS MEMBROS SUPERIORES / Sandro Buzacchi. -- 2015.
104 f.

Orientadora: Leia Bernardi Bagesteiro.

Trabalho de conclusão de curso (Graduação) -- Universidade Federal do Rio Grande do Sul, Escola de Engenharia, Curso de Engenharia Elétrica, Porto Alegre, BR-RS, 2015.

1. Sensores inerciais. 2. Biomecânica. 3. Membros superiores. I. Bagesteiro, Leia Bernardi, orient. II. Título.

Elaborada pelo Sistema de Geração Automática de Ficha Catalográfica da UFRGS com os dados fornecidos pelo(a) autor(a).

Sandro Buzacchi

AVALIAÇÃO DE UM SISTEMA DE ACELEROMETRIA DE BAIXO CUSTO EM ANÁLISE DE MOVIMENTOS DOS MEMBROS SUPERIORES

Este trabalho de conclusão de curso foi analisado e julgado adequado para a obtenção do grau de Bacharel em Engenharia Elétrica e aprovado em sua forma final pelo Orientador e pela Banca Examinadora designada pelo Departamento de Engenharia Elétrica da Universidade Federal do Rio Grande do Sul

_____

Orientadora: Prof.ª Dra. Leia Bernardi Bagesteiro

_____

Chefe do Departamento: Prof. Dr. Ály Flores Ferreira Filho

Aprovado em: ___/___/____

Banca Examinadora:

Prof.ª Dra. Leia Bernardi Bagesteiro _____

Doutora pela University of Surrey

Prof. Dr. Marcelo Götz _____

Doutor pela Universität Paderborn

Prof. Dr. Rafael Antônio Comparsi Laranja _____

Doutor pela Universidade Federal do Rio Grande do Sul

Porto Alegre, 2015

DEDICATÓRIA

Dedico este trabalho aos meus pais por sempre acreditarem em mim e pelo apoio constante, e a minha namorada que teve compreensão e paciência nos momentos difíceis.

AGRADECIMENTOS

Aos meus pais que apesar da distância sempre me deram conselhos valiosos e se mostraram presentes.

À minha namorada por estar acompanhando esta jornada mais de perto e sempre ter me dado apoio.

Aos meus colegas de curso e a equipe do Laboratório de Instrumentação Eletro-Eletrônica (IEE) que quando solicitados mostraram-se disponíveis. Em especial a André Vieira Pigatto, Raissan Chedid e Daniel Corrêa pelos valiosos conhecimentos compartilhados.

À empresa Kinetec pelo apoio às minhas atividades nos momentos em que estive ausente.

RESUMO

Um sistema de acelerometria de baixo custo pode viabilizar análises biomecânicas, como por exemplo, no ambiente clínico, para avaliações e diagnóstico de deficiências e acompanhamento de processos de reabilitação. Este projeto tem como objetivo avaliar o sistema de unidade inercial presente em módulos *ArduIMU,* no contexto da biomecânica, aplicado ao estudo de membros superiores do corpo humano. A análise do sistema é realizada através da comparação das medições da plataforma de acelerometria de baixo custo *ArduIMU* com um sistema comercial de referência (sensor de posição), *FOB (Flock Of Birds)*. Além disso, levantou-se as curvas de resposta em dinâmica e estática do sistema utilizando-se um acelerômetro de referência e um transdutor de posição potenciométrico (*LVDT*). Dessa forma, comparou-se os dados adquiridos com o módulo *ArduIMU,* durante a realização movimentos de membros superiores, com os obtidos, simultaneamente, a partir do sistema de referência. Os movimentos utilizados para comparação são feitos no espaço tridimensional. A máxima diferença entre a medida de posição, obtida a partir dos dados adquiridos com o sistema *ArduIMU*, e a obtida com o sistema de referência, foi de 8,96% para a direção X, 10,13% para a direção Y e 9,51% para a direção Z. Os ensaios dinâmicos apresentaram uma variação máxima de 6,1%, em amplitude, em comparação com transdutor de posição (*LVDT*) e de 7,5% em aceleração, em relação ao acelerômetro de referência, para a faixa de 10 a 50Hz.

Palavras-chave: sensor inercial, biomecânica, membros superiores.

ABSTRACT

Low cost accelerometry systems can be used in biomechanics analysis, for example, in the clinical environment, for evaluation and diagnosis of deficiencies and the follow up of rehabilitation process. This project aims to evaluate the inertial measurement system present on ArduIMU modules, in the biomechanics context, applied in the study of upper limb movements. The analysis of the system is accomplished through a comparison between the low cost accelerometry platform ArduIMU and a commercial reference system (position sensor), FOB (Flock Of Birds). Furthermore, the dynamic and static responses of the system were computed using a reference accelerometer and a piezoelectric position transducer (LVDT). In this way, the data acquired with the ArduIMU during the execution of the upper limb movements were simultaneously compared with the ones obtained from the reference systems. Tridimensional movements were used for the comparisons. The maximum difference between the measured position, acquired with the ArduIMU and the reference system was 8,96% in the X direction, 10,13% in the Y direction and 9,51% in the Z direction. The dynamic tests revealed a maximum difference of 6,1% in amplitude, in comparison with the positional transducer (LVDT), and 7,5% in acceleration in comparison with the reference accelerometer, in a range between 10 and 50Hz.

Keywords: inertial sensor, biomechanics, upper limbs.

SUMÁRIO

1. INTRODUÇÃO

A biomecânica dos movimentos humanos pode ser definida como uma área interdisciplinar que descreve, analisa e avalia os movimentos humanos. Uma vasta gama de movimentos está envolvida: da marcha humana até esportes praticados por atletas profissionais. Os princípios físicos envolvidos são os mesmos em todos estes casos, a diferença está nos movimentos específicos de cada tarefa e no nível de detalhe que é exigido à cerca da execução de cada movimento. (WINTER, 2009). Diversas áreas da ciência utilizam análises biomecânicas: na obtenção de modelos virtuais que representem o movimento total ou parcial do corpo humano em jogos computacionais (RITTERFELD, 2009), na análise de marcha em estudos da eficiência de implantes de órteses (BARTONEK, 2015), e na avaliação do desempenho de atletas durante a prática esportiva (ROUSANOGLOU *et al.*, 2014), dentre tantas outras.

Existem diversas maneiras de realizar este tipo de análise biomecânica, podendo-se utilizar diferentes equipamentos para obter os parâmetros de interesse (deslocamento, velocidade, aceleração, força, momento, etc.) envolvidos nos movimentos do corpo humano: plataformas de força, câmeras para fotogrametria, e acelerômetros, entre outros. A mensuração direta pela acelerometria tem permitido a implementação de sucesso de sensores eletrônicos de baixo consumo e baixo custo que têm sido empregados em ambientes clínicos e cotidianos para o constante monitoramento de pacientes e seus controles. Os dados qualitativos e quantitativos fornecidos por estes sensores tornam possível que engenheiros, clínicos e médicos trabalhem em conjunto e consigam ajudar seus pacientes na superação de suas deficiências físicas. (GODFREY *et al*., 2008).

O objetivo deste trabalho é avaliar sensores de aceleração de baixo custo que poderão ser usados na análise biomecânica de movimentos dos membros superiores. O maior interesse está em reconstruir digitalmente os movimentos realizados pelos segmentos do braço, antebraço e mão, para isso é necessário integrar duas vezes os parâmetros de aceleração coletados com os sensores posicionados nestes segmentos. A utilização de sistemas de acelerometria pode ser usada como ferramenta única na quantificação cinemática dos membros do corpo humano, como por exemplo, no ambiente clínico para avaliações e diagnóstico de deficiências e acompanhamento de processos de reabilitação (CULHANE *et al.*, 2005; BALBINOT *et al.*, 2015). Desta forma, tem-se uma alternativa mais barata, cerca de US$ 100,00 (para uma unidade) em comparação com os sistemas de acelerometria comerciais

disponíveis hoje no mercado (em torno de US$ 2500,00 para uma unidade) (LETSENSE, 2015).

O sensor utilizado neste trabalho é o *ArduIMU*, que é uma unidade de medidas inerciais (do inglês, *Inertial Measuremet Unit*) contendo diferentes sensores: de aceleração, velocidade angular e orientação magnética. Acelerômetros são sensores ou transdutores que fornecem em sua saída uma medida quantitativa proporcional à aceleração, à vibração e à choques mecânicos aplicados sobre o mesmo (BALBINOT e BRUSAMARELLO, 2010). Podem ser uni, bi ou tri-axiais e tem uma ampla gama de aplicações. Giroscópios, por sua vez, são sensores que fornecem em sua saída níveis de tensão proporcionais à velocidade de giro empregada no dispositivo onde estão inseridos. Enquanto que magnetômetros são dispositivos que captam a orientação magnética da terra, e são utilizados em diversas aplicações sendo as mais comuns como uma bússola digital (de onde pode-se obter a direção de um movimento) e como detector de metais ferromagnéticos (SPARKFUN, 2015).

A placa *ArduIMU+ V3,* fabricada pela empresa *DIY Drones*, tem 2,54 cm de largura por 3,81 cm de comprimento e possuí acelerômetros e giroscópios tri-axiais do fabricante *Invensense* modelo MPU-6000 do tipo MEMS (*Microelectromechanical Systems*), bem como um magnetômetro tri-axial do fabricante *Honeywell* modelo HMC-5883L, controlados pelo microcontrolador ATmega 328 do fabricante *Atmel*. Neste projeto os dados do magnetômetro não são utilizados.

Como os movimentos dos membros superiores acontecem num espaço tridimensional relativamente restrito, ou seja, ao redor do corpo; o sistema *ArduIMU* é testado num espaço similar. Por exemplo, um dos movimentos testados foi o de descrever um formato de "S" nesse ambiente que tem amplitude limitada, avaliando-se assim um deslocamento máximo de 70cm e uma aceleração de até 3,8m/s² (FREIVALDS, 2004), numa faixa de 0 a 50Hz. Os dados do sistema de acelerometria *ArduIMU* são comparados com um sistema baseado em sensores magnéticos de posição com 6 graus-de-liberdade *FOB* (*Flock Of Birds*) (ASCENSION TECHNOLOGY, 2015), aqui considerado de referência, e comumente utilizado em pesquisas científicas para avaliação de movimentos humanos (BAGESTEIRO e SAINBURG, 2002) e também na área de desenvolvimento de jogos computacionais (RITTERFELD, 2009).

2. REVISÃO BIBLIOGRÁFICA

Esse capítulo visa revisar e aprofundar os conhecimentos pertinentes à utilização dos sistemas empregados e apresentar os conceitos para o entendimento do desenvolvimento deste trabalho

2.1. Acelerometria

A acelerometria é um método de análise cinemática do movimento (ROBERTSON *et al*., 2004) de forma não-invasiva. É comumente utilizada em análises biomecânicas do movimento humano e permite através da utilização de um sensor, mensurar as acelerações provocadas e sofridas pelos segmentos corporais onde estes estão alocados. Como os movimentos do corpo humano ocorrem em mais de um eixo, utilizam-se preferencialmente acelerômetros tri-axiais, que permitem medir a aceleração instantânea em cada um dos eixos ortogonais, e que em combinação reproduzem a aceleração resultante do movimento em estudo.

Acelerômetros são baseados no princípio físico da ação da aceleração em uma massa para produzir força. Esta relação básica é expressa pela segunda lei de Newton (BALBINOT e BRUSAMARELLO, 2010) como mostra a Equação (1).

$$F = m * a \tag{1}$$

onde F representa a força [N], m a massa [kg] e a é a aceleração [m/s²].

Algumas vantagens de se utilizar um sistema de acelerometria (WINTER, 2009) são:

- Sinal de aceleração disponível imediatamente para gravação e processamento em um computador;
- Baixo custo em relação a outros equipamentos para avaliação biomecânica;
- Fácil implementação (preparação do sujeito de forma rápida, com fácil posicionamento dos sensores).

Possíveis desvantagens:

- As medidas de aceleração e variação angular são relativas à sua posição no membro do corpo;
- A obtenção da posição através da integração do sinal de aceleração pode acumular erros numéricos e apresentar dados não precisos;
- Se utilizados em grande número podem sobrecarregar o movimento;
- A massa do sistema pode resultar em artefatos do movimento, especialmente em movimentos muito rápidos ou que envolvem impacto.

Os acelerômetros da plataforma *ArduIMU* fazem a amostragem da aceleração aplicada sobre o circuito através de um conversor analógico/digital integrado. Associados à essa amostragem estão uma frequência de amostragem, em Hz, e um número de bits, que determina a amplitude máxima do sinal amostrado.

2.2. Avaliação de movimentos dos membros superiores

Com o objetivo de utilizar o *ArduIMU* para avaliar movimentos realizados com os membros superiores e apresentadas as desvantagens da utilização de sistemas de acelerometria na seção 2.1 é de interesse utilizar o menor número possível de unidades inerciais para quantificar e analisar estes movimentos.

Considerando que os segmentos do corpo humano podem ser considerados como segmentos rígidos (WINTER, 2009), mantendo suas dimensões constantes ao longo do tempo de realização dos movimentos, e assumindo um modelo do membro superior de dois segmentos (braço e antebraço - BAGESTEIRO e SAINBURG, 2002), pode-se utilizar somente uma unidade inercial para cada segmento para realizar a avaliação do movimento. Com as medidas antropométricas e as informações de posição e angulação do centro de massa do segmento é possível calcular as posições das extremidades de cada segmento do membro superior (braço e antebraço) em cada instante de execução do movimento avaliado.

Portanto, para os movimentos do membro superior utiliza-se duas unidades inercias, uma para cada segmento. A Figura 1 mostra um a representação do modelo de membros superiores adotado. Os segmentos são considerados rígidos e homogêneos, com uma ligação

entre si e livres para se movimentar conforme os sentidos de giro ilustrados, realizando movimentos de flexão/extensão, adução/abdução, pronação/supinação e rotação, dependendo da articulação em movimento.

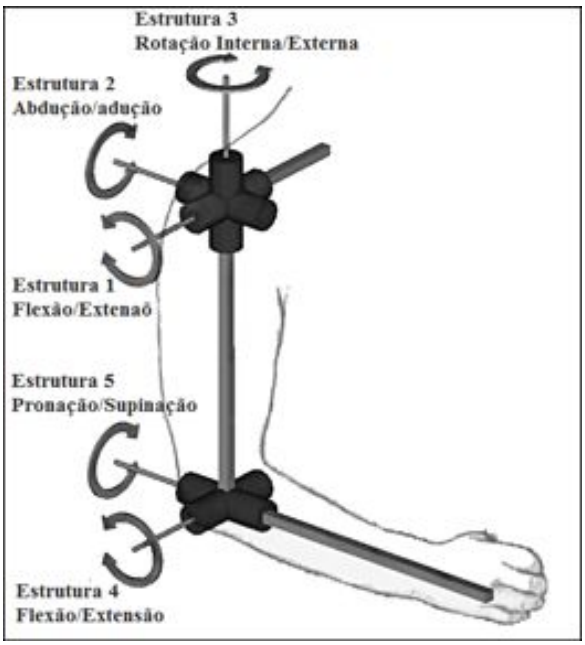


Figura 1 – *Representação do modelo de membros superiores adotado.*

FONTE: *ABDULLAH, 2007. Livre adaptação.*

Além destas definições o modelo considera que a massa de cada segmento está concentrada no centro de massa daquele segmento (braço e antebraço-mão). Para localizar o centro de massa dos segmentos é necessário conhecer o comprimento de cada segmento individualmente. Através das proporções (ROBERTSON *et al*., 2004) mostradas na Tabela 1 é possível localizar o centro de massa (CM) a partir das regiões distal (R_{distal}) e proximal ($R_{proximal}$).

Tabela 1 – *Proporções antropométricas para localizar o centro de massa.*

Segmento	**CM/comprimento do segmento**	
	$R_{proximal}$	R_{distal}
Braço	0,436	0,564
Antebraço-mão	0,682	0,318

As posições dos centros de massa em relação às medidas dos segmentos também são utilizadas para posicionar a sistema de acelerometria do $ArduIMU$ corretamente nestes

segmentos. Desta forma as posições do CM no braço e no antebraço coincidem com o posicionamento dos sensores utilizados.

As proporções da Tabela 1 também podem ser utilizadas como uma aproximação para localizar as extremidades de cada segmento a partir do CM. As extremidades do antebraço são definidas como: comprimento entre sensor (mesma localização do CM) e dedos (região distal) e comprimento entre sensor e cotovelo (região proximal). Já as extremidades do braço são definidas como: comprimento entre sensor e cotovelo (região distal) e comprimento entre sensor e ombro (região proximal). Com estas extremidades conhecidas é possível, a partir dos dados coletados com o *ArduIMU*, utilizar equações de transformação de coordenadas para reconstruir digitalmente os segmentos avaliados. Para estas transformações é necessário utilizar as Equações de (2) a (7).

$$dedos_x = x0 + (\cos(\theta) * \cos(\varphi) * comprimento_{sensor-dedos}) \quad (2)$$

$$dedos_y = y0 + (\text{sen}(\theta) * \cos(\varphi) * comprimento_{sensor-dedos}) \quad (3)$$

$$dedos_z = z0 + (\cos(\varphi) * comprimento_{sensor-dedos}) \quad (4)$$

$$cotovelo_x = x0 + (\cos(\theta) * \cos(\varphi) * -comprimento_{sensor-cotovelo}) \quad (5)$$

$$cotovelo_y = y0 + (\text{sen}(\theta) * \cos(\varphi) * -comprimento_{sensor-cotovelo}) \quad (6)$$

$$cotovelo_z = z0 + (\cos(\varphi) * -comprimento_{sensor-cotovelo}) \quad (7)$$

onde $dedos_x$, $dedos_y$ e $dedos_z$ representam as coordenas x, y e z calculadas para a posição da ponta do dedo médio, $cotovelo_x$, $cotovelo_y$ e $cotovelo_z$ representam as coordenas x, y e z calculadas para a posição do cotovelo, $x0$, $y0$ e $z0$ são as coordenadas da posição inicial (posição do sensor), θ é o ângulo obtido para rotações e torno do eixo Z e φ é o ângulo obtido para rotações em torno do eixo X.

2.3. Transmissão de dados

A forma de transmissão dos dados adquiridos com o *ArduIMU* pode ser feita de diferentes maneiras:

1. Armazenamento local em uma unidade de memória *flash* (cartões micro SD);
2. Pelo cabo de programação do microprocessador;
3. Por meios de transmissão sem fio, como módulos *bluetooth* ou de radiofrequência.

A primeira opção é interessante quando não é necessária a obtenção dos dados para visualização instantânea dos dados coletados, pois como os dados são gravados em um cartão de memória só estarão acessíveis ao término do registro dos mesmos. A alternativa de envio de dados por cabo certamente é a menos interessante, pois restringe os movimentos realizados pelo sujeito. Optou-se pela transmissão dos dados por um sistema sem fio de radiofrequência: os *XBees*.

2.3.1. Módulos de radiofrequência *XBee*

Módulos de radiofrequência *XBee* são soluções embarcadas que fornecem conectividade sem fio para dispositivos em diversas aplicações (DIGI INTERNATIONAL, 2015). O modelo utilizado é o *XBee Series 1* da empresa *Digi International* para a transmissão dos dados e o *XBee-PRO Series 1* na recepção dos dados. Para a transmissão de dados na utilização destes dispositivos são necessárias duas unidades *XBee* mais um adaptador *USB* para ser estabelecida a comunicação: uma delas é conectada ao *ArduIMU* para enviar os dados de aceleração (transmissor), e a outra é conectada a um computador (através do adaptador *USB*) para receber estes dados (receptor ou coordenador). Os módulos também são chamados de transceptores quando não diferenciados entre transmissor e receptor. Na interface receptor-computador a comunicação é feita pela porta *USB* através de um *XBee Exlplorer Dongle*, que nada mais é do que um adaptador para conectar o transceptor ao computador. A Figura 2 mostra um destes adaptadores, denominados CON-USBBEE fabricados pela empresa Rogercom.

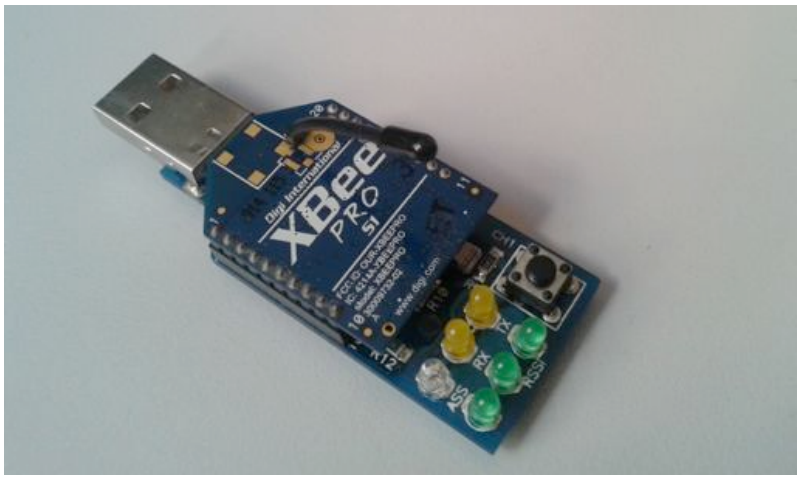


Figura 2 – *Adaptador para conectar o XBee à porta USB do computador (USB Explorer) da Rogercom.*

Os módulos *XBee* trabalham utilizando o protocolo de comunicação IEEE 801.15.4, sendo capazes de criar redes *WPAN* (*Wireless Personal Area Network*). As principais vantagens destes dispositivos são: baixo consumo, suporte a diferentes configurações de rede, comunicação serial *UART*, configurações para gerenciamento de energia, entradas/saídas digitais, *ADCs* de 10 bits e saída *PWM* dedicada. Algumas de suas especificações podem ser vistas na Tabela 2 (dados retirados do manual do fabricante).

Tabela 2 – *Especificações dos módulos XBee* (*SUPPORT DIGI INTERNATIONAL,* 2015).

Especificação	***XBee***	***XBee-PRO***
Desempenho		
Alcance Interno	Até 30m	Até 60m
Alcance Externo	90m	750m
Potência de Transmissão	1mW	63mW (18dBm) 10mW (10dBm)
Velocidade de Transmissão RF	250.000 bps	250.000 bps
Velocidade de Transmissão Serial	1200 bps - 250 kbps	1200 bps - 250 kbps
Sensibilidade de Recepção	-92 dBm (1% taxa de erro de pacote)	-100 dBm (1% taxa de erro de pacote)
Alimentação		
Tensão de alimentação	2.8 – 3.4 V	2.8 – 3.4 V
Consumo Típico	50mA (@ 3.3 V)	55mA (@ 3.3 V)
Generalidades		
Frequência de Operação	ISM 2.4 Ghz	ISM 2.4 Ghz
Dimensões	2,438 cm x 2,761 cm	2.438cm x 3.294cm
Temperatura de Operação	-40 a 85º C	-40 a 85º C

Continuação da ***Tabela 2*** *– Especificações dos módulos XBee* (*SUPPORT DIGI INTERNATIONAL,* 2015).

Redes e Segurança		
Topologias Suportadas	Point-to-point, Point-to-multipoint & Peer-to-peer	-
Número de Canais	16 Canais em sequência direta	12 Canais em sequência direta
Opções de Endereçamento	PAN ID, Canal e Endereço	PAN ID, Canal e Endereço

Os modos possíveis de serem utilizados para a transmissão de dados são: *AT* e *API*. O modo *AT* é também chamado de "modo transparente". Neste modo qualquer dado enviado ao módulo *XBee* é imediatamente enviado ao coordenador identificado pelo endereço de destino. Já o modo *API* baseia-se em quadros (*frames*): os dados transmitidos e recebidos são inseridos em quadros, os quais definem operações ou eventos dentro do módulo. Através desse modo de operação, são informados diversos detalhes sobre a comunicação: endereço da fonte, endereço de destino, nome de um determinado nó, sinal *RSSI* (Intensidade do Sinal Recebido), estado, etc. A explicação das informações contidas nos pacotes *API* será detalhada no capítulo 3.2.

2.4. Sistema de Referência FOB

O sistema *Flock of Birds* (*FOB*) é um dispositivo de medição com 6 graus-de-liberdade que pode ser configurado para rastrear simultaneamente a posição e a orientação de múltiplos sensores para medida de posições e ângulos nos três eixos ortogonais (6 graus de liberdade) e por um transmissor. Cada sensor é capaz de realizar de 20 a 140 medidas por segundo de sua posição e orientação em três dimensões quando o sensor está localizado dentro do limite de ±1,22 metros distante de seu transmissor. Este limite pode ser estendido com um transmissor opcional que aumenta o alcance de operação para até ±2,44 metros. O *FOB* funciona pela transmissão de um campo magnético DC pulsado que é simultaneamente medido por todos os sensores do grupo. Das características de cada campo magnético medido, cada sensor independentemente calcula sua posição e orientação e torna esta informação disponível para um computador gerenciador.

3. METODOLOGIA EXPERIMENAL

O sistema objeto de estudo consiste em uma placa *ArduIMU* que contém o sensor de aceleração, responsável pela transdução da aceleração em sinais elétricos e posteriormente em sinais digitais através de um *ADC* (*Analog Digital Converter*) e o microprocessador que é o responsável por gerenciar as tarefas de leitura do sensor e coordenar os demais dispositivos ligados a ele. Para o funcionamento do *ArduIMU* são necessários outros dois elementos: uma fonte de energia para alimentar o sistema e um método de obtenção dos dados adquiridos. Para o primeiro elemento foi escolhida uma bateria convencional de 9V por apresentar a tensão adequada de suprimento de energia ao sistema que é na faixa de 6 a 12V. O estudo sobre outras fontes de alimentação do circuito é apresentado na seção 3.1.3.

3.1. Hardware

Essa subseção descreve individualmente os itens que compõem o *hardware* do sistema de acelerometria utilizado.

3.1.1. *ArduIMU V3*

O *ArduIMU V3* é uma unidade de medida inercial com sensores e circuitos de filtros em hardware em conjunto com um microprocessador compatível com *Arduino*. Esta placa é composta por acelerômetro e giroscópio tri-axiais, dois reguladores de tensão (3.3V e 5V), entrada para GPS, microprocessador Atmega328 de 16MHz e diversos LEDs de indicação dos estados de operação. É a unidade de medida inercial mais barata do mercado, segundo o fabricante (CODE GOOGLE, 2015). Faz parte de um projeto homônimo *ArduIMU*, de código aberto sobre a licença CC (*Creative Commons*) - que são licenças de *copyright* -, promovido pela *DIY Drones*, com o objetivo de desenvolver um sistema de medidas inerciais para servir de base no desenvolvimento de sistemas de navegação e controle em veículos aéreos não tripulados. A placa encontra-se atualmente em sua quarta versão. A placa utilizada neste projeto é o modelo da terceira versão.

Especificações:

- Sensor de velocidade angular (giroscópio) tri-axial com sensibilidade de até 131 LSB/dps (bytes por graus por segundo) e fundo de escala selecionável entre ±250, ±500, ±1000, e ±2000dps;
- Acelerômetro tri-axial com fundo de escala programável nos alcances de ±2g, ±4g, ±8g e ±16g;
- Fornecimento de corrente em modo inativo: 5μA;
- Temporizador interno com ±1% de variação em frequência;
- Tolerância contra choques de até 10,000g.

Outras características do *ArduIMU V3*:

- Compatível com placas de teste (*protoboards*). Versões anteriores não eram compatíveis;
- Tamanho reduzido: 2,54 x 3,81 cm;
- 6 pinos para entradas analógicas disponíveis;
- Comunicação I2C com tradução para lógica 3.3V;
- Entrada para GPS com auto seleção FTDI.

As características elétricas completas do acelerômetro podem ser vistas na Tabela 3 conforme manual do fabricante. A Figura 3 mostra uma placa do *ArduIMU*.

Tabela 3 *– Características elétricas do acelerômetro MPU-6000* (*InvenSense,* 2015).

Parâmetro	Condição	Mín.	Típico	Máx.	Unidade
Sensibilidade					
Fundo de escala	Faixa de Seleção 0		±2		g
	Faixa de Seleção 1		±4g		g
	Faixa de Seleção 2 *		±8g		g
	Faixa de Seleção 3		±16g		g
Comprimento ADC	Complemento de dois		16		bits
Escala de Sensibilidade	Faixa de Seleção 0		16.384		LSB/g
	Faixa de Seleção 1		8.192		LSB/g
	Faixa de Seleção 2 *		4.096		LSB/g
	Faixa de Seleção 3		2.048		LSB/g
Tolerância de Calibração Inicial			±3		%
Alteração da Sensibilidad vs. Temperatura	Faixa de Seleção 0, -40° a +85°C		±0,02		%/°C

Continuação da ***Tabela 3*** *– Características elétricas do acelerômetro MPU-6000* (*InvenSense,* 2015).

Não linearidade	Melhor aproximação por reta	**0,5**		**%**
Sensibilidade entre eixos		±2		%
Saída em medição de zero g				
Tolerância de Calibração Inicial	Eixos X e Y	±50		mg
	Eixo Z	±80		mg
Alteração vs. Temperatura	Eixos X e Y, 0 a +70°C	±35		mg
	Eixo Z, 0 a +70°C	±60		mg
Desemprenho a Ruído				
Densidade Espectral de Potência	Faixa de Seleção 0 10Hz	400		µg/√Hz
Resposta a Filtro Passa Baixas				
	Faixa Programável	5	260	Hz
Taxa de Saída de Dados				
	Faixa Programável	4	1.000	Hz

** Faixa de seleção escolhida.*

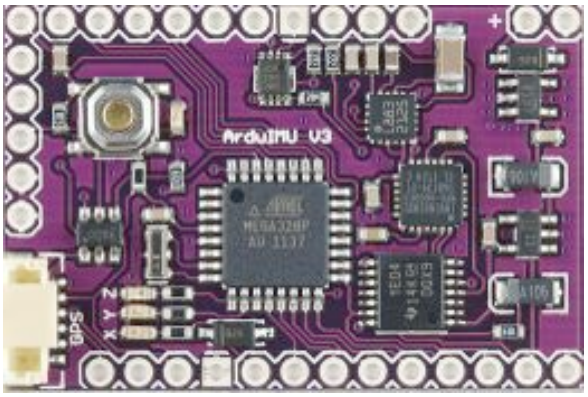


Figura 3 *– Placa comercial do ArduIMU V3.*
FONTE: *https://cdn.sparkfun.com//assets/parts/6/4/2/6/11055-02a.jpg*

3.1.2. Transmissão de dados

Como método de transmissão de dados foram utilizados os módulos de radiofrequência *XBee*. O modo escolhido para transmissão de dados entre os *XBees* é o de pacotes *API* que serão detalhados na seção 3.2.

3.1.3. Alimentação do Circuito

A investigação de um método mais eficiente de alimentação do sistema do que as baterias de 9V foi realizada, porém, para a estrutura atual do *ArduIMU*, a melhor solução continua sendo as baterias de 9V. Uma pesquisa de mercado foi feita procurando diferentes alternativas. Testes foram feitos com baterias do tipo botão CR-2032 e baterias de 12V do tipo A23 e A27, porém estas não se mostraram eficientes em suprir corrente suficiente para o sistema. Outra opção interessante seriam as baterias do tipo botão CR-2477 de 3V e 1000mAh, mas estas são de maior tamanho e peso do que as CR-2032 e acabam por não apresentar vantagem em relação as baterias de 9V pois são necessárias no mínimo 3 unidades para obter a tensão de entrada mínima do *ArduIMU* que fica na faixa de 6 a 12V. A Tabela 4 apresenta um comparativo de dimensões e massa das baterias testadas.

Tabela 4 – *Lista de dimensões e peso das baterias testadas para alimentação a placa do ArduIMU com XBee.*

Tipo	Tensão [V]	Capacidade Típica [mAh]	Dimensões [mm]				Massa [g]
			Diâmetro	Altura	Largura	Profund.	
Botão CR2032	3	190	20	3,2	-	-	3
Botão CR2477	3	1000	24	7,7	-	-	10
Bateria 9V	9	280* 520**	-	48,5	26,5	17,5	46
Bateria A23	12	55	10,3	28,5	-	-	8
Bateria A27	12	22	8	28,2	-	-	4,4

**Bateria de 9V Ni-Mh Recarregável. **Bateria de 9V Alcalina Não Recarregável.*

A grande dificuldade para ser proposta uma alternativa às baterias de 9V é a tensão de entrada exigida pela placa. Sugestões de melhoria neste tópico são apresentadas no capítulo 6.

3.1.4. Integração do Hardware

Como os sensores comerciais já estão embarcados sobre uma placa que contém um microprocessador, as únicas implementações em hardware necessárias são as de ligação com o transceptor *XBee* e a escolha da forma de alimentação. Trabalhos anteriores (CORREA, 2015) já utilizaram o *ArduIMU* como base de seus estudos, e as etapas de implementação de hardware destes estudos serviram como base para o presente trabalho.

O equipamento foi montado conforme o esquema da Figura 4. O circuito é alimentado por uma bateria de 9V convencional que tem seus terminais V+ e V- (caminhos em preto) conectados respectivamente aos pinos 30 (Vin) e 1 (*GND*) da placa. A bateria supre energia para *ArduIMU* (tensão de entrada de 6 a 12V) e para o *XBee* (tensão de alimentação 3.3V). Tanto os sensores presentes na placa quanto o microprocessador possuem tensão de alimentação de 5V. Para esta tensão ser alcançada o *ArduIMU* possuí reguladores de tensão embarcados que rebaixam a tensão de alimentação para 5V e 3.3V quando necessário.

As demais conexões necessárias são referentes ao *XBee*, que além da alimentação (3.3V representado pelo caminho em vermelho, e Terra representado pelo caminho em azul) é ligado ao pino Tx (caminho em verde) do *ArduIMU* responsável pela transmissão de dados na forma serial I2C.

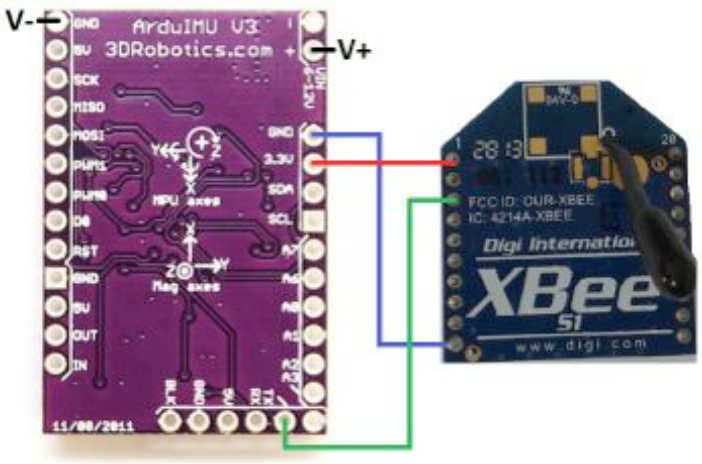


Figura 4 – *Esquema de montagem do hardware utilizado. Os caminhos em preto representam as ligações feitas com uma bateria de 9V e, os demais caminhos representam as conexões com o módulo XBee: alimentação (caminho vermelho), terra (caminho azul) e comunicação (caminho verde).*

FONTE: *https://s3.amazonaws.com/3dr.production/294/large/KT-ArduIMU-30-4.jpg?1441214785), livre adaptação.*

A miniaturização do hardware incluí o estudo de consumo do sistema para procurar alternativas à utilização da bateria de 9V empregada atualmente devido as suas dimensões e peso. A fim de proteger o equipamento e fornecer melhor acabamento ao mesmo, o *ArduIMU* e o *XBee* foram acondicionados em uma caixa plástica e fixados a mesma com fita dupla-face. Devido as suas dimensões a bateria de 9V não coube no interior da caixa plástica. As dimensões finais do sistema são de 450 x 190 x 350mm. As Figuras 5 e 6 mostram a caixa plástica fechada ao lado da bateria de 9V e as ligações feitas entre *ArduIMU* e *XBee* respectivamente.

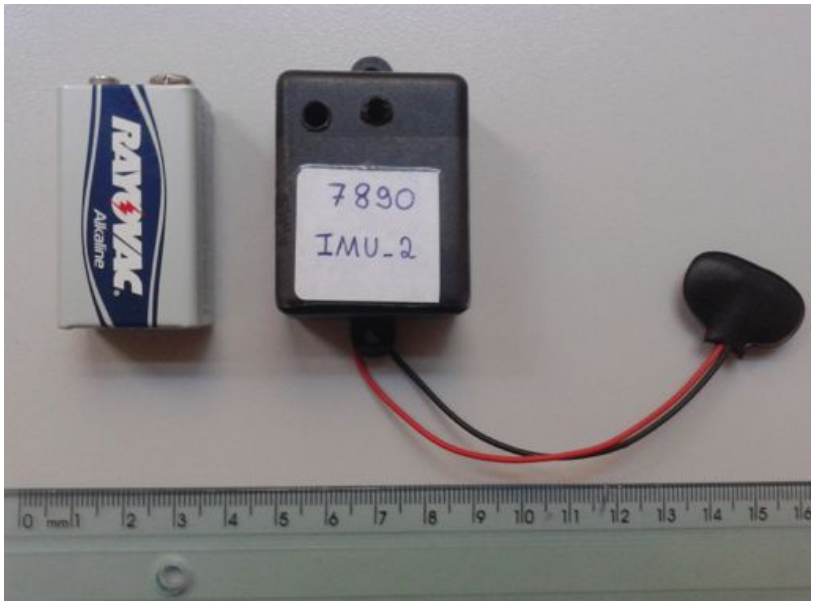


Figura 5 – *Montagem final do sistema ArduIMU e XBee; bateria de 9V e conector dos pinos de alimentação.*

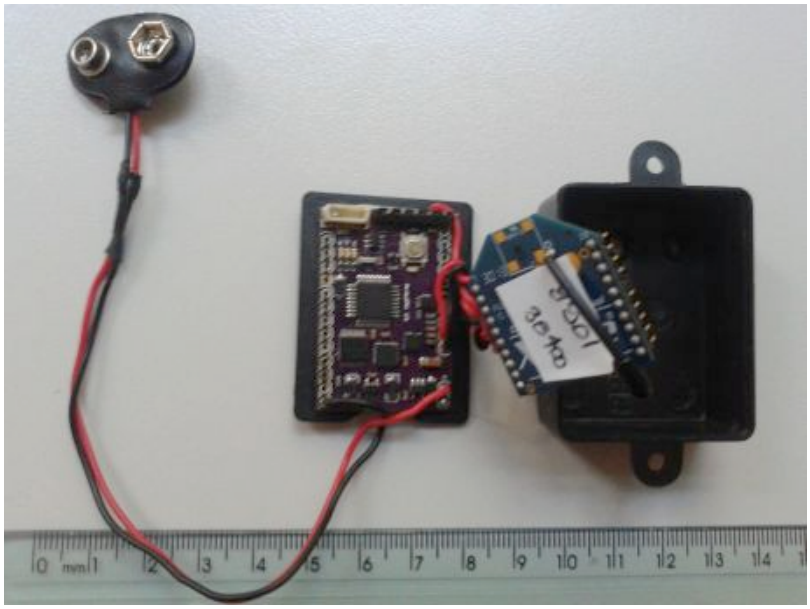


Figura 6 – *ArduIMU e XBee conectados e conector para a bateria de 9V.*

3.2. Software

Os procedimentos de gravação dos códigos de controle no microprocessador, bem como a descrição dos códigos de desencapsulamento dos pacotes *API* (e como estes funcionam) e de processamento dos dados adquiridos com ambos os sistemas são descritos a seguir. Os códigos foram escritos com *MATLAB* e a *IDE* (*Integrated Development Environment*) do *Arduino*.

Após a montagem do hardware é necessária a programação de um código que é escrito no microcontrolador através de um cabo *FTDI* (*Future Technology Devices International*). Este é um cabo adaptador com conexão *USB* em uma das pontas e divisão em seis pinos (interface *FTDI*) em sua outra extremidade, o que possibilita a programação de dispositivos que utilizam da tecnologia de circuitos integrados *FTDI Chip*. O código desenvolvido realiza a amostragem dos sensores embarcados na placa para a aquisição de dados e também será responsável pelo envio dos sinais através dos módulos de radiofrequência.

No código escrito com a *IDE* do *Arduino* (Anexo A), os sensores são amostrados a uma taxa de 500Hz e é aplicada a correção de calibração na mesma taxa. A transmissão dos dados adquiridos é feita a 125Hz.

As variáveis temporárias que armazenam os valores amostrados podem receber valores de até 16 bits (2 bytes) em complemento de dois, compreendendo a faixa de -32.768 a 32.767 em base decimal. É importante que os dados enviados tenham sempre o mesmo comprimento em bits, pois o código de desencapsulamento dos pacotes *API* conta com o posicionamento fixo dos dados dentro do pacote. Para assegurar que os valores amostrados e enviados apresentem sempre o mesmo comprimento, foi implementado o sistema de *bit shifting* e *bit masking*, que divide os 2 bytes de cada valor em dois e envia um de cada vez: parte mais significativa (com 1 byte de comprimento) e parte menos significativa (também com 1 byte de comprimento).

Primeiramente o valor amostrado é deslocado para a direita de 8 bits (1 byte) que corresponde ao processo de *bit shifting*. Isto faz com que somente os 8 bits da parte mais significativa do valor amostrado seja enviado.

Posteriormente o mesmo valor (não deslocado) é operado bit a bit com o operador *AND* e o valor 0xff (11111111 em binário), correspondendo ao processo de *bit masking*. O operador *AND* funciona conforme a álgebra booleana, obedecendo à seguinte regra: se ambos os bits forem 1, o resultado é igual a 1, caso contrário o resultado é 0. Desta forma somente os

8 bits da parte menos significativa do valor amostrado são enviados. O esquemático da Figura 7 ilustra este processo.

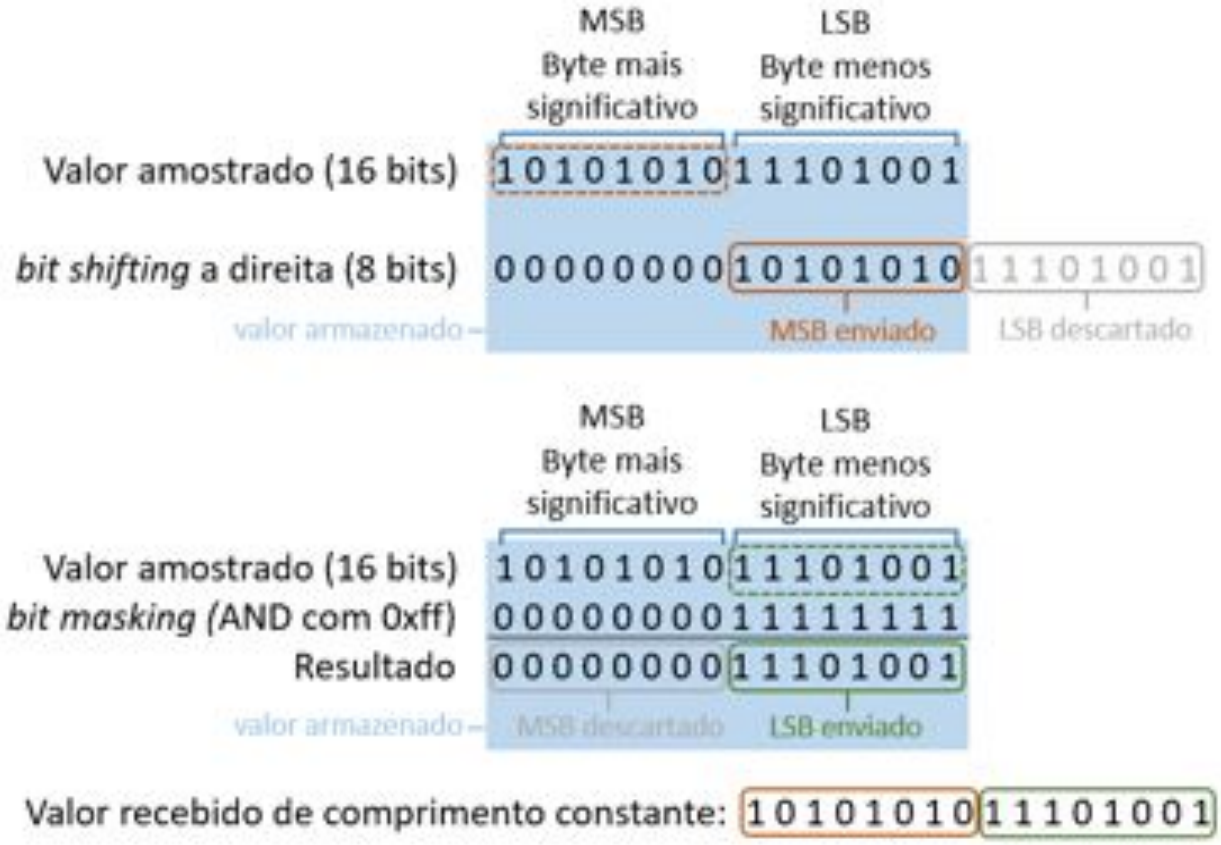


***Figura 7** – Processos de bit shifting e bit masking aplicados aos valores amostrados pelo sensor de acelerometria e resultado final recebido na interface receptor-computador.*

Na interface receptor-computador o software utilizado é o *XCTU,* também da *Digi International,* fabricante dos módulos *XBee.* Este software é responsável pela configuração dos módulos *XBee* e pelo recebimento e gravação dos pacotes *API* em um arquivo com extensão.*log*. As configurações importantes de serem feitas nos *XBees* são:

- Endereço de envio compatível com o endereço do coordenador, para enviar informação na rede correta;
- Canal;
- Habilitar o modo *API;*
- *Baud Rate* - a velocidade com a qual dois dispositivos se comunicam. As *Baud Rates* utilizadas foram: 38400 entre *XBee* e *ArduIMU* e 115200 entre *XBee* e computador.

A estrutura dos quadros do pacote API é ilustrada na Figura 8.

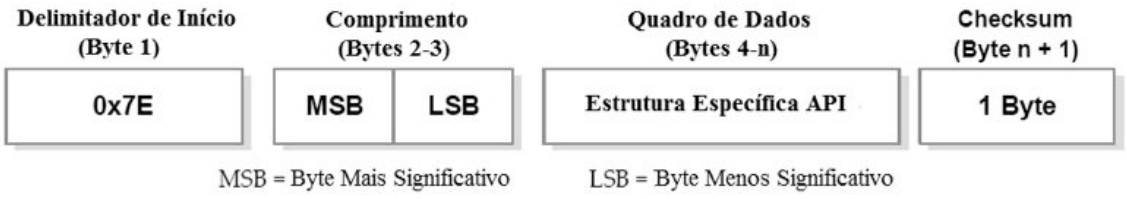


Figura 8 – *Sequência de bits que formam a estrutura dos pacotes API.*

FONTE: *http://www.digi.com/products/xbee-rf-solutions/modules/xbee-series1-module, livre adaptação.*

Dentro desta estrutura o Quadro de Dados é expandido com a sequência de bytes ilustrados na Figura 9.

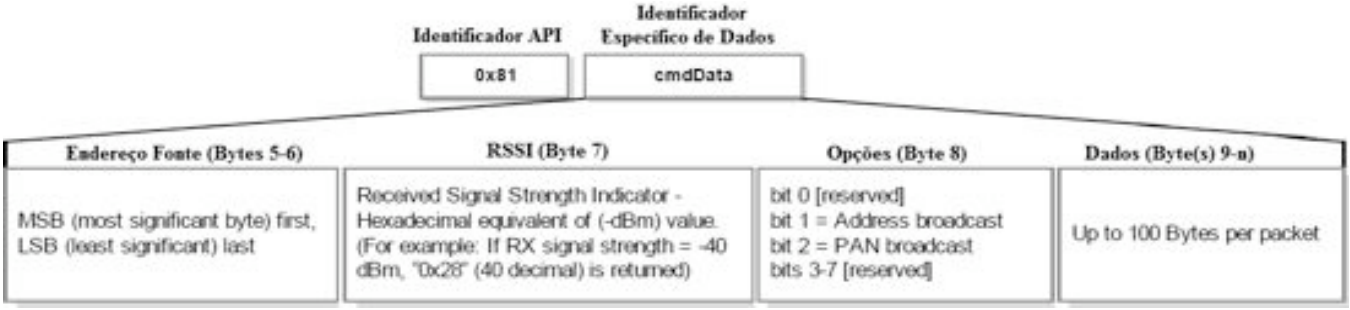


Figura 9 – *Detalhamento dos bytes contidos dentro do quadro de dados*

FONTE: *http://www.digi.com/products/xbee-rf-solutions/modules/xbee-series1-module, livre adaptação.*

O Delimitador de Início é um valor fixo em hexadecimal (0x7E) que identifica o começo do pacote e é o primeiro byte a ser recebido; na sequencia são encontrados 2 bytes que correspondem ao Comprimento do pacote, que começa a ser contado a partir do Endereço Fonte e vai até o byte *Checksum*. *RSSI* é um byte que quantifica a força do sinal recebido pelo coordenador. Opções é um byte dedicado para configurações internas do *XBee.* Dados é a parte reservada para o posicionamento dos dados que são enviados neste pacote; podem ser acumulados até 100 bytes de dados para serem enviados em um pacote. Caso um conjunto de dados exceda este limite de comprimento, são enviados dois pacotes ao invés de um, com a informação dividida entre estes dois pacotes. Por este motivo não é aconselhado que as informações que se deseja enviar superem os 100 bytes. Finalmente, o *Checksum* é um byte para verificação da integridade do pacote e dos dados contidos nele. Caso algum pacote esteja corrompido, o *Checksum* identifica o problema e envia uma mensagem de retorno ao endereço do *XBee* de origem do pacote. Problemas de pacotes corrompidos são extremamente raros e portanto esta funcionalidade não foi utilizada.

Os pacotes *API* recebidos pelo *XBee* coordenador são gravados no computador utilizando-se o software *XCTU*.

Na janela deste software são mostrados instantaneamente todos os pacotes que estão sendo recebidos pelo *XBee* coordenador. Setas em vermelho indicam os pacotes sendo recebidos. Uma coluna de identificação (*ID*) identifica os pacotes na ordem em que foram recebidos. O tempo em que os pacotes foram recebidos também é registrado em uma das colunas. Por fim o comprimento e o tipo de quadro também aparecem na janela de identificação de quadros.

A janela *Frame details* apresenta o pacote *API* completo selecionado na janela *Frames log.* Aqui é possível ver a estrutura completa do pacote abaixo da aba em azul *RX (Receive) Packet 16-bit Address* e pode ser visto em destaque na Figura 10.

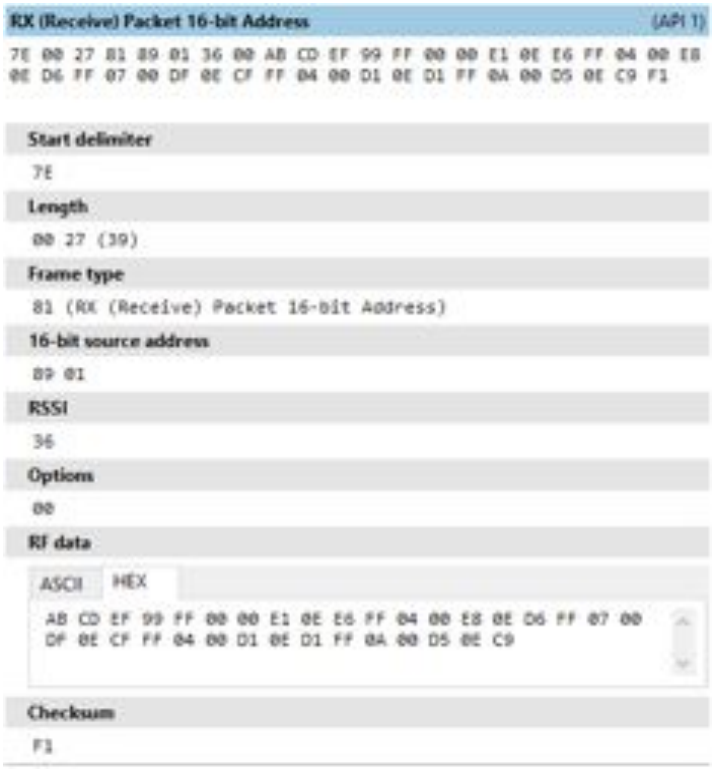


Figura 10 – *Destaque do quadro API recebido. O pacote recebido é mostrado abaixo da aba em azul RX (Receive) Packet 16-bit Address e é detalhado byte a byte nas abas abaixo.*

Como descrito anteriormente, o byte inicial é composto pelo delimitador constante 7E. A informação sobre os dados amostrados do sensor está contida na aba *RF data* e é expressa em valores hexadecimais. Cada pacote *API* é composto por cinco leituras de cada um dos três eixos do acelerômetro. Diferente do modo *AT,* o modo *API* não envia imediatamente as informações coletadas, mas aguarda que uma quantidade preestabelecida de dados seja

agrupada para então enviar os dados correspondentes. Esta quantidade preestabelecida depende: de quantas iterações são aguardadas até a execução do comando de envio, e do comprimento dos dados de uma amostragem. Anteriormente foi mencionado que cada variável temporária que recebe os dados amostrados tem 2 bytes de comprimento, sendo assim, para os três eixos tem-se 2*3 = 6 bytes. O número de iterações é igual ao número de leituras feitas por eixo até o envio, como são efetuadas cinco leituras a quantidade de dados agrupados é igual a 6*5 = 30 bytes.

Junto com cada pacote também é enviado um segundo delimitador que tem por função simplificar o código e desencapsulamento dos dados. Este delimitador é o caractere especial ABCDEF99 e pode ser visto no começo da aba *RF data* na Figura 10; ele tem comprimento de 4 bytes, que somados aos 30 bytes de dados resulta em 34 bytes de informação enviados por pacote. A especificação do tempo não foi enviada junto com os valores amostrados do sensor pois a interface do *XCTU* já fornece esta informação.

A Figura 11 ilustra o processo de leitura e envio de dados através do fluxograma. Somente após 5 amostragens dos sensores serem agrupadas (dados que compõem o pacote *API*) o pacote é enviado. A explicação da escolha das cinco interações, sua utilidade e o funcionamento da Frequência Emulada é descrita no capítulo 4.1.

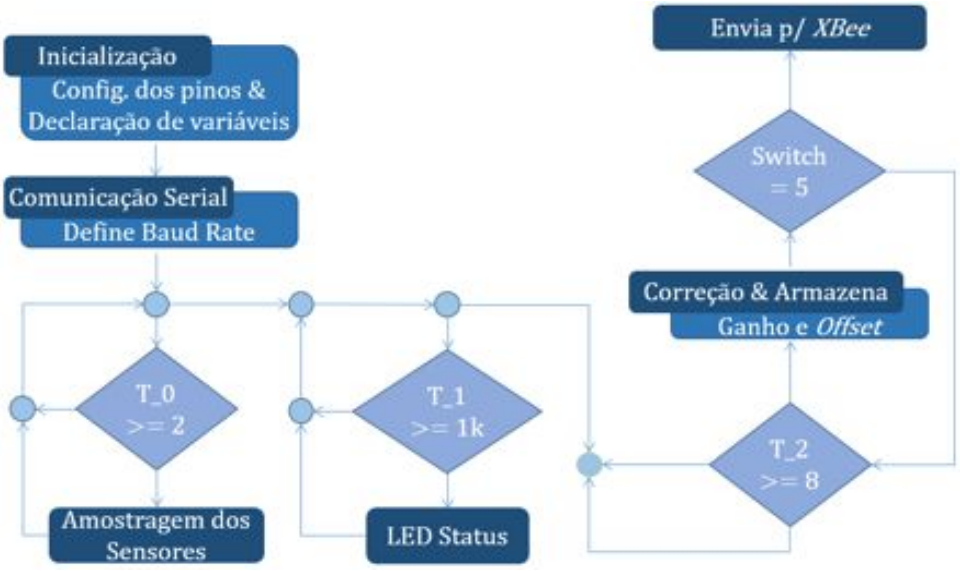


Figura 11 – *Ilustração da amostragem dos três eixos, são esperadas cinco interações para o posterior envio das informações agrupadas para forma o pacote API.*

Os dados recebidos pelo *XCTU* são armazenados em um arquivo de extensão *.log*, estes arquivos podem ser abertos com o bloco de notas e sua estrutura é mostrada na Figura 12.

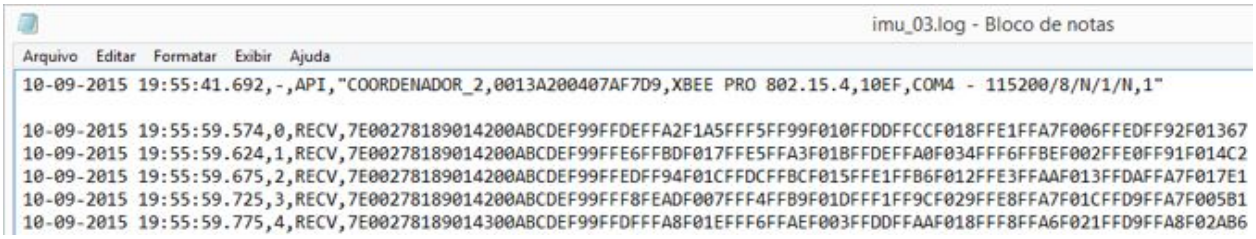


***Figura 12** – Arquivo .log onde são armazenados os dados coletados com o ArduIMU. Pode-se perceber que cada linha corresponde a um pacote API. A linha começa com a data e hora de recebimento do pacote. São apresentados também a identificação da ordem de recebimento, o estado (recebido ou enviado) e finalmente a sequência de bytes do pacote.*

O código (Anexo B3) para o desencapsulamento dos pacotes *API* foi escrito em *MATLAB*. Este código importa o arquivo *.log* para o ambiente do *MATLAB* e o separa em duas matrizes, uma com a informação do tempo e outra com os dados. Esta separação é feita com os delimitadores de tabulação, vírgula e o delimitador especial ABCDEF99 enviado junto com o pacote *API.* Posteriormente a matriz de dados é desmembrada mais uma vez e organiza os dados em quatro matrizes diferentes conforme o endereço fonte do *XBee* que transmitiu os dados. Em cada uma destas matriz os valores amostrados de cada um dos três eixos são separados em três colunas, respectivamente, para os eixos X, Y e Z.

3.3. Calibrações estática e dinâmica dos sensores

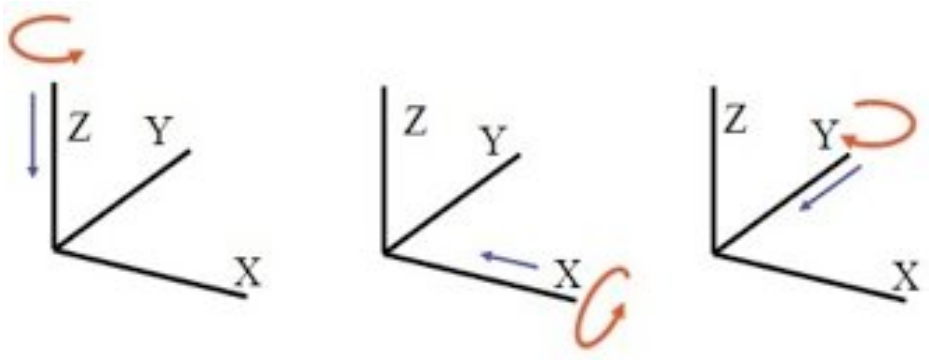
A utilização de qualquer tipo de sensor requer procedimentos de calibração, que são responsáveis por adequar a aquisição de dados dos sensores à situação em que estes serão empregados, fazendo com que suas medições sejam fidedignas à realidade. Esta seção descreve os procedimentos que foram utilizados para a calibração do *ArduIMU*.

3.3.1. Calibração estática do giroscópio

Para calibração do giroscópio, conforme (MARCIAS *et al*., 2012), foram adquiridos os valores de velocidade angular com o *ArduIMU* parado na posição inicial. Deste modo as medidas obtidas para cada um dos eixos com o sistema estático representam o *offset* da velocidade angular. Este *offset* é obtido pela Equação (8) quando amostrados os valores do giroscópio.

$$Calibração = valores\ amostrados - offset \quad (8)$$

A calibração foi feita momentos antes da execução dos movimentos específicos no espaço e ocorreu da seguinte forma: antes de começar a coleta do movimento permaneceu-se na posição inicial durante 30 segundos. Após este intervalo inicial procedeu-se com a execução do movimento. Na etapa de processamento dos dados foi calculada a média dos valores amostrados durante estes 30 segundos iniciais. Então, o valor desta média foi subtraído da matriz de dados do movimento subsequente para remover o valor de *offset* conforme descrito na Equação (8). Desta forma a posição inicial do movimento sempre é a origem do sistema de rotações mostrado na Figura 13, ou seja, com o sentido de giro na direção anti-horária.



(a) *(b)* *(c)*

***Figura 13** – Sentidos de giro do ArduIMU (indicados pelas setas laranja): (a) eixo Z, (b) eixo X e (c) eixo Y. (as setas azuis indicam o sentido de observação dos sentidos de giro).*

Com os dados de velocidade angular instantânea do giroscópio foi realizado o procedimento de integração para obtenção da posição angular, permitindo o conhecimento da quantidade de graus que o sensor rotacionou em cada um dos eixos em relação a posição inicial.

3.3.2. Calibração estática do acelerômetro

O procedimento de calibração estática que foi utilizado nos sensores do *ArduIMU* foi semelhante ao desenvolvido em (CORREA, 2015) e é descrito na documentação de um circuito embarcado comercial da *Texas Instruments* (MARCIAS *et al.*, 2012), que também é um *IMU*. No procedimento descrito faz-se uso da amostragem da aceleração empregada pela gravidade, tanto para a correção do ganho quanto para a compensação na defasagem da leitura do ponto de zero ou *offset*. A placa foi posicionada de modo a ficar perpendicularmente

alinhada com seus eixos de medição, um de cada vez, com o eixo da gravidade. As direções de cada eixo do *ArduIMU* estão impressas na própria placa e podem ser vistos da Figura 14.

***Figura 14** – Informação de orientação dos eixos de medição do acelerômetro e giroscópio (MPU-6000).*
***FONTE:** https://s3.amazonaws.com/3dr.production/294/large/KT-ArduIMU-30-4.jpg?1441214785), livre adaptação.*

Os valores de *offset* e ganho são calculados como descrito pelas Equações (9) e (10) respectivamente

$$Offset = \frac{dados_{máx} + dados_{mín}}{2} \tag{9}$$

$$Ganho = \frac{valor\ esperado}{dados_{máx} - Offset} \tag{10}$$

onde $Offset$ é o valor a ser empregado na compensação de defasagem do ponto zero, $dados_{máx}$ é a leitura corresponde a uma gravidade no eixo que aponta na mesma direção da gravidade, $dados_{mín}$ é a leitura correspondente a uma gravidade no eixo que aponta na direção contrária a gravidade. $Ganho$ é a correção a ser aplicada no ganho dos sensores, $valor\ esperado$ é o valor equivalente a uma gravidade pois é a medida que espera-se encontrar.

A partir do cálculo dos valores de *Offset* e *Ganho* calculados, pode-se aplicar as correções de calibração nos valores amostrados pelo sensor de aceleração com a Equação (11).

$$Calibração = Ganho * (valores\ amostrados - Offset) \tag{11}$$

onde a $Calibração$ é aplicada em todos os valores amostrados pelos sensores (índice $valores\ amostrados$).

Para o correto posicionamento da placa, o *ArduIMU* foi preso a uma morsa com 6 graus-de-liberdade, podendo assim ser ajustada para os diferentes posicionamentos dos eixos.

O nivelamento na morsa foi feito com um nível e inclinômetro da BOSCH e pode ser visto na Figura 15. A placa presa na morsa pode ser vista na Figura 16.

Figura 15 – *Detalhe do nível e inclinômetro utilizado para nivelamento da morsa na fixação da placa do ArduIMU durante a calibração dos eixos do acelerômetro.*

Figura 16 – *Detalhe da placa do ArduIMU fixada na morsa (perpendicular ao eixo da gravidade para cada um dos eixos do acelerômetro).*

Quando posicionado desta forma, as leituras feitas em cada um dos eixos alinhados com a direção da aceleração da gravidade devem apresentar valores iguais a uma gravidade, que com seu valor truncado equivale a 9,81 m/s² ou, conforme informado no manual do MPU-6000 e a faixa de seleção utilizada, apresentar o valor em bytes de 4096. Os valores de *Offset* e *Ganho* foram calculados como descrito nas Equações (9) e (10).

Após o *ArduIMU* ser corretamente posicionado e fixado pela morsa, deu-se início a coleta dos dados para cada um dos seis diferentes posicionamentos. Foram adquiridos 30 segundos de dados e então calculada sua média para compor os valores $dados_{máx}$ e $dados_{mín}$. Com os valores de *Offset* e *Ganho* calculados, um novo código foi gravado no *ArduIMU* (Anexo A - Aba 2: *Calibration.h*), agora contemplando as correções aplicadas por estes valores e descritos pela Equação (11) conforme instruído na página do fabricante.

Os valores de *Ganho* e O*ffset* são calculados com o código descrito no Anexo B1 a partir das medidas coletadas com o procedimento descrito acima.

3.3.3. Calibração dinâmica do acelerômetro

Conforme visto previamente, os movimentos realizados com os membros superiores analisados neste trabalho acontecem numa faixa restrita de medidas em termos de deslocamento, velocidade e aceleração. Portanto para fins de verificação da resposta dos sensores no domínio da frequência pode-se trabalhar com uma faixa de 0 a 50Hz, pois é necessário realizar a calibração dinâmica destes sensores, que serão utilizados na avaliação dos movimentos, verificando sua resposta em frequência e observando se apresentam um comportamento linear.

Uma configuração experimental que permite fazer essa verificação é submeter o sensor a um movimento conhecido (nesse caso uma oscilação senoidal, com frequência e amplitude conhecidas), monitorar sua saída, e verificar sua curva de resposta em frequência.

Para esse procedimento experimental foram usados os seguintes equipamentos:

- Mesa vibratória: dispositivo eletromecânico que converte a energia elétrica em energia mecânica, através de campos magnéticos, na forma de deslocamentos controlados em frequência e amplitude na direção vertical. A mesa vibratória utilizada foi do modelo St5000/300 fabricada pela *VEB*;
- Gerador de funções analógico: equipamento responsável por fornecer o formato, frequência e amplitude do sinal desejado (referência para as vibrações da mesa). O gerador utilizado foi do modelo FG-200D fabricado pela *Dawer*;

- Amplificador de potência: amplifica o sinal especificado pelo gerador de funções para possibilitar a aplicação do mesmo na mesa vibratória. O amplificador utilizado foi do modelo DBK 6000 fabricado pela *Ciclotron*.
- Acelerômetro de referência piezoelétrico uni-axial: modelo *DeltaTron* 4514-B-004 fabricado pela *Brüel & Kjaer*;
- Transdutor de posição potenciométrico (*LVDT*): modelo PY-2-F-025-S01M fabricado pela *GEFRAN*.
- Plataforma de aquisição: para a coleta e gravação em um computador dos dados amostrados foi utilizada uma plataforma de aquisição com conversores de sinais analógico para digital. As plataformas utilizadas foram dos modelos NI 9234 para o acelerômetro *DeltaTron* e NI 9205 para o transdutor de posição, ambas fabricadas pela *National Instruments.*

Um esquema do conjunto de equipamentos utilizados é apresentado na Figura 17. No gerador de funções são escolhidos: o formato, frequência e amplitude da onda (movimento) de interesse. O sinal determinado com o gerador passa pelo amplificador de potência e é então aplicado na mesa vibratória. As placas do *ArduIMU* (blocos em roxo – Fig.18) foram fixadas à mesa com fita dupla-face. O acelerômetro *DeltaTron* (círculo azul – Fig.18) possuí uma cavidade rosqueada onde foi possível sua fixação com parafuso ao eixo central da mesa. O transdutor de posição potenciométrico (blocos verde – Fig.18) foi cuidadosamente fixado a uma haste com altura ajustável de modo a ficar com metade de seu curso utilizado em contato com a mesa em repouso. Desta forma pode-se aplicar amplitudes de até ± 10mm (20mm pico-pico) para não comprometer o sensor atingindo o fim de seu curso utilizável.

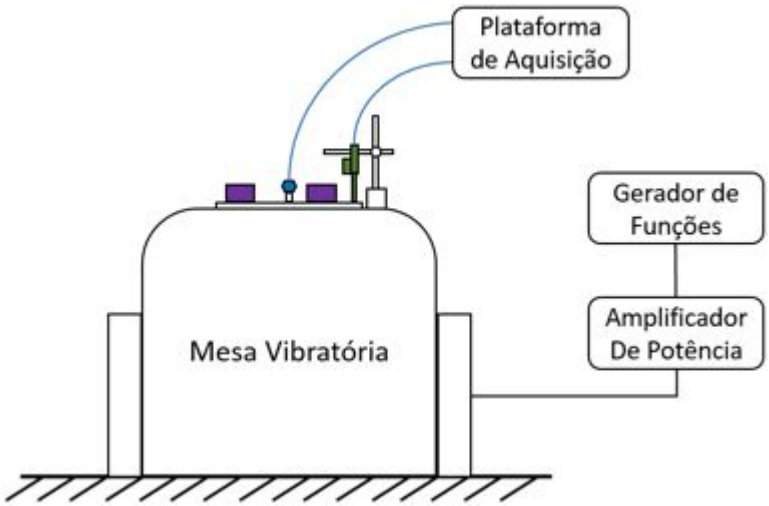


Figura 17 – *Esquema do procedimento experimental (e respectivos equipamentos) utilizados para calibração dinâmica.*

As características do acelerômetro de referência e do transdutor de posição (LVDT) são apresentadas nas Tabelas 5 e 6, respectivamente.

Tabela 5 – *Especificação do acelerômetro uni-axial DeltaTron 4514-B-004 da Brüel & Kjaer.*

Característica	**Valor**
Sensibilidade no eixo de medição [mV/g]	50,05
Faixa de Frequência [Hz]	1~10k
Linearidade	±0,2%
Sensibilidade Transversal ao eixo	< 5%
Faixa de medida [g]	±100 (pico)

FONTE: Datasheet DeltaTron 4514-B-004, Brüel & Kjaer, 2009.

Tabela 6 – *Especificação do transdutor de posição potenciométrico (LVDT) PY-2-F-025-S01M da GEFRAN.*

Característica	**Valor**
Curso utilizável [mm]	25
Resistência [kΩ]	1
Frequência de Ressonância [kHz]	32
Sensibilidade Transversal ao eixo	< 5%
Faixa de vibrações [Hz]	5~2k

FONTE: Datasheet PY2 series, GEFRAN, 2015.

Os sensores foram fixados na mesa vibratória juntamente com as unidades *ArduIMU*. Foram executados 10 conjuntos de testes para cada eixo de medição de aceleração do

ArduIMU em cinco valores de frequência (10, 20, 30, 40 e 50Hz,) e em duas amplitudes (5 e 15mm).

A fim de utilizar as unidades *ArduIMU* para monitoramento de posição é necessário aplicar métodos de integração numérica nas séries discretas de dados. Esses métodos podem acumular erros e degradar o sinal, fazendo com que os dados da posição obtida das sequências de cálculo de integração da aceleração, não correspondam à localização real do objeto em estudo (membros superiores). Para verificar o efeito destas integrações nos dados de aceleração do *ArduIMU*, foi utilizado o transdutor de posição (*LVDT*) na configuração da Figura 17 para comparação entre os dados.

O transdutor de posição potenciométrico é responsável por amostrar os deslocamentos em amplitude das oscilações senoidais aplicadas à mesa vibratória. Este sensor foi utilizado em paralelo com a placa do *ArduIMU* para a aquisição dos dados, e integrando duas vezes os dados de aceleração do *ArduIMU* obtêm-se os valores de posição que podem então ser comparados com os obtidos pelo *LVDT*.

3.4. Sistema de Referência *FOB* (*Flock of Birds*)

O *FOB* pode ser configurado para se adequar a diversas aplicações diferentes, desde uma única unidade com um único sensor até configurações mais complexas, com várias combinações de transmissores e sensores (*Ascension Technology,* Manual do Usuário disponibilizado em CD). A Tabela 7 apresenta as especificações técnicas do *FOB*.

Tabela 7 – *Especificações técnicas do sistema FOB de 6 graus-de-liberdade (Ascension Technology, Manual do Usuário disponibilizado em CD).*

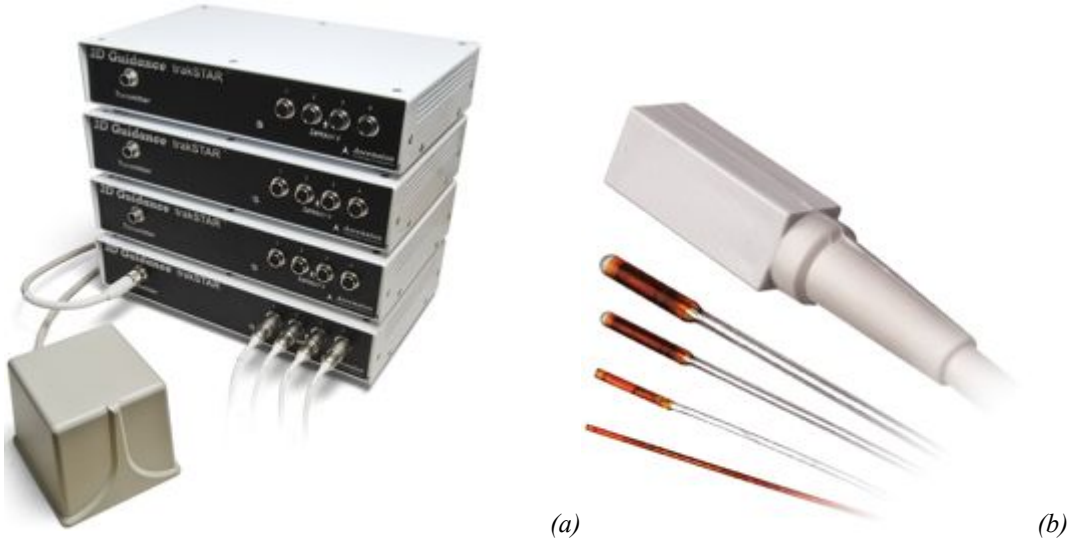
Especificações Físicas	
Transmissor	9,52cm³ (montado no interior do encapsulamento) c cabo de 3m
Sensor	2,54 x 2,54 x 2cm cabo de 3m
Encapsulamento do transmissor	24,13 x 29,21 x 6,6cm
Especificações Técnicas	
Alcance de posição	±14,6m em qualquer direção
Alcance angular	±180° rotações nos eixos X e Y
	±90° rotações no eixo Z
Precisão em posição estática	0,1778cm RMS*

*Continuação da **Tabela 7** – Especificações técnicas do sistema FOB de 6 graus-de-liberdade (Ascension Technology, Manual do Usuário disponibilizado em CD).*

Resolução de posição	**0,0762cm @ 30,48cm**
Precisão em angulação estática	0,5° RMS*
Resolução de angulação	0,1° RMS @ 30,48cm
Taxa de atualização	100 medições/segundo
Saídas	X, Y, Z coordenadas de posição & ângulos de orientação, matriz de rotação, ou quatérnions
Interface	RS232: *Baud rate* de 2.400 ou 115.200 RS485: *Baud rate* de 57.600 ou 500.000
Formato	Binário
Especificações Elétricas	
Requerimentos de alimentação	+5 VDC @ 2.45A méd., 3.85A pico +12 VDC @ 0.53A méd., 0.63A pico -12 VDC @ 0.34A méd., 0.46A pico

**Precisão verificada para a faixa de (20,3cm a 76,2cm) em orientação constante.*

A Figura 18 mostra unidades de rastreamento com um transmissor e alguns modelos de sensores.



(a) *(b)*

***Figura 18** – (a) Unidades de rastreamento sensor e transmissor do FOB (3D Guidance trakSTAR), (b) sensores FOB de diferentes tamanhos.*

***FONTE:** http://www.ascension-tech.com/wp-content/themes/atc/images/atc_stack.jpg*

3.5. Comparação dos movimentos realizados com os sistemas *FOB* e *ArduIMU*

3.5.1. Configuração do Experimento

O sistema *FOB* foi montado numa mesa experimental em madeira que permite ao sujeito sentado apoiar os membros superiores e realizar movimentos na superfície horizontal (plano XY). O sistema conta com quatro sensores magnéticos, sendo usualmente empregados dois sensores para cada membro superior (2 segmentos: braço e antebraço; e lados esquerdo e direito) para medição do movimento de alcance dos membros superiores. Por isso foram testadas quatro unidades *ArduIMU* para a coleta dos dados simultânea dos dois sistemas. No caso de movimentos unimanuais usa-se duas unidades *ArduIMUs*.

Em estudos de avaliação de movimentos de alcance (BAGESTEIRO and SAINBURG, 2002) utiliza-se o sistema numa estação experimental e configuração do sujeito como a vista na Figura 19. O sujeito que participa do experimento senta em uma cadeira ajustável com um ou os dois braços apoiados por talas de PVC imobilizadoras (fixando a articulação do punho) combinadas com um sistema de jatos de ar para reduzir o atrito e os efeitos da gravidade em uma superfície horizontal para movimentos no plano XY. A posição e orientação de cada segmento são amostrados usando os sensores magnéticos *FOB*, posicionados no antebraço e no braço, totalizando quatro sensores para ambos os membros superiores. Os sensores são posicionados de forma a ficarem aproximadamente no centro de cada segmento.

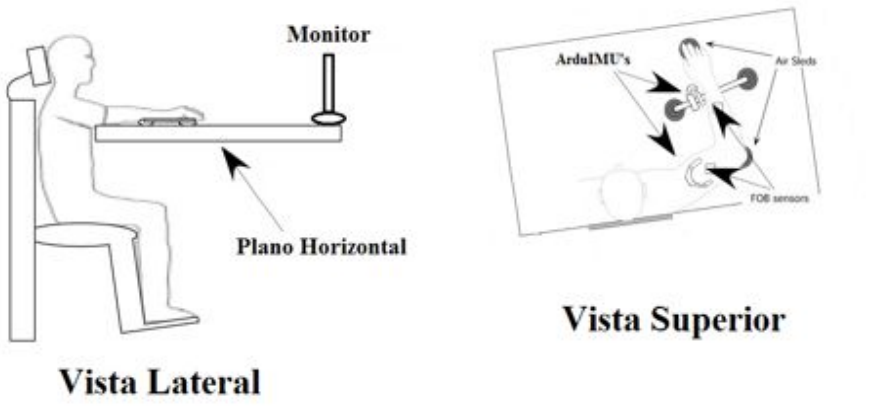


Figura 19 – *Estação experimental para medição do movimento de alcance dos membros superiores.*
FONTE: *BAGESTEIRO and SAINBURG, 2002, livre adaptação.*

3.5.2. Descrição dos movimentos realizados para a comparação entre os sistemas *FOB* e *ArduIMU*

Para o estudo de verificação e avaliação dos sistemas colocou-se o sistema *ArduIMU* e *XBee* colado com fita dupla-face exatamente sobre os sensores magnéticos do *FOB*. A bateria ficou fixada com fita dupla face em uma das hastes de plástico da tala imobilizadora. As Figuras 20 e 21 mostram a fixação da caixa plástica com *ArduIMU* e *XBee* e a bateria nos sensores *FOB* acoplados nas partes plásticas.

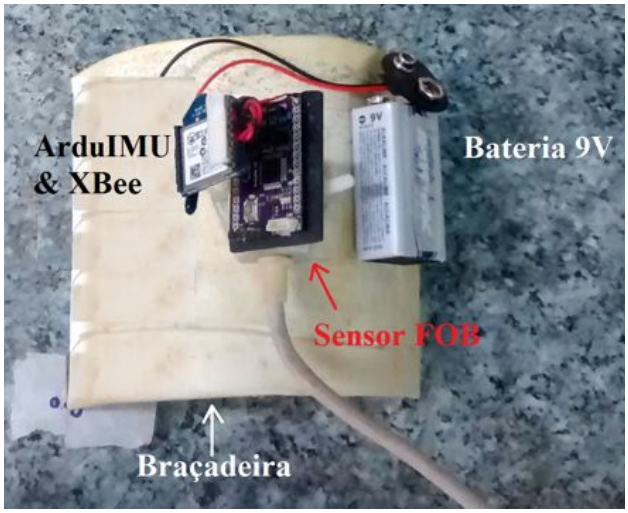
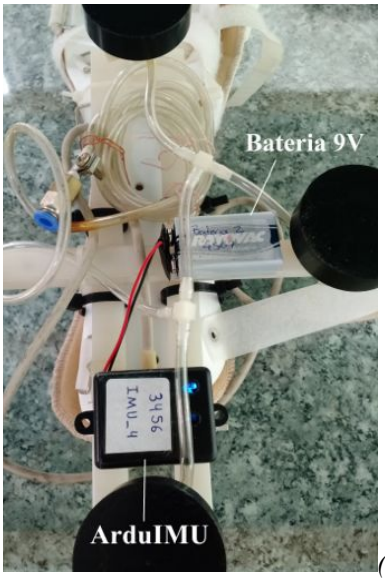
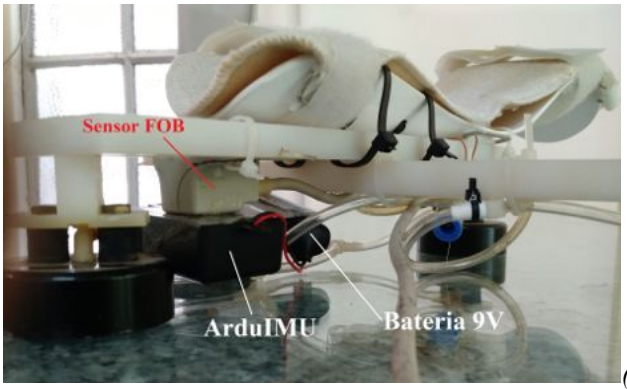


Figura 20 – *Fixação do ArduIMU na braçadeira de plástico ajustável utilizada para fixar um dos sensores FOB no braço do sujeito.*



(a)

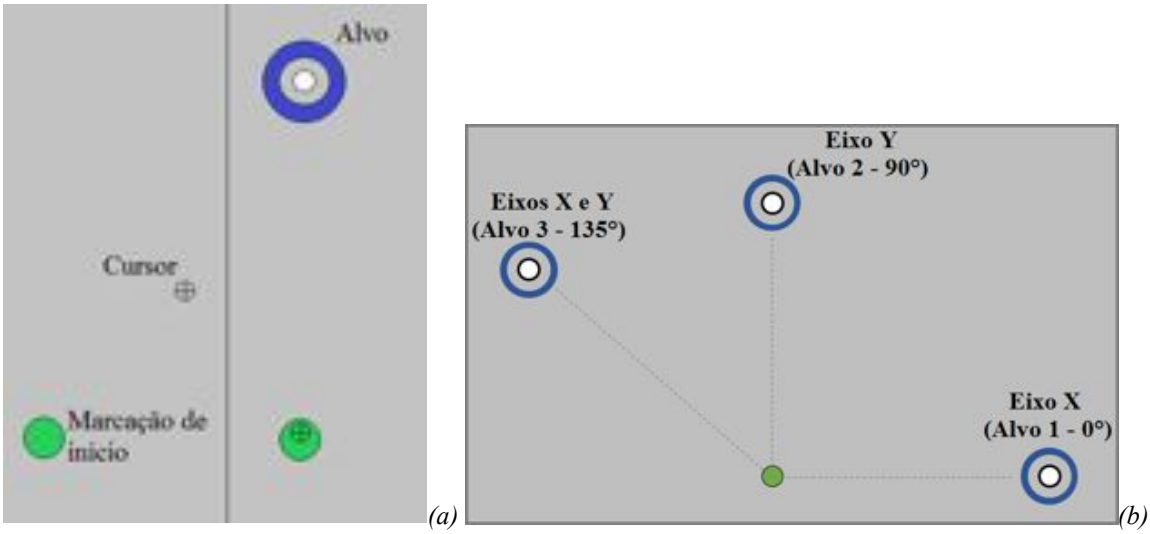


(b)

Figura 21 – *Duas vistas da fixação do conjunto ArduIMU, XBee e bateria na tala imobilizadora.*

Para estudo comparativo dos dados utilizou-se uma série de movimentos padrões e executados em duas condições diferentes. Usando o software gerenciador do *FOB* (desenvolvido especificamente para estudos dos movimentos dos membros superiores - BAGESTEIRO e SAINBURG, 2002) e seu sistema de alvos foram feitas coletas das unidades inerciais *ArduIMU*.

Para os testes foram configurados três alvos que descrevem um comprimento de 30cm: um alvo em 0° (movimento somente no eixo X); um alvo a 90° (movimento somente no eixo Y); e um alvo a 135° (movimento com componentes no plano XY). A configuração de alvos pode ser vista na Figura 22.



(a) *(b)*

***Figura 22** – (a) Detalhe da tela de apresentação do software gerenciador do FOB; (b) Posição dos três alvos para comparação dos sistemas.*

A orientação dos eixos XYZ para os sistemas *FOB* e *ArduIMU* (no plano da mesa experimental) é visto na Figura 23.

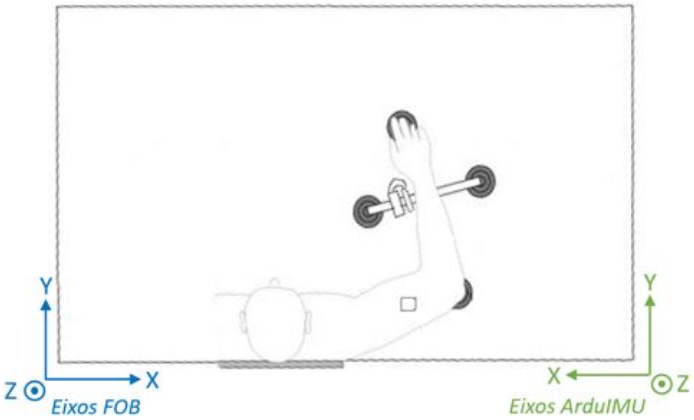


***Figura 23** – Orientação dos eixos XYZ dos sistemas FOB e ArduIMU na mesa experimental.*

***FONTE:** BAGESTEIRO e SAINBURG, 2002, livre adaptação.*

As aquisições dos movimentos padrões (ilustrados na Figura 24) para teste com todas as unidades inerciais do *ArduIMU* foram: (1) movimento somente com componente X dos dois sistemas (alvo 1) (linha vermelha); (2) movimento somente na componente Y dos dois sistemas (alvo 2) (linha amarela); (3) movimento com as componentes X e Y dos dois sistemas (alvo 3) (linha branca); (4) movimento somente na componente Z dos dois sistemas (haste em azul de 30cm).

Estes quatro movimentos foram repetidos em duas condições diferentes: *FOB* e *ArduIMU* operando em paralelo e somente com *ArduIMU* adquirindo os dados (*FOB* desligado).

***Figura 24** – Representação dos movimentos realizados nos testes das unidades ArduIMU (componente Z (linha azul), componente Y (linha vermelha), componente X (linha amarela), componentes X e Y (linha branca)).*

Com os dados coletados de ambos os sistemas, foi realizado o procedimento de integração numérica. Os dados de aceleração adquiridos com o *ArduIMU* são integrados duas vezes a fim de obter-se o deslocamento (dados de posição) correspondente do movimento realizado. A média e os desvios padrão das tentativas são calculadas através do código descrito no Anexo B2. Depois destes cálculos os dados do *ArduIMU* são processados (códigos dos Anexos B3 e B4) e comparados com a faixa de desvios padrões obtidos do *FOB*

(código do Anexo B5) para os respectivos movimentos; e então são calculadas as máximas diferenças entre essas medidas para fins de comparação entre os sistemas.

Além desses movimentos, outras duas combinações de movimento no espaço foram testadas somente com as unidades *ArduIMU*. O primeiro consiste em mover o conjunto de braço-antebraço percorrendo a diagonal principal de um cubo com aresta de 30cm, resultando em uma distância percorrida de 52cm (reta roxa - Figura 24). O segundo movimento é uma trajetória em formato de S para frente. Inicia-se com o braço direito em repouso apoiado no plano da mesa na marcação de 0,0 (marcação verde - Figura 25) deslocando o braço para a direita em um movimento de semicírculo com sua face côncava apontando para a esquerda (curva vermelha - Figura 25), em seguida desloca o braço para a esquerda desenhando outro semicírculo, porém agora com a sua face côncava apontando para a direita (curva amarela - Figura 25). Ao longo deste movimento também é necessário aumentar gradualmente a altura do braço de modo que ao final do movimento atinja-se a altura de 30cm (haste azul - Figura 25). O movimento foi repetido também com o braço esquerdo. Foram coletadas quatro sessões de 10 tentativas para cada membro superior. O movimento foi realizado com o conjunto de quatro *ArduIMUs*, *XBees* e baterias de 9V presos no braço e antebraço esquerdo e direito com uma fita específica para este fim (ver Figura 26). No total foram utilizados 4 sensores inerciais *ArduIMU* trabalhando em conjunto durante esta aquisição.

Figura 25 – *Ilustração dos movimentos no espaço 3D com coleta de dados somente das unidades ArduIMU.*

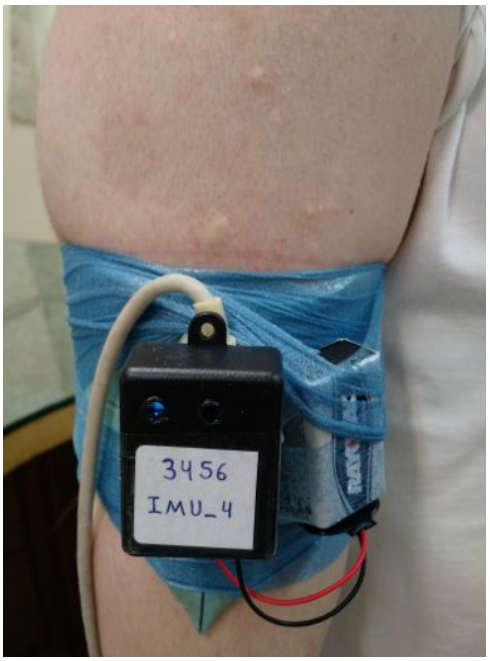


Figura 26 *– Posicionamento e fixação do conjunto (caixa plástica com ArduIMU e XBee e bateria de 9V) no braço do sujeito com fita específica para este fim.*

3.6. Reconstrução dos segmentos corporais

Conforme mencionado anteriormente (seção 2.2), para a reconstrução do membro superior é usado um modelo simplificado de 2 segmentos: (1) segmentos do braço (articulação do ombro) e antebraço-mão (articulação do cotovelo e punho). Além dos dados do *ArduIMU* posicionado no segmento (posição e angulação instantâneas) é necessário o comprimento de cada segmento. Como exemplo, pode-se obter esta informação medindo (com uma fita métrica) o comprimento do segmento do antebraço-mão (distância entre o centro da articulação do cotovelo - região proximal e a ponta do dedo médio - região distal). Segundo dados antropométricos (ROBERTSON *et al*., 2004), o centro de massa deste segmento é obtido pelas Equações (12) e (13).

$$R_{proximal} = 0{,}682 * comprimento\ do\ segmento \quad (12)$$

$$R_{distal} = 0{,}318 * comprimento\ do\ segmento \quad (13)$$

onde $R_{proximal}$ e R_{distal} são as posições do centro de massa do antebraço partindo-se das posições das regiões proximal e distal, respectivamente. Para o indivíduo do nosso exemplo tem-se um comprimento (entre cotovelo e ponta do dedo médio) de 47cm, a posição proximal do centro de massa do segmento antebraço-mão, está localizada em $R_{proximal} = 0{,}682 *$

$47 = 32{,}05cm$. Já a posição distal do centro de massa, está localizada em $R_{distal} = 0{,}318 * 47 = 14{,}95cm$, que somadas resultam no comprimento do segmento antebraço-mão de 47cm.

De forma análoga as medidas antropométricas foram usadas para a reconstrução do segmento do braço (entre as articulações do ombro e cotovelo). O centro de massa deste segmento tem sua posição conforme indicado nas Equações (14) e (15).

$$R_{proximal} = 0{,}436 * comprimento\ do\ segmento \quad (14)$$

$$R_{distal} = 0{,}564 * comprimento\ do\ segmento \quad (15)$$

Com a informação da localização do cotovelo e ponta do dedo médio em relação ao posicionamento do sensor, pode-se calcular a posição de reconstrução destes pontos com base nos dados de posição obtidos do sensor. Para isto utiliza-se as Equações de (2) a (7).

As posições para a correta reconstrução do segmento do braço (ombro-cotovelo) é obtida de forma análoga ao segmento do antebraço (cotovelo-dedos) porém substituindo-se adequadamente os valores para o segmento em questão.

4. RESULTADOS E DISCUSSÕES

4.1. Testes de Perdas de Pacotes

Com a utilização do recurso de transmissão de dados pelos módulos *XBee* é possível que ocorram perdas das informações enviadas por algum motivo, seja ele limiar do alcance de transmissão, bateria fraca ou concorrência do envio de pacotes entre os diversos módulos inseridos na rede. O *ArduIMU* determina a taxa de amostragem dos sensores e também a taxa com a qual os pacotes dos dados amostrados são enviados. Como estas duas taxas são fixas o coordenador deve receber o mesmo número de pacotes que são enviados pelo transmissor; e caso esta condição não seja atendia caracteriza-se a perda de pacotes.

Para a realização dos testes foram estabelecidas diversas configurações combinando diferentes valores de: número de pacotes enviados por segundo, tamanho destes pacotes e número de dispositivos (*ArduIMU* e *XBee*) operando simultaneamente. Os testes consistiram de 8 ensaios para cada configuração, tendo cada ensaio uma duração de 60 segundos. Foram testadas as frequências de transmissão de dados de 200Hz, 150Hz, 125Hz e 100Hz. Os resultados completos para as frequências de 200Hz, 150Hz e 100Hz foram omitidos para não estender este tópico além do necessário, sendo que a Tabela 8 apresenta somente os piores resultados encontrados. Os resultados obtidos para a frequência de transmissão escolhida de 125Hz são apresentados na Tabela 9 (configuração escolhida destacada em azul).

A Tabela 9 apresenta 7 colunas com os dados de: (1) *IMU's On Net,* (2) *Pacotes por Segundo [Hz],* (3) *Interações/Envio,* (4) *Leituras/Interação,* (5) *Tamanho do Pacote [Bytes],* (6) *Média de Pacotes Perdidos por Segundo e* (7) *Máxima Perda de Pacotes por Segundo.*

A coluna (1) *IMU's On Net* representa o número de dispositivos operando na rede simultaneamente; nas linhas que apresentam um asterisco ao lado da quantidade de unidades foram feitas duas redes de *XBees* com dois coordenadores. A separação em duas redes mostrou-se eficiente quando na utilização de 4 módulos simultaneamente e em especial quando a frequência de amostragem é maior do que 150Hz.

A coluna (2) *Pacotes por Segundo [Hz]* representa a taxa de pacotes enviados em um segundo com os dados armazenados de cinco interações. Constatou-se que para pacotes acima de 80 bytes de dados e frequências acima de 40Hz na utilização com mais de uma unidade operando simultaneamente os *XBees* não trabalham de forma correta, dando prioridade a uma das unidades e perdendo completamente os pacotes das demais unidades. Em função disto utilizou-se o recurso de armazenamento parcial dos dados na memória interna do

microprocessador. Desta forma ao invés de as informações dos sensores serem enviadas no exato momento em que são amostradas, armazenam-se cinco leituras de cada eixo no microprocessador, e somente depois destas cinco interações o pacote é enviado. Este método permite utilizar uma determinada frequência de amostragem para os sensores mas emula uma frequência de transmissão de dados que é cinco vezes menor do que a frequência de amostragem, possibilitando assim a operação de forma correta de mais de uma unidade simultaneamente.

A coluna (3) *Interações/Envio* representa o número de interações a serem esperadas até o envio dos dados de um pacote, portanto é o divisor da frequência de amostragem para o cálculo da frequência emulada.

Em (4) *Leituras/Interação* tem-se o número de leituras que são feitas após o total de interações. Quando o número de *Leituras/Interação* é menor do que o número de *Interações/Envio*, existem interações restantes que são contadas somente para a divisão da frequência emulada. Isto foi feito para testar pacotes de diferentes tamanhos de informação para uma mesma frequência emulada.

Em (5) *Tamanho do Pacote [Bytes]* tem-se o tamanho total de cada pacote que é enviado; o modo de transmissão dos *XBees* permite pacotes com o tamanho máximo de 100 bytes.

A coluna (6) *Média de Pacotes Perdidos por Segundo* e a (7) *Máxima Perda de Pacotes por Segundo* representam os valores da média dos 8 ensaios e a máxima taxa de perda de pacotes dentre os 8 ensaios, respectivamente.

Tabela 8 – *Piores resultados de perdas de pacotes para as frequências de 100, 150 e 200Hz.*

Frequência [Hz]	IMU's On Net	Média de Pacotes Perdidos por Segundo	Máxima Perda de Pacotes por Segundo
200	1	21,67	24,95
	2	12,25	15,37
	4	7,91	10,23
	4*	4,36	6,13
150	1	15,14	19,58
	2	9,57	12,37
	4	6,54	9,58
	4*	3,98	5,79

*Continuação da **Tabela 8** – Piores resultados de perdas de pacotes para as frequências de 100, 150 e 200Hz.*

100	**1**	**13,21**	**17,67**
	2	7,98	9,53
	4	5,74	8,67
	4*	2,87	4,25

***Tabela 9** – Resultados obtidos nos testes de perdas de pacotes para a frequência de 125Hz.*

IMU's On Net	Pacotes por Segundo [Hz]	Interações/ Envio	Leituras/ Interação	Tamanho do Pacote [Bytes]	Média de Pacotes Perdidos por Segundo	Máxima Perda de Pacotes por Segundo
1	25	5	5	34	0,37	0,54
1	25	5	4	28	0,28	0,43
1	25	5	3	22	0,23	0,36
1	31,35	4	4	28	0,32	0,51
1	31,35	4	3	22	0,27	0,46
1	41,67	3	3	22	0,31	0,57
1	125	1	1	10	10,91	12,30
2	25	5	5	34	1,74	2,21
2	25	5	4	28	1,65	2,12
2	25	5	3	22	1,47	1,95
2	31,35	4	4	28	1,70	2,17
2	31,35	4	3	22	1,63	1,99
2	41,67	3	3	22	1,66	2,04
2	125	1	1	10	13,87	16,53
4	25	5	5	34	5,83	6,14
4	25	5	4	28	4,62	5,89
4	25	5	3	22	4,25	5,37
4	31,35	4	4	28	5,13	6,43
4	31,35	4	3	22	4,97	5,85
4	41,67	3	3	22	5,08	5,97
4	125	1	1	10	19,35	21,82
4*	25	5	5	34	2,15	2,96
4*	25	5	4	28	1,86	2,74
4*	25	5	3	22	1,79	2,65
4*	31,35	4	4	28	2,22	3,02
4*	31,35	4	3	22	1,83	2,69
4*	41,67	3	3	22	2,17	2,86
4*	125	1	1	10	15,75	17,68

Com a escolha da frequência de amostragem a 125Hz decidiu-se por enviar 5 leituras de cada um dos três eixos do acelerômetro por pacote, contendo 34 bytes de informação. Esta

forma (mesmo não sendo a configuração otimizada para a menor perda de pacotes possível) foi escolhida pois mantém a possibilidade de serem acrescentados os dados do giroscópio e do magnetômetro, pois sendo o limite de envio de informações por pacote *API* de 100 bytes o comprimento final com os dados de acelerômetro, giroscópio e magnetômetro totalizam 94 bytes [(30*3) + 4 = 94].

A perda de pacotes transmitidos pelos *XBees* tem um impacto negativo na amostragem dos sinais de aceleração. Cada pacote contém os dados armazenados de cinco amostragens sucessivas de cada um dos eixos do sensor. Para compor a frequência de amostragem do sistema multiplica-se o número de pacotes por segundo (pps) por estas cinco amostras. Desta forma tem-se a frequência de amostragem de 125Hz pela Equação (16). Então uma taxa de perda de pacotes igual representa uma redução na frequência de amostragem do sistema de 5Hz, resultando em uma frequência de amostragem total de 120Hz conforme a Equação (17), mais baixa do que a desejada.

$$25_{pps} * 5_{amostragens} = 125Hz \quad (16)$$

$$1_{pps} * 5_{iterações} = 5Hz \therefore 125 - 5 = 120Hz \quad (17)$$

Como a máxima perda de pacotes encontrada durante a utilização de 4 unidades simultaneamente foi de 6,14 pacotes por segundo, a máxima redução na frequência de amostragem do sistema será de 30,7Hz, resultando em uma frequência de amostragem total de 94,3Hz, conforme a Equação (18).

$$6{,}14_{pps} * 5_{amostragens} = 30{,}7Hz \therefore 125 - 30{,}7 = 94{,}3Hz \quad (18)$$

Esta diminuição indesejada da frequência de amostragem pode acarretar numa sub amostragem do fenômeno que se deseja quantificar. Para os movimentos de membros superiores analisados neste trabalho, que contém componentes em frequência abaixo de 50Hz, uma taxa de amostragem recomendada é de até duas vezes este valor, ou seja, de 100Hz. Durante a máxima perda de pacotes encontrada a frequência de amostragem total é de 94,3Hz, abaixo do valor ideal, o que pode degradar a reconstrução digital dos segmentos dos membros superiores. Para evitar este impacto negativo, ao serem utilizadas 4 unidades simultaneamente, deve-se dividir a rede em duas, resultando em uma frequência de

amostragem total de 110,2Hz, visto que a máxima perda de pacotes para 4 unidades operando em duas redes é de 2,96 pacotes por segundo.

4.2. Experimento para verificação dos sensores *ArduIMU* no domínio da frequência

Através do cálculo da transformada de Fourier nos dados coletados pelas unidades *ArduIMU,* verificou-se o correto valor obtido nas diferentes frequências aplicadas durante o experimento de calibração dinâmica. Como exemplo, a Figura 27 mostra as curvas de resposta em frequência obtidas numa mesma amplitude (5mm) para a direção X do *ArduIMU* para as 5 frequências de oscilação testadas.

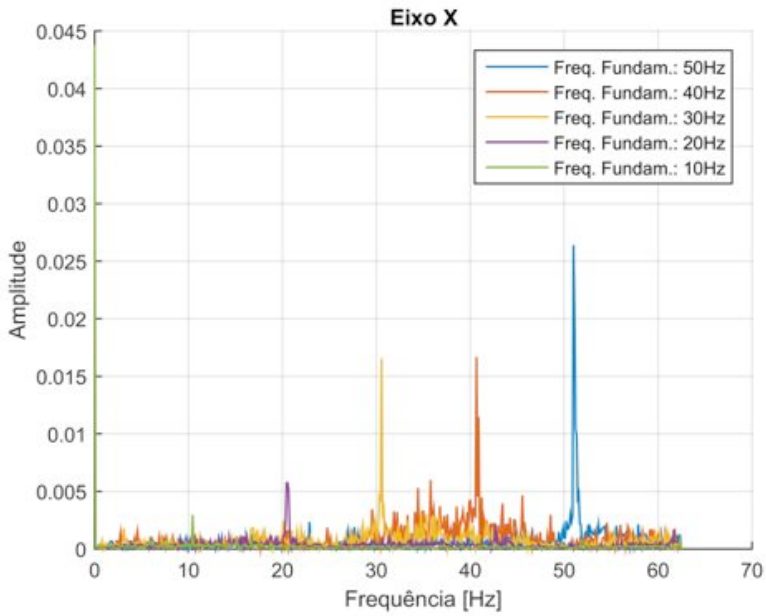
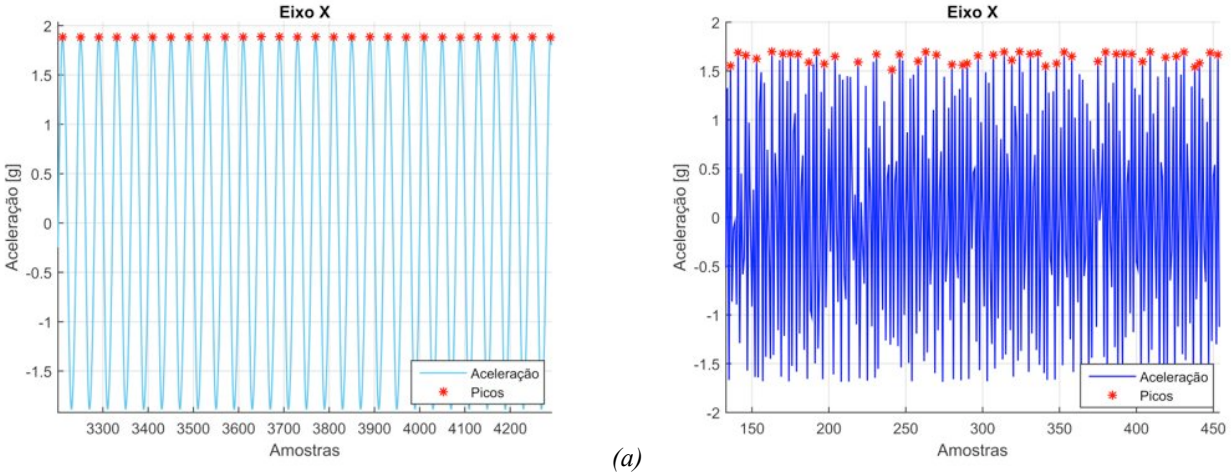


***Figura 27** – Resposta em frequência na direção X do ArduIMU para as5 frequências de oscilação avaliadas.*

Percebe-se pela Figura 27 que as frequências de aceleração adquiridas com o *ArduIMU* correspondem às frequências aplicadas pela mesa vibratória. Os valores de pico encontrados foram: 51,0Hz (curva azul), 40,7Hz (curva laranja), 30,6Hz (curva amarela), 20,4Hz (curva roxa) e 10,4Hz (curva verde).

Para verificação do comportamento em frequência dos sensores de aceleração *ArduIMU* foram comparadas as amplitudes de aceleração adquiridas em cada um dos conjuntos de testes com o valor obtido pelo acelerômetro de referência *DeltaTron* 4514-B-004. Obteve-se a média dos valores da amplitude dos picos de aceleração de cada leitura para

cada um dos sensores (*ArduIMUs* e acelerômetro *DeltaTron*) e dividiu-se os resultados, obtendo-se assim uma proporção entre os dois sistemas. Para verificar se as unidades do *ArduIMU* mantêm uma resposta linear a proporção deve permanecer constante dentro da faixa de frequências empregada nos testes. A Figura 28 ilustra duas curvas de aceleração adquiridas para a frequência de 50Hz na direção X, em *(a)* tem-se a curva do acelerômetro *DeltaTron* (azul claro), e (b) a curva de aceleração de uma das unidades do *ArduIMU* (azul escuro). Os picos são identificados pelos asteriscos em vermelho.



(a) *(b)*

Figura 28 – *Aceleração na frequência de 50Hz para direção X: (a) curva do acelerômetro DeltaTron (azul claro); (b) curva do ArduIMU (azul escuro) (Picos de aceleração marcados em vermelho).*

A Tabela 10 apresenta as máximas diferenças percentuais em aceleração (dentre as cinco frequências avaliadas), quando comparando as medidas de aceleração obtida com acelerômetro *DeltaTron* e as unidades *ArduIMU* nas duas amplitudes.

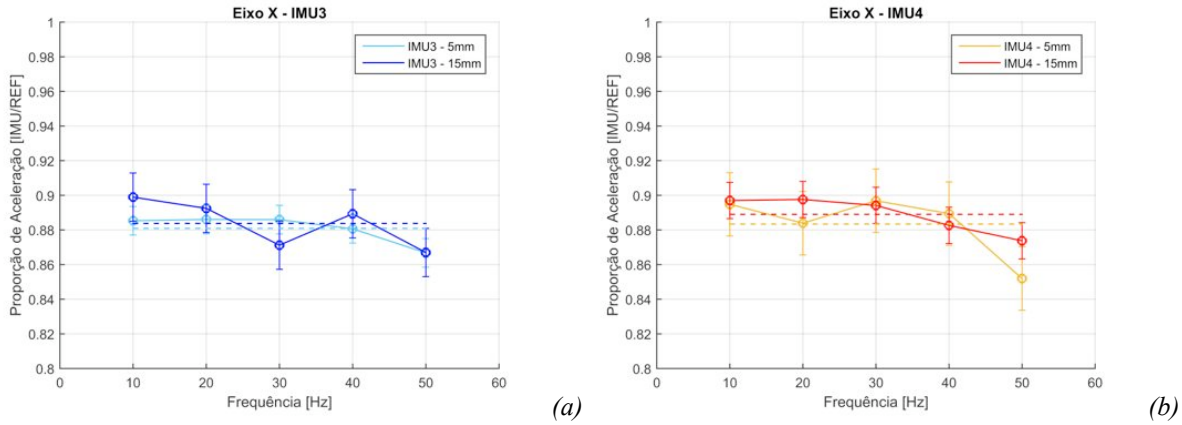
Tabela 10 – *Valores de máxima variação percentual entre as medidas de aceleração do ArduIMU em comparação com a aceleração do acelerômetro DeltaTron.*

Unidade (*ArduIMU*)	**Eixo**	**Amplitude [mm]**	**Máxima variação em aceleração [%]**
IMU_3	X	5	7,5
		15	6,7
	Y	5	7,3
		15	7,4
	Z	5	6,9
		15	6,2

*Continuação da **Tabela 10** – Valores de máxima variação percentual entre as medidas de aceleração do ArduIMU em comparação com a aceleração do acelerômetro DeltaTron.*

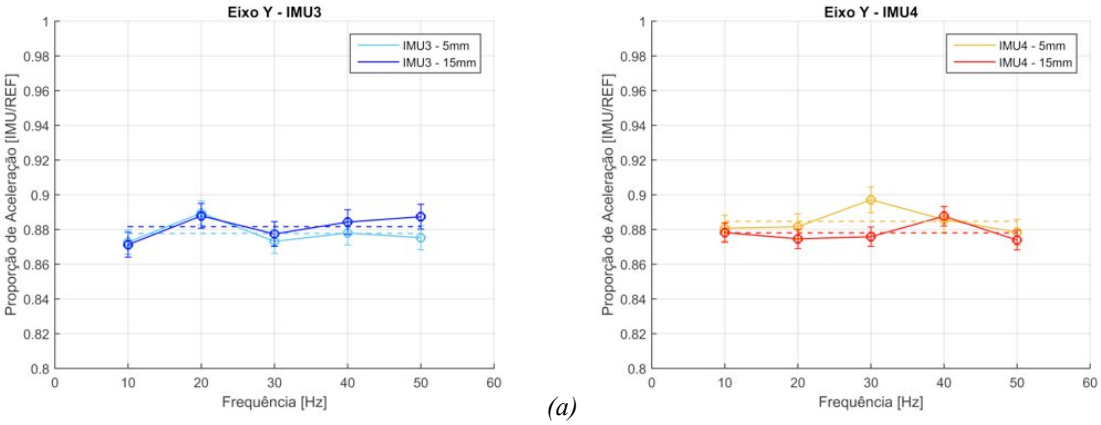
IMU_4	**X**	5	6,8
		15	6,8
	Y	5	6,8
		15	7,4
	Z	5	7,1
		15	5,9

As Figuras 29, 30 e 31 apresentam os resultados para a proporção de amplitude de aceleração calculada entre as unidades do *ArduIMU* e o acelerômetro de referência *DeltaTron* para as cinco frequências testadas e para cada uma das direções (X, Y e Z), respectivamente. As linhas pontilhadas representam a média da proporção de posição e as barras verticais representam os desvios padrão.



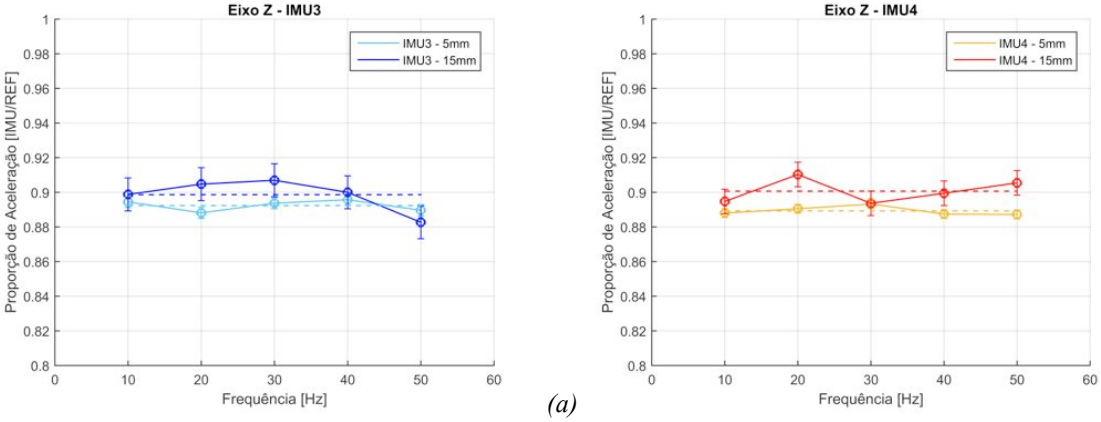
(a) *(b)*

***Figura 29** – Relação de proporção entre as medidas das unidades do ArduIMU e o acelerômetro DeltaTron para a direção X nas duas amplitudes avaliadas: 5 e 15mm.*



(a) *(b)*

***Figura 30** – Relação de proporção entre as medidas das unidades do ArduIMU e o acelerômetro DeltaTron para a direção Y nas duas amplitudes avaliadas: 5 e 15mm.*



(a) *(b)*

***Figura 31** – Relação de proporção entre as medidas das unidades do ArduIMU e o acelerômetro DeltaTron para a direção Z nas duas amplitudes avaliadas: 5 e 15mm.*

A Tabela 11 apresenta as variações máximas das proporções calculadas para cada frequência. Verifica-se que os valores obtidos mantiveram-se igual ou abaixo dos 5% para todos os eixos de medição das unidades *ArduIMU* conforme as condições estabelecidas para este ensaio.

Tabela 11 – *Máxima variação em aceleração na resposta em frequência de aceleração do ArduIMU.*

Unidade (*ArduIMU*)	**Eixo**	**Amplitude [mm]**	**Máxima variação em aceleração [%]**
IMU_3	X	5	2,2
		15	3,6
	Y	5	1,9
		15	1,9
	Z	5	1,8
		15	2,7
IMU_4	X	5	5,0
		15	2,6
	Y	5	2,1
		15	1,5
	Z	5	1,7
		15	1,8

Em seguida, para verificação do comportamento em frequência dos sensores *ArduIMU* quando avaliando seus resultados para os dados de posição (deslocamento) é necessária a integração numérica dos dados e então a comparação das amplitudes obtidas com uma medida de referência através do transdutor de posição PY-2-F-025-S01M (*LVDT*).

Como exemplo, é mostrado na Figura 32 o resultado das duas integrações dos dados de aceleração (no eixo X) de uma unidade do *ArduIMU* (curva verde) em comparação com as medidas do LVDT (curva azul) para a frequência de oscilação de 20Hz e amplitude de 5mm.

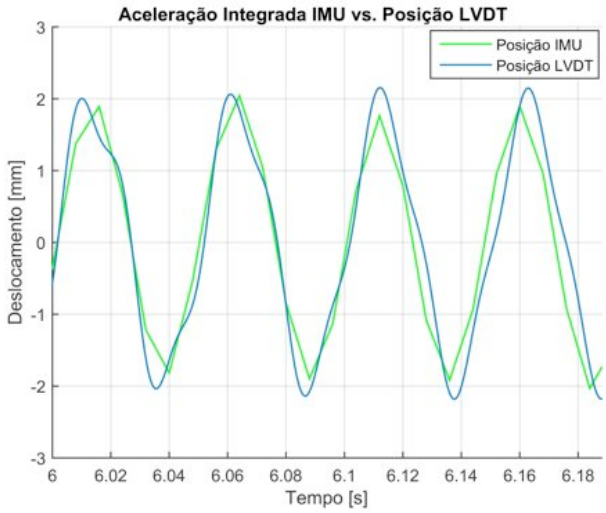
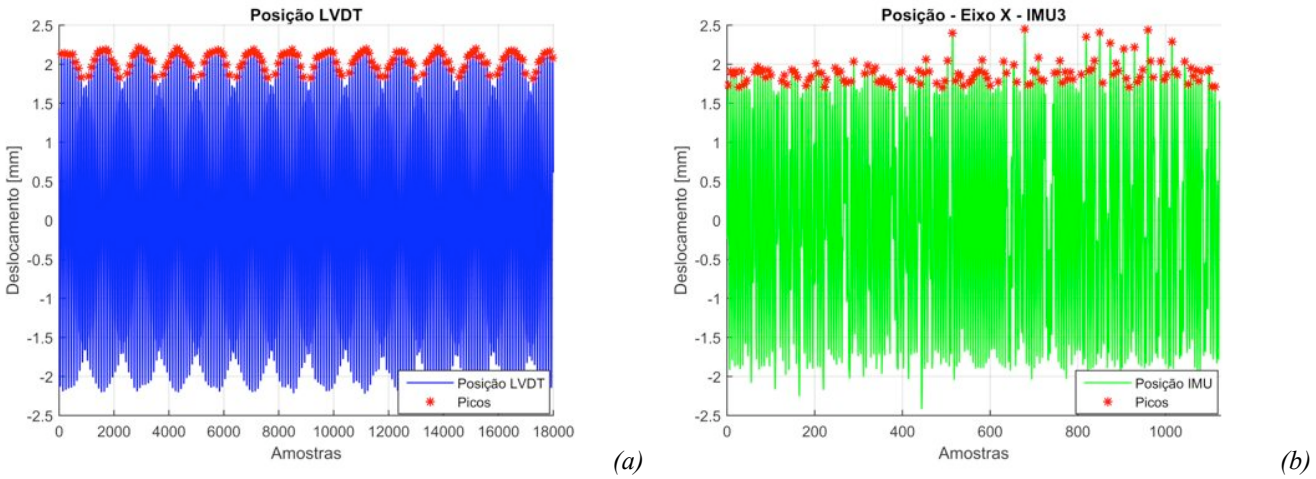


Figura 32 *– Curvas dos dados de posição (integração dupla dos dados de aceleração) do ArduIMU e do LVDT.*

Para realizar a comparação dos dados de posição obteve-se a média dos valores da amplitude dos picos de posição de cada leitura para cada um dos sensores (*ArduIMUs* e transdutor de posição *LVDT*) e dividiu-se os resultados, obtendo-se assim uma proporção entre os dois sistemas e verificando-se se as unidades do *ArduIMU* mantêm uma resposta linear, ou seja, proporção constante dentro da faixa de frequências empregada nos testes. A Figura 33 ilustra duas curvas de posição adquiridas para a frequência de 20Hz e amplitude de 5mm na direção X, em *(a)* tem-se a curva do transdutor de posição (azul), e (b) a curva da aceleração integrada de uma das unidades do *ArduIMU* (verde). Os picos são identificados pelos asteriscos em vermelho.



(a) *(b)*

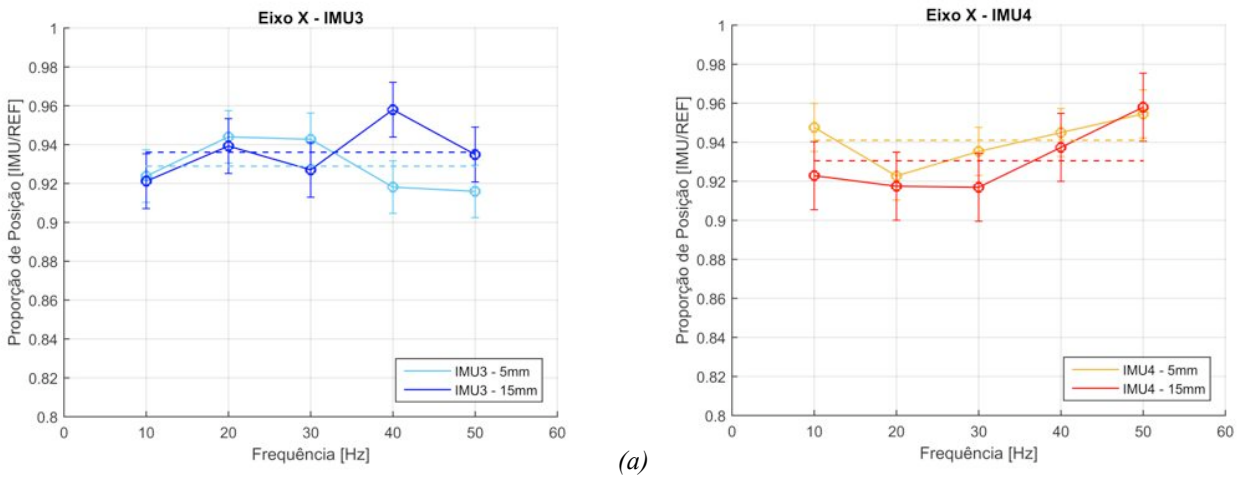
Figura 33 – *Curvas de posição na frequência de 20Hz e amplitude de 5mm para direção X: (a) transdutor de posição (azul); (b) ArduIMU (verde) (Picos marcados em vermelho).*

A Tabela 12 apresenta as máximas diferenças percentuais nos dados de posição, em relação a posição obtida com o transdutor de posição *LVDT*, encontradas para as frequências e amplitudes testadas e para cada uma das unidades do *ArduIMU*.

Tabela 12 – *Máxima variação percentual dos dados de posição do ArduIMU em comparação com a posição do transdutor de posição LVDT.*

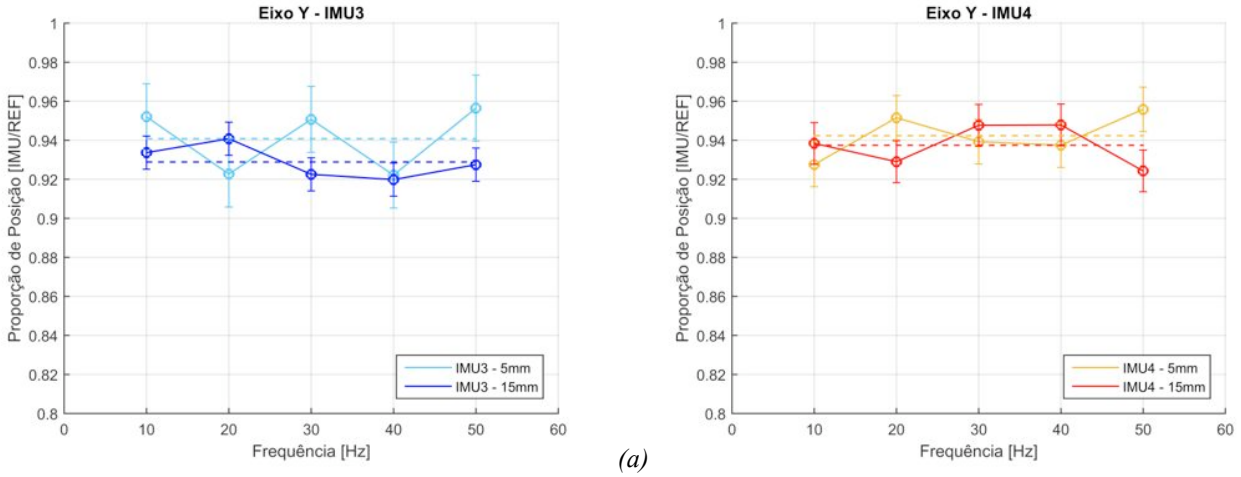
Unidade (*ArduIMU*)	**Eixo**	**Amplitude [mm]**	**Máxima variação em posição [%]**
IMU_3	X	5	5,6
		15	4,2
	Y	5	4,7
		15	5,9
	Z	5	5,1
		15	5,3
IMU_4	X	5	4,5
		15	4,6
	Y	5	4,4
		15	5,2
	Z	5	5,4
		15	6,1

As Figuras 34, 35 e 36 apresentam os resultados obtidos do cálculo da proporção de amplitude de posição entre as unidades do *ArduIMU* e o transdutor de posição *LVDT* para as cinco frequências testadas e para cada uma das direções (X, Y e Z). As linhas pontilhadas representam a média da proporção de posição e as barras verticais representam os desvios padrão.



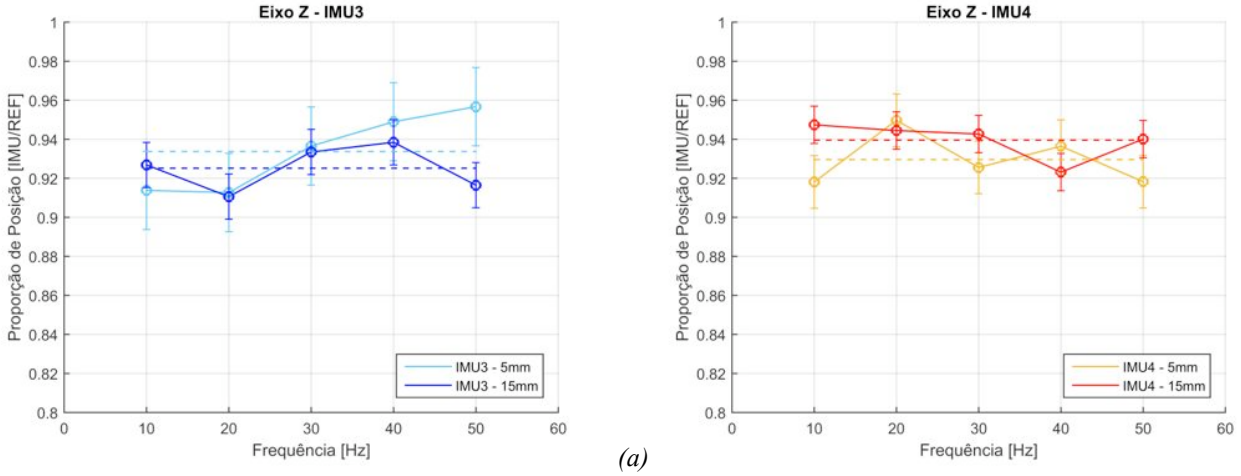
(a) *(b)*

Figura 34 – *Relação de proporção entre as medidas das unidades do ArduIMU e o transdutor de posição LVDT para a direção X nas duas amplitudes avaliadas: 5 e 15mm.*



(a) *(b)*

Figura 35 – *Relação de proporção entre as medidas das unidades do ArduIMU e o transdutor de posição LVDT para a direção Y nas duas amplitudes avaliadas: 5 e 15mm.*



(a) *(b)*

Figura 36 – *Relação de proporção entre as medidas das unidades do ArduIMU e o transdutor de posição LVDT para a direção Z nas duas amplitudes avaliadas: 5 e 15mm.*

A Tabela 13 apresenta as variações percentuais máximas das proporções (posição *ArduIMU* dividida pela posição do *LVDT*) calculadas para cada frequência nos dados de posição. Verifica-se que os valores obtidos mantiveram-se igual ou abaixo dos 4,6% para todos os eixos de medição das unidades *ArduIMU* conforme as condições estabelecidas para este ensaio.

Tabela 13 – *Máxima variação percentual dos dados de posição do ArduIMU em comparação com a posição do transdutor de posição LVDT.*

Unidade (*ArduIMU*)	**Eixo**	**Amplitude [mm]**	**Máxima variação em posição [%]**
IMU_3	X	5	2,9
		15	3,8
	Y	5	3,6
		15	2,2
	Z	5	4,6
		15	3,0
IMU_4	X	5	3,3
		15	4,2
	Y	5	2,9
		15	2,5
	Z	5	3,3
		15	2,6

4.3. Resultados dos testes com movimentos

Com os dados coletados e processados de cada um dos movimentos padrões e situações descritas nas seções 3.5 e 3.6, apresentam-se aqui os resultados encontrados.

A Figura 37 mostra a aceleração bruta coletada com o *ArduIMU* relativa ao movimento padrão no eixo X correspondente ao alvo 1. Esta é a forma característica dos dados antes se serem tratados. Pode-se observar que existem 10 picos na amplitude da aceleração, correspondentes as 10 tentativas dos movimentos executados. A Figura 38 mostra a curva de aceleração (bruta) para uma tentativa (curva vermelha) juntamente a curva de

aceleração filtrada digitalmente através do *MATLAB,* (filtro passa baixas 4ª ordem *Butterworth* - frequência de corte 20Hz).

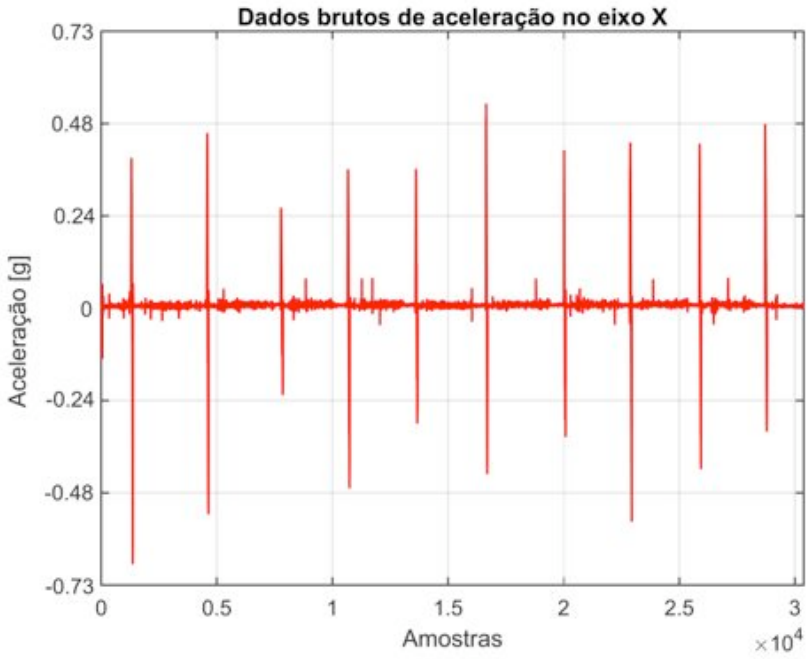


Figura 37 – *Curva característica de aceleração adquirida com a unidade ArduIMU mostrado os 10 picos de aceleração corresponde às 10 tentativas de realização do movimento padrão no eixo X relativo ao alvo 1.*

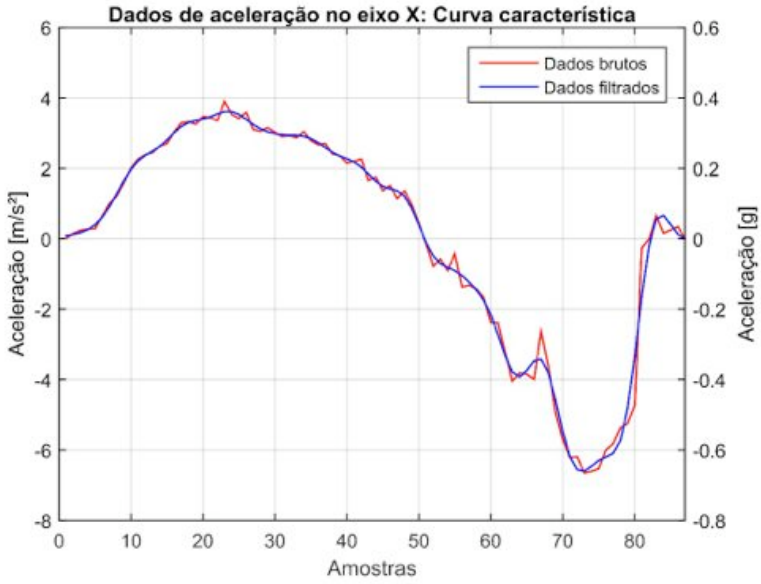
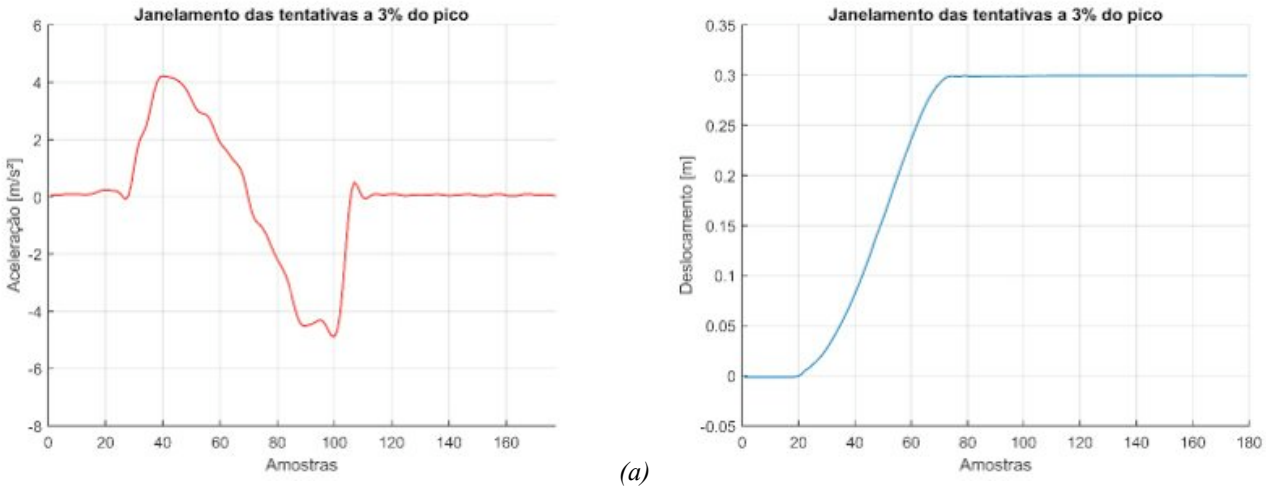


Figura 38 – *Curva de aceleração bruta (vermelha) para uma tentativa e curva dos dados da aceleração filtrados (azul) em 20Hz (filtro passa baixas 4° ordem).*

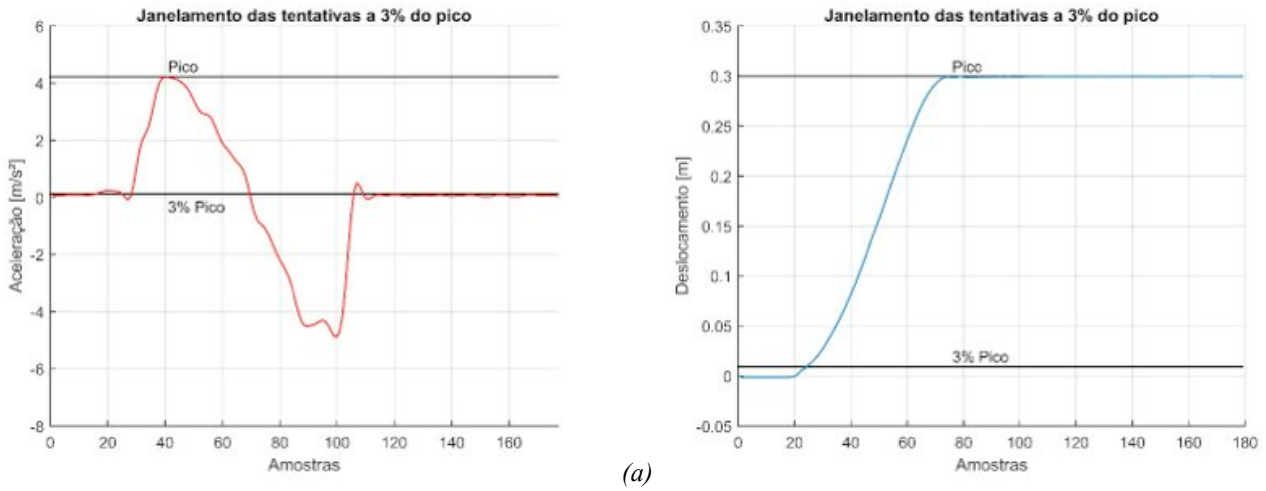
4.3.1. Janelamento a 3% do pico

É necessário um método de janelamento das tentativas sequenciais adquiridas pela unidade *ArduIMU* para a seleção dos trechos de interesse dos sinais adquiridos. O sistema *FOB* já possuí no software gerenciador um algoritmo que separa as tentativas e adquirir somente os dados de interesse do movimento. Normalmente utiliza-se um método de janelamento para definir os pontos de início e fim do movimento. As Figuras de 39 a 42 ilustram o método escolhido. A partir das curvas características de cada sistema (Figura 39), localiza-se o pico local de cada tentativa, identifica-se também, o ponto correspondente ao valor de 3% do pico da respectiva tentativa (característica usada para definir o ponto de início e fim de movimento) (Figura 40). Em seguida obtêm-se os pontos de aceleração correspondentes às intersecções em 3% do pico da curva do sinal (Figura 41), representando-se assim os limites esquerdo e direito (Figura 42) do janelamento da curva de interesse.



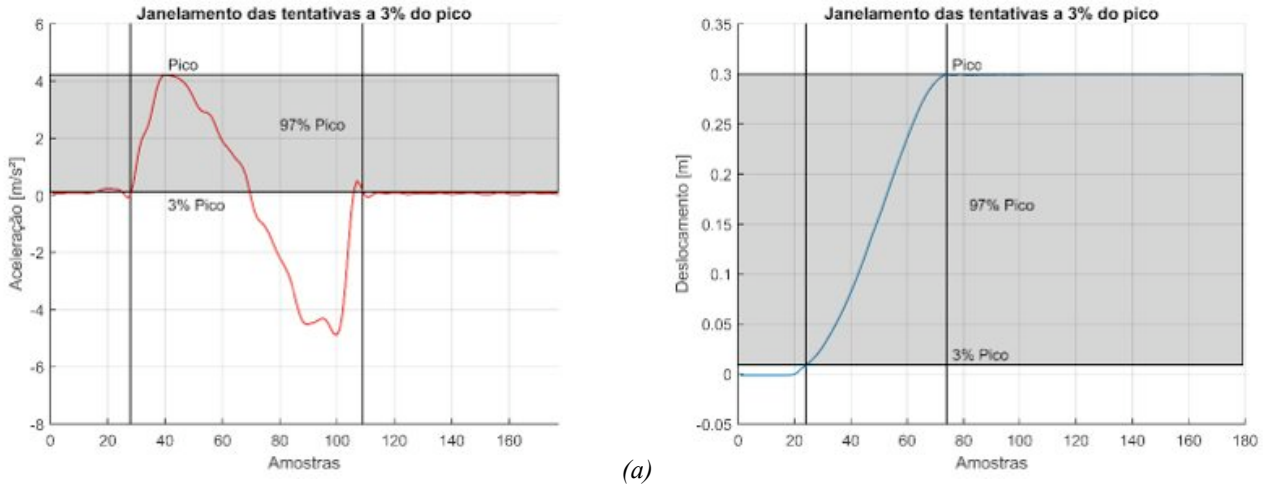
(a) *(b)*

Figura 39 – *Curvas características de aceleração do ArduIMU (a) e posição do FOB (b).*



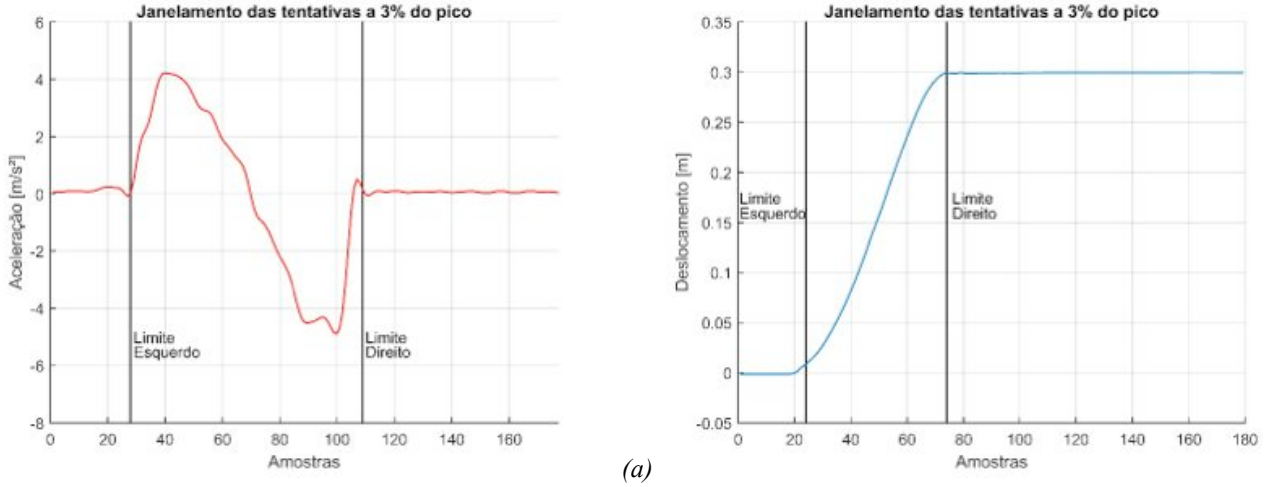
(a) *(b)*

Figura 40 – *Identificação e marcação do ponto equivalente a 3% do pico de cada uma das tentativas. Aceleração do ArduIMU (a) e posição do FOB (b).*



(a) *(b)*

Figura 41 – *Pontos correspondentes 3% do pico. Aceleração do ArduIMU (a) e posição do FOB (b).*



(a) *(b)*

Figura 42 – *Identificação dos limites de início e fim de movimento. Aceleração do ArduIMU (a) e posição do FOB (b).*

Após o processo de janelamento ser aplicado em todas as amostras coletadas, pôde-se iniciar as sucessivas integrações da aceleração do *ArduIMU* para obter os dados de posição. Antes da primeira integração pelo método trapezoidal é removida a média de todas as tentativas janeladas para obter uma série de dados com média zero. Aplicando-se a primeira integração, é obtida a velocidade, e em seguida remove-se a média dos dados janelados para obter uma série de dados com média zero. Aplica-se a segunda integração e são obtidos os dados de posição do movimento. A Figura 43 mostra o resultado da segunda integral. Os processos de integração numérica afetam a amplitude dos dados com a adição de constante não conhecidas. Por este motivo, para realizar a comparação com dados de posição do *FOB*, as séries de dados integradas do *ArduIMU* são normalizadas em amplitude. A normalização do tempo também foi tomada como prática pois os dois sistemas apresentam taxas de amostragem diferentes (103Hz para o *FOB* e 125Hz para o *ArduIMU*). Antes da normalização porém, é necessário remover o *drift* dos dados. *Drift* é uma mudança gradual em qualquer característica quantitativa que deveria permanecer constante (HOLMBERG e ARTURSSON, 2002).

Para a remoção do drift foi utilizada uma aproximação por mínimos quadrados que determinam os coeficientes do polinômio da curva de drift. Conhecidos estes coeficientes, o polinômio foi removido da série de dados. Para os dados janelados de aceleração, a aproximação do *drift* por um polinômio de segundo grau se mostrou eficiente; polinômios de maior ordem foram testados mas não apresentaram diferença ao polinômio de segundo grau.

A Figura 44 ilustra o processo de remoção do *drift* da série de dados da Figura 43, e também a curva do polinômio encontrado (curva rosa) pelo método dos mínimos quadrados e sua equação correspondente.

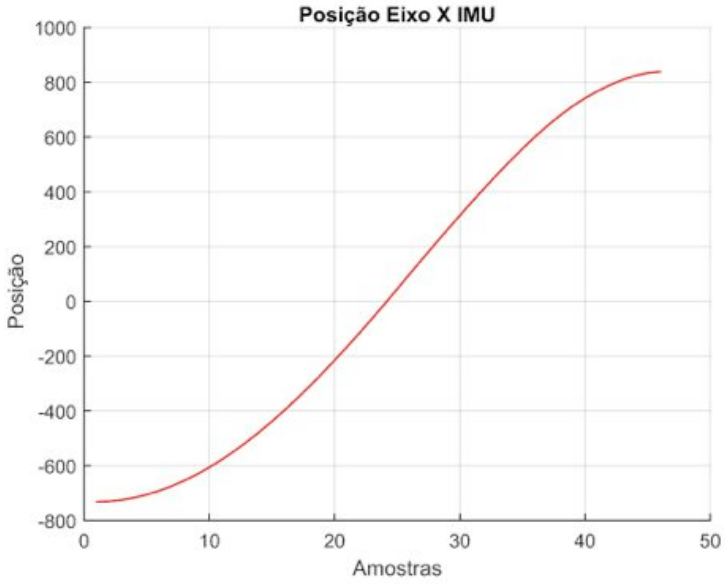


Figura 43 – *Curva Resultante das integrações da aceleração do ArduIMU.*

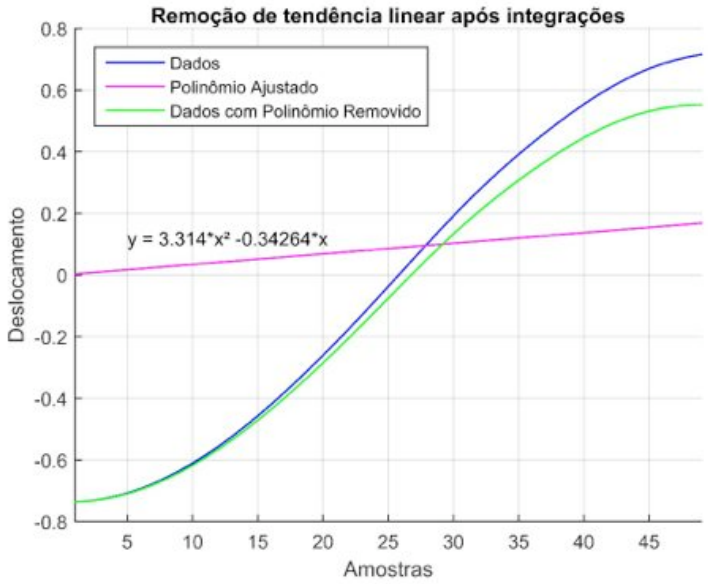
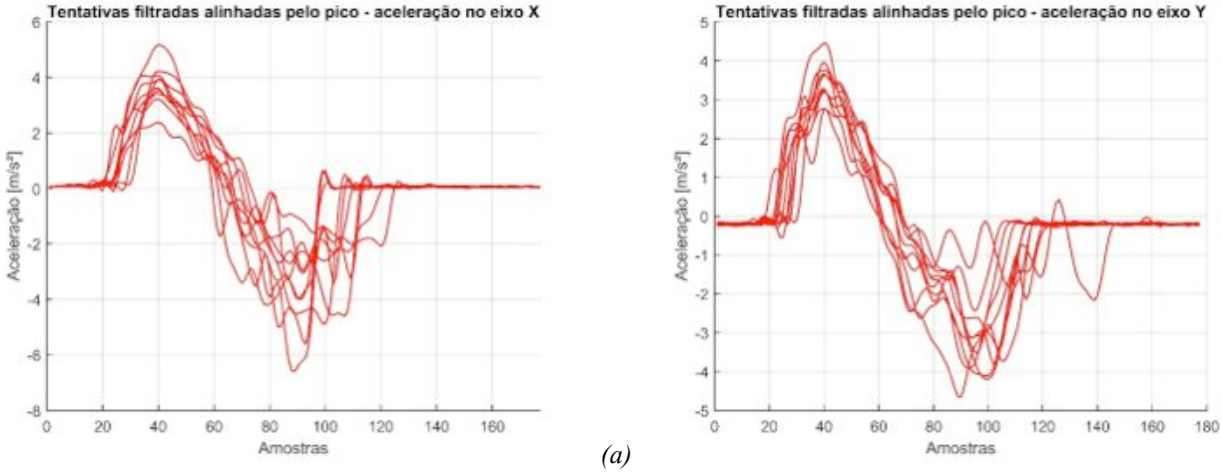


Figura 44 – *Curvas resultantes do Processo de remoção de drift dos dados de aceleração do ArduIMU.*

4.3.2. Movimentos na direção X e na direção Y

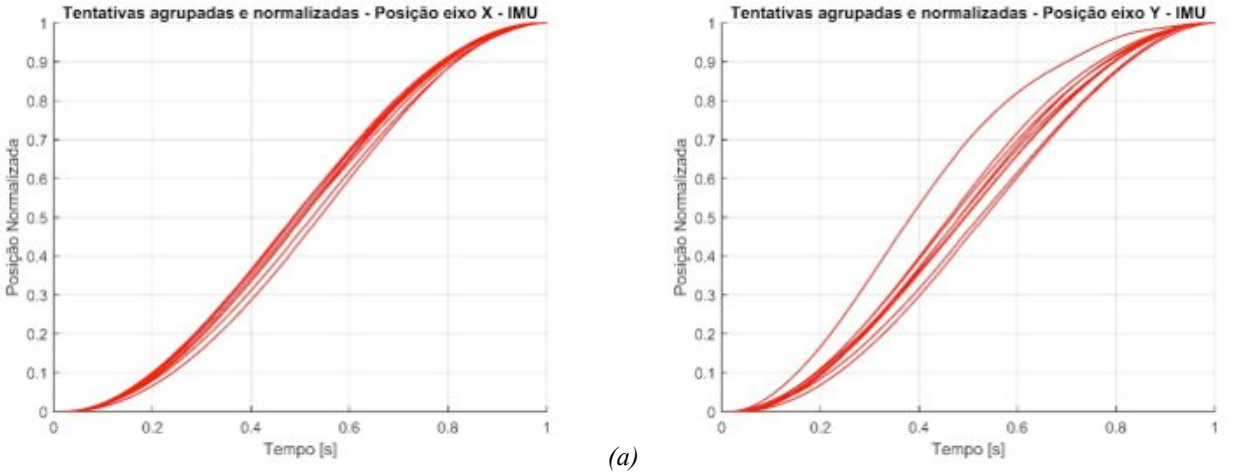
Com estes processos de tratamento dos dados pode-se calcular a posição integrada da aceleração do *ArduIMU.* As Figuras de 45 a 47 mostram o resultado das integrações para os movimentos na direção X (alvo 1) e na direção Y (alvo 2). Foram coletadas 40 tentativas para cada eixo com o sistema *FOB* operando em paralelo com o sistema do *ArduIMU* e mais 40

tentativas somente com o sistema *ArduIMU* (sistema *FOB* desligado). As Figuras de 45 a 47 mostram as 10 tentativas que representam os dados da situação do *ArduIMU* em paralelo com o *FOB*. A Figura 45 mostra as tentativas com repetições agrupadas, filtradas e alinhadas pelo pico. Na Figura 46 são apresentadas as tentativas da posição resultante após as duas integrações e remoção de *drift,* normalizadas em amplitude (dividindo-se pelo pico) e no tempo (dividindo-se pela taxa de amostragem, 125Hz). Na Figura 47 são comparadas as médias destas duas situações.



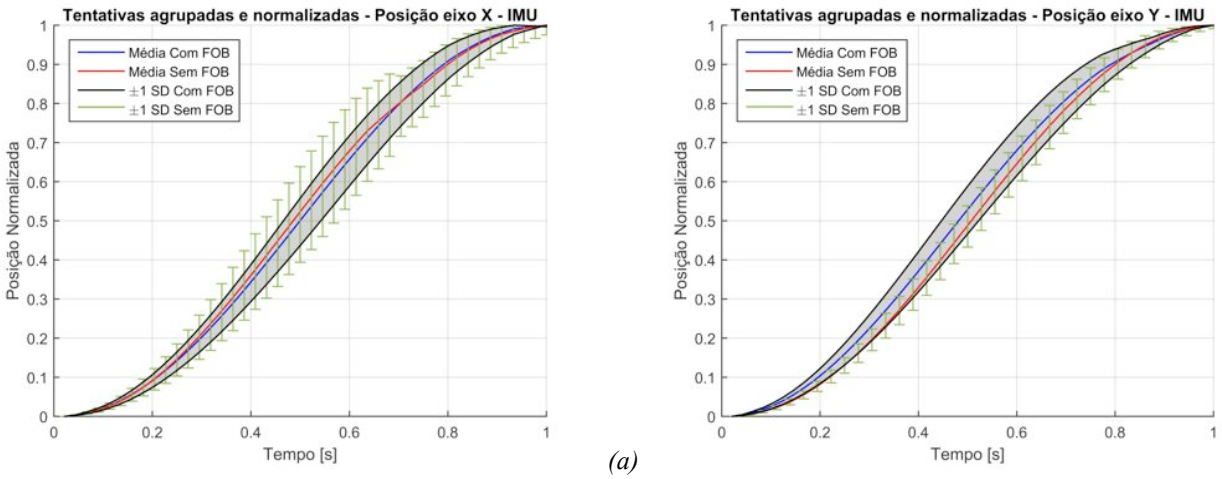
(a) *(b)*

Figura 45 – *Curvas das tentativas agrupadas, filtradas e alinhadas pelo pico. (a) direção X (alvo 1), (b) direção Y (alvo 2).*



(a) *(b)*

Figura 46 – *Curvas dos dados das integrações da aceleração das tentativas agrupadas, filtradas e alinhadas pelo pico. (a) direção X (alvo 1), (b) direção Y (alvo 2).*



(a) *(b)*

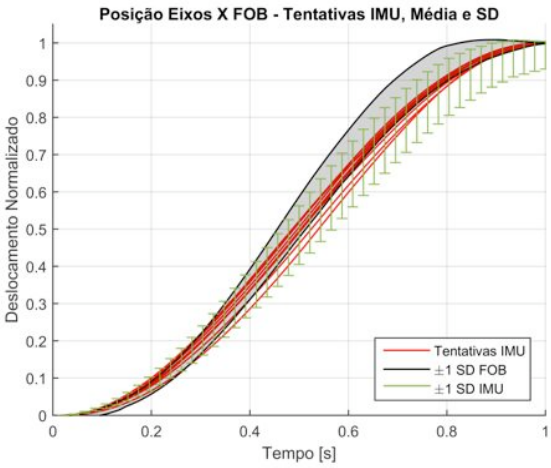
Figura 47 – *Curvas das Médias (e desvios padrão) das tentativas com e sem o sistema FOB rodando em paralelo. (a) direção X (alvo 1), (b) direção Y (alvo 2Y.*

A Tabela 14 mostra as máximas diferenças percentuais encontradas entre as tentativas de posição comparadas nas situações com e sem *FOB* e a média destas.

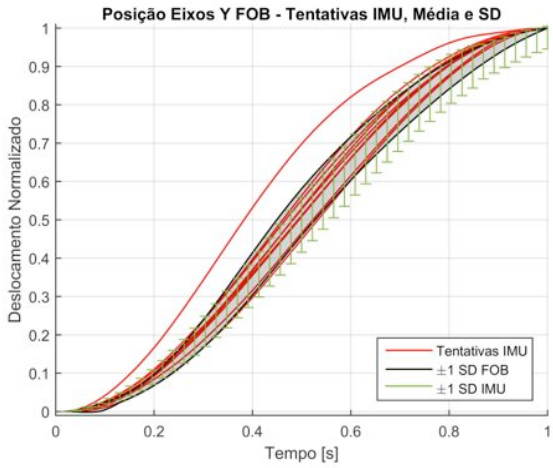
Tabela 14 – *Valores de Máxima diferença percentual entre as curvas e entre a média das curvas de posição com e sem FOB.*

Unidade Inercial	Eixo	Máxima diferença entre tentativas [%]	Máxima diferença na média [%]
IMU_1	X	7,82	6,72
5678	Y	7,98	7,03
IMU_2	X	8,77	7,21
7890	Y	10,13	8,94
IMU_3	X	8,36	7,39
8901	Y	9,13	8,06
IMU_4	X	7,53	6,91
3456	Y	9,74	8,33

As Figuras 48 e 49 as tentativas e média da posição integrada da aceleração do *ArduIMU* são sobrepostas na região delimitada por 1 e 2 desvios padrão, respectivamente, calculados para a posição do *FOB*. Pode-se observar que as curvas das tentativas do *ArduIMU* sobrepostas a 2 desvios padrão estão inseridas quase em sua totalidade nas regiões delimitadas pelos desvios padrões do *FOB*.

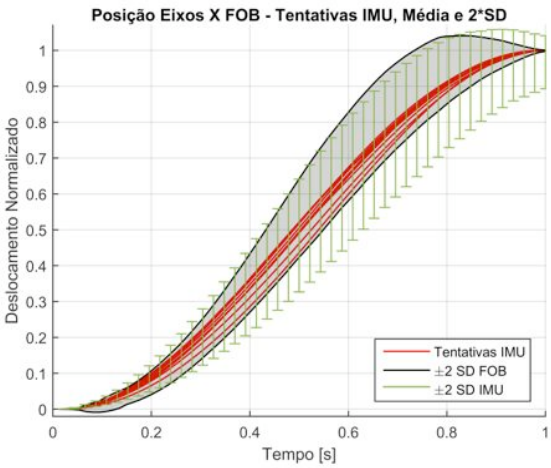


(a)

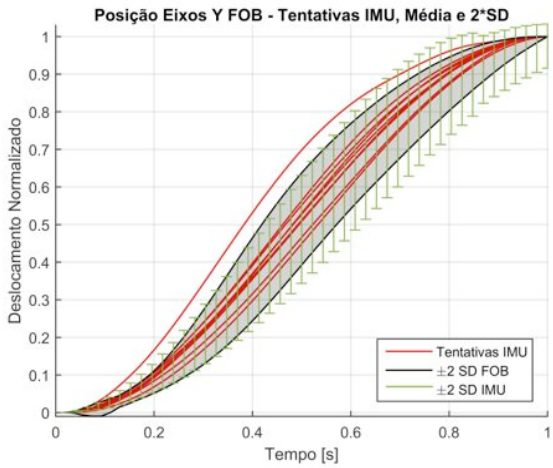


(b)

Figura 48 – *Curvas média das tentativas do sistema ArduIMU em um desvio padrão do sistema FOB. ArduIMU na direção X (a) e direção Y (b).*



(a)



(b)

Figura 49 – *Curvas média das tentativas do sistema ArduIMU em dois desvios padrão do sistema FOB. ArduIMU na direção X (a) e direção Y (b).*

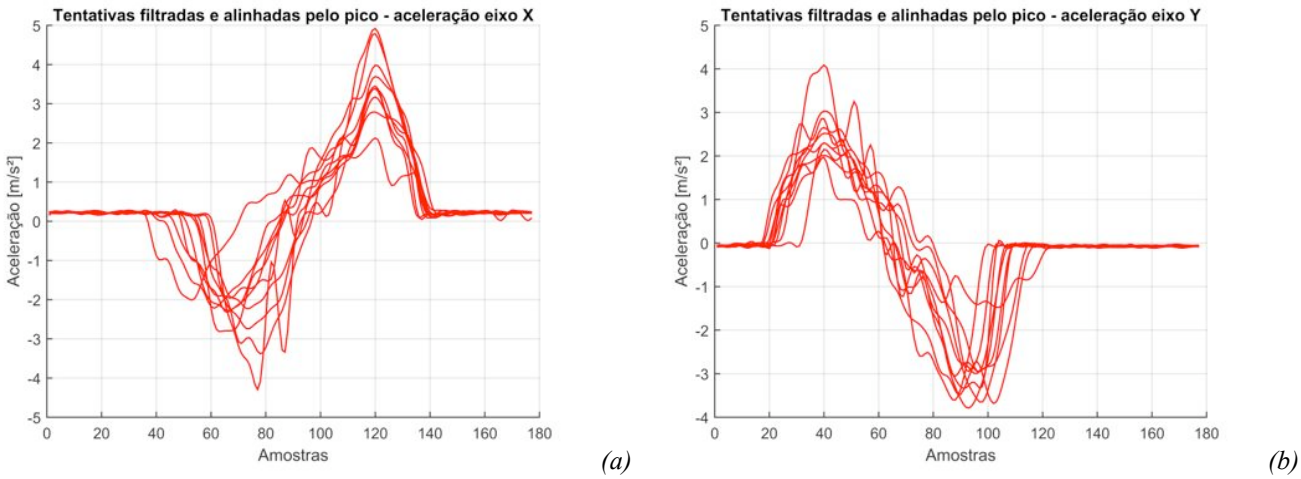
Os valores de máxima diferença percentual encontrados entre as tentativas de posição do sistema *FOB* e do sistema *ArduIMU* e a máxima diferença encontrada no valor médio destas é mostrada na Tabela 15.

***Tabela 15** – Máxima diferença entre as curvas e entre a média das curvas de posição com e sem ArduIMU.*

Unidade Inercial	Eixo	Máxima diferença entre tentativas [%]	Máxima diferença na média [%]
IMU_1	X	7,89	7,04
5678	Y	8,93	7,97
IMU_2	X	7,58	7,13
7890	Y	9,03	8,75
IMU_3	X	8,73	7,47
8901	Y	7,85	7,23
IMU_4	X	8,96	7,87
3456	Y	10,01	8,58

4.3.3. Movimento Combinados na direção X e Y

As Figuras 50 e 51 representam os dados do sistema *ArduIMU* em paralelo com o sistema *FOB*. A Figura 50 mostra as tentativas com as repetições agrupadas, filtradas e alinhadas pelo pico para as direções X e Y. Na Figura 51 são apresentadas as tentativas da posição resultante após as duas integrações e remoção de *drift,* normalizadas em amplitude (dividindo-se pelo pico) e no tempo (dividindo-se pela taxa de amostragem, 125Hz).



(a) *(b)*

***Figura 50** – Curvas das tentativas agrupadas, filtradas e alinhadas pelo pico. (a) direção X (alvo 1), (b) direção Y (alvo 2).*

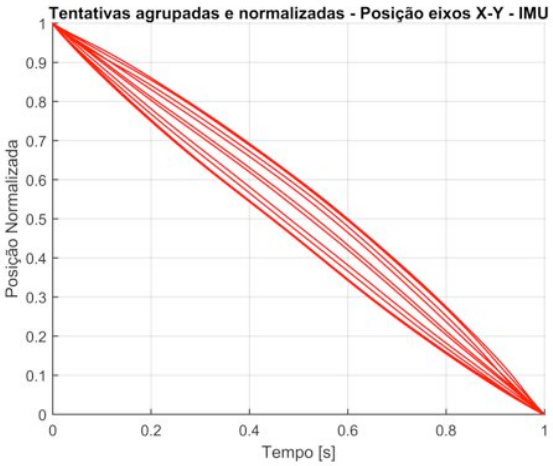


Figura 51 – *Curvas dos dados das integrações da aceleração das tentativas agrupadas, filtradas, e normalizadas em amplitude e no tempo e alinhadas pelo pico na direção XY (combinada). .*

As máximas e mínimas acelerações desenvolvidas nos dois eixos que foram amostrados pelo sistema *ArduIMU* durante o movimento referente ao alvo 3 (movimento combinado nas direções X e Y) são mostradas na Tabela 16.

Tabela 16 – *Máximas e mínimas acelerações encontradas nos dois movimentos.*

Unidade Inercial	Eixo	Máxima aceleração [m/s²]	Máxima aceleração [g]	Mínima aceleração [m/s²]	Mínima aceleração [g]
IMU_15678	X	4,24	0,43	-1,80	-0,18
	Y	3,84	0,39	-1,75	-0,17
IMU_27890	X	4,75	0,48	-1,69	-0,17
	Y	4,16	0,42	-1,83	-0,18
IMU_38901	X	3,97	0,40	-1,78	-0,18
	Y	4,26	0,43	-1,81	-0,18
IMU_43456	X	3,79	0,38	-1,76	-0,18
	Y	4,18	0,42	-1,80	-0,18

Nas Figuras de 52 a 54 são mostradas as comparações entre as aquisições de posição do sistema *FOB* com os dados obtidos pelas integrações da aceleração do sistema *ArduIMU*. Nestas Figuras é mostrado o deslocamento correspondente ao alvo 3 (direções X e Y).

Na Figura 52 são comparadas as médias da posição das tentativas do *ArduIMU* e do *FOB*. Já nas Figuras 53 e 54 são sobrepostas as tentativas e média da posição do *ArduIMU* na região delimitada por 1 e 2 desvios padrão da posição do *FOB*.

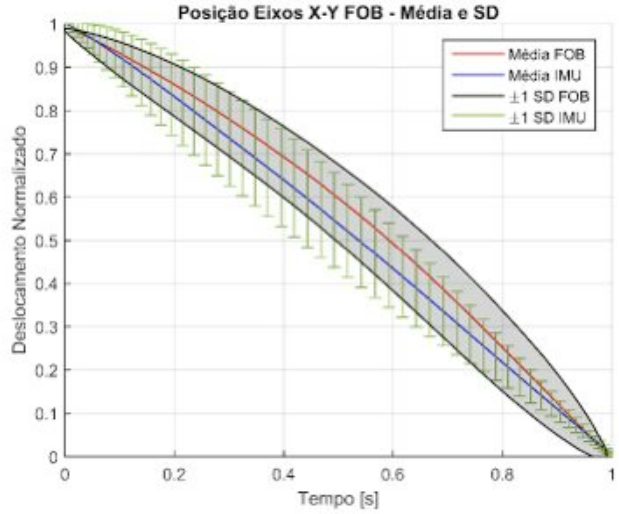


Figura 52 – *Comparação da média da posição do sistema FOB e da média da posição integrada da aceleração do sistema ArduIMU.*

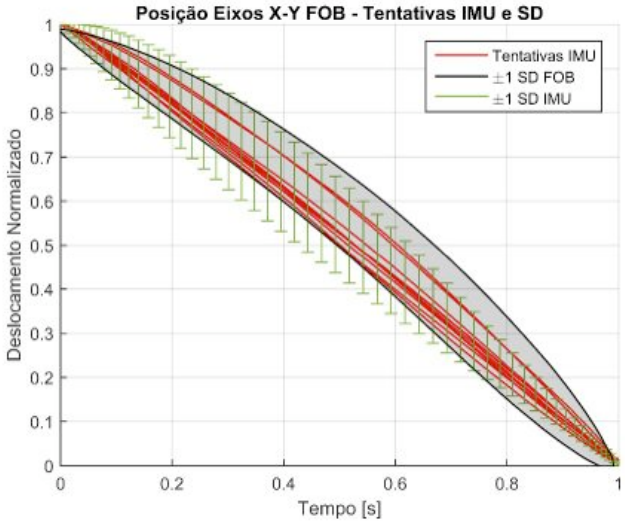


Figura 53 – *Curvas média das tentativas do sistema ArduIMU em um desvio padrão do sistema FOB.*

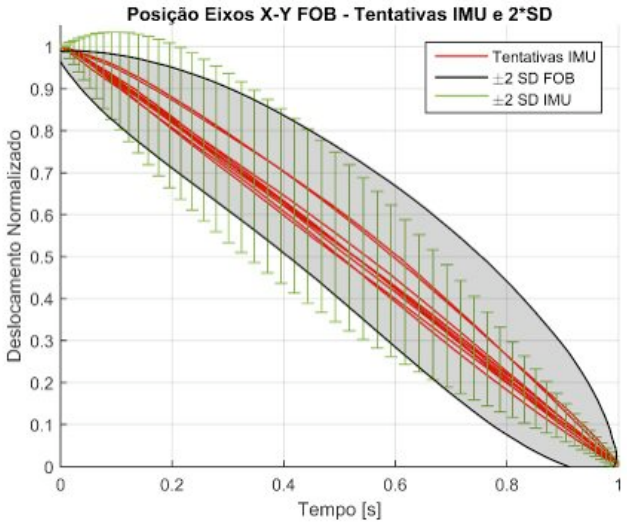


Figura 54 – *Curvas média das tentativas do sistema ArduIMU em um desvio padrão do sistema FOB.*

Os valores percentuais da máxima diferença encontrada entre as tentativas de posição do sistema *FOB* e do sistema *ArduIMU* e a máxima diferença encontrada na média destas é mostrada na Tabela 17.

Tabela 17 – *Máxima diferença entre as curvas e entre a média das curvas de posição com e sem ArduIMU.*

Unidade Inercial	Eixo	Máxima diferença entre tentativas [%]	Máxima diferença na média [%]
IMU_1 5678	X	8,95	7,12
	Y	7,89	6,98
IMU_2 7890	X	7,58	6,23
	Y	8,52	7,33
IMU_3 8901	X	8,72	7,81
	Y	10,03	7,20
IMU_4 3456	X	7,95	6,96
	Y	10,12	8,32

4.3.4. Movimento na direção Z

A Figura 55 mostra as tentativas de aceleração com as repetições agrupadas, filtradas e alinhadas pelo pico. A Figura 56 apresenta as tentativas da posição resultante após as duas

integrações e remoção de *drift,* normalizadas em amplitude (dividindo-se pelo pico) e no tempo (dividindo-se pela taxa de amostragem, 125Hz).

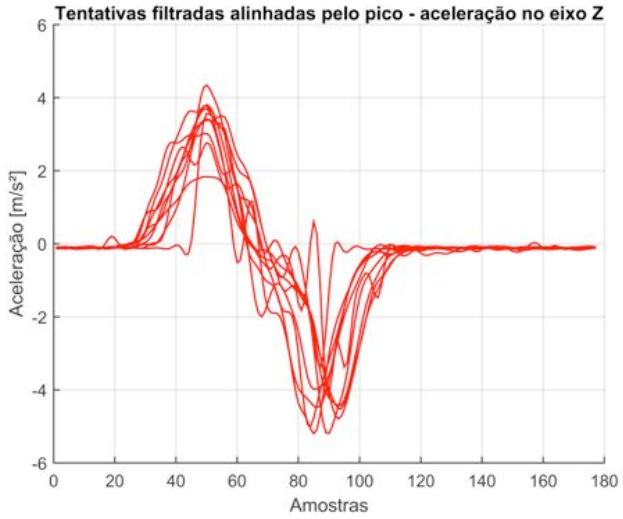


Figura 55 – *Curvas das tentativas agrupadas, filtradas e alinhadas pelo pico na direção Z.*

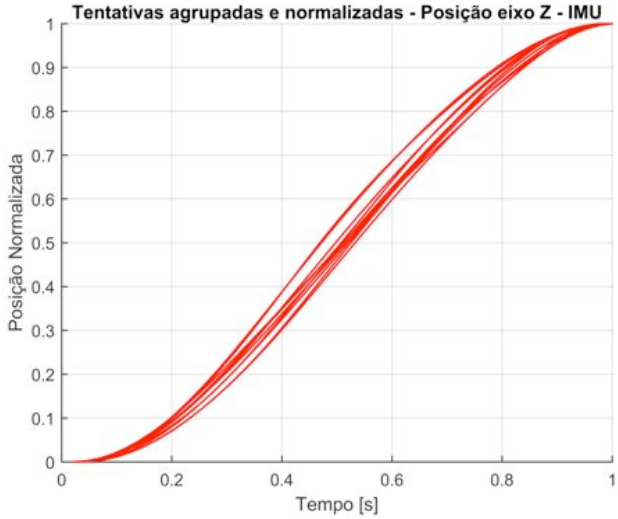


Figura 56 – *Curvas dos dados das integrações da aceleração das tentativas agrupadas, filtradas, e normalizadas em amplitude e no tempo e alinhadas pelo pico na direção Z.*

Os valores das máximas e mínimas acelerações desenvolvidas que foram amostradas pelo sistema *ArduIMU* durante o movimento na direção Z são mostradas na Tabela 18.

Tabela 18 – *Valores Máximos e mínimos de acelerações encontradas nos dois movimentos.*

Unidade Inercial	Eixo	Máxima aceleração [m/s²]	Máxima aceleração [g]	Mínima aceleração [m/s²]	Mínima aceleração [g]
IMU_1 5678	Z	5,23	0,53	-2,07	-0,21
IMU_2 7890		4,96	0,51	-2,22	-0,23
IMU_3 8901		5,09	0,52	-2,16	-0,22
IMU_4 3456		5,17	0,53	-2,09	-0,21

Nas Figuras de 57 a 59 são mostradas as comparações entre as aquisições de posição do *FOB* contra as posições integradas da aceleração do *ArduIMU*.

Na Figura 57 são comparadas as médias da posição das tentativas do *ArduIMU* e do *FOB*. Já nas Figuras 58 e 59 são sobrepostas as tentativas e média da posição do *ArduIMU* na região delimitada por 1 e 2 desvios padrão da posição do *FOB*.

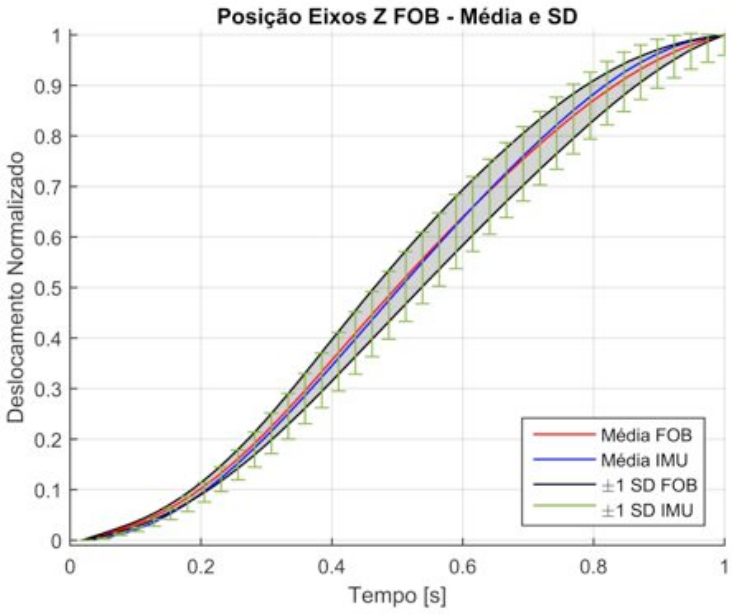


Figura 57 – *Comparação da média da posição do sistema FOB e da média da posição integrada da aceleração do sistema ArduIMU.*

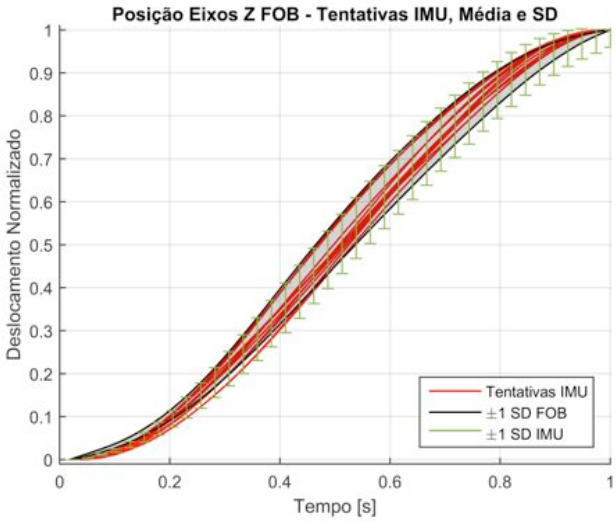


Figura 58 – *Sobreposição das tentativas do sistema ArduIMU na região delimitada por um desvio padrão do sistema FOB.*

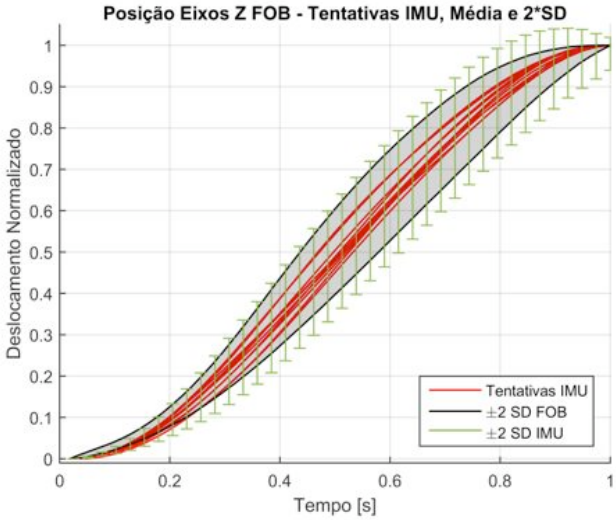


Figura 59 – *Sobreposição das tentativas do sistema ArduIMU na região delimitada por dois desvios padrão do sistema FOB.*

Os valores da máxima diferença encontrada entre as tentativas de posição do sistema *FOB* e sistema *ArduIMU* e a máxima diferença encontrada na média destas é mostrada na Tabela 19.

Tabela 19 – *Máxima diferença entre as curvas e entre a média das curvas de posição com e sem ArduIMU.*

Unidade Inercial	Eixo	Máxima diferença entre tentativas [%]	Máxima diferença na média [%]
IMU_1 5678	Z	9,05	7,55
IMU_2 7890		8,26	7,41
IMU_3 8901		8,73	6,97
IMU_4 3456		7,84	7,01

4.4. Movimentos em S no espaço 3D com o sistema *ArduIMU*

Com as posições dos dedos, cotovelo e ombro calculadas pode-se mostrar as posições dos segmentos reconstruídos, em cada instante do tempo, durante a execução do movimento em S no espaço (3D). As Figuras 60 e 61 mostram o trajeto descrito pelo movimento do sistema *ArduIMU* (para os braços esquerdo e direito, respectivamente), da ponta do dedo médio, do cotovelo e do ombro, bem como a representação do segmento do antebraço (cotovelo-dedos) (linhas em azul) e do segmento do braço (ombro-cotovelo) (linhas em vermelho).

A Tabela 20 mostra os valores das máximas e mínimas acelerações encontradas durante este movimento nos eixos X, Y e Z para os dois braços.

Tabela 20 – *Máximas e mínimas acelerações encontradas para o movimento em S em três dimensões.*

Antebraço	Eixo	Máxima Aceleração [m/s²]	Máxima Aceleração [g]	Mínima Aceleração [m/s²]	Mínima Aceleração [g]
Esquerdo	X	3,66	0,37	-2,99	-0,30
	Y	2,19	0,22	-2,50	-0,25
	Z	3,80	0,39	-2,73	-0,28
Direito	X	3,49	0,36	-2,86	-0,29
	Y	2,05	0,21	-2,37	-0,24
	Z	3,79	0,39	-2,54	-0,26

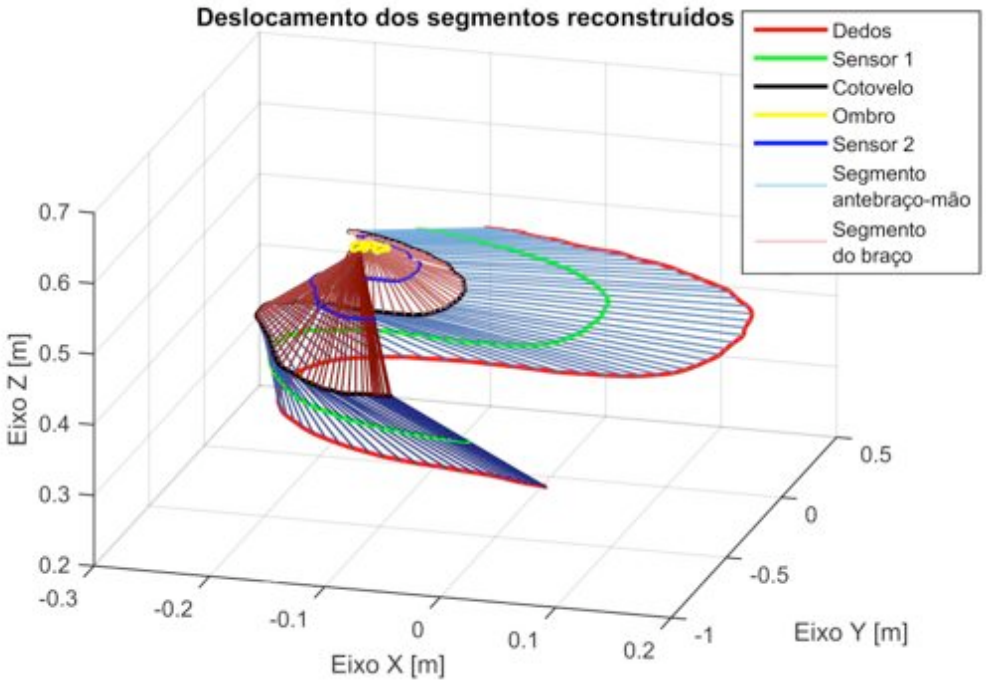


Figura 60 – *Trajeto do movimento em S no espaço (3D) para o membro superior esquerdo. Posição dos dedos (curva em vermelho); posição do sensor do antebraço (curva em verde); posição calculada do cotovelo (curva em preto); sensor do braço (curva em azul); posição calculada do ombro (curva em amarelo); segmento cotovelo-dedos reconstruídos computacionalmente (retas em azul: azul escuro = início do movimento, azul claro = fim do movimento); segmento do braço reconstruído computacionalmente (retas em vermelho: vermelho escuro = fase inicial do movimento vermelho claro = fase final do movimento).*

Finalmente pode-se concluir que o *ArduIMU* pode ser utilizado em aplicações de avaliações biomecânicas em membros superiores considerando os erros em cada direção de medição conforme mostra a Tabela 21. E em movimentos com amplitudes compreendidas no intervalo de 2 a 70cm, que é a faixa de amplitudes utilizada neste estudo.

Tabela 21 – *Máximos erros encontrados em cada um dos três eixos para movimentos avaliados com o sistema ArduIMU.*

Eixo	Erro Máximo [%]
X	8,96
Y	10,13
Z	9,51

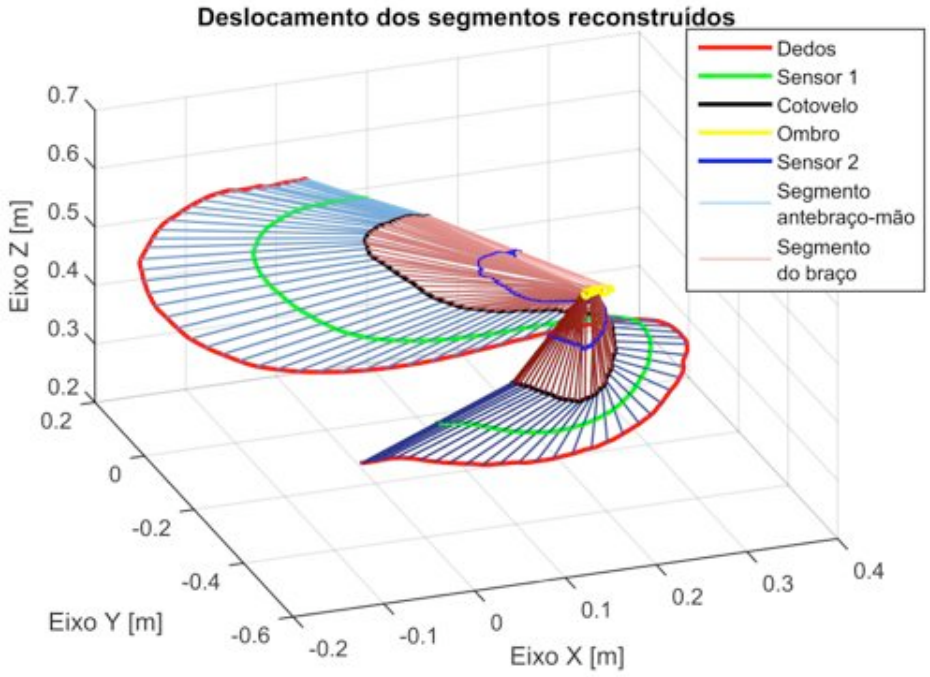


Figura 61 – *Trajeto do movimento em S no espaço (3D) para o membro superior direito. Posição dos dedos (curva em vermelho); posição do sensor do antebraço (curva em verde); posição calculada do cotovelo (curva em preto); sensor do braço (curva em azul); posição calculada do ombro (curva em amarelo); segmento cotovelo-dedos reconstruídos computacionalmente (retas em azul: azul escuro = início do movimento, azul claro = fim do movimento); segmento do braço reconstruído computacionalmente (retas em vermelho: vermelho escuro = fase inicial do movimento vermelho claro = fase final do movimento).*

Um breve manual com instruções de utilização do *ArduIMU* com os softwares *MATLAB*, *XCTU* e *Arduino IDE* está descrito no Anexo C.

5. CONSIDERAÇÕES FINAIS

Os objetivos determinados no início deste trabalho foram alcançados. A avaliação de um sistema de acelerometria para medição de posição em comparação com um sistema *FOB* de sensores magnéticos de posição foi efetuada com sucesso com algumas ressalvas. A necessidade do cálculo das integrais numéricas em grandes quantidades de dados estão sujeitas a acúmulos de erros computacionais, também conhecidos como "erros de máquina", e aproximação dos métodos de cálculo também demostram que a comparação entre os dois conjuntos de dados pode não vir a ser adequada.

Mesmo com estas limitações, os erros máximos obtidos ficaram abaixo dos 10% em quase todas as ocasiões em comparação com o sistema *FOB*: para a direção X o máximo erro encontrado nos dados do sistema *ArduIMU* em relação ao sistema *FOB* foi de 8,96%, n a direção Y de 10,13%, e na direção Z de 9,51%. O formato do movimento de um exercício feito com o braço (no espaço 3D - movimento em "S") foi preservado, indicando que o sensor estudado reproduz adequadamente a posição dos membros em que for fixado, desde que observados os erros encontrados e também que observando-se as pequenas amplitudes dos movimentos dos segmentos corporais.

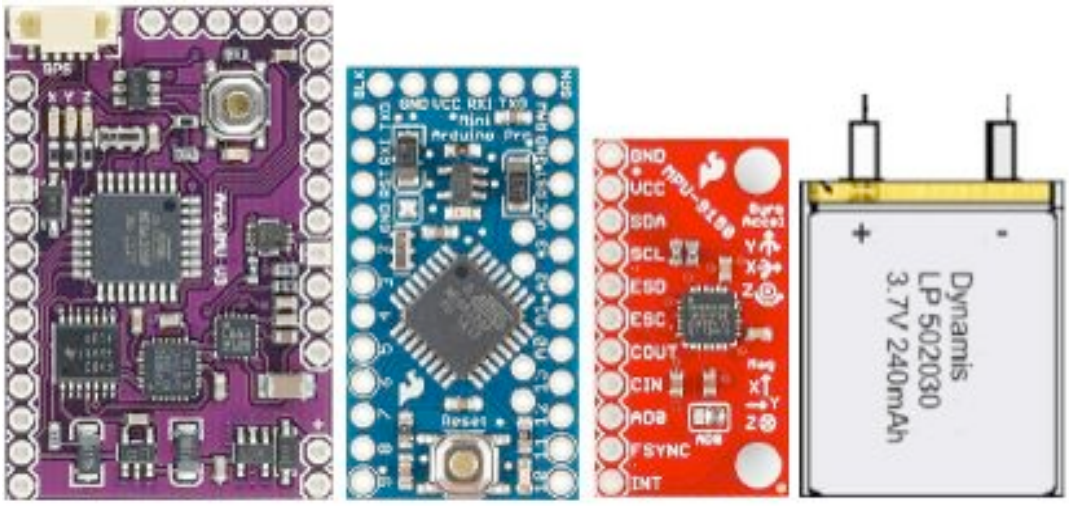
Os experimentos para a calibração dinâmica (numa faixa de 10 a 50Hz) do acelerômetro apresentaram uma variação máxima de 6,1%, em amplitude, em comparação com o *LVDT*. Já a variação máxima em aceleração foi de 7,5% em comparação ao acelerômetro de referência.

Quatro sensores inerciais estão em funcionamento e podem ser utilizados em conjunto para avaliações biomecânicas e obtenção de aceleração e posição em movimentos dos segmentos do membro superior.

O custo total dos componentes utiliza 4% (US$ 100,00) do custo de sensores inerciais comerciais (US$ 2500,00) vendido atualmente.

6. PROPOSTA PARA TRABALHOS FUTUROS

Durante o desenvolvimento deste trabalho constatou-se que é de interesse miniaturizar o sistema de aquisição formado por *ArduIMU*, *XBee* e bateria de 9V. O componente crítico para o sucesso da redução de tamanho do sistema é a bateria de 9V. A troca da fonte de alimentação só é possível se outra placa que dispõem de um *IMU* for utilizada, já que o *ArduIMU* restringe a faixa de alimentação entre 6 e 12V. Existe no mercado uma placa com o sensor MPU-9150, da mesma fabricante (InvenSense) do sensor MPU-6000 utilizado no *ArduIMU*. Esta outra placa possui no mesmo encapsulamento acelerômetros, giroscópios e magnetômetros tri-axiais. Esta placa em combinação com um *Arduino Mini* de lógica 3.3V e uma bateria do tipo LP502030 de 3.7V (DYNAMIS BATTERIEN, 2015) podem miniaturizar o hardware atual de forma a embuti-lo totalmente dentro da caixa plástica utilizada atualmente e que não comporta as dimensões da bateria de 9V. Um comparativo entre as dimensões (em escala) dos componentes citados pode ser visto na Figura 62. Da esquerda para direita tem-se, em primeiro o *ArduIMU,* em segundo *Arduino Mini,* em terceiro o sensor MPU-9150 e por último a bateria LP502030 de 3.7V.



(a) ***(b)*** ***(c)*** ***(d)***

Figura 62 - *Componentes sugeridos para trabalhos futuros. (a) ArduIMU; (b) Arduino Mini; (c) sensor MPU-9150; (d) bateria LP502030 (3,7V).*

Fonte Imagens: (a) https://cdn.sparkfun.com//assets/parts/6/4/2/6/11055-02a.jpg;

(b) https://cdn.sparkfun.com//assets/parts/7/3/7/6/11486-04.jpg;

(c) https://cdn.sparkfun.com//assets/parts/6/5/4/0/11114-02a.jpg;

(d) http://www.dynamis-batterien.de/pdfs/batt/parts/DYNAMIS-LITH-POLY-LINE-LP502030_rev1-1.pdf?langSel=0

7. REFERÊNCIAS BIBLIOGRÁFICAS

BARTONEK, A. (2015). **The use of orthoses and gait analysis in children with AMC.** Journal of Children's Orthopaedics, Vol. 9: 437-447.

ROUSANOGLOU, E.; NOUTSOS, K.; BAYIOS, I.; BOUDOLOS, K. (2014). **Ground Reaction Forces and Throwing Performance in Elite and Novice Players in Two Types of Handball Shot.** Journal of Human Kinetics, Vol. 404: 49-55.

FREIVALDS, A. **Biomechanics Of The Upper Limbs: Mechanics, Modeling, and Musculoskeletal Injuries**. Corporate Blvd., Boca Raton, Florida, 2004.

ABDULLAH, H.A.; TARRY, C.; DATTA, R.; MITTAL, G.S.; ABDERRAHIM, M. (2007). **Dynamic biomechanical model for assessing and monitoring robot-assisted upper-limb therapy.** Journal of Rehabilitation Research & Development, Vol. 44: 43-62.

BALBINOT, A.; BRUSAMARELLO V. J. **Instrumentação e Fundamentos de Medidas (Instrumentation and Measurement Fundamentals)**, Vol. II, 2. ed. Livros Técnicos e Científicos, Brasil, Rio de Janeiro, cap. 11, p. 165-249, 2010.

ASCENSION TECHNOLOGY. **FOB - Flock Of Birds**, 2015. Disponível em:
http://www.ascension-tech.com/
Acesso em: 28 Out. 2015.

BAGESTEIRO, LB.; SAINBURG, RL. (2002). **Handedness: Dominant Arm Advantages in Control of Limb Dynamics**. Journal of Neurophysiology, Vol. 88: 2408-2421.

BALBINOT, A.; FREITAS, J.; CORREA, D. (2015) **Use of inertial sensors as devices for upper limb motor monitoring exercises for motor rehabilitation**. Health and Technology, Vol.5 (2), pp.91-102.

CODE GOOGLE. **Introduction to ArduIMU V3**, 2015. Disponível em:

https://code.google.com/p/ardu-imu/wiki/IntroductionPage
Acesso em: 07 Nov. 2015.

CORREA, D. **Realidade Virtual e Sensores Inerciais no Desenvolvimento da Tecnologia Assistiva: Um sistema para estudo da marcha humana baseado em fusão de sensores inerciais**. 2015. 115 p. Dissertação de mestrado (Engenharia Elétrica) - Universidade Federal do Rio Grande do Sul, Porto Alegre, 2015.

CULHANE K. M., O'CONNOR M., LYONS D., LYONS G. M. (2005). **Accelerometers in rehabilitation medicine for older adults**. Age and Ageing, Vol. 34: 556-560.

WINTER D. A., **Biomechanics and motor control of human movement**, Hoboken: John Wiley & Sons, 4th ed, xiv, cap. 3.2, p. 48-81, 2009.

DYNAMIS BATTERIEN. **LP502030**, 2015. Disponível em:
http://www.dynamis-batterien.de/pdfs/batt/parts/DYNAMIS-LITH-POLY-LINE-LP502030_rev1-1.pdf?langSel=0
Acesso em: 05 Nov. 2015.

LETSENSE. **Wiva Biomech Tracker**, 2015. Disponível em:
http://www.e-wiva.com/store.html#!/Wiva-Biomech-Tracker/p/49680340/category=13005250
Acesso em: 13 Nov. 2015.

DIGI INTERNATIONAL. **XBee® 802.15.4 Multipoint Wireless Networking RF Module**, 2015. Disponível em:
http://www.digi.com/products/xbee-rf-solutions/modules/xbee-series1-module
Acesso em: 03 Nov. 2015.

SUPPORT DIGI INTERNATIONAL, **XBee / XBee-PRO RF Modules,** 2015.
Disponível em:
http://ftp1.digi.com/support/documentation/90000982_S.pdf
Acesso em: 03 Nov. 2015.

GODFREY A; CONWAY R; MEAGHER D; ÓLAIGHIN G. **Direct measurement of human movement by accelerometry**. Medical Engineering & Physics. 2008, 30:1364-1386.

HOLMBERG M., ARTURSSON T., **Drift Compensation, Standards and Calibration Methods**, in T. C. Pearce, S. S. Schiffman, H. T. Nagle and J. W. Gardner (Eds.), Handbook of Machine Olfaction: Electronic Nose Technology, Weinheim, Germany: Wiley-VCH, 2002, pp.325-346.

INVENSENSE. **MPU-6000**, 2015. Disponível em:
http://store.invensense.com/datasheets/invensense/MPU-6050_DataSheet_V3%204.pdf
Acesso em: 09 Nov. 2015.

MARCIAS, E.; TORRES, D.; RAVINDRAN, S. **Nine-Axis Sensor Fusion Using the Direction Corsine Matrix Algorithm on the MSP430F5xx Family**. Dallas: Texas Instruments, 2012.

ROBERTSON, GORDON; CALDWELL, GRAHAM; HAMILL, JOSEPH; KAMEN, GARY; WHITTLESEY, SANDY**. Research Methods in Biomechanics.** Champaign, IL: Human Kinetics, 2004.

SPARKFUN, **Triple Axis Magnetometer - HMC5883L Breakout Quickstart Guide**, 2015. Disponível em:
https://www.sparkfun.com/tutorials/301
Acesso em: 23 Out. 2015.

RITTERFELD U., CODY U. M., VORDERER P., **Serious Games: Mechanisms and Effects**. Routledge 1st ed. cap. 24, 2009.

ANEXO A - CÓDIGO EM *ARDUINO*

Aba 1: *ACC_DATA*

```
#define RLED_PIN  5
#define BLED_PIN  6
#define YLED_PIN  7

#include "MPU6000.h"
#include "Calibration.h"

long current_time = 0;
long timer_0 = 0;
long timer_1 = 0;
long timer_2 = 0;

int accelX_Corretion = 0;
int accelY_Corretion = 0;
int accelZ_Corretion = 0;

int accelValueX0 = 0;  int accelValueX1 = 0;  int accelValueX2 = 0;  int accelValueX3
= 0;  int accelValueX4 = 0;
int accelValueY0 = 0;  int accelValueY1 = 0;  int accelValueY2 = 0;  int accelValueY3
= 0;  int accelValueY4 = 0;
int accelValueZ0 = 0;  int accelValueZ1 = 0;  int accelValueZ2 = 0;  int accelValueZ3 =
0;  int accelValueZ4 = 0;

int iteration = -1;

float ini_time = 0.0;
int bled_on_off = 0;

void TimelFloatPrint(float f)
{
  byte * b = (byte *) &f;
  Serial.write(b[0]);
  Serial.write(b[1]);
  Serial.write(b[2]);
  Serial.write(b[3]);
}

void setup()
{
  Serial.begin(38400);      // Baud-rate. Do not change!

  MPU6000_Init();

  pinMode(RLED_PIN, OUTPUT); // Red LED
```

```
  pinMode(BLED_PIN, OUTPUT); // Blue LED
  pinMode(YLED_PIN, OUTPUT); // Yellow LED

  iteration = -1;

  ini_time = millis();
}

void loop()
{
  current_time = millis();

  if((current_time - timer_0) >= 2)  // Sampling Sensors loop runs at 500Hz
  {
    timer_0 = current_time;
    read_sensors();
  }

  if((current_time - timer_1) >= 1000) // LED loop runs at 1Hz
  {
    timer_1 = current_time;
    bled_blink();
  }

  if((current_time - timer_2) >= 8) // Sending data loop runs at 125Hz
  {
    timer_2 = current_time;
    iteration++;

    accelX_Corretion = floor(ACCELEROMETER_X_SCALE*(accelX -
ACCELEROMETER_X_OFFSET));
    accelY_Corretion = floor(ACCELEROMETER_Y_SCALE*(accelY -
ACCELEROMETER_Y_OFFSET));
    accelZ_Corretion = floor(ACCELEROMETER_Z_SCALE*(accelZ -
ACCELEROMETER_Z_OFFSET));

    switch (iteration)
    {
          case(0):
          accelValueX0 = accelX_Corretion;
          accelValueY0 = accelY_Corretion;
          accelValueZ0 = accelZ_Corretion;
          break;

        case(1):
          accelValueX1 = accelX_Corretion;
          accelValueY1 = accelY_Corretion;
          accelValueZ1 = accelZ_Corretion;
          break;
```

```
          case(2):
          accelValueX2 = accelX_Corretion;
          accelValueY2 = accelY_Corretion;
          accelValueZ2 = accelZ_Corretion;
          break;

        case(3):
          accelValueX3 = accelX_Corretion;
          accelValueY3 = accelY_Corretion;
          accelValueZ3 = accelZ_Corretion;
          break;

          case(4):
          accelValueX4 = accelX_Corretion;
          accelValueY4 = accelY_Corretion;
          accelValueZ4 = accelZ_Corretion;

          output_serial();
          iteration = -1;
          break;
    }
  }
}

void read_sensors()
{
  MPU6000_Read();
}

void output_serial()
{
//  Delimiter
    Serial.write(171);
    Serial.write(205);
    Serial.write(239);
    Serial.write(153);

//  Iteration 0
//  Accelerometer Data
    Serial.write(accelValueX0 >> 8);
    Serial.write(accelValueX0 & 0xff);
    Serial.write(accelValueY0 >> 8);
    Serial.write(accelValueY0 & 0xff);
    Serial.write(accelValueZ0 >> 8);
    Serial.write(accelValueZ0 & 0xff);

//  Iteration 1
//  Accelerometer Data
    Serial.write(accelValueX1 >> 8);
    Serial.write(accelValueX1 & 0xff);
```

```
    Serial.write(accelValueY1 >> 8);
    Serial.write(accelValueY1 & 0xff);
    Serial.write(accelValueZ1 >> 8);
    Serial.write(accelValueZ1 & 0xff);

//  Iteration 2
//  Accelerometer Data
    Serial.write(accelValueX2 >> 8);
    Serial.write(accelValueX2 & 0xff);
    Serial.write(accelValueY2 >> 8);
    Serial.write(accelValueY2 & 0xff);
    Serial.write(accelValueZ2 >> 8);
    Serial.write(accelValueZ2 & 0xff);

//  Iteration 3
//  Accelerometer Data
    Serial.write(accelValueX3 >> 8);
    Serial.write(accelValueX3 & 0xff);
    Serial.write(accelValueY3 >> 8);
    Serial.write(accelValueY3 & 0xff);
    Serial.write(accelValueZ3 >> 8);
    Serial.write(accelValueZ3 & 0xff);

//  Iteration 4
//  Accelerometer Data
    Serial.write(accelValueX4 >> 8);
    Serial.write(accelValueX4 & 0xff);
    Serial.write(accelValueY4 >> 8);
    Serial.write(accelValueY4 & 0xff);
    Serial.write(accelValueZ4 >> 8);
    Serial.write(accelValueZ4 & 0xff);
}

void bled_blink()
{
  if(bled_on_off == 0)
  {
    bled_on_off = 1;
    digitalWrite(BLED_PIN, HIGH);
  }

  else
  {
    bled_on_off = 0;
    digitalWrite(BLED_PIN, LOW);
  }
}
```

Aba 2: ***Calibration.h***

```
#define ACCELEROMETER_X_OFFSET 0.0
#define ACCELEROMETER_Y_OFFSET 0.0
#define ACCELEROMETER_Z_OFFSET 0.0
#define ACCELEROMETER_X_SCALE 1.0
#define ACCELEROMETER_Y_SCALE 1.0
#define ACCELEROMETER_Z_SCALE 1.0
#define GYROSCOPE_X_OFFSET 0.0
#define GYROSCOPE_Y_OFFSET 0.0
#define GYROSCOPE_Z_OFFSET 0.0
#define GYROSCOPE_X_SCALE 1.0
#define GYROSCOPE_Y_SCALE 1.0
#define GYROSCOPE_Z_SCALE 1.0
#define GYROSCOPE_X_TRIGGER_READ 500.0
#define GYROSCOPE_Y_TRIGGER_READ 500.0
#define GYROSCOPE_Z_TRIGGER_READ 500.0
#define MAGNETOMETER_X_OFFSET 0.0
#define MAGNETOMETER_Y_OFFSET 0.0
#define MAGNETOMETER_Z_OFFSET 0.0
#define MAGNETOMETER_X_SCALE 1.0
#define MAGNETOMETER_Y_SCALE 1.0
#define MAGNETOMETER_Z_SCALE 1.0
#define USE_FILTER_25HZ true
```

Aba 3: ***MPU6000.h***

```
// MPU6000 support for AruduIMU V3
#include <SPI.h>

#define MPU6000_CHIP_SELECT_PIN 4  // MPU6000 CHIP SELECT

// MPU 6000 registers
#define MPUREG_WHOAMI        0x75
#define MPUREG_SMPLRT_DIV    0x19
#define MPUREG_CONFIG        0x1A
#define MPUREG_GYRO_CONFIG   0x1B
#define MPUREG_ACCEL_CONFIG  0x1C
#define MPUREG_INT_PIN_CFG   0x37
#define MPUREG_INT_ENABLE    0x38
#define MPUREG_ACCEL_XOUT_H  0x3B
#define MPUREG_ACCEL_XOUT_L  0x3C
#define MPUREG_ACCEL_YOUT_H  0x3D
#define MPUREG_ACCEL_YOUT_L  0x3E
#define MPUREG_ACCEL_ZOUT_H  0x3F
#define MPUREG_ACCEL_ZOUT_L  0x40
#define MPUREG_TEMP_OUT_H    0x41
#define MPUREG_TEMP_OUT_L    0x42
#define MPUREG_GYRO_XOUT_H   0x43
```

```
#define MPUREG_GYRO_XOUT_L    0x44
#define MPUREG_GYRO_YOUT_H    0x45
#define MPUREG_GYRO_YOUT_L    0x46
#define MPUREG_GYRO_ZOUT_H    0x47
#define MPUREG_GYRO_ZOUT_L    0x48
#define MPUREG_USER_CTRL      0x6A
#define MPUREG_PWR_MGMT_1     0x6B
#define MPUREG_PWR_MGMT_2     0x6C

// Configuration bits  MPU 6000
#define BIT_SLEEP                 0x40
#define BIT_H_RESET               0x80
#define BITS_CLKSEL               0x07
#define MPU_CLK_SEL_PLLGYROX      0x01
#define MPU_CLK_SEL_PLLGYROZ      0x03
#define MPU_EXT_SYNC_GYROX        0x02
#define BITS_FS_250DPS            0x00
#define BITS_FS_500DPS            0x08
#define BITS_FS_1000DPS           0x10
#define BITS_FS_2000DPS           0x18
#define BITS_FS_MASK              0x18
#define BITS_DLPF_CFG_256HZ_NOLPF2  0x00
#define BITS_DLPF_CFG_188HZ       0x01
#define BITS_DLPF_CFG_98HZ        0x02
#define BITS_DLPF_CFG_42HZ        0x03
#define BITS_DLPF_CFG_20HZ        0x04
#define BITS_DLPF_CFG_10HZ        0x05
#define BITS_DLPF_CFG_5HZ         0x06
#define BITS_DLPF_CFG_2100HZ_NOLPF  0x07
#define BITS_DLPF_CFG_MASK        0x07
#define BIT_INT_ANYRD_2CLEAR          0x10
#define BIT_RAW_RDY_EN                0x01
#define BIT_I2C_IF_DIS            0x10

// global variables
volatile uint8_t MPU6000_newdata;

//Sensor variables
int accelX;
int accelY;
int accelZ;

int gyroX;
int gyroY;
int gyroZ;
```

Aba 4: ***MPU6000***

```
// MPU6000 support for ArduIMU V3
#include <SPI.h>

// MPU6000 SPI functions
byte MPU6000_SPI_read(byte reg)
{
  byte dump;
  byte return_value;
  byte addr = reg | 0x80; // Set most significant bit
  digitalWrite(MPU6000_CHIP_SELECT_PIN, LOW);
  dump = SPI.transfer(addr);
  return_value = SPI.transfer(0);
  digitalWrite(MPU6000_CHIP_SELECT_PIN, HIGH);
  return(return_value);
}

void MPU6000_SPI_write(byte reg, byte data)
{
  byte dump;
  digitalWrite(MPU6000_CHIP_SELECT_PIN, LOW);
  dump = SPI.transfer(reg);
  dump = SPI.transfer(data);
  digitalWrite(MPU6000_CHIP_SELECT_PIN, HIGH);
}

// MPU6000 INTERRUPT ON INT0
void MPU6000_data_int()
{
  MPU6000_newdata++;
}

// MPU6000 Initialization and configuration
void MPU6000_Init(void)
{
    // MPU6000 chip select setup
    pinMode(MPU6000_CHIP_SELECT_PIN, OUTPUT);
    digitalWrite(MPU6000_CHIP_SELECT_PIN, HIGH);

    // SPI initialization
    SPI.begin();
    SPI.setClockDivider(SPI_CLOCK_DIV16);      // SPI at 1Mhz (on 16Mhz clock)
    delay(10);

    // Chip reset
    MPU6000_SPI_write(MPUREG_PWR_MGMT_1, BIT_H_RESET);
    delay(100);
    // Wake up device and select GyroZ clock (better performance)
```

```
   MPU6000_SPI_write(MPUREG_PWR_MGMT_1, MPU_CLK_SEL_PLLGYROZ);
   delay(1);
   // Disable I2C bus (recommended on datasheet)
   MPU6000_SPI_write(MPUREG_USER_CTRL, BIT_I2C_IF_DIS);
   delay(1);
   // SAMPLE RATE
   MPU6000_SPI_write(MPUREG_SMPLRT_DIV,0x01);     // Sample rate = 200Hz
Fsample= 1Khz/(1+1) = 500Hz
   delay(1);
   // FS & DLPF   FS=2000º/s, DLPF = 20Hz (low pass filter)
   MPU6000_SPI_write(MPUREG_CONFIG, BITS_DLPF_CFG_98HZ);
   delay(1);
   MPU6000_SPI_write(MPUREG_GYRO_CONFIG,BITS_FS_2000DPS);  // Gyro
scale 2000º/s
   delay(1);
   //MPU6000_SPI_write(MPUREG_ACCEL_CONFIG,0x08);           // Accel scale 4g
(4096LSB/g)
   MPU6000_SPI_write(MPUREG_ACCEL_CONFIG,0x10);           // Accel scale 8g
   delay(1);
   // INT CFG => Interrupt on Data Ready
   MPU6000_SPI_write(MPUREG_INT_ENABLE,BIT_RAW_RDY_EN);         //
INT: Raw data ready
   delay(1);
   MPU6000_SPI_write(MPUREG_INT_PIN_CFG,BIT_INT_ANYRD_2CLEAR);  //
INT: Clear on any read
   delay(1);
   // Oscillator set
   //
MPU6000_SPI_write(MPUREG_PWR_MGMT_1,MPU_CLK_SEL_PLLGYROZ);
   delay(1);

   // MPU_INT is connected to INT 0. Enable interrupt on INT0
   attachInterrupt(0,MPU6000_data_int,RISING);
}

// Read gyros and accel sensors on MPU6000
void MPU6000_Read()
{
 int byte_H;
 int byte_L;

 // Read AccelX
   byte_H = MPU6000_SPI_read(MPUREG_ACCEL_XOUT_H);
   byte_L = MPU6000_SPI_read(MPUREG_ACCEL_XOUT_L);
   accelX = (byte_H<<8)| byte_L;
   // Read AccelY
   byte_H = MPU6000_SPI_read(MPUREG_ACCEL_YOUT_H);
   byte_L = MPU6000_SPI_read(MPUREG_ACCEL_YOUT_L);
   accelY = (byte_H<<8)| byte_L;
   // Read AccelZ
```

```
    byte_H = MPU6000_SPI_read(MPUREG_ACCEL_ZOUT_H);
    byte_L = MPU6000_SPI_read(MPUREG_ACCEL_ZOUT_L);
    accelZ = (byte_H<<8)| byte_L;

    // Read Temp
    //byte_H = MPU6000_SPI_read(MPUREG_TEMP_OUT_H);
    //byte_L = MPU6000_SPI_read(MPUREG_TEMP_OUT_L);
    //temp = (byte_H<<8)| byte_L;

    // Read GyroX
    byte_H = MPU6000_SPI_read(MPUREG_GYRO_XOUT_H);
    byte_L = MPU6000_SPI_read(MPUREG_GYRO_XOUT_L);
    gyroX = (byte_H<<8)| byte_L;
    // Read GyroY
    byte_H = MPU6000_SPI_read(MPUREG_GYRO_YOUT_H);
    byte_L = MPU6000_SPI_read(MPUREG_GYRO_YOUT_L);
    gyroY = (byte_H<<8)| byte_L;
    // Read GyroZ
    byte_H = MPU6000_SPI_read(MPUREG_GYRO_ZOUT_H);
    byte_L = MPU6000_SPI_read(MPUREG_GYRO_ZOUT_L);
    gyroZ = (byte_H<<8)| byte_L;
}
```

ANEXO B - CÓDIGOS EM *MATLAB*

B1 - Calibração

```
g_expected = 4096;

% Sorting Elements
sort_xp = sort(IMU_3XP(:,2),1,'descend');
sort_xn = sort(IMU_3XN(:,2),1,'ascend');

sort_yp = sort(IMU_3YP(:,3),1,'descend');
sort_yn = sort(IMU_3YN(:,3),1,'ascend');

sort_zp = sort(IMU_3ZP(:,4),1,'descend');
sort_zn = sort(IMU_3ZN(:,4),1,'ascend');

% Get 20% of the highest elements
XP_20 = floor(size(IMU_3XP,1)*0.2);
XN_20 = floor(size(IMU_3XN,1)*0.2);

YP_20 = floor(size(IMU_3YP,1)*0.2);
YN_20 = floor(size(IMU_3YN,1)*0.2);

ZP_20 = floor(size(IMU_3ZP,1)*0.2);
ZN_20 = floor(size(IMU_3ZN,1)*0.2);

% Calculate the mean of the 20% highest elements
XP = mean(IMU_3XP(1:XP_20,2));
XN = mean(IMU_3XN(1:XN_20,2));

YP = mean(IMU_3YP(1:YP_20,3));
YN = mean(IMU_3YN(1:YN_20,3));

ZP = mean(IMU_3ZP(1:ZP_20,4));
ZN = mean(IMU_3ZN(1:ZN_20,4));

% Calculate OFFSET
off_x = (XP + XN)/2;
off_y = (YP + YN)/2;
off_z = (ZP + ZN)/2;

% Calculate SCALE
scale_x = g_expected/(XP - off_x);
scale_y = g_expected/(YP - off_y);
scale_z = g_expected/(ZP - off_z);
```

B2 - Cálculo de Média e Desvio Padrão para dados do *IMU*

```
%% Mean and SD of Elements

v = NORM_3XI;

m = max(cellfun(@length,v));
n = length(v);
```

```
V = zeros(m,n);
mask = zeros(m,n);

for i = 1:length(v)
    vi = v{i};
    V(1:length(vi),i) = vi;
    mask(1:length(vi),i) = ones(size(vi));
end

MeanV = sum(V.*mask,2)./sum(mask,2);

for i = 1:m
    P(i,:) = (V(i,:) -  MeanV(i)).^2;
end

MeanP = sqrt(sum(P.*mask,2)./sum(mask,2));
```

B3 - Importação de Dados do *IMU*

```
% clear all; close all; clc;

extension = '*.log';
title     = 'Select the XCTU .log file';
directory = 'C:';

[FileName,PathName] = uigetfile(extension,title,directory);

[time,ID,data] = importfile(FileName,2,inf);

% REF     Adsress  Node Identifier
% IMU_1 = 0x5678 - Sensor D
% IMU_2 = 0x7890 - Sensor (IMU_2)
% IMU_3 = 0x8901 - Sensor C
% IMU_4 = 0x3456 - Sensor B

%%

i_1 = 0;
i_2 = 0;
i_3 = 0;
i_4 = 0;

for i = 1:size(data,1)     % Varredura das linhas

        % IMU 1
        if ID{i,1}(1,9:12) == '5678'

            i_1 = i_1 + 1;
            % Interation 0                                      % DATA
            IMU_1(i_1,2)  = shex2dec(data{i,1}(1,1:4));     % ACC - X
            IMU_1(i_1,3)  = shex2dec(data{i,1}(1,5:8));     % ACC - Y
            IMU_1(i_1,4)  = shex2dec(data{i,1}(1,9:12));    % ACC - Z

            i_1 = i_1 + 1;
            % Interation 1                                      % DATA
            IMU_1(i_1,2)  = shex2dec(data{i,1}(1,13:16));   % ACC - X
```

```
    IMU_1(i_1,3)  = shex2dec(data{i,1}(1,17:20));   % ACC - Y
    IMU_1(i_1,4)  = shex2dec(data{i,1}(1,21:24));   % ACC - Z

    i_1 = i_1 + 1;
    % Interation 2                                  % DATA
    IMU_1(i_1,2)  = shex2dec(data{i,1}(1,25:28));   % ACC - X
    IMU_1(i_1,3)  = shex2dec(data{i,1}(1,29:32));   % ACC - Y
    IMU_1(i_1,4)  = shex2dec(data{i,1}(1,33:36));   % ACC - Z

    i_1 = i_1 + 1;
    % Interation 3                                  % DATA
    IMU_1(i_1,2)  = shex2dec(data{i,1}(1,37:40));   % ACC - X
    IMU_1(i_1,3)  = shex2dec(data{i,1}(1,41:44));   % ACC - Y
    IMU_1(i_1,4)  = shex2dec(data{i,1}(1,45:48));   % ACC - Z

    i_1 = i_1 + 1;
    % Interation 4                                  % DATA
    IMU_1(i_1,2)  = shex2dec(data{i,1}(1,49:52));   % ACC - X
    IMU_1(i_1,3)  = shex2dec(data{i,1}(1,53:56));   % ACC - Y
    IMU_1(i_1,4)  = shex2dec(data{i,1}(1,57:60));   % ACC - Z

end

% IMU 2
if ID{i,1}(1,9:12) == '7890'

    i_2 = i_2 + 1;
    % Interation 0                                  % DATA
    IMU_2(i_2,2)  = shex2dec(data{i,1}(1,1:4));     % ACC - X
    IMU_2(i_2,3)  = shex2dec(data{i,1}(1,5:8));     % ACC - Y
    IMU_2(i_2,4)  = shex2dec(data{i,1}(1,9:12));    % ACC - Z

    i_2 = i_2 + 1;
    % Interation 1                                  % DATA
    IMU_2(i_2,2)  = shex2dec(data{i,1}(1,13:16));   % ACC - X
    IMU_2(i_2,3)  = shex2dec(data{i,1}(1,17:20));   % ACC - Y
    IMU_2(i_2,4)  = shex2dec(data{i,1}(1,21:24));   % ACC - Z

    i_2 = i_2 + 1;
    % Interation 2                                  % DATA
    IMU_2(i_2,2)  = shex2dec(data{i,1}(1,25:28));   % ACC - X
    IMU_2(i_2,3)  = shex2dec(data{i,1}(1,29:32));   % ACC - Y
    IMU_2(i_2,4)  = shex2dec(data{i,1}(1,33:36));   % ACC - Z

    i_2 = i_2 + 1;
    % Interation 3                                  % DATA
    IMU_2(i_2,2)  = shex2dec(data{i,1}(1,37:40));   % ACC - X
    IMU_2(i_2,3)  = shex2dec(data{i,1}(1,41:44));   % ACC - Y
    IMU_2(i_2,4)  = shex2dec(data{i,1}(1,45:48));   % ACC - Z

    i_2 = i_2 + 1;
    % Interation 4                                  % DATA
    IMU_2(i_2,2)  = shex2dec(data{i,1}(1,49:52));   % ACC - X
    IMU_2(i_2,3)  = shex2dec(data{i,1}(1,53:56));   % ACC - Y
    IMU_2(i_2,4)  = shex2dec(data{i,1}(1,57:60));   % ACC - Z

end

% IMU 3
```

```
if ID{i,1}(1,9:12) == '8901'

    i_3 = i_3 + 1;
    % Interation 0                                      % DATA
    IMU_3(i_3,2)  = shex2dec(data{i,1}(1,1:4));     % ACC - X
    IMU_3(i_3,3)  = shex2dec(data{i,1}(1,5:8));     % ACC - Y
    IMU_3(i_3,4)  = shex2dec(data{i,1}(1,9:12));    % ACC - Z

    i_3 = i_3 + 1;
    % Interation 1                                      % DATA
    IMU_3(i_3,2)  = shex2dec(data{i,1}(1,13:16));   % ACC - X
    IMU_3(i_3,3)  = shex2dec(data{i,1}(1,17:20));   % ACC - Y
    IMU_3(i_3,4)  = shex2dec(data{i,1}(1,21:24));   % ACC - Z

    i_3 = i_3 + 1;
    % Interation 2                                      % DATA
    IMU_3(i_3,2)  = shex2dec(data{i,1}(1,25:28));   % ACC - X
    IMU_3(i_3,3)  = shex2dec(data{i,1}(1,29:32));   % ACC - Y
    IMU_3(i_3,4)  = shex2dec(data{i,1}(1,33:36));   % ACC - Z

    i_3 = i_3 + 1;
    % Interation 3                                      % DATA
    IMU_3(i_3,2)  = shex2dec(data{i,1}(1,37:40));   % ACC - X
    IMU_3(i_3,3)  = shex2dec(data{i,1}(1,41:44));   % ACC - Y
    IMU_3(i_3,4)  = shex2dec(data{i,1}(1,45:48));   % ACC - Z

    i_3 = i_3 + 1;
    % Interation 4                                      % DATA
    IMU_3(i_3,2)  = shex2dec(data{i,1}(1,49:52));   % ACC - X
    IMU_3(i_3,3)  = shex2dec(data{i,1}(1,53:56));   % ACC - Y
    IMU_3(i_3,4)  = shex2dec(data{i,1}(1,57:60));   % ACC - Z

end

% IMU 4
if ID{i,1}(1,9:12) == '3456'

    i_4 = i_4 + 1;
    % Interation 0                                      % DATA
    IMU_4(i_4,2)  = shex2dec(data{i,1}(1,1:4));     % ACC - X
    IMU_4(i_4,3)  = shex2dec(data{i,1}(1,5:8));     % ACC - Y
    IMU_4(i_4,4)  = shex2dec(data{i,1}(1,9:12));    % ACC - Z

    i_4 = i_4 + 1;
    % Interation 1                                      % DATA
    IMU_4(i_4,2)  = shex2dec(data{i,1}(1,13:16));   % ACC - X
    IMU_4(i_4,3)  = shex2dec(data{i,1}(1,17:20));   % ACC - Y
    IMU_4(i_4,4)  = shex2dec(data{i,1}(1,21:24));   % ACC - Z

    i_4 = i_4 + 1;
    % Interation 2                                      % DATA
    IMU_4(i_4,2)  = shex2dec(data{i,1}(1,25:28));   % ACC - X
    IMU_4(i_4,3)  = shex2dec(data{i,1}(1,29:32));   % ACC - Y
    IMU_4(i_4,4)  = shex2dec(data{i,1}(1,33:36));   % ACC - Z

    i_4 = i_4 + 1;
    % Interation 3                                      % DATA
    IMU_4(i_4,2)  = shex2dec(data{i,1}(1,37:40));   % ACC - X
    IMU_4(i_4,3)  = shex2dec(data{i,1}(1,41:44));   % ACC - Y
```

```
            IMU_4(i_4,4)  = shex2dec(data{i,1}(1,45:48));   % ACC - Z

            i_4 = i_4 + 1;
            % Interation 4                                  % DATA
            IMU_4(i_4,2)  = shex2dec(data{i,1}(1,49:52));   % ACC - X
            IMU_4(i_4,3)  = shex2dec(data{i,1}(1,53:56));   % ACC - Y
            IMU_4(i_4,4)  = shex2dec(data{i,1}(1,57:60));   % ACC - Z

        end
end
```

B4 - Processamento de Dados do *IMU*

```
run('log_xctu_5P_AC');
set(0,'defaultlinelinewidth',1);

%% Filtro

cut_off = 20;
sample_rate = 125;
n = 4;
Wn = cut_off/(sample_rate/2);
[b,a] = butter(n,Wn,'low');

%% Equivalência dos Eixos

g = 9.81;
ang = 38.1;
fix = 90 - ang;

sam_imu_3 = [1:1:size(IMU_3,1)]';

% EQU
EQU_3(:,2) = -IMU_3(:,2).*g./4096;
EQU_3(:,3) =  IMU_3(:,3).*g./4096;
EQU_3(:,4) =  IMU_3(:,4).*g./4096;

% EQU Filtrado
EQU_3f(:,2) = (filtfilt(b,a,-IMU_3(:,2))).*g./4096;
EQU_3f(:,3) = (filtfilt(b,a, IMU_3(:,3))).*g./4096;
EQU_3f(:,4) = (filtfilt(b,a, IMU_3(:,4))).*g./4096;

%% Separação das tentativas

clear row_3X
row_3X{1}  = 955:1049;
row_3X{2}  = 4246:4325;
row_3X{3}  = 7691:7776;
row_3X{4}  = 11049:11135;
row_3X{5}  = 14807:14871;
row_3X{6}  = 18150:18227;
row_3X{7}  = 21703:21783;
row_3X{8}  = 25102:25190;
row_3X{9}  = 28872:28950;
row_3X{10} = 32278:32355;
```

```
clear SEP_3X SEP_3Y SEP_3Z SEP_3Xf SEP_3Yf SEP_3Zf
for rep = 1:10
    SEP_3X{rep}  = EQU_3(row_3X{rep},3);
    SEP_3Y{rep}  = EQU_3(row_3X{rep},3);
    SEP_3Z{rep}  = EQU_3(row_3X{rep},4);

    SEP_3Xf{rep}  = EQU_3f(row_3X{rep},3);
    SEP_3Yf{rep}  = EQU_3f(row_3X{rep},3);
    SEP_3Zf{rep}  = EQU_3f(row_3X{rep},4);
end

for rep = 1:10
    SS(:,rep) = size(SEP_3X{rep},1);
    [max_s(rep), loc_s(rep)] = max(SEP_3X{rep});
    [max_sf(rep), loc_sf(rep)] = max(SEP_3Xf{rep});
end

center = 40;
fig = 1;
figure(fig)
for rep = 2
    hold on
    [ax,h1,h2] =
plotyy(1:SS(:,rep),SEP_3Xf{rep},1:SS(:,rep),SEP_3Xf{rep}/g);
end

hold off
delete(h2)

clear title
xlabel('Amostras');
ylabel(ax(1),'Aceleração [m/s²]');
ylabel(ax(2),'Aceleração [g]');
grid on

%% Integração

for rep = 1:10
    M_1X{rep} = cumtrapz(SEP_3X{rep} - mean(SEP_3X{rep},1));   %
Integra aceleração sem média = velocidade
    M_2X{rep} = cumtrapz(M_1X{rep} - mean(M_1X{rep},1));       %
Integra velocidade sem média = posição
    M_3X{rep} = M_2X{rep} - mean(M_2X{rep},1);                 %
Posição sem média

    M_1Y{rep} = cumtrapz(SEP_3Y{rep} - mean(SEP_3Y{rep},1));   %
Integra aceleração sem média = velocidade
    M_2Y{rep} = cumtrapz(M_1Y{rep} - mean(M_1Y{rep},1));       %
Integra velocidade sem média = posição
    M_3Y{rep} = M_2Y{rep} - mean(M_2Y{rep},1);                 %
Posição sem média

    M_1Z{rep} = cumtrapz(SEP_3Z{rep} - mean(SEP_3Z{rep},1));   %
Integra aceleração sem média = velocidade
    M_2Z{rep} = cumtrapz(M_1Z{rep} - mean(M_1Z{rep},1));       %
Integra velocidade sem média = posição
    M_3Z{rep} = M_2Z{rep} - mean(M_2Z{rep},1);                 %
Posição sem média
```

```
    clear xdata_1 ydata_1 x0_1 fun_1 x_1 xdata_2 ydata_2 x0_2 fun_2
x_2 xdata_3 ydata_3 x0_3 fun_3 x_3
    x0_1 = [0, 0];
    xdata_1 = (1:SS(:,rep))';
    ydata_1 = M_3X{rep};
    fun_1 = @(x_1,xdata_1)(x_1(2)^2)*xdata_1 + x_1(1)*xdata_1;
    x_1 = lsqcurvefit(fun_1, x0_1, xdata_1, ydata_1);

    x0_2 = [0, 0];
    xdata_2 = (1:SS(:,rep))';
    ydata_2 = M_3Y{rep};
    fun_2 = @(x_2,xdata_2)(x_2(2)^2)*xdata_2 + x_2(1)*xdata_2;
    x_2 = lsqcurvefit(fun_2, x0_2, xdata_2, ydata_2);

    x0_3 = [0, 0];
    xdata_3 = (1:SS(:,rep))';
    ydata_3 = M_3Z{rep};
    fun_3 = @(x_3,xdata_3)(x_3(2)^2)*xdata_3 + x_3(1)*xdata_3;
    x_3 = lsqcurvefit(fun_3, x0_3, xdata_3, ydata_3);

    DRIFTX{rep} = M_3X{rep} - fun_1(x_1,xdata_1);
    DRIFTY{rep} = M_3Y{rep} - fun_2(x_2,xdata_2);
    DRIFTZ{rep} = M_3Z{rep} - fun_3(x_3,xdata_3);
end

% Alinhando as tentativas
center_x = 1;
center_y = 0;
for rep = 1:10

    [max_drift(rep), loc_max_drift(rep)] = max(DRIFT{rep});
    [min_drift(rep), loc_min_drift(rep)] = min(DRIFT{rep});

    TRIALS_I{rep} =
DRIFT{rep}(loc_min_drift(rep):loc_max_drift(rep),1);
    [max_trials_i(rep), loc_max_trials_i(rep)] = max(TRIALS_I{rep});
    [min_trials_i(rep), loc_min_trials_i(rep)] = min(TRIALS_I{rep});

    NORM_3XI{rep} = circshift((TRIALS_I{rep} -
min_trials_i(rep))/(max_trials_i(rep) - min_trials_i(rep)) + center_y,
[center_x - loc_min_trials_i(rep), 0]);
    fim_i(rep) = loc_max_trials_i(rep) - loc_min_trials_i(rep);
    sam_i{rep} = 1:(fim_i(rep)+1);
end

run('test_resample');
N = 1;
sd_p_i = (MeanV + N*MeanP )';
sd_n_i = (MeanV - N*MeanP )';
go = length(MeanV);

fig = 2;
figure(fig)
for rep = 1:10
    hold on
    plot(sam_i{rep}/(fim_i(rep)+1), NORM_3XI{rep},'r')
%
plot(sam_i_2{rep}/(fim_i_2(rep)+1),NORM_3XI_2{rep},'b','DisplayName','
Tentativas Sem FOB')
end
```

```
fill([(1:go)/go fliplr((1:go)/go)],
[sd_p_i_5(1:go)/max(sd_p_i_5(1:go))
fliplr(sd_n_i_5(1:go)/max(sd_n_i_5(1:go)))], 'k') % Plota a área
hachurada.
alpha(.20);
% Transparência da área hachurada. 1 = 100% Opaco, 0 = 100%
Transparente

plot((1:go)/go, MeanV(1:go)/max(MeanV(1:go)),'b')             % Média
IMU
plot((1:go_2)/go_2, MeanV_2(1:go_2)/max(MeanV_2(1:go_2)),'r') % Média
IMU_COMP
plot((1:go)/go, sd_p_i_5(1:go)/max(sd_p_i_5(1:go)),'k')       % + N
Desvio(s) Padrão(ões)
plot((1:go)/go, sd_n_i_5(1:go)/max(sd_n_i_5(1:go)),'k')       % - N
Desvio(s) Padrão(ões)
hold off

xlabel('Tempo [s]');
ylabel('Posição Normalizada')
grid on
```

B5 - Importação e Processamento de Dados do *FOB*

```
%% Inicialização dos arquivos a serem processados.

extension = '*.*';
title     = 'Select the FOB + IMU acquisition file';
directory = 'C:';

[FileName,PathName] = uigetfile(extension,title,directory);
var = '%04d';
half_path = strcat(PathName,var);
full_path = strrep(half_path, '\', '/');

REP = 10;   % Repetições

srow = 52;  % Linha de começo dos dados
erow = inf; % Linha de término dos dados

% Importa os dados de todas as repetições e armazena na matriz A_2
i = 0;
for(j=0:REP-1)
    i = i + 1;
    X = sprintf(full_path,j);
    dados = import_fob(num2str(X),srow,erow); % Sintaxe:
importfile(filename, startRow, endRow)
    A_2{i} = dados;
end

%% Sensores
```

```
X  = 2; Y  = 3; Z  = 4;  % Posição dos sensores 1 e 3 (antebraço)
AZ = 8; EL = 9; RL = 10; % Ângulo dos sensores 1 e 3 (antebraço)

% X  = 5; Y  = 5; Z  = 6;    % Posição dos sensores 2 e 4 (braço)
% AZ = 11; EL = 12; RL = 13; % Ângulo dos sensores 2 e 4 (braço)

%% Filtro

cut_off = 5;
sample_rate = 103;
n = 4;
Wn = cut_off/(sample_rate/2);
[b,a] = butter(n,Wn);

%% Posição

% Detecta a menor dimensão para poder concatenar os vetores
for rep_2 = 1:10;
    dim_2(1,rep_2) = size(A_2{rep_2},1);
end
short_2 = min(dim_2);
sam_d_2 = 1:short_2;

% Concatenação para possibilitar o cálculo da média de sd
for rep_2 = 1:10;
    Dx_2(:,rep_2) = A_2{rep_2}(sam_d_2,X);
    Dy_2(:,rep_2) = A_2{rep_2}(sam_d_2,Y);
    Dz_2(:,rep_2) = A_2{rep_2}(sam_d_2,Z);
end

% Média (tem que ser horizonatal, usar ')
m_x_2 = mean(Dx_2,2)';
m_y_2 = mean(Dy_2,2)';
m_z_2 = mean(Dz_2,2)';

% +- N Desvio(s) Padrão(ões)
N_2 = 1; % Número de desvios padrões
sd_p_x_2 = (m_x_2' + N_2*abs(std(Dx_2,1,2)))';
sd_n_x_2 = (m_x_2' - N_2*abs(std(Dx_2,1,2)))';
sd_p_y_2 = (m_y_2' + N_2*abs(std(Dy_2,1,2)))';
sd_n_y_2 = (m_y_2' - N_2*abs(std(Dy_2,1,2)))';
sd_p_z_2 = (m_z_2' + N_2*abs(std(Dz_2,1,2)))';
sd_n_z_2 = (m_z_2' - N_2*abs(std(Dz_2,1,2)))';

%% Erros Máximos das Médias

% Média X
dif_x = abs(m_x - m_x_2)';
[max_err_x,loc_x] = max(dif_x);
err_x = abs(m_x_2(loc_x) - m_x(loc_x))*100/m_x(loc_x);

% Média Y
dif_y = abs(m_y - m_y_2)';
[max_err_y,loc_y] = max(dif_y);
err_y = abs(m_y_2(loc_y)-m_y(loc_y))*100/m_y(loc_y);

% Média Z
dif_z = abs(m_z - m_z_2)';
```

```
[max_err_z,loc_z] = max(dif_z);
err_z = abs(m_z_2(loc_z) - m_z(loc_z))*100/m_z(loc_z);

%%
% Posição Eixo X
fig = fig + 1;
figure(fig)
for rep = 1:REP
    hold on
    plot(sam_d,Dx(:,rep),'Color', [0, 0.4470, 0.7410]
    plot(sam_d_2,Dx_2(:,rep),'b')
end

fill([sam_d fliplr(sam_d)], [sd_p_x fliplr(sd_n_x)], 'k'); % Plota a
área hachurada.
alpha(.20);                                                 %
Transparência da área hachurada.

plot(sam_d,m_x,'Color',[0.6350, 0.0780, 0.1840],'LineWidth',1)
% Média FOB
plot(sam_d_2,m_x_2,'r','LineWidth',1)
% Média IMU + FOB
errorbar(sam_d(loc_x),m_x(loc_x),max_err_x,'Color',[0.4660, 0.6740,
0.1880]) % Erro máximo

plot(sam_d,sd_p_x,'k') % + N Desvio(s) Padrão(ões)
plot(sam_d,sd_n_x,'k') % - N Desvio(s) Padrão(ões)

hold off
clear title
title('Posição Y FOB');
xlabel('Amostras');
ylabel('Deslocamento [m]');
grid on


%%
% Posição Eixo Y
fig = fig + 1;
figure(fig)
for rep = 1:REP
    hold on
    plot(sam_d,Dy(:,rep),'Color',[0.9290, 0.6940, 0.1250])
    plot(sam_d_2,Dy_2(:,rep),'y')
end

fill([sam_d fliplr(sam_d)], [sd_p_y fliplr(sd_n_y)], 'k'); % Plota a
área hachurada.
alpha(.20);                                                 %
Transparência da área hachurada.

plot(sam_d,m_y,'Color',[0.6350, 0.0780, 0.1840],'LineWidth',1)
% Média FOB
plot(sam_d_2,m_y_2,'r','LineWidth',1)
% Média IMU + FOB
errorbar(sam_d(loc_y),m_y(loc_y),max_err_y,'Color',[0.4660, 0.6740,
0.1880]) % Erro máximo

plot(sam_d,sd_p_y,'k') % + N Desvio(s) Padrão(ões)
```

```
plot(sam_d,sd_n_y,'k') % - N Desvio(s) Padrão(ões)

hold off
clear title
title('Posição Y FOB');
xlabel('Amostras');
ylabel('Deslocamento [m]');
grid on

%%
% Posição Eixo Z
fig = fig + 1;
figure(fig)
for rep = 1:REP
    hold on
    plot(sam_d,Dz(:,rep),'Color',[0.4940, 0.1840, 0.5560])
    plot(sam_d_2,Dz_2(:,rep),'m')
end

fill([sam_d fliplr(sam_d)], [sd_p_z fliplr(sd_n_z)], 'k'); % Plota a
área hachurada.
alpha(.20);                                                 %
Transparência da área hachurada.

plot(sam_d,m_z,'Color',[0.6350, 0.0780, 0.1840],'LineWidth',1)
% Média FOB
plot(sam_d_2,m_z_2,'r','LineWidth',1)
% Média IMU + FOB
errorbar(sam_d(loc_z),m_z(loc_z),max_err_z,'Color',[0.4660, 0.6740,
0.1880]) % Erro máximo

plot(sam_d,sd_p_z,'k') % + N Desvio(s) Padrão(ões)
plot(sam_d,sd_n_z,'k') % - N Desvio(s) Padrão(ões)

hold off
clear title
title('Posição Z FOB');
xlabel('Amostras');
ylabel('Deslocamento [m]');
grid on
```

ANEXO C - PROCEDIMENTO DE UTILIZAÇÃO DO *ARDUIMU*

1. Conectar a extremidade correspondente do cabo FTDI ao *ArduIMU* e a outra extremidade à porta USB do computador;

2. Lançar a *IDE* do *Arduino* e carregar as quatro abas do código descrito no ANEXO A - CÓDIGO EM *ARDUNIO* e fazer o *upload* destes códigos no microprocessador como indicado pelo destaque em vermelho mostrado na Figura C-2;

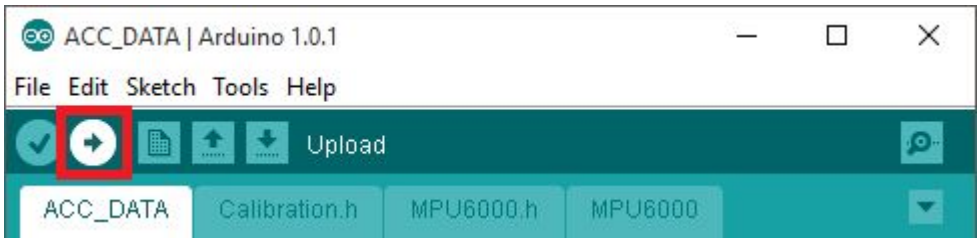


Figura C-2 - *Indicação com destaque em vermelho do botão para fazer o upload dos códigos no microprocessador.*

3. Quando necessário recalibrar os *ArduIMUs*, procedendo conforme descrito no capítulo 3.3 e executar o código em *MATLAB* descrito no ANEXO B - CÓDIGOS EM *MATLAB*, Calibração;

4. Obtidos os valores de O*ffset* e Ganho alterar os mesmos na Aba 2: *Calibration.h* do ANEXO A - CÓDIGO EM *ARDUNIO* nas variáveis correspondentes:

```
#define ACCELEROMETER_X_OFFSET 0.0
#define ACCELEROMETER_Y_OFFSET 0.0
#define ACCELEROMETER_Z_OFFSET 0.0
#define ACCELEROMETER_X_SCALE 1.0
#define ACCELEROMETER_Y_SCALE 1.0
#define ACCELEROMETER_Z_SCALE 1.0
```

e fazer o *upload* dos códigos com os valores de *Offset* e Ganho atualizados no microprocessador;

5. Quando todas as unidades estiverem com os devidos códigos gravados do microprocessador já podem ser utilizadas;

6. Para adquirir os dados de aceleração com as unidades conectar o *XBee* coordenador com o *USB Explorer* à uma porta USB do computador;

7. Lançar o software *XCTU* para gravar as informações que são transmitidas pela porta USB adicionando o módulo coordenador *XBee* como descrito na Figura C-7. Clicar em *Finish* na próxima janela e resetar o *USB Explorer* clicando no botão presente em sua base quando solicitado;

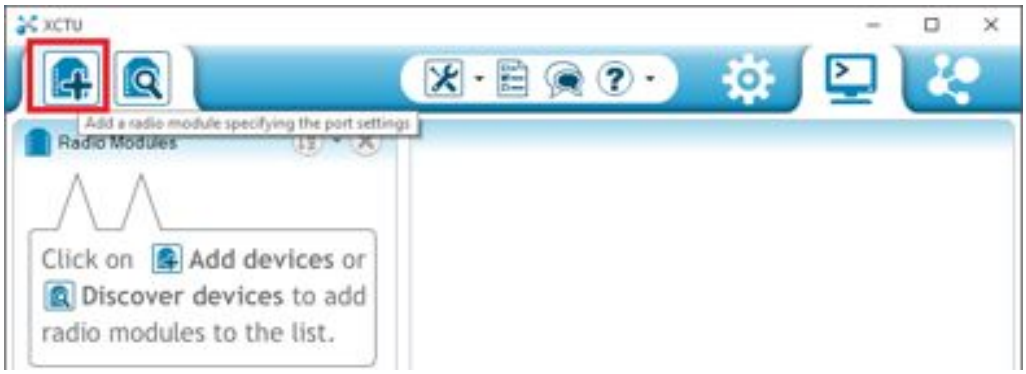


Figura C-7 - *Botão destacado em vermelho para adicionar módulos XBee conectados ao computador.*

8. Selecionar a aba do *console* marcada com o número 1 em vermelho na Figura C-8;

9. Abrir a conexão serial com o módulo coordenador clicando na marcação do número 2 em vermelho mostrado na Figura C-8;

10. Gravar os dados recebidos em um arquivo *.log* clicando na marcação do número 3 em vermelho mostrado na Figura C-8;

Figura C-8 - *Marcação de número 1 para selecionar a aba do console. A marcação de número 2 abre a conexão serial entre computador e módulo XBee conectado. A marcação de número 3 grava os dados recebidos pela porta USB do computador.*

11. Quando a aquisição for concluída clicar novamente na marcação de número 3 para encerrar a gravação dos dados;

12. Para importar os dados adquiridos com *ArduIMU* executar o código descrito no ANEXO B - CÓDIGOS EM *MATLAB*, Importação de Dados do *IMU* selecionando o arquivo *.log* criado no item 10;

13. Para processar os dados do *ArduIMU* executar o código descrito no ANEXO B - CÓDIGOS EM *MATLAB*, Processamento de Dados do *IMU* e alterá-lo conforme desejado;

14. Para importar e processar os dados adquiridos com o *FOB* executar o código descrito no ANEXO B - CÓDIGOS EM *MATLAB*, Importação e Processamento de Dados do *FOB* e alterá-lo conforme desejado;