INTERFACE GESTUAL DE PERCUSSÃO

Nycholas Maia, José Eduardo Fornari Novo Junior
Universidade Estadual de Campinas

Resumo:
Enquanto instrumentos reais de percussão são grandes, caros e pouco flexíveis à exploração de novas sonoridades, como aquelas proporcionadas por técnicas estendidas, instrumentos virtuais de percussão são portáveis, baratos e totalmente flexíveis à exploração de novas sonoridades. O objetivo desse projeto foi criar uma interface de baixo custo financeiro, para a criação de um instrumento virtual de percussão que seja acessível a todos os estudantes e profissionais de música. Para tanto, este projeto também criou novos modelos computacionais de interpretação de gestos musical, a fim de que o músico usuário possa utilizar esta interface gestual como um instrumento virtual de percussão, da mesma forma que utiliza um instrumento de percussão real, sem grande necessidade de adaptação de sua técnica ou de seu gesto musical.

Palavras-chave: Interfaces Gestuais, Síntese Sonora, Modelos Computacionais

1 Introdução

Atualmente, apesar dos avanços tecnológicos, ainda são escassas as ferramentas computacionais que auxiliam estudantes de música a expressarem a sua visão estética e a sua técnica performática na execução de uma obra musical. Por isso, há a necessidade de se criar ferramentas computacionais, onde a sensibilidade e os conhecimentos do usuário sobre a obra musical possam ser testados e implementados simultaneamente à execução da música, onde o resultado sonoro é gerado em tempo real (ou seja, durante a performance musical).

BALDÉ (2008) desenvolveu um *modelo computacional* utilizando tecnologia de comunicação sem fio (*Bluetooth)* cujo protocolo foi descrita por KARDACH (2000) e ALCORN (2011) para a manipulação de informações e valores de diversos sensores com a finalidade de exploração de novas sonoridades em música experimental. No entanto, este *sistema* não é específico para promover a interpretação musical e sua aquisição não é acessível à grande maioria dos estudantes que buscam o aperfeiçoamento da técnica de instrumentos musicais de percussão.

Recentemente surgiram no comércio algumas interfaces gestuais, tais como o controle-remoto do videogame Wii (Wiimote), lançado em 2006. Com a queda do custo viabilizou-se o acesso à utilização artística desses sensores, tais como acelerômetros, giroscópios e câmeras de infravermelho. Com o uso de tais sensores, é possível capturar os gestos de interpretes; não apenas músicos, mas também de outras áreas artísticas, como: dança, teatro, artes plásticas, e utilizá-los de modo a proporcionar a interatividade de discursos artísticos multimodais. O uso de interfaces gestuais combinado com algoritmos para gerar sons compõe um instrumento virtual, o qual pode assumir inúmeras sonoridades dependendo apenas de sua programação.

MALLOCH e WANDERLEY (2007) desenvolveram recentemente uma família de instrumentos virtuais chamados "T-Stick", os quais são compostos por um hardware dedicado que envia sinais para um computador, que, por sua vez, possui o software MAX para gerar múltiplos timbres.

XENIA, ERIKA e STEWART (2012) criaram um projeto intitulado "*Orquestra Digital*", o qual integra diversos instrumentos virtuais e executa um repertório próprio, encomendando suas obras a compositores contemporâneos.

KIEFER, COLLINS e FITZPATRICK (2008) publicaram um trabalho chamado "*HCI Methodology For Evaluating Musical Controllers*", no qual são discutidos alguns exemplos de "interfaces indivíduo-máquina" no âmbito da música, visando controlar e modificar sons computadorizados através de gestos do usuário.

Recentemente, HOLLAND, WILKIE, **MULHOLLAND** e SEAGO (2013) publicaram um livro que discute a interação musical, por meios computacionais, na criação de intrumentos virtuais e tecnologias de música interativa. O projeto aqui apresentado tem como diferencial criar uma interface gestual de baixo custo financeiro, que seja acessível a todos os estudantes e profissionais de música. Além disso, esta nova interface se propõe a criar novos modelos computacionais de interpretação de gestos musicais, a fim de que o usuário possa utilizá-la da mesma forma que utiliza um instrumento de percussão real, ou seja, sem grande necessidade de adaptação de técnica ou de gesto. Uma outra característica dessa interface é a facilidade de manipulação e a conexão com inúmeros softwares comerciais ou livres.

O trabalho envolvido em criar uma nova interface gestual de percussão é motivado pelo desejo observado entre os estudantes e profissionais da área de música de possuir uma alternativa de baixo custo, fácil transporte e armazenamento, de instrumentos musicais de percussão.

Muitos desses instrumentos musicais reais de percussão são de grande porte e frágeis, sendo necessário muito cuidado no transporte dos mesmos, além de exigir proteções especiais chamadas "*bags"* ou *"cases"* e transportes exclusivamente dedicados a esse tipo de locomoção.

Além da dificuldade do transporte, por serem muito sensíveis, tais instrumentos têm que ser guardados em grandes salas, longe do sol, da humidade e de outras intempéries que possam danificar a estrutura física desses instrumentos musicais. Por isso, esse trabalho também visou minimizar essas grandes dificuldades para os alunos de percussão, as quais são: transporte, portabilidade, armazenamento e conservação dos instrumentos musicais reais de percussão.

A nova interface de percussão para a criação de instrumentos virtuais de percussão aqui descrita visa também explorar novas sonoridades de percussão através de um sistema de síntese sonora em tempo real, porém mantendo a fidedignidade técnica de um instrumento de percussão real, para facilitar a adaptação do usuário. Sendo assim, as características técnicas musicais desta interface são as mesmas do instrumento de percussão real.

Desta forma, espera-se que esta nova interface seja eficiente e suficiente para que estudantes e profissionais da área da música possam ter em seus estudos ou apresentações artísticas um instrumento de percussão virtual, gerando diversas sonoridades eletrônicas, além de poder simular alguns timbres já conhecidos como, por exemplo, tímpanos sinfônicos, marimbas e bumbos sinfônico.

É importante ressaltar que o timbre musical gerado por tal modelo computacional é muito flexível, pois é sintetizado em tempo real e utiliza o protocolo

MIDI como forma de parametrização do processo computacional de geração sonora.

Este protocolo, descrito por HECKROTH (1995), é compatível com a grande maioria de sintetizadores, *samplers* e outros equipamentos tecnológicos de modelamento sonoro, como, por exemplo, MAX (2014) e PureData (2014).

Assim, a interface aqui apresentada é capaz de produzir inúmeras sonoridades distintas, sendo da escolha do usuário o timbre que deseja sintetizar e utilizar em cada *performance*, o que dá uma ampla liberdade para o músico na criação, modelamento e mixagem deste áudio em seus estudos e performances musicais.

O funcionamento da interface é detalhado nas seções que se seguem.

2 A Construção da Interface Gestual

Para a realização deste projeto, foi inicialmente construída uma placa de dimensões 6 cm x 14cm com 172 LEDs que emitem raios de luz infravermelha, a qual constitui um campo de luz invisível de aproximadamente 2,5m x 2,5m, a uma distância de 3 metros do usuário. Este constitui o campo gestual de atuação do usuário.

Na Figura 1 são apresentados o projeto e o protótipo da placa contendo os LED. Esta foi projetada utilizando o software Eagle (2014).

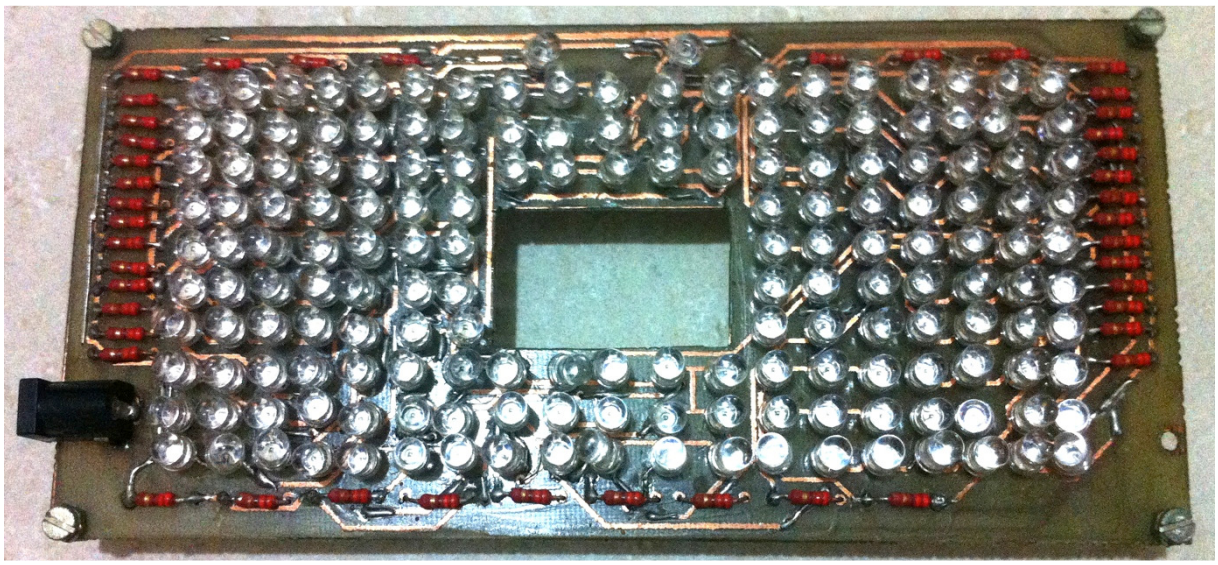

Figura 1. Placa de LEDs infravermelhos

O usuário interage com o sistema através de uma baqueta modificada. Na ponta desta baqueta existe um pequeno refletor de luz infravermelha, o qual reflete no sentido oposto o feixe de luz infravermelha gerado pela placa da Figura 1. Este é captado pela câmera do controle-remoto do *Wiimote* ilustrado na Figura 2.

Figura 2. Visão em diversas perspectivas do controle-remoto Wiimote, onde a perspectiva frontal (acima) apresenta a câmera de captação de luz infravermelha.

Quando a câmera do *Wiimote* capta a luz infravermelha refletida pela baqueta, ela rastreia a posição desse ponto e o transmite para o computador, via Bluetooth. Os parâmetros passados são: a posição da ponta da baqueta (ponto) no eixo X (horizontal) e a posição no eixo Y (vertical) em relação à câmera. Esses parâmetros são enviados continuamente, numa frequência de 24Hz, uma vez que a taxa de captura da câmera infravermelha do Wii é de 24 quadros por segundo.

No computador, esses parâmetros são processados por um modelo computacional desenvolvido pelo autor principal, em linguagem C#. Este modelo se incumbe de interpretar os gestos do usuário e gerar uma saída de dados no protocolo MIDI.

Essas mensagens MIDI são recebidas pelo *sampler* virtual *Kontakt 5 (2014);* um aplicativo comercial que contém um banco de dados de sons amostrados, entre estes, sons de instrumentos de percussão. O fluxograma processual desta interface é mostrado na Figura 3.

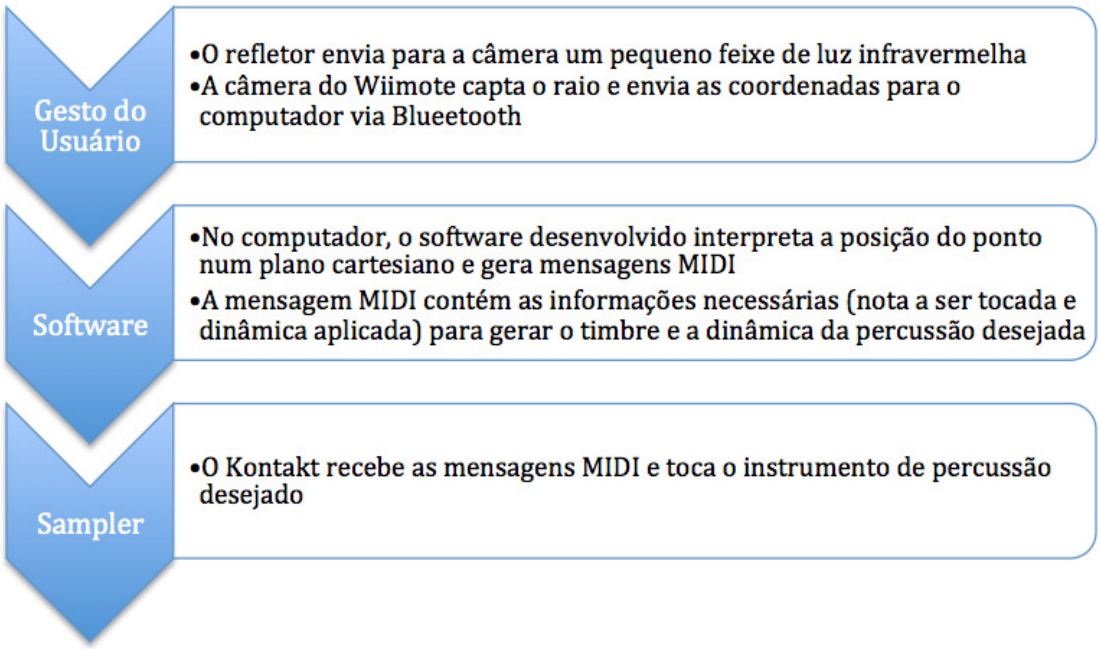

Figura 3. Visão geral da Interface: fluxograma do sistema

Como citado acima, para o desenvolvimento do modelo computacional que interpreta os gestos do usuário, foi usada a linguagem de programação C# dentro do ambiente de desenvolvimento Microsoft Visual Studio (2012). As configurações de hardware do notebook PC usado foram: processador Intel i7 2.66GHz, 8Gb de memória RAM, 500Gb de HD e sistema operacional Windows 7.

2.1 Algoritmos Desenvolvidos

Visando o aperfeiçoamento deste modelo computacional, foram testados vários algoritmos diferentes para a aquisição e interpretação dos gestos. Os mais relevantes são descritos abaixo.

2.1.1 Primeiro Algoritmo

Neste primeiro algoritmo, pensou-se em simular o som do golpe da baqueta na pele de um instrumento de percussão membranofônico, usando uma lógica de comparação de posição cartesiana entre dois pontos: o “ponto atual” e o ”ponto anterior”. Dessa forma, o “ponto atual” é a última posição cartesiana do feixe de luz infravermelha, registrada pelo software, e o “ponto anterior” é a penúltima posição cartesiana deste mesmo feixe de luz infravermelha registrada pelo software.

O algoritmo verifica reiteradamente o valor das posições do “ponto atual” e do “ponto anterior” no eixo Y. Caso o “ponto atual” seja menor que o “ponto anterior”, o software interpreta que o gesto da mão do usuário está sendo descendente. Caso o “ponto atual” seja maior que o “ponto anterior”, o software interpreta que o gesto da mão do usuário está sendo ascendente.

No momento da inversão do sentido descendente para o sentido ascendente, ou seja, no ataque do instrumento de percussão, o software gera uma mensagem MIDI, a qual irá disparar um som através do sampler Kontakt 5. Para obter a dinâmica do gesto, o software utiliza do armazenamento dos pontos de máximo e mínimo de

cada gesto. A diferença entre o ponto de máximo e o ponto de mínimo é dimensionada numa escala de 0 à 127 (padrão MIDI) e transformada em mensagem MIDI. Essa mensagem MIDI é recebida pelo Kontakt juntamente com a mensagem de ataque do instrumento de percussão, que gera assim um som de percussão com ataque e dinâmica controlada em tempo real através do gesto do usuário.

O problema encontrado neste algoritmo é que o disparo da mensagem MIDI somente ocorre depois da inversão de sentido e não exatamente no ataque do instrumento de percussão. A latência ou atraso do sistema desejado é de aproximadamente 50 milissegundos, mas com esse algoritmo o atraso foi de aproximadamente 200 milissegundos, o que inviabilizou a sua utilização.

2.1.2 Segundo Algoritmo

Este é um aprimoramento do primeiro algoritmo, no qual a detecção da batida do instrumento de percussão não é dada pela inversão do sentido do movimento do gesto, mas sim com o resultado nulo da derivada da curva do espaço em relação ao tempo gerado pela somatória dos pontos do gesto do usuário.

Quando o resultado dessa derivada é nula, entende-se que houve um “ponto de mínimo” no gesto. Sendo assim, neste momento o software gera a mensagem MIDI de ataque juntamente com a mensagem MIDI de dinâmica, sendo que a lógica para o cálculo da dinâmica segue inalterada.

O problema encontrado neste algoritmo foi que o número de pontos que geram a curva do espaço (24 pontos por segundo) não é suficiente para um cálculo preciso dessa derivada, o que gera algumas mensagens MIDI aleatórias.

2.1.3 Terceiro Algoritmo

Neste algoritmo pensou-se em delimitar um nível de limiar (*threshold*) para o disparo da mensagem MIDI associada ao ataque. Caso o “ponto atual” do deslocamento atravessasse este limiar (como que formando uma linha horizontal virtual) num gesto descendente, o software emitiria a mensagem MIDI de ataque do instrumento de percussão. A lógica para o cálculo da dinâmica segue inalterada em relação ao algoritmo anterior.

O problema encontrado neste algoritmo foi também relacionado à pequena taxa de 24 por segundo. A probabilidade de um ponto captado coincidir com o limiar atribuído é bem pequena. Sendo assim, em muitos ataques gestuais processados por este algoritmo não são geradas as devidas mensagens MIDI para o ataque da percussão.

2.1.4 Quarto Algoritmo

Neste algoritmo pensou-se em usar a aceleração do gesto como modo de disparo da mensagem MIDI. No exato momento em que a baqueta do usuário atinge o instrumento de percussão, a aceleração do “ponto atual” é zero. Sendo assim, quando a aceleração do “ponto atual” for nula, o software dispara a mensagem MIDI de ataque. Para o cálculo da dinâmica também foram usados os valores de aceleração do “ponto atual”. A cada gesto do usuário, o software registra o valor máximo de aceleração do “ponto atual”. Esse valor é dimensionado numa escala de 0 a 127 (padrão MIDI) e transformado em mensagem MIDI.

O problema encontrado neste algoritmo foi o fato de que o controle da aceleração é não é trivial. Um exemplo disso é quando se tem um gesto descendente

em movimento uniforme, ou seja, velocidade constante e aceleração nula durante todo o gesto. Neste exemplo, são geradas inúmeras mensagens MIDI no decorrer do gesto, o que inviabiliza o uso deste algoritmo.

3 Interface II

De acordo com o que foi constatado nas implementações anteriormente descritas dos 4 algoritmos, o sistema teve que ser remodelado. Nessa seção será abordada a remodelagem do sistema, visando resolver as limitações de hardware e software já testados anteriormente.

Durante os experimentos realizados, constatou-se que havia uma limitação do hardware do controle *Wiimote*, usado como câmera, o qual captura suas imagens numa taxa de amostragem de 24 quadros por segundo. Isso corresponde a 41.66 milissegundos de latência no sistema somente por parte da câmera, o que compromete a execução adequada do instrumento musical de percussão virtual.

Outra limitação de hardware é o fato de que o *Wiimote* envia seus dados via Bluetooth para o computador numa frequência de 100Hz. Sendo assim, temos 10 milissegundos de latência somente por parte da transferência de dados entre a câmera e o computador.

Desse modo, analisando somente o atraso proveniente do hardware usado, temos aproximadamente 50 milissegundos de latência. Uma latência alta é muito prejudicial a sistemas que tem por prioridade a execução em tempo real. Quando ela cresce, menos o usuário tem a sensação de interação com o sistema.

A linguagem de programação utilizada foi o C#, dentro do ambiente de desenvolvimento Microsoft 2012. O estudo e comparação entre as linguagens, mostrou que o C# não possui um bom desempenho para aplicações de alto processamento em tempo real. Além disso, o ambiente Visual Studio gera vários arquivos e linhas de código desnecessárias, o que aumenta a complexidade, peso e a ineficiência do software. A junção do ambiente de desenvolvimento Visual Studio com a linguagem de programação C# produz um atraso de resposta no sistema de aproximadamente 150 milissegundos, um valor muito alto para que o usuário tenha uma sensação de execução em tempo real.

Visando a redução da latência/atraso do sistema, espera-se encontrar um novo hardware e um novo software que seja mais veloz e mais eficiente do que os anteriormente utilizados. Para a nova câmera, torna-se necessário uma taxa de amostragem de, pelo menos, 100 quadros por segundo.

Assim, tornou-se necessário utilizar um novo ambiente de programação que não gere arquivos ou linhas de código desnecessários e uma linguagem de programação mais eficiente para o tratamento de dados em tempo real. Além disso, será necessária a criação de um novo algoritmo mais eficiente evitando os erros anteriores.

Neste sentido, encontrou-se um hardware que possui as características necessárias ao desenvolvimento de um protótipo: o *PS Move*.

O PS Move é um equipamento para controle do videogame PlayStation 3.

Este equipamento atendeu o requisito desejado de baixa latência, mas exigiu uma reestruturação nas bases do novo sistema desenvolvido.

O controle *PS Move* substituiu a baqueta com o refletor de infravermelho.

Deste modo, o novo sistema se restringiu, apenas, a um computador e o *PS*

Move.

Uma vantagem desse novo sistema é o baixo custo; aproximadamente, R$ 150,00 (cotação em junho de 2013) e a sua fácil obtenção em qualquer loja de videogames. Com esse novo conjunto de equipamentos e software criou-se a "Interface II".

A Figura 4 mostra a imagem do PS Move.

Figura 4. Controle *PS Move: O controle acende a bola azul mostrando que está ligado*

Um estudo sobre as linguagens de programação mostrou que a melhor solução seria a linguagem C/C++ por sua velocidade e eficiência no tratamento de dados em tempo real. Desta vez, não foi utilizado qualquer ambiente de desenvolvimento. Ao invés disso, todo o código foi escrito num editor de texto padrão (aplicativo *Notepad*) e compilado via linha de comando (aplicativo *Terminal*). Visando um processamento mais rápido, o notebook PC foi substituído por um MacBook Pro, com processador Intel i7 2.66GHz e 8Gb de memória RAM, rodando o sistema operacional Mac OSX 10.8 (Mountain Lion).

Com essas modificações de hardware, obtivemos um novo sistema baseado

em quatro etapas, como mostra a Figura 5.

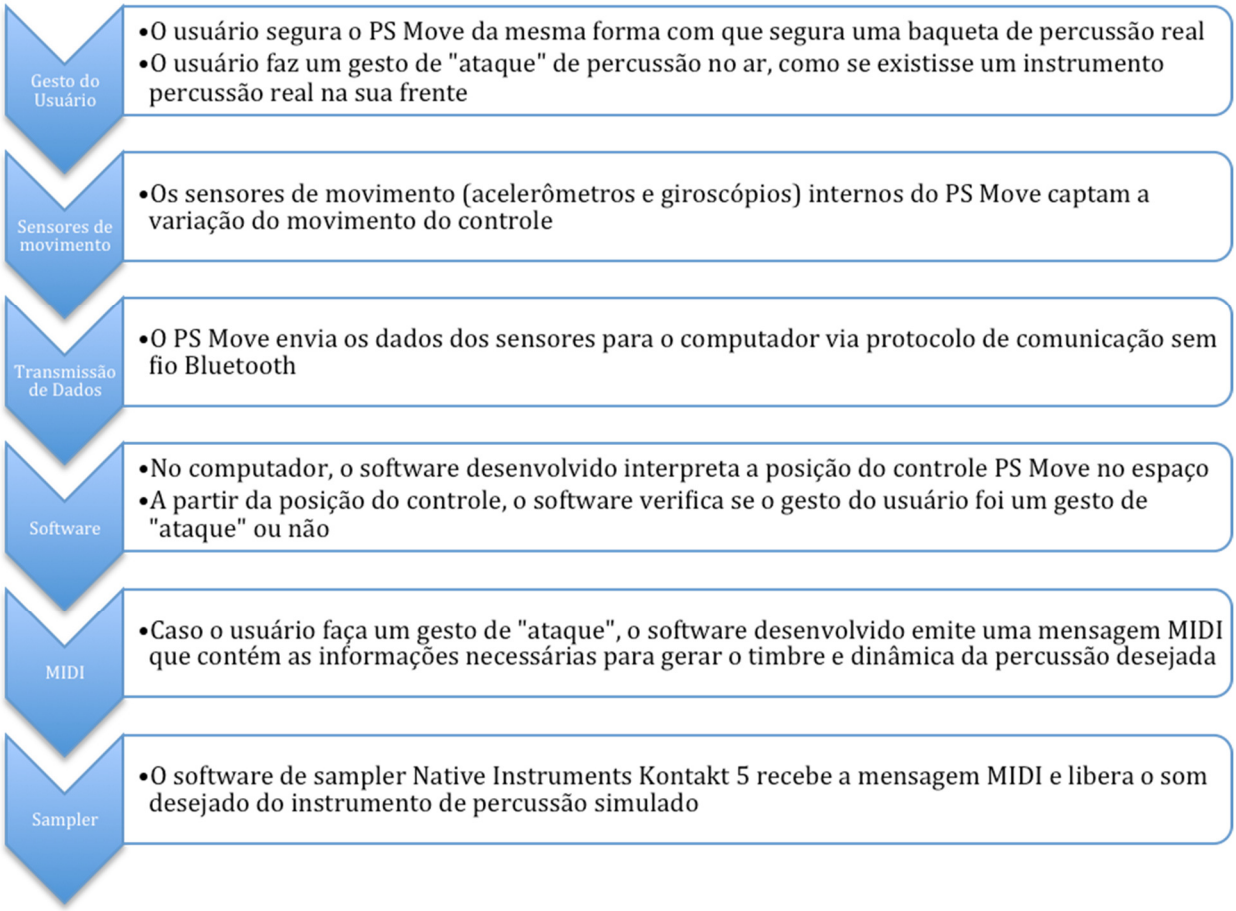

Figura 5. Esquema geral da Interface II

O algoritmo foi criado de acordo com a prática comum de instrumentos de percussão que ficam posicionados com a pele horizontal (virada para cima) como, por exemplo, a caixa clara e o tímpano, como mostra a Figura 6. De acordo com os alunos citados acima, o importante ao tocar esses instrumentos é a manutenção do ângulo reto (90º) entre o antebraço e o braço do usuário. Além disso, quando a baqueta atinge a pele do instrumento, ela deve estar praticamente paralela à pele, ou seja, o ângulo entre a baqueta e a pele é próximo a 0º.

Figura 6a. Baqueta paralela à pele da caixa

Figura 6b. PS Move funcionando

Com essas informações, o algoritmo tornou-se mais simples e foi possível aplicá-lo de acordo com o seguinte procedimento:

Inicialmente, é necessário calibrar os sensores de posição e movimento internos do controle *PS Move* (acelerômetros e giroscópios). Acelerômetros são sensores que medem a aceleração e giroscópios são sensores que medem a velocidade angular. Para isso, basta deixar o controle do *PS Move* sob uma mesa reta e informar ao software para iniciar a calibração. Neste processo, o software calibra os acelerômetros e giroscópios, criando uma referência de "ângulo 0º", ou seja, a referência de paralelismo com a pele do instrumento como mostrado na figura 6b. Ocorre também uma redução de ruído de sinais indesejados no processamento apenas nivelando o sinal já existente (sinal ruidoso) a uma "referência 0º", ou seja, igualando ele a zero.

Este processo leva aproximadamente três segundos. Quando ele está finalizado, o PS Move acende o globo de luz, que fica na sua extremidade, emitindo uma cor verde, o que sinaliza que o equipamento terminou o processo de calibração e que o usuário pode começar a tocá-lo.

Uma vez calibrado, o modelo computacional fica aguardando o momento em que o usuário irá fazer o gesto de ataque do instrumento de percussão, cruzando o "ângulo 0º". Quando isto ocorre, o computador dispara uma mensagem MIDI para o software Kontakt 5. Este recebe a mensagem MIDI e toca o instrumento de percussão virtual selecionado previamente no próprio Kontakt pelo usuário.

Com o remodelamento da primeira interface, conseguiu-se reduzir o atraso sonoro (latência) entre o gesto e o som disparado, viabilizando o uso dessa nova Interface como um verdadeiro instrumento de percussão virtual. Além disso, conseguiu-se reduzir os custos de implementação e facilitar a acessibilidade e a aquisição deste hardware para o usuário final.

Mesmo com estes benefícios, ainda ocorreu um problema referente ao hardware. Quando a interface remodelada foi testada, por alunos de graduação de Percussão do Instituto de Artes da UNICAMP, estes sentiram a falta do rebote da baqueta, ou seja, da força em sentido contrário ao gesto provocado pela reação da pele na baqueta tradicional. Isso ocorreu porque, até aquele momento, a Interface II baseava-se somente em movimentos do *PS Move* sem qualquer amortecimento ou impulso causados pela interação da baqueta com a pele de instrumentos musicais reais.

Desta forma, para que os percussionistas pudessem gerar o som ocasionado pelo rebote, eles eram obrigados fazer repetidamente o gesto de ida e de volta do *PS Move*, o que é inconveniente para o músico.

A partir dessa constatação desenvolveu-se uma nova versão do hardware e respectivamente um novo software, para obter o som e a sensação de rebote da baqueta no instrumento virtual. Este foi chamado de Interface III.

4 Desenvolvimento da Interface III

Uma vez que os percussionistas estão habituados a sentir o rebote da pele dos seus instrumentos reais, um instrumento virtual deve emular tanto essa sensação sonora quanto a realimentação tátil.

Neste sentido, buscou-se criar uma pele física que, ao ser tocada, reproduzisse o rebote esperado.

4.1 Criação da Pele de Percussão

O estudo de vários tipos de materiais, flexibilidade e custo, revelou que o plástico de polietileno, usado para a confecção de *banners*, seria uma boa opção para a confecção da pele de baixo custo. Esse material é flexível, barato e suporta altas tensões. Para a criação do instrumento, foi usado um aro de pneu de bicicleta, de 26 polegadas, de alumínio, onde foi duplicado o número de furos de raio, totalizando em 52 pequenos furos internos, como mostra a Figura 7b.

Figura 7a. Aro de bicicleta de 26 polegadas.

Figura 7b. Detalhe dos furos do aro

Para fixá-la e tencioná-la no aro, a pele foi recortada em forma de círculo e furada nas extremidades para a colocação de ilhós de metal. Os 56 ilhós foram usados para que a pele não rasgasse em altas tensões. Para amarrar a pele no aro foi usada linha de pesca de alta tensão, como mostra a Figura 8a.

Figura 8a. Fixação da pele por meio de Ilhós e linha de pesca

Figura 8b. Visualização da pele esticada

O protótipo do aro com a pele deu um bom resultado final, mas houve a necessidade de esticar a pele ainda mais, para que obtivéssemos um rebote parecido com o do instrumento real.

Outros meios foram experimentado para esticar a pele, inclusive utilização de uma catraca mecânica. Mesmo assim, ela não esticava a pele por igual e deixava irregularidades nesta superfície.

O objetivo inicial era possibilitar que o usuário pudesse usar uma baqueta de

percussão comum, sem a necessidade de hardware especial, facilitando ainda mais o uso da interface.

Dessa vez, usamos o aro de bicicleta porém sem a pele de polietileno. Ao invés dela, fixamos 52 LEDs de luz infravermelha no furos do aro, como mostra em detalhe a Figura 10. Desses 52 LEDs, 26 eram emissores de luz infravermelha, posicionados em uma metade do aro, e os outros 26 eram receptores de luz infravermelha, posicionados na metade oposta a dos emissores. Deste modo, tínhamos para cada LED emissor, um receptor correspondente, perfeitamente posicionado na direção de recepção do emissor.

Este sistema cria um plano de infravermelho, no lugar da pele de percussão.

Quando o sistema é ligado, o LED emissor emite a luz infravermelha para o seu LED receptor correspondente. Enquanto os 26 LEDs receptores recebem luz dos seus 26 LEDs emissores, o plano da pele não é atingido pela baqueta. Caso houvesse interrupção desse fluxo emissor-receptor pela baqueta, o plano da pele era atingida e o software dispararia uma mensagem MIDI. A vantagem deste sistema é que pode-se colocar qualquer material embaixo do plano de infravermelho para obter o efeito "rebote".

Em suma, os componentes usados para criar o protótipo deste sistema foram:
- Um aro de bicicleta de 26 polegadas, 26 LEDs emissores de infravermelho, 26 LEDs receptores de infravermelho, fios de cobre para a conexão do circuito, estanho para solda dos componentes, um transformador de corrente alternada para corrente contínua de 3 Volts e uma Placa Arduino Uno (2013) para a conexão do circuito ao computador

O circuito elétrico foi dividido em duas partes. A primeira consiste de LEDs emissores ligados em circuito paralelo. Assim, todos os polos positivos dos LEDs foram soldados juntos e conectados na ponta positiva do transformador, o qual foi alimentado por uma tomada 110 Volts. Os polos negativos também foram soldados juntos e conectados na ponta negativa do transformador.

4.2 Criação do novo hardware

O segundo circuito da Interface III foi formado de LEDs receptores, também, ligados em paralelo. Desta forma, seus polos positivos foram soldados juntos e conectados numa porta analógica da placa Arduino Uno, que por sua vez foi conectada no computador via conexão USB. Os polos negativos também foram soldados juntos e conectados na ponta negativa (*ground*) da placa Arduino Uno (Figura 9).

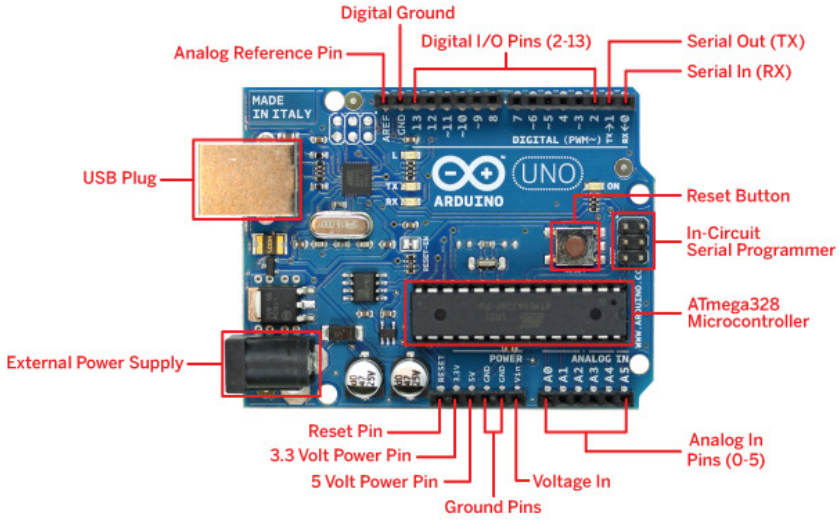

Figura 9. Arduino Uno Board, utilizado na Interface III

Para fazer o isolamento elétrico e para que não houvesse interferência do aro de alumínio no circuito, usamos uma borracha autoadesiva que cobriu todo o circuito, como mostra a figura 10a. O circuito final é também apresentado na Figura 10b.

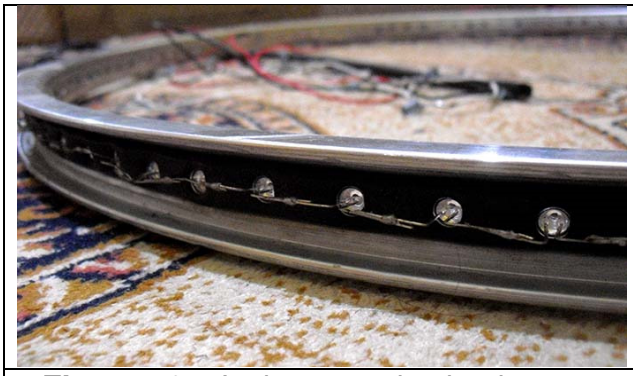

Figura 10a. Isolamento do circuito com borracha autoadesiva

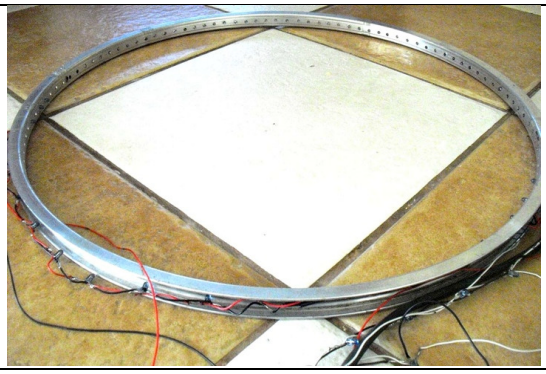

Figura 10b. Protótipo final sem a pele

4.3 Criação do Modelo Computacional

O algoritmo desenvolvido foi baseado na variação de tensão, de 0 a 5 volts, medida entre os polos positivo e negativo do circuito dos LEDs receptores. Enquanto a baqueta não ultrapassava o plano de infravermelho, o qual substituía a pele tradicional, a tensão era a máxima (5 volts), porém quando a baqueta interrompia o fluxo emissor-receptor, a tensão caia drasticamente, dependendo da distancia do centro do aro.

Dessa forma, o computador recebe a todo tempo a medida de 5 volts, e isso é interpretado como "pele estática". Quando a tensão cai, o computador interpreta que a pele foi atingida pela baqueta (o fluxo emissor-receptor foi interrompido) e então manda uma mensagem MIDI para o Kontakt. O Kontakt por sua vez toca o som da

percussão escolhido previamente pelo usuário.

Algo interessante deste algoritmo é a gama de possibilidades timbrísticas que se pode obter, já que cada posição de interrupção do fluxo emissor-receptor dentro do aro interrompe um número diferente de LEDs. Quando a baqueta atinge perfeitamente o centro do aro, ela interrompe todo o fluxo emissor-receptor, pois todos os feixes de infravermelho passam pelo centro do aro antes de atingirem seus receptores. Quanto mais a baqueta se distancia do centro do aro, ela vai interrompendo um numero menor dos 26 feixes de luz infravermelha emitidos, o que implica numa diminuição gradual da tensão elétrica. Isso pode ser mapeado e parametrizado a fim de se conseguir uma variedade timbrística do instrumento.

O algoritmo foi implementado com sucesso conseguindo extrair vários timbres diferentes ao tocar a baqueta em regiões diferentes do aro.

5 Análise das 3 Interfaces

As três interfaces virtuais de percussão desenvolvidas foram concluídas com sucesso, mostrando potencial para a criação deste tipo de instrumento musical. Cada uma delas tem seus pontos positivos e também seus pontos negativos, os quais estão detalhados na Tabela 1.

No entanto, surgiu um problema elétrico na Interface III que não foi possível resolver no tempo da pesquisa. Por algum motivo desconhecido, a tensão variava mesmo quando o sistema estava estático, talvez causado por um "fenômeno bobina" originado pela corrente elétrica no aro de alumínio ou talvez algo referente ao isolamento de todas as fontes de infravermelho do ambiente, no entanto, o problema se mostrou de difícil resolução. Por causa desse problema eletromagnético, as vezes o sistema emitia o som da percussão, mesmo não sendo tocado nem interrompido por qualquer barreira, o que inviabilizou este modelo.

Tabela 1. Análise e comparação das três interfaces

Características	Interface I	Interface II	Interface III
Custo aproximado	R$ 280,00	R$ 150,00	R$ 40,00
Montagem	Demorada	Média	Rápida
Resposta sonora aproximada	200 ms	60 ms	30 ms
Velocidade do algoritmo	lento	média	rápido
Variedade timbrística	nenhuma	nenhuma	26 possibilidades
Tipo de problema encontrado	alta latência	nenhum	eletromagnético

Analisando a Tabela 1 podemos ver que a Interface I teve o pior desempenho comparado com as outras interfaces por seu custo elevado e principalmente por não ter uma resposta em tempo real que satisfizesse os percussionistas. A Interface II tem um custo mediano e oferece um tempo de latência aceitável à prática de

instrumentos de percussão, o que viabiliza o seu uso em todas as áreas pretendidas. Na tentativa de criar uma pele para o instrumento de percussão virtual obtivemos a Interface III, a qual tem o menor custo e um excelente tempo de resposta, porém alguns problemas eletromagnéticos inviabilizaram o seu uso como instrumento virtual.

6 Conclusão

Este artigo apresentou a concepção, o desenvolvimento e a implementação de três interfaces gestuais de percussão, cujo objetivo inicial foi criar um equipamento de baixo custo para ser utilizado por estudantes e profissionais de música, como substituto de baixo custo para os instrumentos de percussão acústicos.

Como se pode ver na Tabela 1, a interface que melhor apresenta resultados compatíveis com um instrumento de percussão real e sem nenhum problema de implementação é a Interface II. Infelizmente, a Interface II não possui variação de timbre, o que é esteticamente muito importante para os percussionistas.

Outro aspecto vantajoso das interfaces virtuais é o seu uso em técnicas estendidas, performances remotas e instalações multimodais, formas de arte muito exploradas nos últimos anos.

Espera-se continuar o desenvolvimento de interfaces gestual de percussão em um possível trabalho futuros de pós-graduação. Dessa forma será possível obter as soluções dos problemas encontrados neste trabalho de iniciação científica e, assim, obter um produto ou interface que seja eficaz para o desenvolvimento de alunos e profissionais que utilizam instrumentos de percussão.

Agradecimentos

Agradecemos ao Conselho Nacional de Desenvolvimento Científico e Tecnológico - CNPq, pelo Programa Institucional de Bolsas de Iniciação Científica – PIBIC, o qual financiou o desenvolvimento desse projeto.

Referências Bibliográficas

ALCORN, E. **Windows Embedded Compact: Bluetooth Architecture Overview.**

2011. Disponível em: <http://www.scribd.com/doc/96687022/Bluetooth-Architecture-

Overview> Acesso em: mar. 2014.

ARDUINO. Arduino Uno. Arduino. Disponível em:

<http://arduino.cc/en/Main/ArduinoBoardUno >. Acesso em: 19 mar. 2013.

BALDÉ, F.; WAISVISZ, M. JunXion 4 Manual. Stein Foundation. Amsterdam, 2008,

p. 45. Disponível em: < http://www.steim.org/support/manuals/jXv4_Manual.pdf>.

Acesso em: 18 mar. 2014.

EAGLE PCB SOFTWARE. Versão 7. CadSoft Computer, 2014. Disponível em:

<http://www.cadsoftusa.com >. Acesso em: 22 jul. 2014.

HECKROTH, J. **Tutorial on MIDI and Music Synthesis.** 1995. 81 p. Disponível em: <http://www.idea2ic.com/Manuals/MIDI_Format.pdf>. Acesso em: jul. 2014.

HOLLAND, S.; WILKIE, K.; **MULHOLLAND**, P., SEAGO, A**. Music and Human: Computer Interaction.** London: Springer, 292 p. 2013. Disponível em: <http://www.springer.com/computer/hci/book/978-1-4471-2989-9>. Acesso em: mar. 2014.

KARDACH, J. **Bluetooth Architecture Overview.** Intel Technology Journal. Q2. Mobile Computing Group, Intel Corporation James. 13 p. 2000. Disponível em: < http://www.ieee802.org/11/Tutorial/90538S-WPAN-Bluetooth-Tutorial.pdf>Acesso em: mar. 2014.

KIEFER, C.; COLLINS, N.; FITZPATRICK, G. (2008, June). **HCI Methodology For Evaluating Musical Controllers.** In Proceedings of the 8th international conference on New interfaces for musical expression. Disponível em: < http://nime2008.casapaganini.org/documents/Proceedings/Papers/193.pdf >. Acesso em: abr. 2014.

KONTAKT. Versão 5. Native-instruments, 2011. Disponível em <http://www.native-instruments.com>. Acesso em: 01 de jan. 2014.

MALLOCH, J., WANDERLEY M. M. (2007, June). **The T-Stick: From Musical Interface to Musical Instrument.** In Proceedings of the 7th international conference on New interfaces for musical expression (pp. 66-70). ACM. 2007. Disponível em: < www.idmil.org/_media/wiki/nime07_malloch_tstick.pdf>. Acesso em: jan. 2014.

MAX. Versão 6. Cycling '74. 2014. Disponível em <http://cycling74.com/products/max/>. Acesso em: 19 de mar. 2014.

PUREDATA COMPUTER MUSIC SYSTEM. software livre, Versão 0.43.4. Disponível em: <http://puredata.info/downloads>. 2014. Acesso em: 20 jul. 2014.

VISUAL STUDIO. Versão 2012. Microsoft. Disponível em <http://www.microsoft.com/visualstudio>. Acesso em: 04 de fev. 2013.

XENIA, P.; ERIKA, D.; STEWART, A. **The Digital Orchestra Project: Digital Musical Instruments and Performance Practice.** 2012.

REVISTA BRASILEIRA DE INICIAÇÃO CIENTÍFICA - ISSN 2359-232X
VOL. 2, Nº 01, 2015