Universidade de São Paulo
Instituto de Matemática e Estatísticas
Departamento de Ciência da Computação

Pedro Marcondes

Contrabaixo aumentado com uso de sensores para controle de efeitos sonoros

São Paulo

2017

Pedro Marcondes

Contrabaixo aumentado com uso de sensores para controle de efeitos sonoros

Trabalho de conclusão de curso em Ciência da Computação do Instituto de Matemática e Estatística da Universidade de São Paulo

Orientadores: Carolina Brum Medeiros, Marcelo Gomes de Queiroz e Antonio Deusany de Carvalho Junior

São Paulo

2017

para Luiz Alberto Marcondes e Mônica Renata da Silva

Agradecimentos

Queria agradecer aos meu orientadores, todos do grupo de computação musical e minha família. Primeiro ao Marcelo, meu orientador, por ter me acolhido como orientando, por ter me guiado dentro da computação musical, por todos ensinamentos durante esse percurso, por todo esforço de estar presente e ajudar muito no projeto mesmo quando o tempo parecia escasso. Agradeço ao Antonio Deusany, meu orientador, por toda ajuda com as apresentações, Arduino, patches de *Pure Data*, pesquisas sobre comunição I2C.

Agradeço a todos do grupo de computação musical e do laboratório 118 por esse ano sensacional dentro da universidade e em especial ao Roberto Bodo, por todas as tardes me ajudando com a pesquisa e por todos materiais emprestados. Agradeço ao Felipe Félix e o Felipe Moreira, por terem me apresentado o grupo.

Sobre minha família, agradeço por todos anos de apoio incondicional, em especial minha tia Rosemeire por esses 4 anos que me acomodou em sua casa.

Reservo o último agradecimento para minha orientadora Carolina Brum, que me apresentou o que acabou sendo o tema do meu trabalho e uma nova paixão (sensores), por todos artigos indicados, por todas discussões teóricas, pelas dicas de apresentação, por todo apoio durante as mesmas, pela paciência, por todas as "duras", pelas reuniões em horários não convencionais, por me forçar a sair da zona de conforto, pelos materiais concedidos para realizar o projeto e por outros diversos e incontáveis motivos.

Todos agradecimentos são extremamente sucintos, pois se fosse agradecer do jeito que todos mereciam, não haveriam páginas e nem tempo para escrever o que todos significaram para mim durante esse processo.

"Sinnerman where you gonna run to?"

Nina Simone

"Eu chorei, perdi a paz

Mas o que eu sei é que ninguém nunca teve mais, mais do que eu"

Baden Powell e Vinicius De Moraes

Resumo

Um instrumento aumentado fornece uma diferente forma de interação entre músico e instrumento para geração sonora. Este trabalho apresenta o desenvolvimento de um baixo aumentado em que efeitos sonoros são controlados pelos gestos do músico. Implementamos um algoritmo para estimar o ângulo da articulação do cotovelo, usando sensores MARG para calcular a orientação do braço e antebraço, e alguns exemplos de mapeamentos utilizados entre esta estimativa e o controle de efeitos utilizados no processamento do som gerado pelo instrumento. Discutiremos as técnicas utilizadas para o desenvolvimento do algoritmo, as motivações que nos levaram à escolha das tecnologias utilizadas e o mapeamento realizado. Os resultados nos mostraram que o algoritmo desenvolvido nos permite realizar o controle de áudio de maneira responsiva aos gestos do músico.

Palavras-chave: computação musical; sistema AHRS; sensor MARG; fusão de sensores; instrumento aumentado; baixo elétrico; processamento sonoro em tempo real.

Abstract

An augmented instrument provides a different form of interaction between musician and instrument for sound generation. This work presents the development of an augmented bass where sound effects are controlled by musicians gestures. We implemented a custom algorithm to obtain elbow joint angles from MARG sensors measurements and some examples of mapping between the angle estimation and the control of effects. We'll discuss the techniques utilized during the development of the algorithm, the motivations in the process of choosing the technologies used in this work and the design of the mapping. The results show us that the algorithm enabled us to perform the control of the effects in a manner responsive to the musician gestures.

Keywords: computer music; AHRS system; MARG sensor; sensor fusion; augmented instrument; electric bass; real time sound processing.

Sumário

Capítulo 1

Introdução

1.1 Contextualização e motivação

1.1.1 Instrumentos aumentados

Instrumento musical aumentado é um instrumento cujas possibilidades sonoras são estendidas, para possibilitar novas formas de interação ou novas maneiras de produzir som. Nesse trabalho usaremos dispositivos eletrônicos com objetivo de possibilitar novas formas de interação com o instrumento (baixo elétrico) para modificar o som gerado por ele.

Existem diversas maneiras de estabelecer as relações entre o músico, a máquina e o instrumento, de forma a estabelecer novas maneiras de realizar ou alterar a interação entre o músico e o instrumento. Wanderley (2006) discute o papel dos instrumentos aumentados no meio musical, a metodologia da sua construção e os motivos para seu uso. Segundo esse autor, um instrumento aumentado tem como objetivo dar novas opções expressivas para o músico, não necessariamente melhores ou piores do que as originalmente disponíveis, mas que permitam a produção de diferentes resultados sonoros e o uso em contextos diferentes do original.

Um desafio em construir um instrumento aumentado é garantir sua "tocabilidade" (*playability*), ou seja, que o instrumento continue confortável de ser tocado. A ideia é criar instrumentos que não impactem demasiadamente a performance do músico, relativamente ao instrumento original, em termos de conforto, peso do instrumento aumentado, limitação dos movimentos.

1.1.2 Tecnologias utilizadas

Para criar o baixo aumentado utilizamos dois chips *NXP Precision 9DoF* desenvolvido pela *Adafruit*. Esse modelo é uma combinação de dois sensores de movimento diferentes: o primeiro contém um magnetômetro e um acelerômetro de três eixos e o segundo contém um giroscópio de três eixos. Escolhemos esse modelo por ser economicamente acessível e por ter maior resolução que outros modelos da mesma marca, chegando a possuir maior estabilidade quando o sensor está imóvel[1].

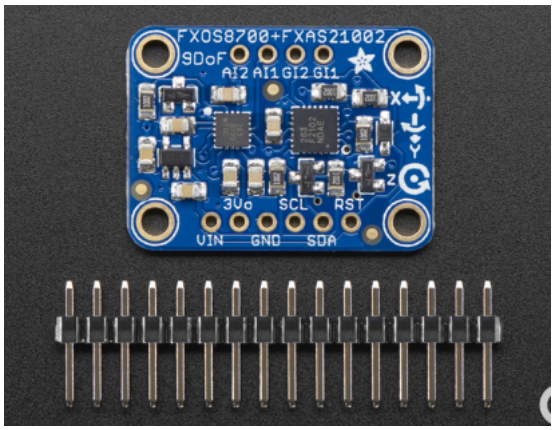


Figura 1.1: NXP Precision 9DoF. Fonte: Adafruit

Além dos sensores usamos também dois Arduinos (modelo Nano, escolhido pela dimensão reduzida) para realizar a comunicação entre os sensores e o computador.

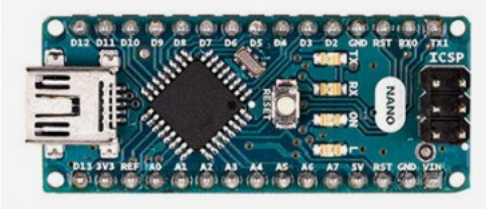


Figura 1.2: Arduino Nano. Fonte: Arduino

1.2 Trabalhos relacionados

Durante a concepção deste projeto diversos trabalhos foram analisados a fim de angariar ideias e motivar o desenvolvimento. Alguns trabalhos foram fundamentais para

[1] Mais detalhes podem ser encontrados em `https://learn.adafruit.com/nxp-precision-9dof-breakout/`

a construção das ideias que servem de base para esse projeto:

- Wanderley (2001) discute diferentes formas em que gestos são usados no controle musical em instrumentos digitais.
- Ramkissoon (2011) desenvolveu um baixo aumentado que usa o movimento dos pés do músico para controlar o processamento do áudio do instrumento e também o processamento de vídeos usados durante a performance em tempo real.
- Gong et al. (2013) apresenta uma técnica para criar adesivos que usam tinta condutiva na criação de interfaces interativas para manipulação física de áudio. Esse design permite uma forma de criar instrumentos aumentados de maneira rápida e facilmente personalizável.

1.3 Objetivos

Esse trabalho possui como objetivo final construir um baixo aumentado que usa o ângulo entre braço e antebraço do músico para controlar efeitos sonoros em tempo real. Esse objetivo final pode ser dividido em 3 objetivos intermediários necessários para atingí-lo:

- Desenvolver um algoritmo para calcular o ângulo entre braço e antebraço, usando sensores, que possua baixo custo computacional no processamento de áudio em tempo real.
- Usar a estimação do ângulo para controlar um efeito sonoro de maneira fluída e responsiva em relação aos movimentos do músico.
- Fazer com que esse sistema seja acessível e com que o seu *setup* seja facilmente reprodutível.

1.4 Estrutura do trabalho

O capítulo 2 apresenta a fundamentação teórica na qual o trabalho se baseou durante todo processo de desenvolvimento, tratando desde o pré-processamento dos dados providos pelos sensores até as representações matemáticas necessárias ao cálculo das rotações.

No capítulo 3 descrevemos o algoritmo desenvolvido para estimar o ângulo da articulação do cotovelo e o controle dos efeitos utilizando este dado. Além disso, apresentamos duas validações experimentais do sistema implementado: a validação do algoritmo de estimação e a validação do uso do instrumento por músicos.

Por fim, no capítulo 4 temos as conclusões sobre o projeto, onde confrontamos os objetivos iniciais com os resultados obtidos e discutimos trabalhos futuros.

Capítulo 2

Fundamentação Teórica

2.1 Instrumentos aumentados

A construção de um instrumento aumentado pode ser dada de diferentes formas, e nesse trabalho a abordagem escolhida foi o uso de sensores para controle de efeitos sonoros em tempo real. O foco foi definido no tratamento dos dados dos sensores e o mapeamento dos mesmos para o controle de efeitos sonoros. Neste trabalho todas as implementações de efeitos foram feitas na linguagem *Pure Data*.

O processamento de áudio em tempo real foi realizado usando o ângulo entre braço e antebraço do músico, dado pelos sensores, como parâmetro do efeito que modula o áudio do instrumento. Samora da Graça (2016) apresenta um maior aprofundamento no processamento de áudio e as técnicas utilizadas para construção de efeitos sonoros.

O uso de sensores para criar um instrumento aumentado está ligado à percepção de que em uma performance o artista não se expressa apenas pelo som mas também por gestos performáticos. A ideia é expandir as capacidades dos instrumentos integrando essas duas formas de expressão de uma maneira fluída, a fim de trazer a performance física do músico para o campo sonoro. Por isso o projeto foi desenvolvido de maneira flexível, possibilitando de uma maneira simples a personalização do mapeamento entre sensores e efeitos.

2.2 Sensores

2.2.1 Fusão de sensores

Fusão de sensores é a combinação de dados sensoriais, sejam eles de dispositivos idênticos ou diferentes, para gerar uma informação mais precisa ou melhor condicionada do que a que se conseguiria com apenas uma fonte de informação (Elmenreich, 2002). Essas técnicas têm a função de contornar as limitações nos sistemas de medição. Segundo Medeiros e Wanderley (2014), podemos observar algumas das principais motivações que justificam o uso dessa abordagem e também as principais limitações das técnicas de fusão de sensores. Algumas dessas motivações são:

- Privação de um sensor: mal funcionamento de um sensor, causando perda de informação;
- Limitações do sensor: a limitação da medição, seja na escala da medição fornecida ou da posição onde foi colocado; a frequência de amostragem, ou seja o tempo necessário pra realizar a medição e transmitir a informação; limitação intrínseca do sensor, como a imprecisão de suas medições;
- Suscetibilidade ao ambiente: a suscetibilidade do sistema com as variações naturais do meio onde está inserido que podem prejudicar sua performance.

As técnicas de fusão de dados nos auxiliam a reduzir o impacto desses problemas e incertezas durante o processo:

- Com mais leituras possibilitamos, mesmo em caso de falhas, o uso de outras medições provenientes de outros sensores, para manter o funcionamento do sistema;
- Podemos contornar as limitações do sensores pois teremos maior cobertura espacial e uma maior confiança nas medições, dado que teremos duas fontes independentes de dados possibilitando uma estimativa e redução do erro e uma validação das medições entre os sensores. Além disso podemos fazer a fusão de dados de sensores que são menos sensíveis a diferentes tipos de interferência, para aumentar a robustez do sistema.

Apesar das vantagens, a fusão de sensores apresenta algumas limitações. Trabalhar com diferentes fontes de dados aumenta a complexidade do sistema, não apenas por termos mais dados para tratar, mas também pois precisaremos de algoritmos mais complexos para processar esses dados. Além disso os algoritmos para a fusão, em sua maior parte, dependem muito de uma modelagem correta e precisa do sistema já que dependem muito do conhecimento do modelo físico adotado: uma modelagem pobre acarretará em imprecisão e instabilidade inviabilizando todo o sistema.

2.2.2 Sistema de referência de atitude e direção (AHRS)

Como dito em 1.1.2, utilizamos sensores MARG (*Magnetic, Angular Rate, and Gravity*), que possuem um magnetômetro, um acelerômetro e um giroscópio. Esses três sensores transmitem suas leituras para então aplicarmos um algoritmo de fusão de sensores e extrair a informação desejada deles.

Como nosso objetivo é conseguir o ângulo entre os dois sensores, iremos usar um sistema de referência de atitude e direção (AHRS - Attitude and Heading Reference System), também chamado de sistema MARG, que é capaz de nos oferecer a orientação do sensor em relação a direção da gravidade e o campo magnético da Terra.

Diversos algoritmos de fusão de sensores são usados na implementação de um sistema AHRS, tais como filtro complementar (Tseng et al., 2011), filtro complementar não-linear (Mahony et al., 2008) e filtro de Kalman (Yun e Bachmann, 2006). Geralmente as soluções baseadas em filtro de Kalman são as mais usadas, porém a regressão linear que é parte essencial do filtro exige uma taxa de amostragem alta além de um vetor grande de estados. Como nesse trabalho iremos processar áudio em tempo real além de calcular a orientação dos sensores, optamos por trabalhar com um algoritmo que usa gradiente descendente (Madgwick et al., 2011), considerando como prioridade que a acessibilidade do projeto para outras pessoas o implementarem sem dificuldade e com baixo custo. Esse algoritmo possui uma precisão semelhante à de soluções baseadas em filtro de Kalman. Além de sua fácil implementação e baixo custo computacional (a técnica de gradiente descendente é uma das mais simples tanto de implementar quanto de computar), outra característica importante é que esse algoritmo nos permite operar com uma taxa de amostragem baixa do sensor. No capítulo 3 iremos detalhar mais a fundo o funcionamento desse algoritmo.

2.3 Ângulos e Rotações

2.3.1 Ângulos de Euler

Formulado por Leonhard Euler em 1748 com o propósito de de descrever a orientação de um corpo rígido num espaço tridimensional, esse sistema considera que toda orientação angular pode ser alcançada através de 3 rotações elementais, comumente chamadas de *roll*, *pitch* e *yaw*, nos eixos x, y e z, como podemos ver na figura 2.1.

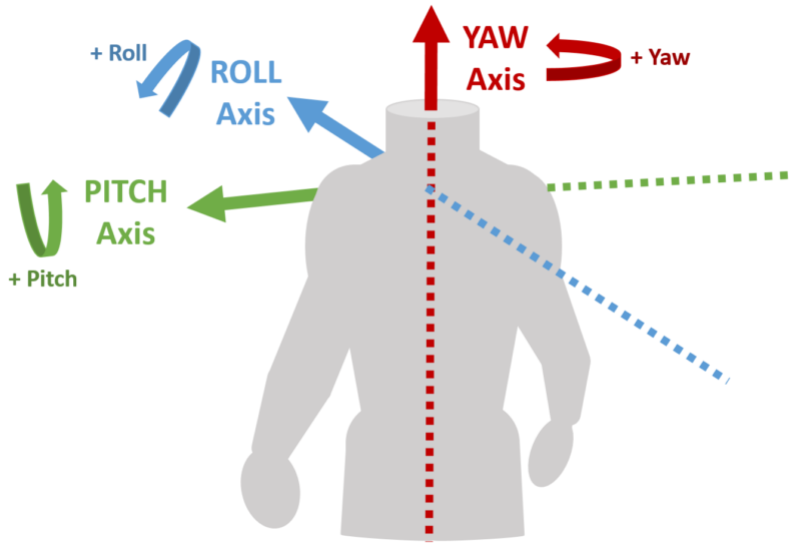


Figura 2.1: Eixos e as orientações as quais são convencionalmente associados. Fonte: Motion Sensor Computing.

Apesar do uso de ângulos de Euler ser a abordagem mais convencional para trabalhar com orientações de objetos em um espaço tridimensional, esse sistema apresenta um problema conhecido como o *Gimbal Lock*.

Este fenômeno é a perda de um grau de liberdade em um mecanismo multidimensional que ocorre quando dois eixos são levados à mesma configuração durante uma rotação elemental. Nas figuras 2.2 e 2.3 podemos ver um caso onde ocorre o *gimbal lock*.

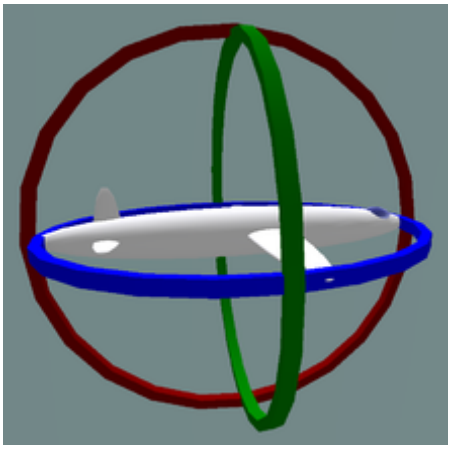


Figura 2.2: Objeto sólido antes de rotação usando ângulos de Euler. Fonte: MathsPoetry.

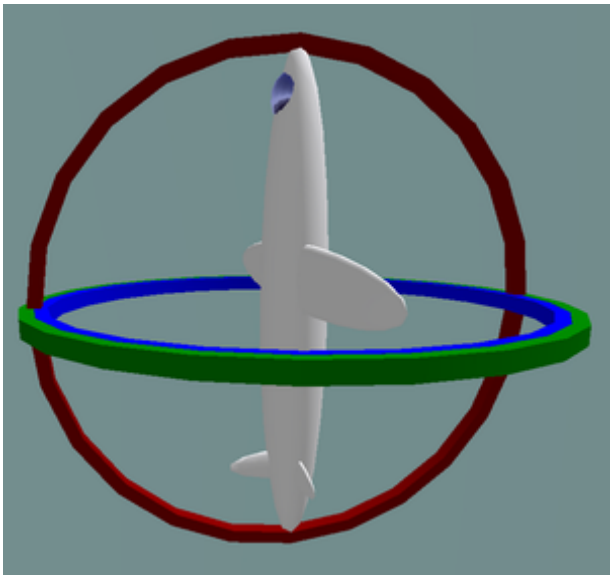


Figura 2.3: Perda de um grau de liberdade após rotação de 90° do eixo *pitch* pois dois gimbals ficaram no mesmo plano. Fonte: MathsPoetry.

Isso faz com que não seja possível percorrer todas as rotações possíveis fazendo uma sequência de rotações elementais. É possível identificar pontos críticos onde o *gimbal lock* ocorre; quando a rotação é em torno de dois eixos, o problema aparece quando o seno da rotação secundária (a segunda rotação elemental realizada para se atingir a posição desejada) é zero; e quando a rotação é em torno de três eixos, o problema aparece quando o cosseno da rotação secundária é zero (Silva, 2014).

Existem algumas soluções para esse problema, como o uso de um quarto gimbal ou uma rotação aleatória dos eixos para resetar o gimbal. A solução mais simples seria apenas usar outra forma de representação diferente dos ângulos de Euler. Matrizes de

rotação aparecem como uma opção válida, porém a representação não é muito compacta e tais matrizes são mais custosas de se trabalhar do que quatérnios (Dam et al., 1998).

2.3.2 Quatérnios

O conjunto dos quatérnios, denotado por $\mathbb{H}$, é uma extensão do conjunto dos complexos que consiste de uma parte escalar e três unidades imaginárias (i, j e k) (Hamilton, 1843). Geralmente um quatérnio é representado da seguinte forma:

$$q = q_0 + q_1 i + q_2 j + q_3 k,$$

onde i, j, k são as unidades fundamentais dos quatérnios e $q_n \in \mathbb{R}, \forall n \in \{0, 1, 2, 3\}$. As unidades dos quatérnios seguem uma regra similar aos números complexos em relação à multiplicação, dada pela fórmula fundamental da multiplicação de quatérnios que nos mostram as identidades descritas por Sir William Rowan Hamilton:

$$i^2 = j^2 = k^2 = ijk = -1.$$

2.3.3 Quatérnios e rotação espacial

Quatérnios são usados em rotações espaciais por alguns motivos:

- Gimbal Lock: diferentemente da representação com ângulos de Euler, a representação de rotações usando quatérnios não apresenta esse problema.
- Construção: um quatérnio é mais compacto que um matriz de rotação e além disso é fácil construir um quatérnio dado um eixo e um ângulo desejado.
- Interpolação: é conveniente o uso de quatérnios para animações usando algumas técnicas de interpolação (Dam et al., 1998), que permitem obter movimentos suaves e contínuos (o que não é tão simples em outras formas de representação).
- Erros de arredondamento: numa sequência de rotações, é normal acumular erros de arredondamento devido à precisão finita dos computadores. Porém um quatérnio com esses pequenos erros ainda representa uma rotação após ser normalizado. Já com matrizes de rotação isso não é necessariamente verdade, pois alguns pequenos erros de arredondamento podem fazer com que a matriz deixe de ser ortogonal, e o processo para re-ortogonalizar a matriz é custoso.

2.3.4 Representação vetorial

Como estamos trabalhando com vetores e rotações espaciais, é conveniente representar vetores 3-dimensionais como quatérnios. Um quatérnio $q = q_0 + q_1 i + q_2 j + q_3 k$ pode ser representado como:

$$q = \begin{bmatrix} q_0 \\ q_1 : q_3 \end{bmatrix} \tag{2.1}$$

Onde q_0 é a parte escalar e $q_1 : q_3$ a parte vetorial.

Para representar um vetor 3-dimensional $\vec{v}$ como um quatérnio, atribui-se $\vec{v}$ a $q_1 : q_3$ e 0 à parte escalar q_0. Como um exemplo, o vetor $\vec{v} = \{5,\ 3,\ 2\}$ é representado pelo quatérnio

$$v = \begin{bmatrix} 0 \\ \vec{v} \end{bmatrix} \tag{2.2}$$

que é o mesmo que:

$$v = 0 + 5i + 3j + 2k.$$

2.3.5 Forma exponencial

Outra forma de representar um quatérnio é usando sua forma exponencial, que é uma extensão da forma de Euler (Carrier et al., 2005)

$$e^{i\theta} = cos(\theta) + i \cdot sen(\theta).$$

Para quatérnios nós temos

$$q = \begin{bmatrix} a \\ Bu \end{bmatrix}, \tag{2.3}$$

onde a representa a parte escalar, u a parte vetorial (imaginária) normalizada ($|u| = 1$) e $B(\mathbb{R}^3)$ são os coeficientes da parte imaginária. Assim podemos estender a forma de Euler para representar um quatérnio da seguinte maneira (Hamilton, 1899):

$$q = e^a e^{Bu} = e^a(cos(B) + u \cdot sin(B)).$$

2.3.6 Operações

Vamos definir algumas operações elementares de quatérnios descritas em Hamilton (1843) e Hamilton (1899) que serão utilizadas nas próximas seções. Para isso considere os dois quatérnios p e q:

- Adição e subtração:

$$q + p = (q_0 + p_0) + (q_1 + p_1)i + (q_2 + p_2)j + (q_3 + p_3)k,$$

$$q - p = (q_0 - p_0) + (q_1 - p_1)i + (q_2 - p_2)j + (q_3 - p_3)k.$$

- Multiplicação, também conhecida como multiplicação de Hamilton, onde $\cdot$ é o produto escalar e $\times$ o produto vetorial no $\mathbb{R}^3$ e q e p em sua forma vetorial:

$$qp = (q_0, \vec{q})(p_0, \vec{p}) = (q_0 p_0 - \vec{q} \cdot \vec{p}, q_0\vec{p} + p_0\vec{q} + \vec{q} \times \vec{p}).$$

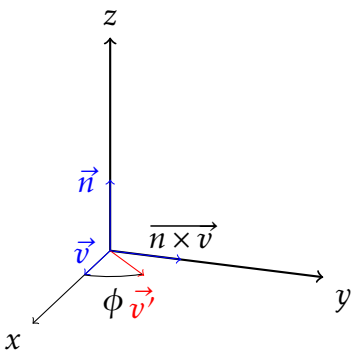
  Vale notar que a multiplicação não é comutativa.

2.3.7 Rotacionando um vetor usando quatérnios

Caso especial

Para demonstrar como rotacionar um vetor, vamos iniciar com um caso especial. Suponha que temos o vetor v no plano xy e queremos realizar a rotação dele por um ângulo ϕ em torno do eixo z; vamos chamar de v' essa versão rotacionada de v.

Seja n o vetor unitário com a orientação do eixo z (e portanto perpendicular a v). Se fizermos o produto vetorial de n e v, o resultado será o vetor n x v que é perpendicular a v e n (Crowe, 1967).



Se projetarmos v' em v e n x v, vamos notar que v' é a soma desses vetores multiplicados por $cos(\phi)$ e $sen(\phi)$ como seus respectivos escalares. Disso temos que:

$$v' = cos(\phi)v + sin(\phi)(n \times v) \tag{1}$$

Usando quatérnio no caso especial

Podemos representar os vetores v, n, n x v e v' através dos respectivos quatérnios v, n, nv e v':

$$\mathsf{v} = \begin{bmatrix} 0 \\ \vec{v} \end{bmatrix}, \quad \mathsf{n} = \begin{bmatrix} 0 \\ \vec{n} \end{bmatrix}, \quad \mathsf{nv} = \begin{bmatrix} 0 \\ \overrightarrow{n \times v} \end{bmatrix}, \quad \mathsf{v}' = \begin{bmatrix} 0 \\ \vec{v'} \end{bmatrix}. \tag{2.4}$$

É importante notar 2 coisas:

- A multiplicação entre os quatérnios n e v é igual ao quatérnio nv (Hamilton, 1843).
- |n| = 1 pois a parte vetorial do quatérnio n é unitária.

Aplicando as representações em quáternios nos vetores de (1) temos:

$$\mathsf{v}' = cos(\phi)\mathsf{v} + sin(\phi)\mathsf{nv},$$

$$\mathsf{v}' = (cos(\phi) + sin(\phi)\mathsf{n})\mathsf{v},$$

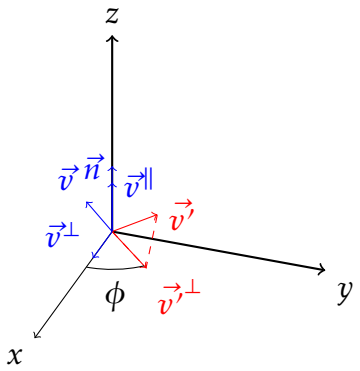
e então podemos usar a forma exponencial para obter:

$$\mathsf{v}' = e^{\phi \mathsf{n}}\mathsf{v} \tag{2}$$

Caso geral

Agora vamos trabalhar com um vetor qualquer. Suponha que temos um vetor $\vec{v}$ e queremos rotacionar ele em torno do eixo z por um ângulo ϕ. Primeiro vamos decompor $\vec{v}$ em $\vec{v}^{\parallel}$, que é paralelo com o eixo z, e $\vec{v}^{\perp}$ que é perpendicular com o eixo z. Lembrando que vamos usar a representação em quatérnio dos vetores durante as fórmulas.

Observamos que vamos precisar apenas rotacionar $\vec{v}^{\perp}$ para conseguir a versão rotacionada de $\vec{v}$, a qual chamaremos de $\vec{v'}$.



Então temos

$$\mathsf{v}' = \mathsf{v}'^{\parallel} + \mathsf{v}'^{\perp},$$

mas $\mathsf{v}^{\parallel} = \mathsf{v}'^{\parallel}$, então

$$\mathsf{v}' = \mathsf{v}^{\parallel} + \mathsf{v}'^{\perp},$$

logo podemos usar a fórmula do caso especial (2):

$$\mathsf{v}' = \mathsf{v}^{\parallel} + e^{\phi n}\mathsf{v}^{\perp}.$$

Para avançar aqui precisamos provar 2 coisas. Já sabemos que a multiplicação de quatérnios não é comutativa. Primeiro vamos mostrar que se $\vec{v}$ e $\vec{n}$ são perpendiculares, temos

$$e^{\phi n}\mathsf{v} = \mathsf{v}e^{-\phi n}.$$

Abrindo o produto da esquerda,

$$(cos(\phi), sen(\phi)n)(0, \mathsf{v}) = (0, \mathsf{v})(cos(\phi), -sen(\phi)n)$$

$$(0, cos(\phi)\mathsf{v} + sen(\phi)(n \times \mathsf{v})) = (0, cos(\phi)\mathsf{v} - sen(\phi)(\mathsf{v} \times n)),$$

mas $(\mathsf{v} \times n) = -(n \times \mathsf{v})$ pois o produto vetorial é anticomutativo, logo temos

$$e^{\phi n}\mathsf{v} = \mathsf{v}e^{-\phi n}.$$

A segunda coisa que iremos precisar é que se $\vec{k}$ e $\vec{n}$ são paralelos, temos

$$e^{\phi n}\mathsf{k} = \mathsf{k}e^{\phi n}.$$

Repetindo os passos anteriores:

$$(cos(\phi), sen(\phi)n)(0, \mathsf{k}) = (0, cos(\phi)\mathsf{k} - sen(\phi)(\mathsf{k} \times n)),$$

mas $(\mathsf{k} \times n)) = (n \times \mathsf{k}) = 0$ pois o produto vetorial de dois vetores paralelos é igual a 0, logo temos

$$e^{\phi n}\mathsf{k} = \mathsf{k}e^{\phi n}.$$

Agora podemos voltar para fórmula de rotação genérica $\mathsf{v}' = \mathsf{v}^{\parallel} + e^{\phi n}\mathsf{v}^{\perp}$ e fazer algumas manipulações algébricas:

$$\mathsf{v}' = e^{(\phi/2)n}e^{-(\phi/2)n}\mathsf{v}^{\parallel} + e^{(\phi/2)n}e^{(\phi/2)n}\mathsf{v}^{\perp}.$$

Como sabemos $\vec{v}^{\parallel}$ é paralelo com $\vec{n}$, logo podemos comutar sem conjugar os fatores, $\vec{v}^{\perp}$ é perpendicular com $\vec{n}$, logo podemos comutar mas temos que conjugar os fatores. Assim vamos ter o seguinte:

$$\mathsf{v}' = e^{(\phi/2)n}\mathsf{v}^{\parallel}e^{-(\phi/2)n} + e^{\phi/2n}\mathsf{v}^{\perp}e^{-(\phi/2)n} = e^{(\phi/2)n}(\mathsf{v}^{\parallel} + \mathsf{v}^{\perp})e^{-(\phi/2)n}.$$

Mas $\mathsf{v} = \mathsf{v}^{\parallel} + \mathsf{v}^{\perp}$, e lembrando que

$$e^{(\phi/2)} = cos(\phi/2) + sen(\phi/2)n_x i + sen(\phi/2)n_y j + sen(\phi/2)n_z k$$

e

$$e^{-(\phi/2)} = cos(\phi/2) - sen(\phi/2)n_x i - sen(\phi/2)n_y j - sen(\phi/2)n_z k,$$

temos nossa fórmula genérica para rotacionar um vetor ao redor de um eixo:

$$\mathsf{v}' = e^{(\phi/2)n}\mathsf{v}e^{-(\phi/2)n}.$$

2.4 Ângulo entre articulações

Sistemas AHRS de baixo custo estão sendo cada vez mais utilizados para rastrear o movimento do corpo humano. A técnica consiste em colocar múltiplos sensores MARG e usá-los para monitorar a orientação de cada parte do corpo separadamente. Depois é usado o conhecimento prévio das articulações do corpo humano para se obter o mapeamento dos movimentos do corpo.

Nosso sistema é uma simplificação desse processo pois queremos apenas rastrear o movimento de um dos braços do músico durante a performance. Inicialmente a ideia era usar 3 sensores: no braço, antebraço e um no ombro. O último seria a referência para os outros dois para obtermos a orientação completa do braço em relação a Terra.

Porém muitos dos movimentos laterais do braço estão ligados à necessidade de percorrer o braço do instrumento para executar certos trechos da música, não sendo necessariamente movimentos de expressão do artista (são movimentos que o músico não possuiria liberdade para variar). Em vista disso, optamos por trabalhar apenas com o ângulo de flexão formado entre o braço e o antebraço(2.4), o que confere ao artista mais liberdade no controle do efeito aplicado ao som.

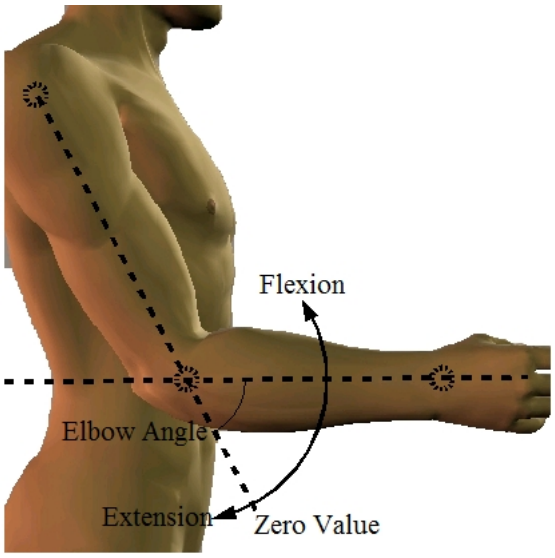


Figura 2.4: Ângulo de flexão da articulação do cotovelo. Fonte: Joint Passive Resistance Database, Hatze,1997.

Para calcular este ângulo, primeiro observaremos que os movimentos durante a performance são limitados: todos os movimentos possíveis realizados pelo músico não alteram a posição relativa entre os eixos dos sensores, ou seja, rotacionar braço e antebraço usando a articulação do ombro não altera a posição relativa dos eixos dos sensores, como podemos observar na figura 2.5.

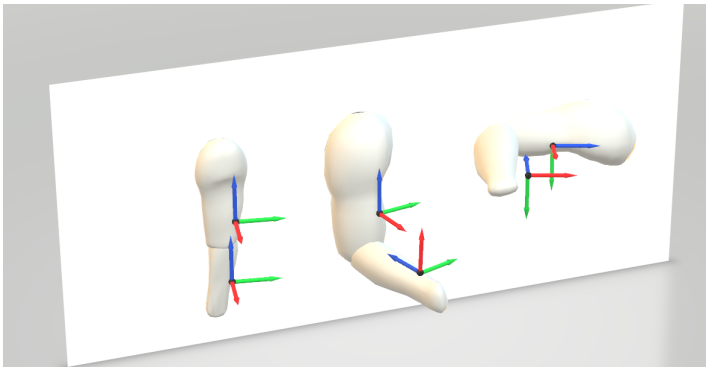


Figura 2.5: Podemos observar que, fixando os sensores na parte interna do antebraço e na parte lateral do braço, movimentos laterais do braço não alteram as posições relativas dos eixos, apenas os movimentos de flexão alteram essas posições.

Vamos utilizar as orientações do braço e antebraço recebidas pelos sistemas AHRS para derivar o ângulo entre a articulação de maneira simples, focando em um dos eixos e calculando a rotação necessária para que possamos ir de uma posição a outra.

Capítulo 3

Desenvolvimento

3.1 Metodologia

3.1.1 Setup dos sensores

Usamos um *setup* simples para captura dos dados dos sensores, dado que a prioridade era o processamento dos dados. Para isso usamos um Arduino nano (escolhemos esse modelo por ser pequeno e leve, já que ficará no corpo do músico) conectado a cada sensor, e do Arduino transmitimos os dados para o computador via USB.

A conexão entre os dois dispositivos é feita com 4 cabos e está ilustrada na figura 3.1. Dois destes cabos são o pino de aterramento (GND ou *ground*) e o pino de alimentação (*VIN*); já para a transmissão de dados usamos o protocolo I2C, um protocolo serial síncrono desenvolvido pela *Philips* (Semiconductors, 2000) que usa uma porta SDA (*serial data*) e uma porta SCL (*serial clock*).

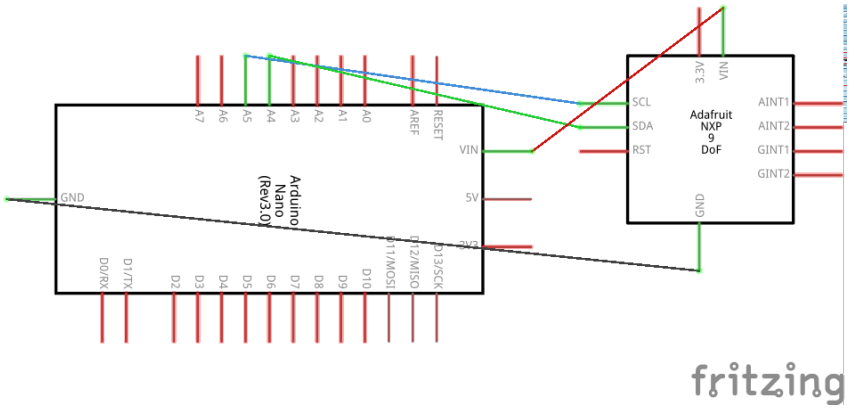


Figura 3.1: Esquema de conexão entre sensor e Arduíno

Para fixarmos o sensor na faixa que será colocada no corpo do performer, os dois

sensores são colocados na mesma posição relativamente à faixa, a fim de estabelecer um eixo de coordenadas similar no caso de o braço estar estendido (apesar de isso não ser uma obrigatoriedade). Na figura 3.2 podemos ver como os sensores foram utilizados durante os experimentos.

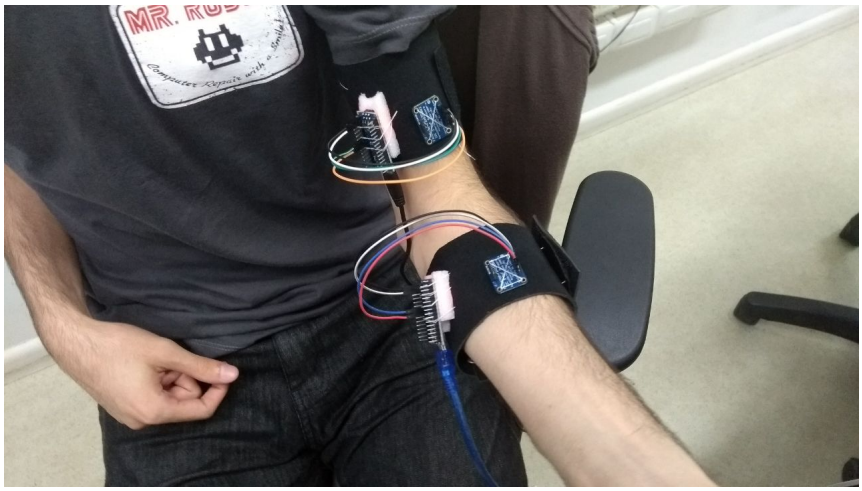


Figura 3.2: Setup usado nos experimentos

Devido à escolha de dispositivos leves e material flexível, conseguimos minimizar a limitação que os equipamentos causavam nos músicos. Na sessão 3.2.2 discutiremos os resultados obtidos durante os testes práticos com os músicos.

3.1.2 Algoritmo AHRS

Como falamos no capítulo 2 escolhemos trabalhar com o algoritmo de Madgwick para extrair a orientação dos sensores (Madgwick et al., 2011). Aqui vamos discutir a implementação feita em *Python* para esse trabalho baseada na implementação do próprio Madgwick[1].

O primeiro passo é extrair a orientação do sensor através da velocidade angular. Para obtermos essa informação usamos os dados do giroscópio para fazer o seguinte cálculo, onde g é o quatérnio $(0, g_x, g_y, g_z)$ contendo as leituras do giroscópio respectivamente nos eixos x, y e z e $\dot{q} = (\dot{q}_1, \dot{q}_2, \dot{q}_3, \dot{q}_4)$ corresponde à orientação estimada a partir dos dados do giroscópio e a estimativa anterior realizada pelo algoritmo $q = (q_1, q_2, q_3, q_4)$, computada como segue:

$$\dot{q} = \frac{1}{2} q \cdot g \tag{3.1}$$

[1] Disponível em `http://x-io.co.uk/open-source-imu-and-ahrs-algorithms/`

Após esse passo iremos calcular a orientação através do acelerômetro e do magnetômetro. Primeiro normalizamos as leituras dos mesmos, ou seja, convertemos $a = (ax, ay, az)$ em $a/\|a\|$ e $m = (mx, my, mz)$ em $m/\|m\|$ (usando a norma Euclideana).

Feito isso passamos para a etapa de calcular a compensação da direção do campo magnético, pois distorções magnéticas causam erros substanciais no cálculo da orientação dos sensores (Bachmann et al., 2004). Essas interferências são causadas pela proximidade com materiais ferro-magnéticos com o sensor. Pela natureza da nossa aplicação, sempre teremos esse tipo de material ao redor do sensor, pois sempre estaremos com Arduinos, computador e instrumento próximo, além da influência dos materiais desse tipo que estarão no ambiente em que o músico estiver.

Interferências nomeadas *hard-iron* podem ser corrigidas através de calibração pois são geradas por materiais que produzem um campo magnético aditivo ao da Terra, porém interferências chamadas *soft-iron*, são causadas por materiais que distorcem o campo magnético mas sem necessariamente gerar um campo próprio, de maneira que sua influência no campo depende da orientação do material em relação ao sensor portanto não podendo ser corrigida apenas por uma constante.

Para corrigir as interferências *hard-iron*, calibramos o sensor usando o algoritmo fornecido pela *Adafruit* para obter os valores que devem ser adicionados à medição do magnetômetro. Já para corrigir as interferências *soft-iron* é necessário calcular sua correção a cada iteração do algoritmo, como descreveremos a seguir.

Erros da estimativa relativos ao plano vertical à superfície da terra podem ser compensados pelo fato de o acelerômetro nos oferecer uma estimativa da orientação adicional. Porém erros da estimativa nos eixos relativos ao plano horizontal da superfície da Terra só podem ser corrigidos se tivermos uma referência de adicional da orientação do campo magnético.

Para isso primeiro calculamos a estimativa da direção do campo magnético da Terra, que chamaremos de emt, usando a medição do magnetômetro, como descrito na equação 3.2. Temos $\dot{m}$ que é o quatérnio $(0, \vec{m})$, onde $\vec{m}$ contém os valores normalizados de m, rotacionado pela orientação estimada na iteração anterior do algoritmo, q (q^* é sua conjugação):

$$emt = (0, emt_x, emt_y, emt_z) = q \times \dot{m} \times q^* \tag{3.2}$$

Podemos corrigir os efeitos de uma estimativa errada da inclinação do campo magnético

da Terra se a direção de referência do campo magnético usada pelo filtro, rmt, for da mesma inclinação. Podemos realizar isso computando rmt como emt normalizado para ter apenas componentes nos eixos x e z:

$$rmt = (0, rmt_x, rmt_y, rmt_z) = (0, \sqrt{emt_x^2 + emt_y^2}, 0, emt_z) \tag{3.3}$$

O passo seguinte do algoritmo é realizar o passo corretivo da orientação com os novos dados usando o método de otimização do gradiente. O método é iterativo e em cada passo ele toma a direção negativa do gradiente, que é a direção de declive máximo, com o intuito de achar um mínimo local. Esse método é um dos mais simples tanto em termos de implementação como em termos de computação.

Se soubermos a orientação do campo magnético da Terra, poderemos usar a medição do magnetômetro para estimar sua orientação em relação a Terra. Entretanto nunca teremos apenas uma única orientação do sensor para cada leitura feita, na verdade teremos infinitas soluções, que são todas as rotações da orientação correta em torno do eixo paralelo em relação a direção do campo magnético. O mesmo pode ser feito com a gravidade e a medição do acelerômetro. Podemos obter uma orientação do sensor se tivermos uma direção de referência do campo gravitacional, porém esta estimativa também encontraria o mesmo problema.

Por isso a estimativa da orientação usando o campo magnético e o campo gravitacional de forma separada são incompletas, conseguimos obter a estimativa em dois eixos da orientação, porém não é possível obter a estimativa correta, dentre todas as infinitas possibilidades, no terceiro eixo. Iremos combinar, através de fusão de sensores, a informação de ambos para conseguir uma orientação completa.

Podemos fazer essa estimativa inicial, incompleta, da orientação do sensor com os dados do magnetômetro através de uma formulação de um problema de minimização da função f_m, onde nos diz que a orientação do sensor é aquela que se alinha com a direção de referência do campo magnético, rmt com a medição realizada pelo sensor e é calculada da seguinte maneira:

$$\min_{q \in \mathbb{R}^4} f_m = \min_{q \in \mathbb{R}^4} (q^* \times rmt \times q - \dot{m}) \tag{3.4}$$

A mesma coisa pode ser feita com o campo gravitacional. Porém usaremos a direção da gravidade, rgt, que sabemos possuir componente apenas no eixo z, substituindo rmt

e usaremos $\dot{g} = (0, \vec{a})$, onde $\vec{a}$ são as leituras normalizadas do acelerômetro, substituindo *emt*:

$$\min_{q \in \mathbb{R}^4} f_a = \min_{q \in \mathbb{R}^4} (q^* \times rgt \times q - \dot{g}) \tag{3.5}$$

Como falamos anteriormente essas duas soluções separadamente não são suficientes para achar uma orientação completa (nos três eixos). Para isso combinamos f_m e f_a para obter a função f que possui um único mínimo dado que $rmt_x \neq 0$. A função f é definida por:

$$f = \begin{bmatrix} f_a \\ f_m \end{bmatrix}.$$

Para calcular o gradiente de f, ∇f, fazemos o seguinte:

$$\nabla f = J_{am}(q, rmt) \times f \tag{3.6}$$

onde $J(q, rmt)$ é o Jacobiano de f, que é a combinação dos Jacobianos de f_m e f_a, $J_m(q, rmt)$ e $J_a(q)$ respectivamente, definido como:

$$J_{am}(q, rmt) = \begin{bmatrix} J_a(q) \\ J_m(q, rmt) \end{bmatrix},$$

onde

$$J_a(q) = \begin{bmatrix} -2q_3 & 2q_4 & -2q_1 & 2q_2 \\ 2q_2 & 2q_1 & 2q_4 & 2q_3 \\ 0 & -4q_2 & -4q_3 & 0 \end{bmatrix}$$

e

$$J_m(q, rmt) = \begin{bmatrix} -2rmt_z q_3 & 2rmt_z q_4 & -4bxq_3 - 2bzq_1 & -4rmt_x q_4 + 2rmt_z q_2 \\ -2rmt_x q_4 + 2rmt_z q_2 & 2rmt_x q_3 + 2rmt_z q_1 & 2rmt_x q_2 + 2rmt_z q_4 & -2rmt_x q_1 + 2rmt_z q_3 \\ 2rmt_x q_3 & 2rmt_x q_4 - 4rmt_z q_2 & 2rmt_x q_1 - 4rmt_z q_3 & 2rmt_x q_2 \end{bmatrix}.$$

Assim podemos dar o passo corretivo da estimativa usando o método do gradiente, onde q' é a novo ângulo estimado e q a estimativa do passo anterior, da seguinte maneira:

$$q' = q - \mu \frac{\nabla f}{\|\nabla f\|} \tag{3.7}$$

O passo μ do método é definido de forma que garanta que a taxa de convergência é limitada por $\dot{q}$ pois assim evitamos ultrapassar a estimativa desejada causada por um

passo muito grande. Assim podemos calcular μ da seguinte forma, onde $freq$ é a taxa de amostragem:

$$\mu = \|\dot{q}\| \frac{1}{freq} \tag{3.8}$$

Após isso fazemos o passo final da fusão de sensores, agora com a informação que obtemos do giroscópio com a informação do magnetômetro já combinada com o acelerômetro, para calcular o passo corretivo

$$\dot{q} = 0.1q' \tag{3.9}$$

onde o valor 0.1 foi obtido experimentalmente, usando resultados apresentados em Madgwick et al. (2011). O valor ótimo desse parâmetro foi definido por ser aquele que é grande o suficiente para minimizar erros na computação de $\dot{q}$ porém pequeno o suficiente para não causar ruído devido a passos muito grandes no método do gradiente.

A fim de adequar o algoritmo ao nosso uso, fizemos algumas modificações. A primeira modificação é consequência do fato de usarmos *Python*: o algoritmo original usava a implementação rápida da inversa da raiz quadrada desenvolvida no jogo *Quake* (Lomont, 2003), porém em *Python* a implementação normal é a mais rápida[2].

Outra modificação feita foi a troca de plataforma. Como iriamos precisar utilizar um computador para processar o áudio, fizemos com que o Arduino assumisse apenas o papel de mensageiro dos dados dos sensores e estimamos a orientação dos sensores usando o Madgwick no computador, para não precisarmos nos preocupar com a limitação de processamento do Arduino Nano, que possui um poder de processamento bem menor que um computador e outros mini computadores como a *Raspberry PI*, e a necessidade de usar dois destes no corpo do músico.

3.1.3 Cálculo do ângulo de flexão

Como discutido no capítulo 2, simplificamos o problema para calcular apenas o ângulo de flexão do músico a fim de evitar que movimentos não conscientemente controláveis alterassem o efeito aplicado ao instrumento. Para realizar isso, primeiro reconvertemos cada quatérnio associado aos sensores para ângulos de Euler[3]. Após feito isso teremos 3 ângulos correspondentes o *heading*, *pitch* e o *yaw*, dando a orientação

[2] `http://ajcr.net/fast-inverse-square-root-python/`

[3] Essa conversão não é problemática pois o risco de *gimbal lock* ao trabalhar com esses ângulos só existe durante o cálculo das rotações.

de cada sensor. Para essa análise ser possível é necessário que os sensores estejam acoplados em posições específicas do corpo onde possamos garantir a estabilidade de seus sistemas de eixos, considerando a musculatura do braço e antebraço. Como podemos ver na figura 3.3, o *pitch* é o ângulo ligado à flexão do braço, de acordo com a orientação dos sensores quando acoplados no corpo.

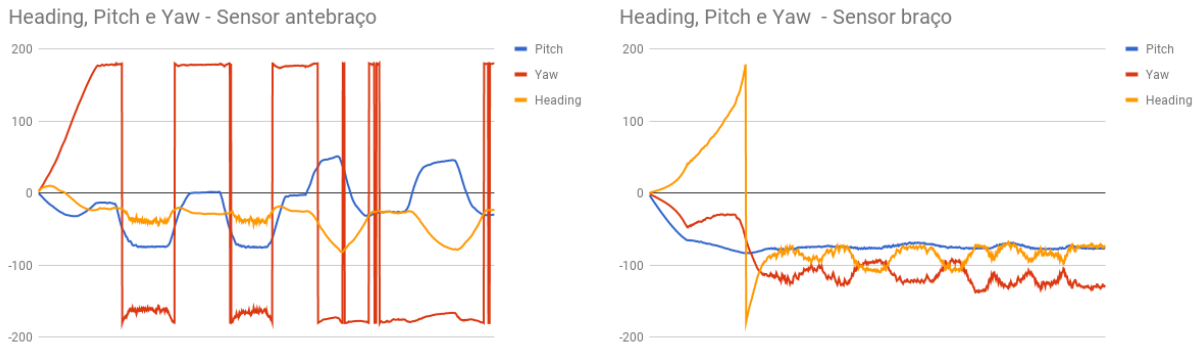


Figura 3.3: Saídas do heading, pitch e yaw dos dois sensores durante o movimento vertical do braço. *À esquerda*: sensor acoplado no antebraço. *À direita*: sensor acoplado no braço. É importante notar que os saltos de 360° que ocorrem no *yaw* do sensor do antebraço e uma vez no *heading* do sensor do braço não são descontinuidades, refletindo meramente a natureza cíclica dos ângulos.

A variação do *pitch* (linha azul) é bem clara quando analisamos o comportamento dos sensores durante o movimento vertical do antebraço, tanto de extensão (dois movimentos executados na primeira metade dos gráficos) quanto de flexão (idem na segunda metade), numa variação total de aproximadamente 180°. Em relação ao sensor do braço (gráfico da direita) o esperado seria que o ângulo ficasse estável durante os movimentos do antebraço; durante os testes observamos que o sensor acoplado no braço se mantém estável ao redor de -90° (linha azul) e o do antebraço varia entre um valor próximo de -75° (quando o antebraço está flexionado e próximo ao braço) a 90° (quando o antebraço está completamente esticado em relação ao braço).

Assim para calcular o ângulo entre o braço e antebraço fazemos a diferença do *pitch* de cada sensor. Porém o ângulo estimado dessa forma possui dois problemas ligados ao sistema AHRS usado em nossa aplicação:

- A demora para estabilizar o resultado no ângulo correto: isso ocorre devido ao algoritmo de Magdwick, que usa informação passada junto com os dados do sensor para se aproximar de forma iterativa da orientação correta, o que exige algumas iterações para obter o ângulo verdadeiro.

- A variação ao redor do ângulo quando o algoritmo se estabiliza: quando o algoritmo alcança o valor correto da orientação do sensor, a estimativa produzida fica variando ao redor do valor correto.

Como estamos trabalhando com um sistema em tempo real e conhecendo a natureza do algoritmo utilizado, fizemos um *oversampling* dos dados do sensor para alimentar o algoritmo e chegar ao ângulo estável mais rapidamente; além desse oversampling, usamos um filtro da média (passa-baixas) causal a fim de suavizar o ângulo estimado. Podemos ver na figura 3.4 a sequência em que as técnicas discutidas são aplicadas durante a estimativa do ângulo:

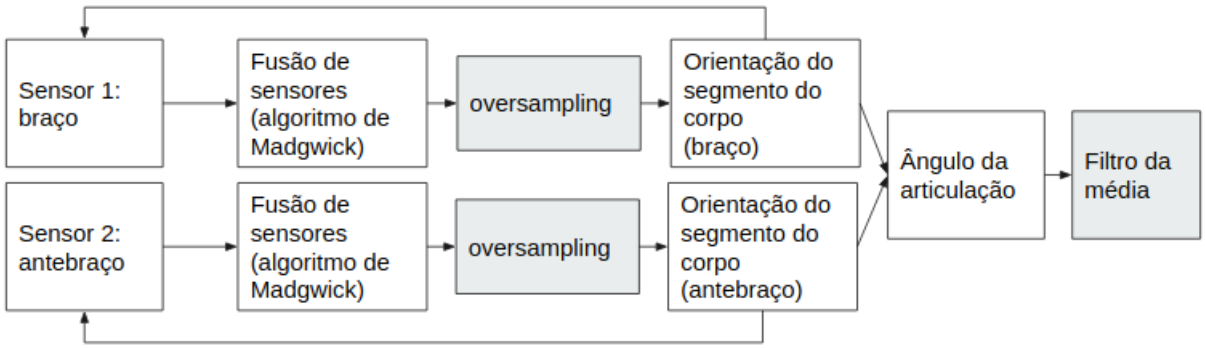


Figura 3.4: Diagrama do algoritmo que realiza a estimativa do ângulo entre braço e antebraço

3.1.4 Controle de efeitos de áudio

Como falamos no capítulo 2 os efeitos de áudio utilizados no nosso desenvolvimento foram implementados no *Pure Data*, que é uma linguagem de programação visual criada com o foco em computação musical. A escolha dessa linguagem se baseia em sua facilidade de uso para o processamento de áudio em tempo real, além de ela ser multiplataforma e amplamente utilizada por comunidades de artistas, que frequentemente disponibilizam suas implementações.

Usamos implementações já conhecidas de vários efeitos, como as do *Guitar Extended*[4], para fazer testes práticos relativos à performance do dispositivo em termos de conforto e manipulação do efeito em tempo real.

Para que pudéssemos usar essas implementações foram necessárias algumas modificações simples, como por exemplo usar comunicação com *sockets* para enviar informações do

[4] `https://guitarextended.wordpress.com/`

programa que estima o ângulo para o *patch* em *Pure Data*. Podemos ver na figura 3.5 um exemplo desse tipo de comunicação, onde o ângulo estimado é usado para controlar o parâmetro do tamanho da sala no efeito *reverb*.

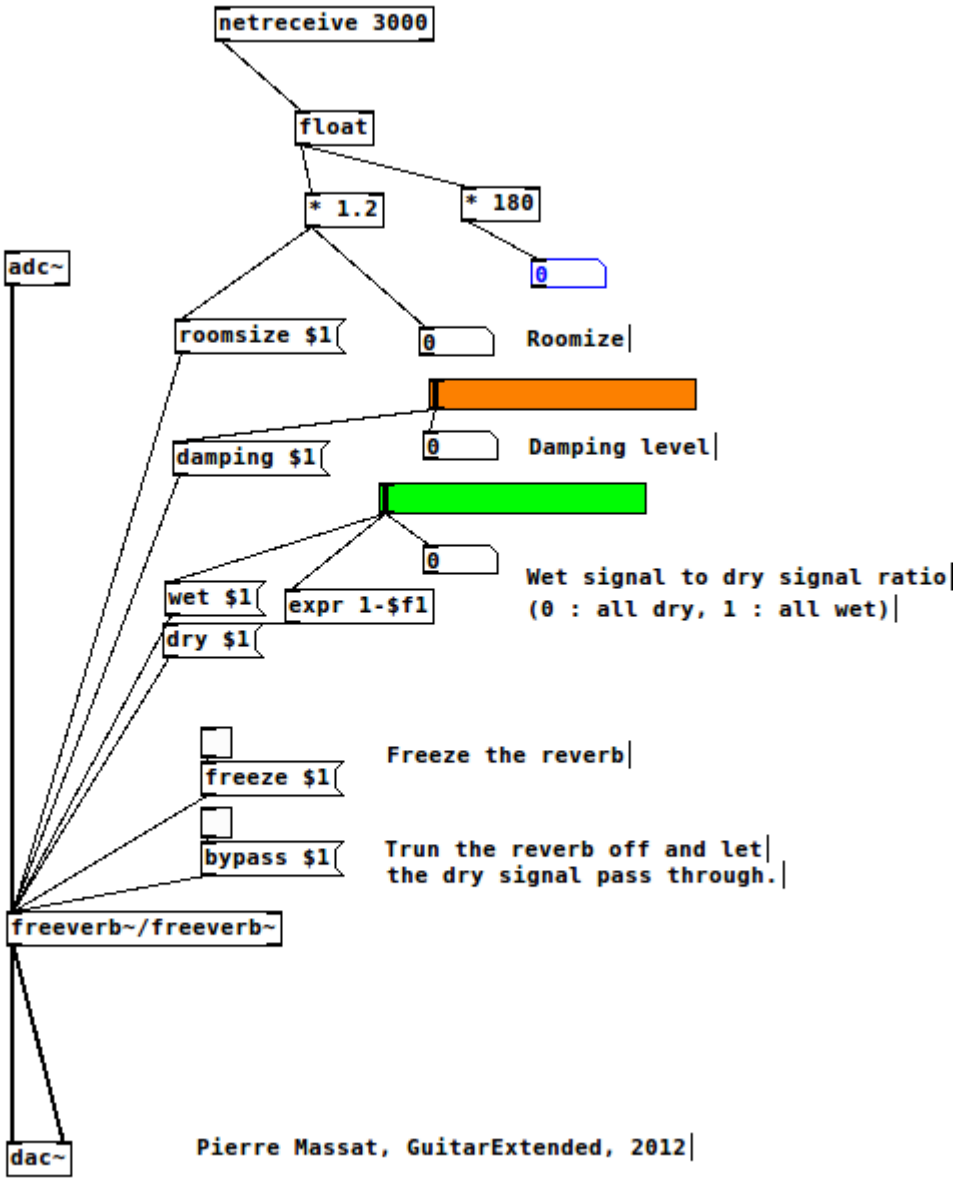


Figura 3.5: Modificação da implementação de um reverb do *Guitar Extended*. Recebemos os dados pelo objeto *netreceive (socket)*, fazemos o *casting* para trabalhar com float e ajustamos a escala para usar esse valor como parâmetro para o efeito. Esse ajuste é feito multiplicando o valor obtido por 1.2, sendo que esse parâmetro foi obtido empiricamente para realçar perceptualmente a alteração no efeito.

Como podemos observar, as alterações necessárias são bem simples de se adicionar em implementações de outros efeitos e fáceis de personalizar para cada efeito. Basi-

camente é necessário mapear os dados dos sensores em valores adequados conforme o parâmetro do efeito que se quer utilizar.

3.2 Experimentos

3.2.1 Experimentos numéricos da estimativa dos ângulos

Para testar a performance do algoritmo com as diferentes técnicas implementadas primeiramente coletamos os dados crus dos sensores acoplados a um músico durante a execução do baixo elétrico. Após esse procedimento, estimamos o ângulo com três diferentes versões do algoritmo: a primeira usando apenas as orientações dos sensores retornadas pelo algoritmo de Madgwick para calcular ângulo de flexão do braço; a segunda usando *oversampling* para alimentar o algoritmo de Madgwick; e a terceira usando, além do *oversampling*, o filtro da média para suavizar as variações em torno do ângulo correto.

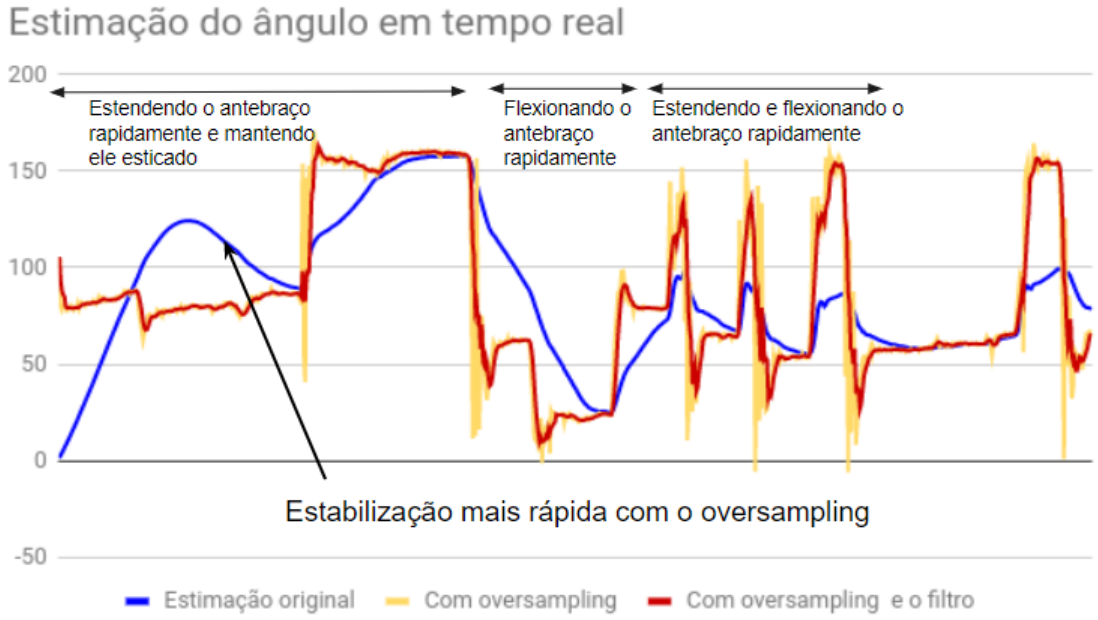


Figura 3.6: Comparação da performance do algoritmo durante o movimento de extensão e flexão do antebraço.

Podemos observar na figura 3.6 que o *oversampling* é importante para que o algoritmo alcance o ângulo desejado em um tempo satisfatório. Na primeira parte do gráfico onde o músico estendeu e depois flexionou seu antebraço e o manteve nessa posição por algum tempo vemos que a estimativa original consegue calcular o ângulo após algum

tempo, porém gastando um tempo muito maior que a versão com *oversampling*. Isso se torna um problema maior quando os movimentos são mais rápidos, como podemos observar na metade final do gráfico: a estimativa original não consegue se estabilizar ao redor do ângulo desejado antes do braço do músico alcançar uma outra posição, tornando problemática a sua aplicação no problema proposto nesse trabalho (devido à falta de conexão entre efeito e gesto).

Após passarmos a barreira inicial de calcular o ângulo desejado em um tempo suficientemente curto para controlar o efeito em tempo real, observamos a necessidade de suavizar a estimativa a fim de evitar que essa flutuação do ângulo, proveniente do algoritmo de estimativa e não do movimento do músico, fosse perceptível durante a performance do artista. Então usamos o filtro da média e pudemos observar uma diminuição dessa variação, principalmente nos momentos de mudança de direção da aceleração do antebraço.

3.2.2 Experimentos musicais com o contrabaixo aumentado

Os experimentos musicais foram feitos em duas etapas: primeiro testamos o nosso algoritmo com alguns efeitos, a fim de encontrar o valor ideal para cada uma das implementações dos efeitos usados (o valor enviado do ângulo estimado para o efeito é normalizado previamente, logo precisamos encontrar um mapeamento ideal para que o músico consiga percorrer toda extensão do efeito durante a performance), a segunda parte foi ajustar o valor de cada parte para cada músico de acordo com o alcance de seus movimentos de flexão do braço e suas preferências estéticas.

Foram feitos testes com três músicos e usamos dois efeitos: *fuzz* e *reverb*. Em termos de calibração dos parâmetros e controle dos efeitos os resultados foram satisfatórios: todos os músicos acharam o processo simples e facilmente reproduzível mesmo sem o auxílio de outra pessoa com conhecimento do algoritmo. Além disso o controle dos efeitos recebeu elogios por sua fluidez.

As maiores críticas foram feitas pela necessidade de cabos entre os Arduinos e o computador. Os participantes mostraram incômodo pela limitação da movimentação no espaço e pela preocupação em se manter próximo ao computador para evitar problemas de conexão entre os equipamentos (ou até mesmo derrubá-lo). Podemos diminuir esse incômodo usando métodos de comunicação que nos permitam eliminar os cabos

entre Arduino e computador: no capítulo 4 discutiremos possíveis soluções como trabalhos futuros.

Sobre o fato de estarem com equipamentos acoplados no braço e antebraço, dois dos participantes disseram que rapidamente se adaptaram à sensação, enquanto o terceiro expressou que seu desconforto se manteve durante todo o experimento.

Capítulo 4

Conclusões e trabalhos futuros

O trabalho consistiu no estudo de técnicas para derivar a orientação de sensores MARG, combinando a estimação do ângulo e sua aplicação em efeitos sonoros em um sistema de processamento de áudio em tempo real. O desenvolvimento desse sistema exigiu o estudo teórico e a aplicação prática de uma vasta gama de conhecimentos sobre processamento de sinais, fusão de sensores, Arduino e processamento de áudio em tempo real.

O projeto pode ser considerado bem sucedido em relação à proposta inicial, tendo atingido quase todos objetivos propostos durante o desenvolvimento do projeto. Conseguimos desenvolver um algoritmo que estima o ângulo entre as duas partes do corpo onde os sensores são acoplados com uma boa precisão, com baixo custo computacional e curto tempo de resposta, o que viabilizou o objetivo principal do projeto que era a criação de um contrabaixo aumentado que usasse o ângulo do braço e antebraço do músico para controlar efeitos aplicados ao som gerado pelo instrumento em tempo real usando dispositivos acessíveis. Além disso os músicos que participaram dos experimentos relataram facilidade em utilizar e personalizar o sistema, mesmo dispondo de poucas instruções iniciais.

Como trabalhos futuros desejamos coletar mais dados para melhor validação do algoritmo de estimação do ângulo da articulação do cotovelo. Uma possível abordagem seria utilizar equipamentos de captura de vídeo ou *motion capture* para realizar a captura de movimentos a fim de estabelecer um *ground truth* e melhor avaliar a precisão do nosso algoritmo. Com esses dados seria possível estudar estratégias para melhorar a precisão da estimativa do ângulo.

Além disso pretendemos melhorar o conforto do usuário no uso do dispositivo. O

primeiro passo seria usar comunicação *bluetooth* para eliminar a necessidade de cabos entre Arduino e computador. A segunda proposta seria utilizar apenas um Arduino para realizar a comunicação entre sensor e computador a fim de diminuir os custos.

Referências Bibliográficas

Bachmann E. R., Yun X., Peterson C. W., An investigation of the effects of magnetic variations on inertial/magnetic orientation sensors, 2004, vol. 2, p. 1115

Carrier G. F., Krook M., Pearson C. E., Functions of a complex variable: Theory and technique. SIAM, 2005

Crowe M. J., A history of vector analysis: The evolution of the idea of a vectorial system. Courier Corporation, 1967

Dam E. B., Koch M., Lillholm M., Quaternions, interpolation and animation. vol. 2, Datalogisk Institut, Københavns Universitet, 1998

Elmenreich W., An introduction to sensor fusion, Vienna University of Technology, Austria, 2002

Gong N.-W., Zoran A., Paradiso J. A., Inkjet-printed conductive patterns for physical manipulation of audio signals, 2013, pp 13–14

Hamilton W. R., On Quaternions; or on a new System of Imaginaries in Algebra (letter to John T. Graves, dated October 17, 1843), Philos. Magazine, 1843, vol. 25, p. 489

Hamilton W. R., Elements of Quaternions. vol. 1, Longmans, Green, and Company, 1899

Lomont C., Fast inverse square root, Tech-315 nical Report, 2003, p. 32

Madgwick S. O., Harrison A. J., Vaidyanathan R., Estimation of IMU and MARG orientation using a gradient descent algorithm, 2011, pp 1–7

Mahony R., Hamel T., Pflimlin J.-M., Nonlinear complementary filters on the special orthogonal group, IEEE Transactions on automatic control, 2008, vol. 53, p. 1203

Medeiros C. B., Wanderley M. M., A comprehensive review of sensors and instrumentation methods in devices for musical expression, Sensors, 2014, vol. 14, p. 13556

Ramkissoon I., The Bass Sleeve: A Real-time Multimedia Gestural Controller for Augmented Electric Bass Performance., 2011, pp 224–227

Samora da Graça V., Efeitos Sonoros em Tempo Real, 2016, p. 49 p.

Semiconductors P., The I2C-bus specification, Philips Semiconductors, 2000, vol. 9397, p. 00954

Silva A. M. d., A representação das matrizes de rotações com o uso dos quatérnios: aplicações à fotogrametria, 2014

Tseng S. P., Li W.-L., Sheng C.-Y., Hsu J.-W., Chen C.-S., Motion and attitude estimation using inertial measurements with complementary filter, 2011, pp 863–868

Wanderley M. M., Gestural control of music, 2001, pp 632–644

Wanderley M. M., Instrumentos Musicais Digitais-Gestos, Sensores e Interfaces, Em Busca da Mente Musical, 2006, vol. 60, p. 21

Yun X., Bachmann E. R., Design, implementation, and experimental results of a quaternion-based Kalman filter for human body motion tracking, IEEE transactions on Robotics, 2006, vol. 22, p. 1216