

## Determinação e simulação do movimento em três dimensões a partir de sensores de baixo custo

Eduardo Barros Guimarães<sup>1</sup> & Gustavo Passos Ribeiro<sup>2</sup>  
(orientador: José Marcelo Gomes)<sup>3</sup>

**Resumo:** Os sistemas eletrônicos modernos permitiram a criação de sensores de baixo custo como o MPU-6050 que podem ser utilizados para medir a aceleração e a velocidade angular. A partir desses dados, é possível obter a posição e a rotação realizada pelo corpo utilizando dos artifícios do cálculo diferencial e integral e das leis da mecânica. O processamento da rotação foi feito utilizando quatérnions a fim de evitar a perda de graus de liberdade do movimento. Com isso, desenvolveu-se um sistema de monitoramento e simulação de movimentos de amplitude menor que 2 m, captados pelo sensor MPU-6050 e gravados com o Raspberry Pi 4, capaz de gerar gráficos em duas e três dimensões. O sistema gera simulações da trajetória, da rotação isolada do movimento e de ambos, podendo criar imagens estáticas ou dinâmicas.

**Palavras Chave:** simulação, acelerômetro, giroscópio, quatérnions, mecânica.

**Abstract:** Modern day electronical systems have allowed the creation of low cost sensors such as MPU-6050, which can be used to measure acceleration and angular velocity. By these data, it is possible to obtain position and rotation performed by the body using the rules of differential and integral calculus and the laws of mechanics. The processing of rotation was done using quaternions, for the sole purpose of avoiding the loss of degrees of freedom. Therefore, a monitoring and motion simulation system was developed using Raspberry Pi 4 and MPU-6050 sensor, for movements in a range of 2 m or less. The system is capable of plotting data in two and three dimensions, and can simulate trajectory, rotation devoid of movement, and both at the same time, inasmuch as creating static and dynamic images.

**Keywords:** simulation, accelerometer, gyroscope, quaternions, mechanics.

### 1. INTRODUÇÃO

A Mostra Brasileira de Foguetes (MOBFOG) da Olimpíada Brasileira de Astronomia e Astronáutica (OBA) é uma olimpíada escolar de caráter experimental, em que são lançados modelos de foguetes de forma oblíqua, partindo de uma base de lançamento. Além disso, o foguete e a base devem ser feitos pelos competidores, em busca de alcançar a maior distância horizontal.

As dificuldades de medição e melhorias nos foguetes usados nessa competição – devido à falta de informações úteis sobre o movimento que ocorre entre o começo e o fim do lançamento – inspiraram, juntamente com os conhecimentos das áreas de Mecânica e Matemática, a criação desse projeto. As leis do cálculo permitem determinar, a partir da aceleração, tanto a velocidade quanto a posição de um corpo. Ademais, é possível descobrir, a partir da velocidade angular, a rotação do corpo em torno de si próprio em cada instante (Halliday, 2012). Para a representação das rotações, existem dois métodos comumente utilizados em simulações na indústria, sendo eles os ângulos de Euler e os quatérnions. Sabendo que o problema chamado “Gimbal Lock”, em que ocorre perda de graus de liberdade, acontece quando se usam esses ângulos, pareceu mais vantajoso trabalhar com os quatérnions. Além disso, os

---

<sup>1</sup> Aluno do 3º ano do Colégio de Aplicação da Universidade Federal de Viçosa(CAp-COLUNI).

<sup>2</sup> Aluno do 3º ano do CAp-COLUNI.

<sup>3</sup> Professor de Física do 3º ano, formado em Ciência da Computação.

trabalhos de Madgwick (2012), Nelli (2020) e Biasi e Gattass (2002) serviram de base para a idealização e realização do projeto.

Visa-se usar sensores de baixo custo capazes de perceber a aceleração e a velocidade angular para determinar os movimentos translacionais e rotacionais realizados por ele. A partir dos dados obtidos e com o auxílio de linguagens de programação, objetiva-se desenvolver um sistema de monitoramento para o sensor que seja capaz de representar os dados em gráficos bidimensionais, bem como a trajetória em gráficos tridimensionais.

O domínio dos movimentos estudados restringe-se a amplitudes menores do que 2m, o que torna o estudo mais prático e é um bom ponto de partida para o desenvolvimento desse sistema, bem como fornece uma base sólida para expansões futuras. Os possíveis usos futuros desse sistema estão atrelados à própria competição supracitada, podendo ser usado para avaliar a trajetória e realizar melhorias nos foguetes, bem como diversas aplicações na robótica, como simulações de realidade virtual e para gravação de movimentos corporais usados em modelagem 3D para filmes e jogos.

## **2. OBJETIVOS**

### **2.1 OBJETIVO GERAL**

O objetivo geral desse estudo é determinar a posição do sensor relativa à origem do movimento com tanta precisão quanto possível, bem como saber sua orientação no espaço por meio dos dados obtidos dos sensores.

### **2.2 OBJETIVOS ESPECÍFICOS**

Tem-se como objetivos específicos montar, criar e desenvolver um sistema de leitura, gravação e renderização de dados de acelerômetros e de giroscópios e um sistema de monitoramento e visualização dos dados obtidos; usar métodos numéricos e estatísticos para determinar a posição e a rotação do sensor com precisão; avaliar os dados obtidos e fazer correções na computação da posição.

## **3. METODOLOGIA**

### **3.1 REPRESENTAÇÃO TEÓRICA DO MOVIMENTO**

A modelagem matemática do sistema de monitoramento toma por premissas as seguintes afirmações:

1. A posição inicial do corpo é sempre a origem;
2. A velocidade inicial do corpo é sempre zero;
3. O espaço, num sentido físico, é bem representado pelo espaço vetorial  $\mathbb{R}^3$ ;
4. A aceleração e a velocidade angular são funções contínuas do tempo.

As medições feitas pelo sensor dizem respeito ao seu plano de referência, portanto, as medições não coincidem com os eixos fixos determinados na origem. Para obter os valores corretos da aceleração são usados os dados da velocidade angular fornecidos pelo giroscópio do sensor MPU-6050. É possível determinar a rotação em torno de cada direção pelas seguintes igualdades:

$$\int \omega_x dt = \theta_x, \quad \int \omega_y dt = \theta_y, \quad \int \omega_z dt = \theta_z.$$

Como o conjunto de dados não é contínuo, utilizou-se o método de integração trapezoidal (EPPERSON, 2013). Dado um conjunto de  $n$  pares ordenados  $(x_i, f(x_i))$ , onde  $1 \leq i \leq n$ , e  $\Delta x_i = x_{i+1} - x_i$ , então vale:

$$\lim_{\Delta x_i \rightarrow 0, n \rightarrow \infty} \sum_{i=1}^{n-1} \frac{(f(x_i) + f(x_{i+1})) \cdot \Delta x_i}{2} = \int_{x_1}^{x_n} f(x) dx.$$

Essa igualdade foi utilizada sempre que foi necessário fazer a integração de um conjunto de pontos. Os ângulos  $\theta_x, \theta_y, \theta_z$  indicam, em relação ao eixo do primeiro instante do sensor, quanta rotação o corpo sofreu nos eixos  $x, y$  e  $z$ . Dentre os meios matemáticos para tratar as rotações, havia três opções disponíveis: os ângulos de Euler, os quatérnions e os ângulos de *Yaw, Pitch & Roll*. O primeiro e o último sofrem de um fenômeno chamado *Gimbal Lock*, ocasionado pela perda de graus de liberdade de um sistema.

Diante disso, optou-se por usar os quatérnions, que são uma extensão dos números complexos. Os quatérnions, em contraste com os números reais, caracterizam-se pela perda da comutatividade na multiplicação, isto é, sobre produto de dois quatérnions  $p$  e  $q$ , não é necessário que  $pq$  seja igual a  $qp$ . Formalmente, esses números, segundo Hamilton (1844), podem ser expressos sob a forma  $a + bi + cj + dk$ , onde  $a, b, c, d \in \mathbb{R}$  e  $i, j$  e  $k$  são unidades imaginárias que respeitam  $i^2 = j^2 = k^2 = ijk = -1$ . Por consequência lógica, é evidente que  $jk = i$  e que  $ij = k$ .

A consequência dessa definição é, justamente, a não comutatividade do produto desses números. Dessa forma, a multiplicação de dois quatérnions  $q = a + bi + cj + dk$  e  $p = a_1 + b_1i + c_1j + d_1k$  é:

$$qp = \begin{bmatrix} a & -b & -c & -d \\ b & a & -d & c \\ c & d & a & -b \\ d & -c & b & a \end{bmatrix} \begin{bmatrix} a_1 \\ b_1 \\ c_1 \\ d_1 \end{bmatrix} \cdot \begin{bmatrix} 1 \\ i \\ j \\ k \end{bmatrix}.$$

Os vetores do  $\mathbb{R}^3$  serão representados, portanto, pelo subespaço formado pelos quatérnions sem parte real, isto é, todas as combinações lineares de  $i, j$  e  $k$ .

Segundo o Teorema da Rotação de Euler, quaisquer três rotações em eixos distintos podem ser descritas por um único eixo unitário  $\hat{u}$  e um ângulo  $\theta$  em torno desse eixo (EULER, 1775). Representando o vetor  $\hat{u}$  pelo quatérnion sem parte real  $bi + cj + dk$ , onde  $b, c, d$  são os respectivos valores do vetor em  $\mathbb{R}^3$ . Pela extensão da Identidade de Euler, vale a seguinte igualdade:

$$q = e^{\frac{\theta}{2}(bi+cj+dk)} = \cos \frac{\theta}{2} + (bi + cj + dk) \sin \frac{\theta}{2}.$$

Seja  $p$  o quatérnion que representa o vetor que se deseja fazer uma rotação horária de ângulo  $\theta$  em torno do eixo  $\hat{u}$ . Então, é correto afirmar que:

$$p_1 = qp\bar{q},$$

onde o vetor  $p_1$  é a rotação de  $p$  em torno do eixo  $\hat{u}$  por um ângulo de  $\theta$  no sentido horário, e  $\bar{q}$  denota a conjugação de  $q$ , isto é, se  $q = a + bi + cj + dk$ , então  $\bar{q} = a - bi - cj - dk$ .

O Teorema de Rotação, junto com a extensão da Identidade de Euler e o produto de quatérnion por um vetor representando uma rotação, permite compreender que os quatérnions do sensor podem ser obtidos interpretando os ângulos  $\theta_x, \theta_y, \theta_z$  como ângulos em torno dos eixos unitários  $x, y, z$  quaisquer, respectivamente. Algebricamente, isso é:

$$\begin{aligned} q_x &= \cos \frac{\theta_x}{2} + x \sin \frac{\theta_x}{2}, & q_y &= \cos \frac{\theta_y}{2} + q_x y \bar{q}_x \sin \frac{\theta_y}{2}, \\ q_z &= \cos \frac{\theta_z}{2} + q_y q_x z \bar{q}_x \bar{q}_y \sin \frac{\theta_z}{2}. \end{aligned}$$

E o quatérnion de rotação final é dado pelo produto  $q = q_x q_y q_z$ . Os ângulos medem quanta rotação houve em cada eixo relativo à configuração inicial, portanto, o quatérnion que representa o vetor de aceleração  $a = a_x i + a_y j + a_z k$  relativo ao sensor, precisa receber a rotação inversa desses ângulos para que aponte para a direção correta nos eixos de referência. Logo, o vetor obtido é  $a_r = qa\bar{q}$ , que é a aceleração apontando para a direção real.

A obtenção da velocidade por meio dessa aceleração foi feita usando o método do trapézio supracitado, faz-se valer:

$$v = v_0 + \int_{t_0}^{t_1} a \, dt.$$

Da mesma forma, a posição é obtida pela seguinte equação:

$$S = S_0 + \int_{t_0}^{t_1} v \, dt.$$

Na reconstrução do conjunto após a integração, perde-se sempre uma unidade de dados, e, para tanto, adiciona-se como primeiro índice da velocidade o vetor nulo e na posição o ponto da origem.

### 3.2 MONTAGEM DO HARDWARE

Visando manter os custos de produção do sistema baixos, foi escolhido o sensor MPU-6050 GY-521, apresentado na Figura 1, encontrado na faixa de preço de R\$ 10,00 a R\$ 20,00 no mercado. Ele possui seis pinos, fora os de alimentação do circuito, sendo os dois principais SDA (Serial Data) e SCL (Serial Clock), que servem para a comunicação I<sup>2</sup>C (Inter-Integrated Circuit). Essa é uma forma de comunicação serial em que os pinos SDA possuem um caminho bidirecional de dados e o SCL envia pulsos de clock indicando quando efetuar uma operação. Esse módulo apresenta

também um redutor de tensão embutido, corrigindo a entrada 5 V para a faixa de 2,38 V a 3,46 V.

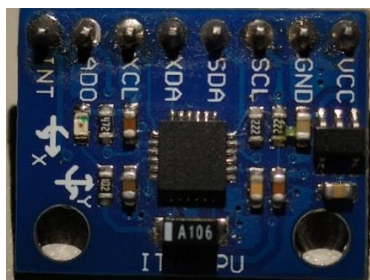


Figura 1: Imagem do Sensor MPU-6050.

O sensor conta com dados de giroscópio e acelerômetro, que medem, respectivamente, a velocidade angular e a aceleração linear em cada eixo. Os dados do giroscópio dizem respeito à velocidade angular em torno dos eixos x,y e z, e é possível escolher entre os intervalos de leitura  $\pm 250$ ,  $\pm 500$ ,  $\pm 1000$  ou  $\pm 2000$  /s. Já o acelerômetro mede a aceleração em cada eixo relativo ao plano do sensor, e a escala de medição pode ser programada entre  $\pm 2g$ ,  $\pm 4g$ ,  $\pm 8g$  ou  $\pm 16g$ , onde  $g$  é a aceleração gravitacional ( $\sim 9,81 \text{ m/s}^2$ ) (INVENSENSE, 2015)..

Além disso, o chip apresenta o módulo DMP (Digital Motion Processor), uma unidade de pré-processamento dos dados do sensor que filtra os dados brutos coletados do giroscópio e do acelerômetro e produz dados corrigidos. Não foi possível descobrir exatamente como esse módulo corrige os dados, porque a programação dele não é em código aberto. Os dados disponibilizados pela unidade são o giroscópio filtrado, a aceleração filtrada, o quatérnion, a temperatura e o magnetômetro, caso haja um instalado.

Implementou-se o sistema com a placa Raspberry Pi 4, a qual custa cerca de R\$ 500,00. Ele possui 2 GB de memória SDRAM, processador Broadcom 1,5 GHz 64-bit. Esse módulo apresenta pinos SDA e SCL para comunicação I2C, que foram utilizados para ler os dados enviados pelo sensor MPU-6050. Uma versão modificada da biblioteca *open-source i2cdevlib* para esse módulo ajudou a fazer o programa de registro de dados para o Raspberry. Assim, os dados são salvos em arquivos de texto, com extensão “.txt”.

Os LEDs e o botão servem de interface entre o operador e o circuito, indicando os estados do sensor. A montagem da placa foi desenvolvida soldando o circuito numa placa de fenolite perfurada da forma como está disposto na Figura 2.

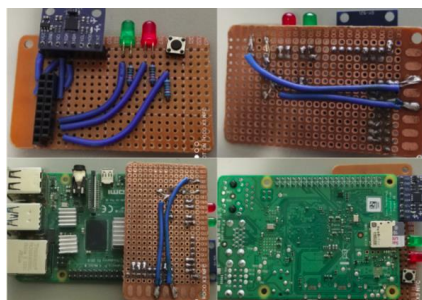


Figura 2: Circuito final, parte inferior.

### 3.3 SOFTWARE PARA VISUALIZAÇÃO DE DADOS

O sistema de coleta desenvolvido gera quatro arquivos que contêm os quatérnions, os dados do giroscópio, a aceleração com gravidade e a aceleração sem

gravidade. Eles estão formatados de forma a possibilitar a distinção entre células e colunas pela biblioteca de Python chamada *pandas*, que transforma esses arquivos em tabelas (MCKINNEY et al., 2021). Convém saber que há uma coluna do tempo decorrido em todos esses arquivos.

Essa biblioteca, junto com a *scipy*, foi largamente usada para conseguir manipular as tabelas no programa e realizar as operações necessárias que tornam os dados utilizáveis, como a integração e filtragem da informação. Além disso, a biblioteca *matplotlib*, que simula os gráficos do programa MATLAB (HUNTER et al., 2021), é a grande responsável pela geração dos gráficos e das imagens e simulações tridimensionais. Ela é capaz de ler as tabelas da *pandas* e criar uma relação entre a coluna do tempo e as colunas de dados, fazendo gráficos em função do tempo. Com os dados da posição, essa biblioteca gera os pontos no espaço e vetores que representam as direções do sensor, para perceber a orientação do corpo.

O programa foi desenvolvido de modo a permitir que o operador possa escolher o tipo e a quantidade de gráficos que ele deseja e se quer ou não salvar uma imagem ou um vídeo dos gráficos.<sup>4</sup>

### 3.4 CORREÇÃO VETORIAL DA GRAVIDADE

Como afirmado na seção 3.1, os dados do sensor são referentes ao seu plano de orientação, o que significa que eles não condizem com os eixos de referência da origem. Além disso, o MPU-6050, se disposto com o eixo z coincidente com o eixo perpendicular de referência, medirá zero nos eixos x e y, e medirá  $\mathbf{g}$ , a aceleração gravitacional na superfície da terra, no eixo z.

Para que a medição seja efetiva, é necessário subtrair o vetor da aceleração gravitacional, porque a medição esperada nessa posição é zero, uma vez que o sensor está parado. Nesse sentido, para obter a influência do vetor da aceleração gravitacional em cada eixo, representa-se esse vetor pelo quatérnion  $9.81k$ . Nesse sentido, a soma dos vetores de aceleração do sensor após a rotação deve ser igual a esse quatérnion, isto é, dado que  $\mathbf{a}_{sensor} = a_x i + a_y j + a_z k$  e  $\mathbf{a}_{real} = q \mathbf{a}_{sensor} \bar{q}$ , onde  $q$  representa o quaternion correspondente.

Diante disso, conclui-se que  $\mathbf{g} = q \mathbf{g}' \bar{q}$ , onde o quaternion  $\mathbf{g}'$  é a influência da gravidade em cada eixo do sensor. O efeito de multiplicar  $q$  à esquerda e à direita é a diferença no sentido da rotação. Com essas informações, obtém-se a seguinte igualdade:

$$\mathbf{g}' = \bar{q} \mathbf{g} q$$

Assim, o vetor  $\mathbf{g}$  sofre rotação no sentido contrário, permitindo o encontro do vetor  $\mathbf{g}'$  percebido pelo sensor. A partir disso, subtrai-se o vetor  $\mathbf{g}'$  do vetor  $\mathbf{a}_{sensor}$ , assim,  $\mathbf{a}_{real} = q(\mathbf{a}_{sensor} - \mathbf{g}') \bar{q}$ .

### 3.5 CORREÇÃO DA ACELERAÇÃO

Após um conjunto de testes e observações, percebeu-se que, apesar de apresentar aceleração nula, o sensor detecta uma aceleração maior do que 0, o que, no longo prazo, provoca um aumento constante na velocidade computada. Nesse sentido, as observações permitiram determinar que, em geral, quando o vetor de aceleração

<sup>4</sup> O código-fonte desenvolvido está disponível na plataforma GitHub no link <<https://github.com/Edubgr/MPU6050-MotionTracking>>.

apresenta magnitude menor do que  $0,05 \text{ m/s}^2$ , o mais provável era que o sensor estivesse parado.

Dessa forma, criou-se uma função  $E(t)$ , na qual se tem:

$$E(t) = \begin{cases} 1, & \text{se } |\vec{a}| > 0,05 \\ 0, & \text{se } |\vec{a}| \leq 0,05 \end{cases}$$

Assim,  $E(t)$  indica se, no instante  $t$ , o sensor estava ou não em movimento. Além disso, para tornar a função mais suave, foram usados os filtros de Low Pass e High Pass, facilitando a detecção de períodos estacionários.

### 3.6 INTERPOLAÇÃO DA VELOCIDADE

O erro propagado em decorrência da aceleração nos períodos estacionários é, em geral, bem representado por uma reta, o que motivou os testes com a interpolação entre os pontos iniciais e finais dos períodos de movimento. Antes disso, a regressão linear foi utilizada para o ajuste do erro acumulado, entretanto, interpolar os pontos de início e de final do movimento, apresentou-se mais eficiente e coerente com os movimentos realizados.

Assim, tomam-se os pontos  $P_i = (t_0, v_0)$ ,  $P_f = (t_1, v_1)$ , o polinômio que melhor interpola esses pontos é:

$$P(t) = \frac{v_1 - v_0}{t_1 - t_0} \cdot (t - t_0) + v_0.$$

Portanto, para corrigir a função da velocidade, para todo  $t$  no intervalo  $[t_0, t_1]$ , é necessário que  $E(t) = 0$ , assim, vale:

$$v(t) = \begin{cases} 0, & \text{se } E(t) = 1 ; \\ -P(t) + \int_{t_0}^{t_1} a \, dt, & \text{se } E(t) = 0, t \in [t_0, t_1]. \end{cases}$$

### 3.7 CORREÇÃO DOS QUATÉRNIONS

Os dados da rotação se apresentaram satisfatórios, sendo a única correção necessária a multiplicação pelo quatérnion inverso do quatérnion do instante zero, uma vez que o sensor apresentou um deslize nos primeiros 20 s de coleta a partir do momento que é ligado. Não foi possível perceber diferença na orientação do gráfico multiplicando à esquerda ou à direita. Para poder observar diferentes representações, foram registrados os ângulos de Euler e os quatérnions.

Portanto, no intuito de corrigir esses quatérnions, efetua-se o produto sobre o conjunto  $D$  de dados:

$$q^{-1}D,$$

onde  $q^{-1}$  é o quatérnion inverso do quatérnion  $q$  coletado no primeiro instante após esses 20 segundos.

### 3.8 MÉTODO EXPERIMENTAL DE MEDIÇÃO

Para medir de forma efetiva os movimentos realizados com o sensor, montou-se uma mesa de dimensões 100 cm x 50 cm, com marcações nas bordas e no centro a cada 25 cm, para manter uma orientação clara. Decidiu-se fazer cinco movimentos,



realizados usando a mão para mover o sensor, para averiguar a qualidade das medições: um movimento feito aproximadamente em linha reta; outro movimento cuja trajetória é aproximadamente um quadrado, a partir das marcações da mesa; mais um em que o corpo realiza um movimento quase parabólico sem sofrer rotação dos seus eixos; outro movimento parabólico, porém com rotação dessa vez; e dois movimentos rotacionais em torno do eixo z, um em que a rotação é contínua até que se complete  $360^\circ$  e outro em que o corpo faz breves paradas a cada  $90^\circ$  e não passa de  $270^\circ$  de rotação. Todos esses movimentos foram gravados para efeitos comparativos.

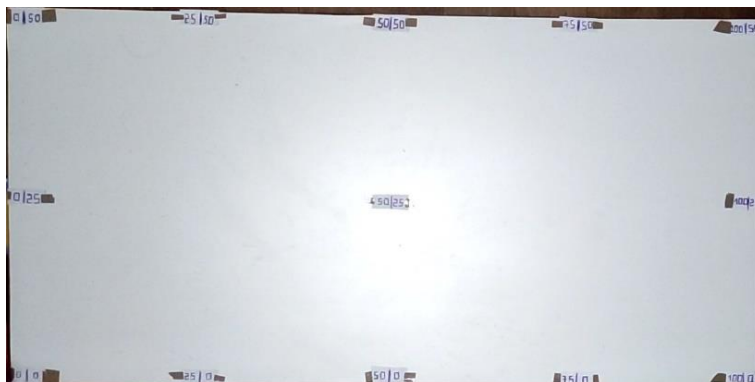
Devido ao fato de os movimentos serem feitos com as mãos, não é possível garantir total equivalência da intenção do movimento com o que de fato foi feito. Apesar disso, acredita-se que, para o contexto do projeto, as medições apresentadas e a forma com que o experimento foi realizado são suficientes para resolver os problemas propostos.

## 4. RESULTADOS

Para poder avaliar o sensor e fazer as correções necessárias, os principais movimentos estudados foram um movimento simples em linha reta, um movimento simulando um quadrado, dois movimentos parabólicos<sup>5</sup> e dois movimentos de rotação em torno do eixo z.<sup>6</sup>

### 4.1 PRIMEIRAS MEDIÇÕES E CALIBRAGEM DO SISTEMA

O primeiro caso de teste foi um movimento linear sobre o plano de uma mesa de **100 cm x 50 cm**, enumerada a cada **25 cm** nas bordas, sendo realizada uma filmagem do sensor para análise. O propósito disso foi testar a qualidade das medições e quanto ruído havia no movimento.



*Figura 3: Mesa de referência.*

Com o auxílio do programa desenvolvido, percebeu-se que, nos momentos estáticos, a medição indicava a existência de aceleração considerável, o que indicou, nos gráficos de velocidade e de posição, valores incoerentes com a realidade, uma vez que o movimento se deu num comprimento de cerca de 100cm no eixo x somente, mas os dados processados indicaram um movimento de cerca de 200 cm no eixo y e recuo no eixo x de cerca de 66 cm.

---

<sup>5</sup> Esses movimentos foram gravados e podem ser acessados em: < <https://www.youtube.com/watch?v=dDBdsNAT8Fs>>

<sup>6</sup> Esse movimento foi gravado e pode ser acessado em: < <https://www.youtube.com/watch?v=ekyY6dQ9t6o&t=2s>>



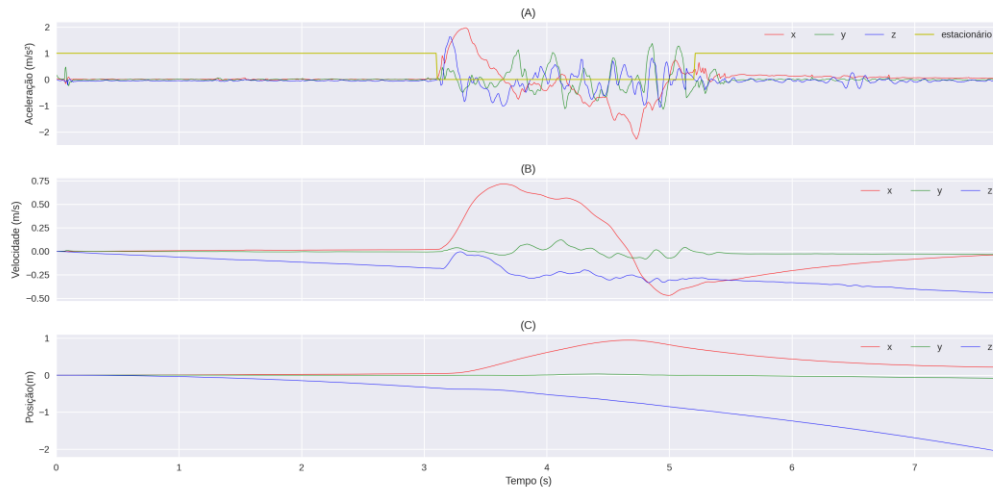


Figura 4: Gráficos contendo aceleração (A), velocidade (B) e posição (C) sem modificações.

Num primeiro momento, a correção desses dados foi feita usando a média da aceleração nos momentos estáticos, e posterior subtração dessa, a fim de aproximar a média de zero nesses momentos. Porém, essa solução não foi muito eficaz, já que a média não era constante em boa parte dos testes e até mesmo variava em um único teste.

Diante disso, decidiu-se implementar os filtros de High Pass e Low Pass para auxiliar na correção dos dados, uma vez que eles conseguem suavizar as curvas e reduzir o ruído da medição. Apesar de não resolverem completamente o problema, esses filtros foram de grande ajuda na correção, dado que conseguiram regularizar os dados. O problema que restou foi que o movimento nos períodos estacionários persistiu, mas a correção se tornou mais fácil.

A solução proposta para esse problema foi a implementação da função estacionária, que identifica se o corpo está ou não em movimento. Após realizar alguns testes com diversos movimentos, notou-se que, de modo geral, quando a magnitude da aceleração é menor que 0,05 m/s, o corpo está parado.

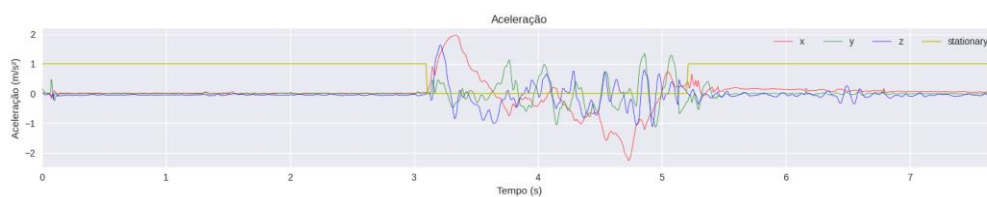


Figura 5: Gráfico da aceleração referente ao movimento retilíneo.

#### 4.2 CORREÇÕES DA VELOCIDADE

Com a função  $E(t)$ , foi possível determinar os momentos estáticos, onde, portanto, a velocidade do sensor é zero, assim, o programa exige  $v(t) = 0$  sempre que  $E(t) = 1$ , reduzindo a propagação do erro na integração. Todavia, o gráfico, por apresentar erro, não consegue fazer com que  $v$  tenda a 0 na vizinhança do último período estacionário, ficando com um salto abrupto para 0.

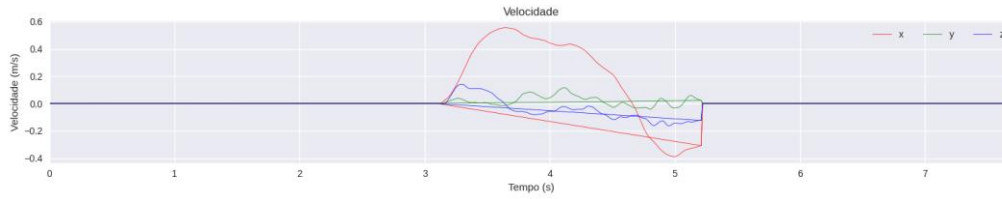


Figura 6: Velocidade sem interpolação entre os pontos iniciais e finais.

Dessa forma, a interpolação se mostrou eficaz por dar uma representação mais fiel ao movimento realizado pelo sensor, como se observa nos gráficos abaixo.

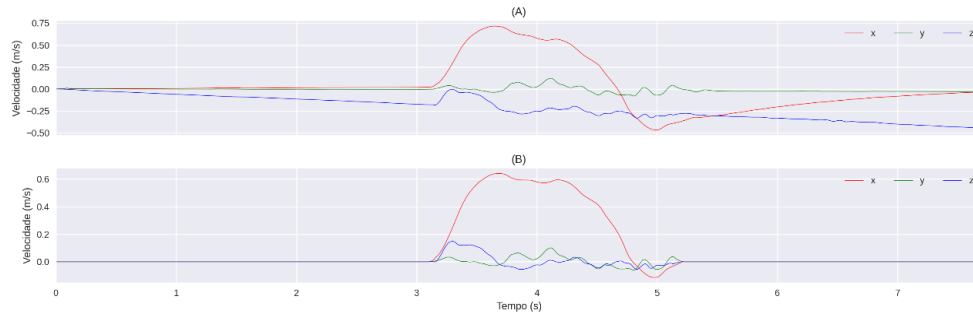


Figura 7: Comparativo entre velocidade pura (A) e processada (B), relativas ao movimento em linha reta.

#### 4.3 POSIÇÃO NORMAL vs CORRIGIDA

Com as correções feitas anteriormente, a integral da velocidade corrigida forneceu um dado mais próximo da realidade, com erro de 6%. No caso particular do projeto, esse método se demonstrou eficiente na resolução desse problema, uma vez que raramente o sensor apresenta velocidade constante.

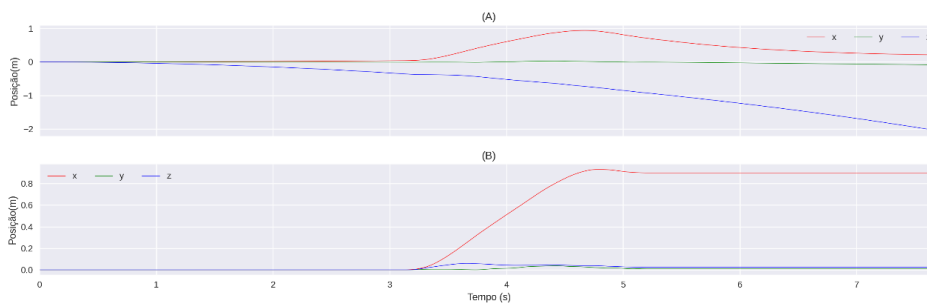


Figura 8: Posição pura (A) comparada com a posição processada (B), referente ao movimento retilíneo.

Os métodos utilizados anteriormente se mostraram eficazes na determinação da posição, em contraste com o dado sem processamento, que mediu, no eixo  $x$ , um movimento de cerca de 50 cm com retorno à posição inicial do eixo, totalmente incoerente com o movimento realizado.

#### 4.4 CORREÇÃO DOS QUATÉRNIONS

Na Figura 9 estão três gráficos comparando os quatérnions antes e após a correção e os ângulos de Euler. Pode-se observar uma simetria nos gráficos, característica da rotação de  $360^\circ$  realizada em torno do eixo z. O gráfico B é o que melhor representa o movimento feito e é obtido a partir da correção do gráfico A, feita de acordo com a Seção 3.7.

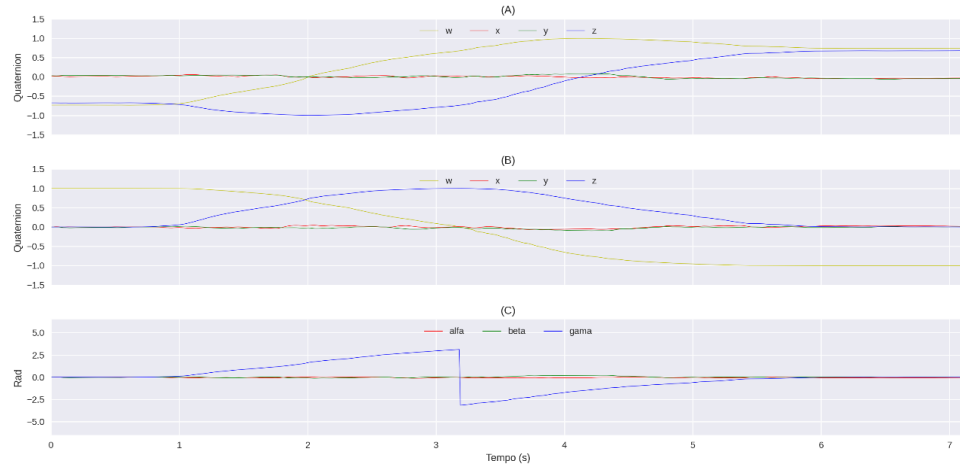


Figura 9: Comparativo entre o quatérnion puro (A), corrigido (B) e ângulos de Euler (C), referente a uma rotação de  $360^\circ$  em torno do eixo z.

#### 4.5. MOVIMENTOS

##### 4.5.1. LINHA RETA

O primeiro movimento selecionado para a avaliação dos dados processados foi um simples movimento retilíneo, em que o sensor parte da posição  $(0,0)$  e encerra o seu movimento na posição  $(100,0)$ . O erro da estimativa do programa foi de 6%.

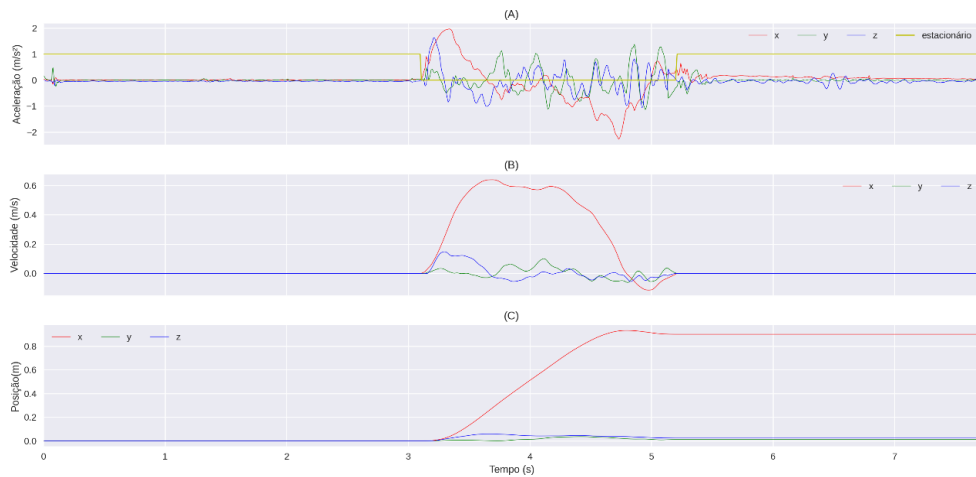


Figura 10: Gráficos do movimento retilíneo contendo a aceleração (A), a velocidade (B) e a posição (C).

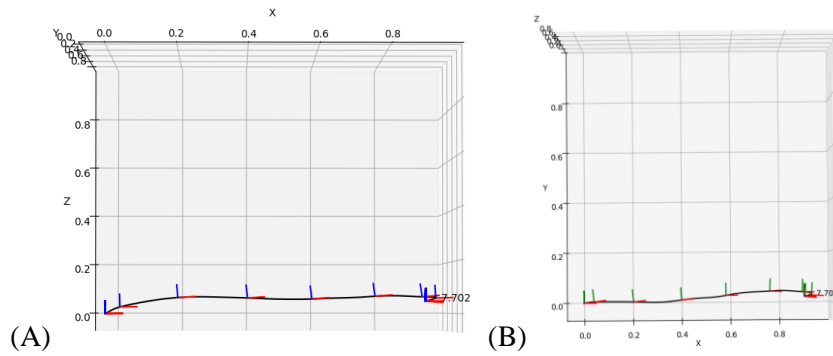


Figura 11: Movimento tridimensional, visto de lado (A) e de cima (B), referente à Figura 10.

Os gráficos da Figura 11.A e da Figura 11.B são, respectivamente, uma vista lateral e uma vista superior do movimento retilíneo. Como se pode observar na Figura 11.A, há pouca variação de altitude e nenhuma rotação sensível, o que é indicado pelos eixos de orientação verde, azul e vermelho. A Figura 11.B confirma a ausência de rotação e exibe a baixa variação de posição no eixo y, enquanto ambas confirmam uma variação de aproximadamente um metro no eixo x, conforme o movimento realizado.

#### 4.5.2 QUADRADO

O segundo movimento selecionado foi simulando o desenho de um quadrado no plano. Nesse dado, o sensor parte de  $(50, 0)$  e se move até  $(50, 50)$ , então indo para  $(100, 50)$ ,  $(100, 0)$  e enfim tornando à posição inicial.

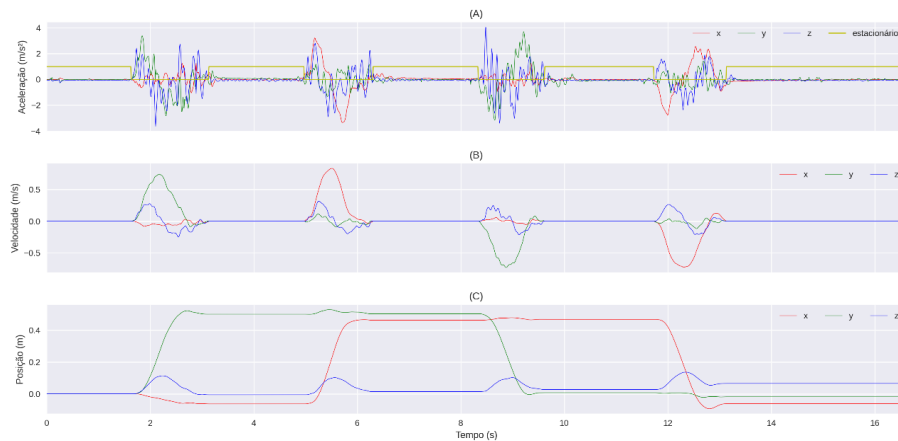


Figura 12: Gráficos de aceleração (A), velocidade (B) e posição (C), e num movimento emulando um quadrado.

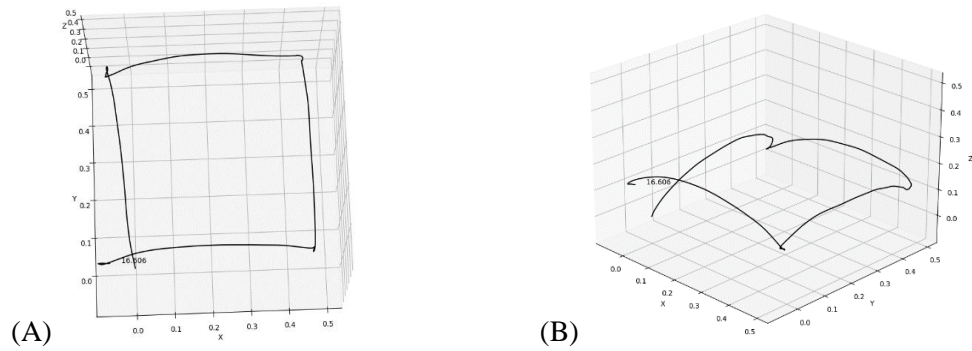


Figura 13: Representação em três dimensões do movimento do quadrado, visto de cima (A) e visto obliquamente (B).

#### 4.5.3. ROTAÇÃO

O último movimento no plano foi uma rotação. O sensor inicia fazendo uma rotação de  $90^\circ$  no sentido anti-horário e para um tempo. Em seguida, ele retoma o movimento e completa  $180^\circ$  de rotação anti-horária e para mais uma vez. Logo após, o sensor percorre  $90^\circ$  de rotação anti-horária, totalizando  $270^\circ$  em relação à posição inicial, e, por fim, o sensor sofre uma rotação de  $90^\circ$  na direção horária, retomando o estado de  $180^\circ$  de rotação relativos à origem.

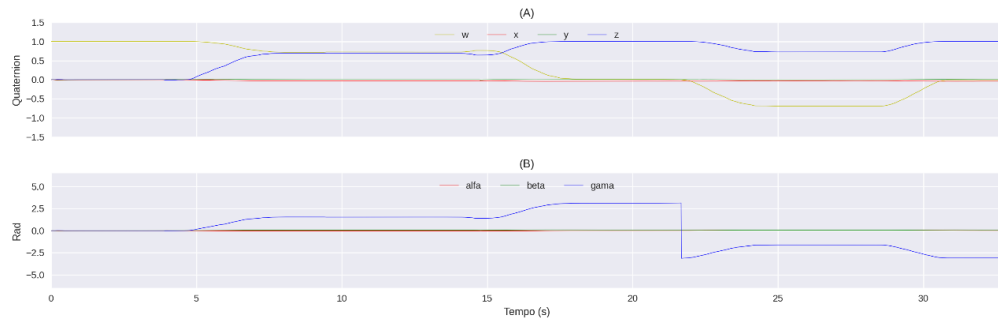


Figura 14: Movimento de rotação no plano representado em quatérnions (A) e em ângulos de Euler (B).

#### 4.5.4. MOVIMENTO PARABÓLICO

Foram feitos dois movimentos parabólicos: um em que o sensor não efetuava rotação alguma e retornava à origem, e outro em que o sensor fazia uma rotação em torno do eixo  $y$ .

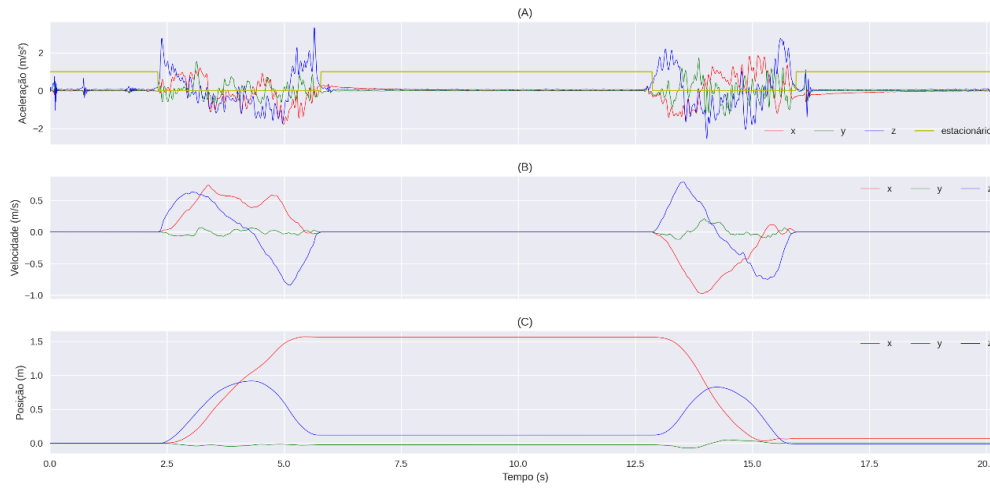


Figura 15: Movimento parabólico de ida e volta sem rotação. Aqui estão a aceleração (A), a velocidade (B) e a posição (C) do corpo.

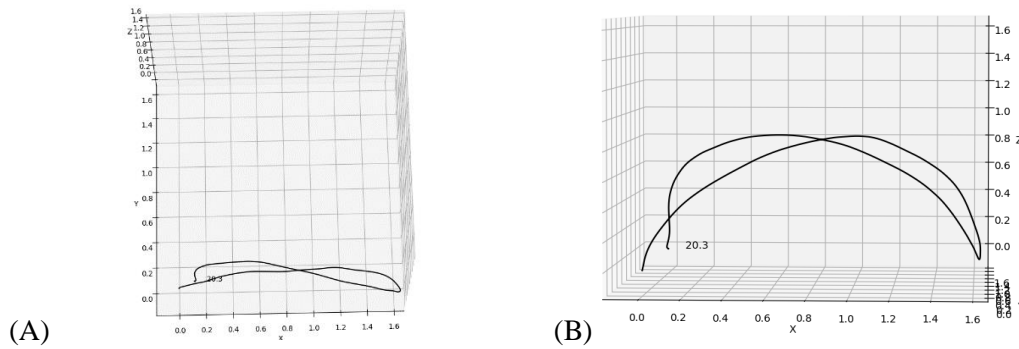


Figura 16: Referente ao movimento da Figura 15, visto de cima (A) e de lado (B).

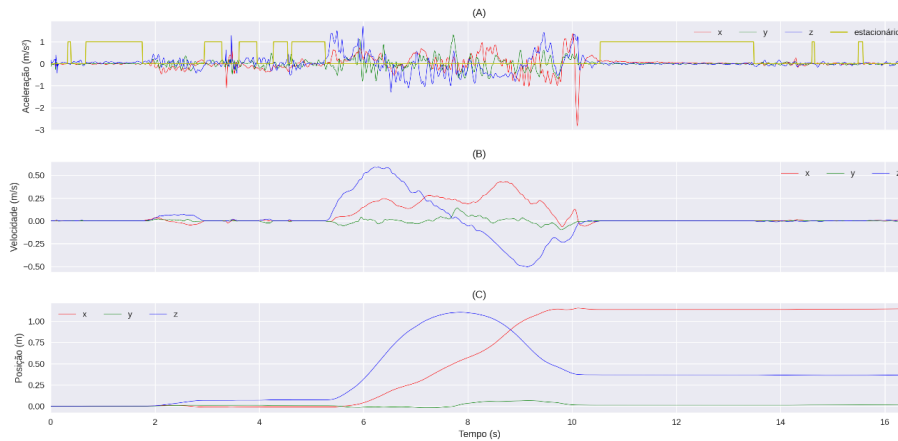


Figura 17: Parábola com rotação em torno do eixo y. Estão representadas a aceleração (A), a velocidade (B) e a posição (C).

Nesse movimento, a altura do início e do fim deveria ser igual, entretanto a diferença pode ser explicada pelo fato de o sensor ser um corpo extenso, assim, a

posição do sensor é diferente devido ao comprimento da placa.

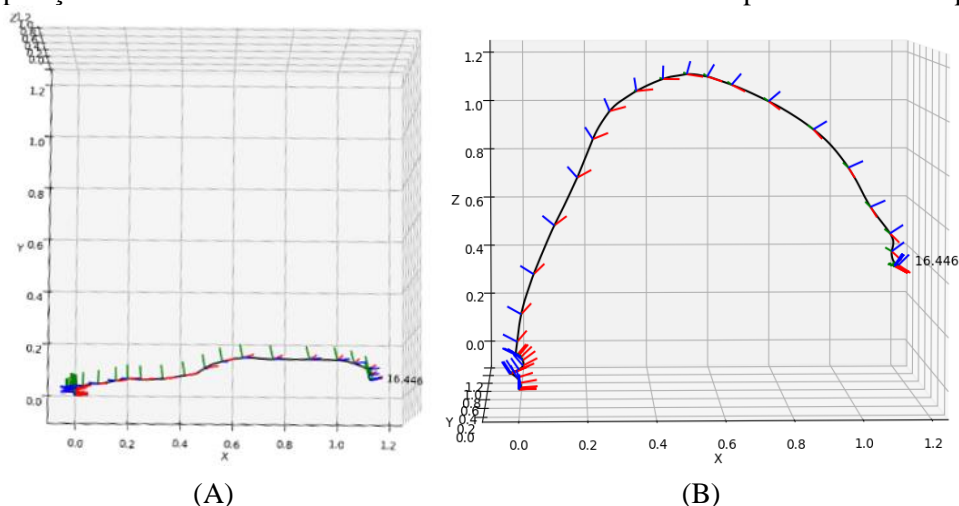


Figura 18: Representação do movimento da Figura 17, visto de cima (A) e de lado (B).

## 5. CONSIDERAÇÕES FINAIS

Usando o sensor MPU-6050 e o embarcado Raspberry Pi 4, o sistema de monitoramento, representação e simulação do movimento foi desenvolvido com sucesso, sendo capaz de processar os dados coletados e gerar gráficos em duas e três dimensões, que foram suficientes para expressar bem o movimento e auxiliaram na interpretação desses dados.

Quanto à determinação da posição e da rotação, o sensor se demonstrou suficientemente preciso nas medições de rotação, entretanto, a determinação da posição envolveu o uso de filtros estatísticos e métodos numéricos. Assim, cumpriram-se os objetivos específicos do projeto. Apesar de ter sido eficiente em medições de curta distância e períodos de movimento e repouso intercalado, devido às restrições de tempo para adaptar o sistema, não foi possível testar a eficiência desse sistema em longas distâncias.

Desta forma, dentre as possíveis continuidades desse projeto está a otimização do programa e verificação da sua qualidade em distâncias longas e desenvolver outros métodos que funcionem com maior generalidade. Principalmente, a redução do custo do projeto pode ser feita com a programação da placa Arduino, uma vez que o Raspberry não é utilizado em todo o seu potencial e é caro em relação ao Arduino.

Por fim, o sistema desenvolvido pode ser útil em jogos de realidade virtual, biotecnologia de próteses, orientação de drones, e, de modo geral, se aplica a qualquer sistema dependente da orientação do corpo em questão.

## 6. REFERÊNCIAS

- [1] EPPERSON, J. F. **An Introduction to Numerical Methods and Analysis**. New Jersey: John Wiley & Sons, Inc, 2013.
- [2] EULER, L. General formulas for any translation of rigid bodies. (p. 11-22) **St. Petersburg Academy**, 1775. Disponível em:



<<http://www.17centurymaths.com/contents/euler/e478tr.pdf>>. Acesso em: 16 jul 2021.

[3] HAMILTON, W. R. **On Quaternions, or a New System of Imaginaries in Algebra**. Londres: Philosophical Magazine, 1844-1850. Disponível em: <<https://www.maths.tcd.ie/pub/HistMath/People/Hamilton/OnQuat/OnQuat.pdf>>. Acesso em: 16 jul 2021.

[4] MADGWICK, S. **Open source IMU and AHRS algorithms**. X-IO Technologies. Disponível em: <<https://x-io.co.uk/open-source-imu-and-ahrs-algorithms/>> Acesso em: 05 ago 2021

[5] BIASI, S. C.; GATTASS, M. **Utilização de quatérnios para representação de rotações em 3D**. Disponível em: <<https://webserver2.tecgraf.puc-rio.br/~mgattass/Quaternios.pdf>> . Acesso em: 05 ago 2021

[6] INVENSENSE. **MPU-6000 and MPU-6050 Product Specification and Revision**. Disponível em: <<https://invensense.tdk.com/wp-content/uploads/2015/02/MPU-6000-Datasheet1.pdf>> Acesso em: 05 ago 2021.

[7] HUNTER, J. et al. **Matplotlib Release 3.4.2**. Disponível em <<https://matplotlib.org/stable/Matplotlib.pdf>> Acesso em: 05 ago 2021.

[8] NELLI, F. **Hamilton's Quaternions and 3D Rotation with Python**. Meccanismo Complesso. Disponível em: <<https://www.meccanismocomplesso.org/en/hamiltons-quaternions-and-3d-rotation-with-python/>> Acesso em: 05 ago 2021.

[9] MCKINNEY, W et al. **pandas: powerful Python data analysis toolkit Release 1.3.1**. Disponível em: <<https://pandas.pydata.org/docs/pandas.pdf>>. Acesso em: 05 ago 2021.

[10] HALLIDAY, D. **Fundamentos de Física Volume 1 - Mecânica (9ª ed)**. Rio de Janeiro: LTC - Livros Técnicos e Científicos, 2012.

Recebido para publicação em 27-09-21; aceito em 14-10-21