Relatório Final de Trabalho de Graduação

Um *framework* de código aberto para jogos de *Air Guitar*

Aluno	**Orientadora**
Lucas Silva Figueiredo	Veronica Teichrieb
lsf@cin.ufpe.br	**vt@cin.ufpe.br**

Recife, Dezembro de 2009

Resumo

Este relatório descreve um *framework* de código aberto voltado para o desenvolvimento de jogos musicais que fazem uso de interação com gestos; mais especificamente, jogos baseados na prática de *Air Guitar* (que é o ato de fingir que se está tocando guitarra). A idéia central é tornar *Air Guitar* uma prática sonoramente ativa. Desta forma, ao dar uma palhetada no ar, tocando as cordas de uma guitarra imaginária, o usuário escutará um som correspondente àquele ato, fazendo-o acreditar que realmente está tocando um instrumento, o que de certo modo é verdade. Além do detalhamento do *framework*, este relatório descreve um estudo de caso usado para validar a plataforma, o qual foi aplicado a um grupo de usuários de distintas idades. Nos testes, todos tiveram que executar corretamente um trecho de música simples e todos eles atingiram tal objetivo em um curto período de tempo, demonstrando a facilidade de uso da ferramenta. Ainda mais importante que isso, todos os indivíduos que participaram dos testes relataram ter gostado da experiência.

Abstract

This document describes an open-source framework to develop music games using gesture interaction; more specifically, games based on Air Guitar practice (that is the act of simulating a guitar being played). The main idea is to make Air Guitar a sonorous active practice. This way, when the user mimics that he is using the guitar pick pulling the strings of an imaginary guitar he will hear a sound that is correspondent to that act, making him believe that he is really playing an instrument, which in a certain way is true. Besides the detailed framework description, this document describes a case study used to validate the platform, which was applied to a group of users of different ages. In the tests, they had to correctly execute a part of a simple song and all of them have achieved this objective in a small period of time, demonstrating the tool's easiness of use. More important than that, all tested subjects reported that they enjoyed the experience.

Agradecimentos

Lembro de ter lido uma vez, em outro trabalho de graduação, que esta era a parte mais difícil a ser escrita, somente hoje vim entender isso, e confesso que concordo. Aqui está feito o impossível, registrar gratidão a todos aqueles que ajustaram o meu caminho durante estes cinco anos de graduando. Obrigado

a Edna, Geber, Ruy e Silvio, cada um da sua forma, por mostrar um pouco de magia na palavra professor.

a Felipe (Soso), Henrique (Lhama), Diego (Chucky), Lucas (Bagaça), Rodrigo (Reaf), Jerônimo (Jera), Antônio (Espanta), Pablo (Pcp), Heitor e todos aqueles que transformaram horas de estudo e noites viradas nos laboratórios em cenas que lembro hoje com nostalgia (não que eu as queira de volta, mas dá gosto de lembrar).

a Artur (Tuzin), que sempre um passo a frente, tantas vezes indicou as direções.

ao GRVM, por me motivar, me mostrar a importância de se trabalhar em grupo, sinto orgulho de fazer parte disto. E em particular a João Marcelo (Joma), João Paulo (Jonga), Judith (Professora) e Veronica (Vt), por me orientarem, cada um à sua maneira, me mostrando onde estão as pedras desta jornada.

a Alan, Diego, Lorena, Rodrigo, Alice, Nadhine, Rômulo, Gabriela, Juliana, Alison (Berg), Heitor (Piauí), Renato (KaraVea), Guilherme (Trop), Eriton (Tricolor), Diego (Undead), Paulo (Pexe), Chico, Manel, Caio, Wil, Ciro, Mari e muitos outros por momentos da mais pura diversão, pois sem esta não se sustenta a alegria e tampouco a vontade.

a Java e Ralph, pela amizade, por mostrarem meus erros e muitas vezes, até sem querer, me ensinarem sobre coisas que achei que já sabia.

a Ela, pois não bastasse todo o resto, ela reaviva e dá forças aos meus sonhos, e sem eles nada faria sentido.

a minha família, Caio, Aninha, Clara, Vera, Lídia, Céu, Figueiredo, Socorro, Chico, Ana, Val e tantos outros. Cada um sabe a importância dos seus pilares e os meus, por ventura, são bem fortes.

a Sônia e Isa, que cuidam tão bem destes pilares.

ao meu pai, por não me deixar esquecer do dever de mudar este mundo tão injusto em que vivemos. Confesso que ainda não encontrei o caminho, mas minha busca está só começando.

Índice

1. Introdução

Ao longo dos anos a indústria de jogos vem incorporando dispositivos periféricos aos seus produtos, com os objetivos principais de aprimorar a jogabilidade e aumentar a imersão nos mesmos. Esses dispositivos demonstraram ter a capacidade de tornar mais estreita a interação entre jogo e jogador, atingindo inclusive grupos de usuários que antes tinham pouca familiaridade com o mundo dos jogos eletrônicos. Um exemplo que ilustra bem esse caso são os tapetes de dança [Hoysniemi 2006], que se popularizaram consideravelmente há poucos anos atrás, atraindo pessoas que gostavam de dançar, mas ainda não tinham encontrado um jogo que explorasse essa habilidade (Figura 1).

Figura 1. Crianças usando um tapete de dança como dispositivo de entrada para um jogo.

Pode-se considerar que esta evolução dos dispositivos de interação tem como um dos seus focos os jogos musicais. Jogos que exploram intensamente o componente sonoro, fazendo com que o mesmo influa diretamente na jogabilidade, aos poucos tiveram seus tradicionais *joysticks* substituídos por controladores com formatos mais próximos da realidade de cada jogo. Periféricos em formatos de guitarra e bateria (Figura 2) se tornaram acessórios comuns para jogadores assíduos de Rock Band [Harmonix *et al*. 2007] e Guitar Hero [RedOctane 2005] (séries bem sucedidas de jogos musicais). Outro exemplo que explora esta metáfora do instrumento como controle se encontra em Frets on Fire [Voodoo 2006] (jogo musical de código aberto para PC), em que o jogador é levado a usar o teclado de uma forma diferente, segurando-o com as duas mãos, simulando o ato de usar uma guitarra.

De fato, muitos trabalhos contribuíram para formalizar o *framework* de *Air Guitar* proposto neste Trabalho de Graduação. Dentre estes, os jogos musicais tiveram sua parcela representada na motivação deste trabalho. Devido à crescente popularidade destes jogos nos últimos anos, eles têm conquistado espaço no atual mercado de entretenimento [Wixon, 2007]. Adagio[Games 2007], GuitarFreaks [Konami *et al*. 1998], FreQuency [Harmonix 2001] e DJHero [FreeStyleGames 2009] são alguns jogos

que podem ser citados, variando desde simples jogos implementados em *flash* até os mais sofisticados feitos sob-medida para a última geração de *consoles*.

Figura 2. Dispositivos de entrada para jogos musicais. Na esquerda, *joysticks* baseados nas guitarras Guibson SG, e na direita, controle que simula uma bateria simplificada.

Na mesma linha de evolução, é possível perceber o surgimento de uma tendência do uso de rastreadores de movimento para aprimorar a interatividade dos jogos. O Nintendo Wii [Nintendo 2006], *console* de sucesso na atualidade, foi um dos pioneiros na popularização da idéia de rastrear movimentos com seu *joystick* que possui acelerômetros acoplados. Reforçando essa tendência, os concorrentes da Nintendo também anunciaram alternativas semelhantes. A Sony, por exemplo, disponibilizará no comércio um dispositivo semelhante para o seu *console*, o Playstation 3 [Engadget 2009]. Para o Xbox 360 surgiu a pouco a promessa da Microsoft sobre o aclamado Project Natal [Microsoft 2009]. Como anunciado na E3 2009 (*Electronic Entertainment Expo* 2009), a ferramenta, além de permitir reconhecimentos de face e de voz, será capaz de rastrear com precisão uma série de movimentos corporais incluindo gestos com as mãos e expressões faciais.

Todas essas mudanças endossam a teoria de que aos poucos a interação via botões dará espaço para outras tecnologias, que exploram movimentos mais naturais ao ser humano. Diante deste contexto, surge o pano de fundo deste trabalho: usar o conceito de interação natural aplicado aos jogos musicais. Desta forma, a idéia de explorar a prática de *Air Guitar* [Turoque 2006] ganha forças, primeiramente por ser uma modalidade intrinsecamente ligada à música e, em seguida, por fazer uso de movimentos culturalmente naturais; muitas vezes uma pessoa faz gestos próprios de *Air Guitar* quase que involuntariamente, em um *show* de *rock*, por exemplo.

É importante ressaltar que *Air Guitar* é uma atividade consolidada, inclusive com disputas em campeonatos por todo o mundo (o mais antigo está na sua 14ª edição). Neste trabalho é feita uma análise da performance dos praticantes de *Air Guitar* no sentido de averiguar quais movimentos poderiam ser incorporados ao *framework* em questão. Refinando um pouco esta idéia, a proposta deste Trabalho de Graduação tem como foco criar um *framework* para jogos de *Air Guitar*, que seja capaz de reproduzir som de forma coerente com os movimentos do usuário. Uma das principais

motivações para o trabalho é criar uma plataforma que permita que leigos musicais possam tocar um instrumento com facilidade e de forma intuitiva. Além disso, a plataforma é de código aberto justamente para abarcar essa tendência da evolução das formas de interação nos jogos musicais, compartilhando conhecimento com a comunidade científica da área, visto que, apesar de existirem trabalhos semelhantes, não há material disponível para ser usado como ponto de partida. Os principais objetivos práticos deste trabalho são:

- A construção de um *framework* de código aberto bem organizado e modularizado. Desta forma, o usuário da ferramenta pode alterá-la e atualizá-la com facilidade. A divisão por módulos permite a substituição dos mesmos para alterar o processamento de etapas específicas, garantindo que as outras fases continuarão funcionando adequadamente. Além disso, torna possível a criação de soluções que não necessariamente fazem uso de todas as funcionalidades do *framework*.
- A confecção de uma aplicação exemplo codificada a partir do *framework*. Esta aplicação deve fazer uso de todos os módulos da ferramenta, explorando de forma objetiva boa parte dos recursos oferecidos para demonstrar o potencial deste protótipo.
- A realização de um estudo de caso para validar todo o trabalho. Este estudo de caso deve levar em conta indivíduos de diferentes idades e com graus variados de experiência com instrumentos musicais.
- Disponibilizar o conteúdo produzido na *web*, possibilitando a colaboração de todos os interessados e divulgando a ferramenta principalmente para o uso acadêmico, no âmbito da pesquisa.

Além destes pontos, a ferramenta em questão deve atingir alguns requerimentos básicos para a obtenção de aplicações robustas e com boa usabilidade. Primeiramente, a interação deve acontecer em tempo real. Dado que o usuário está de alguma forma tocando música, a restrição de tempo é fundamental, pois atrasos ocasionariam respostas incoerentes, tanto sonoras quanto visuais. Em seguida, o *framework* deve disponibilizar métodos que sejam capazes de reconhecer quando um usuário se movimenta com a intenção de reproduzir som, como se estivesse tocando as cordas de uma guitarra imaginária. Por fim, a resposta sonora deve ser realística e coerente com os movimentos do usuário, fazendo-o acreditar que realmente está tocando um instrumento, o que de certa forma é verdade, apesar de ser virtual.

O presente documento contém a descrição completa deste Trabalho de Graduação e está organizado da seguinte forma. No próximo capítulo é apresentado o contexto no qual está inserido este trabalho, listando os principais trabalhos relacionados. O capítulo 3 detalha o *framework*, enquanto o capítulo 4 a aplicação exemplo e o estudo de caso. Finalmente, no capítulo 5, são expostas as conclusões significativas do trabalho e alguns planos para trabalhos futuros, no intuito de aperfeiçoar a ferramenta proposta.

2. Contexto

Antes de descrever o *framework* desenvolvido é necessário fazer um levantamento do contexto no qual o trabalho está inserido. Neste capítulo se encontra uma análise dos principais tópicos que rodeiam este Trabalho de Graduação, que são: interação natural, rastreamento de modelos, rastreamento de mãos e, por fim, uma revisão dos trabalhos similares.

2.1. Interação natural

Uma breve análise da evolução da interação humano-computador nos últimos anos expõe uma tendência nesta área que vem sendo chamada de interação natural [Valli 2005]. Com o passar das décadas, as formas de interação vêm evoluindo sempre no sentido de tornar a relação usuário-máquina mais eficiente e confortável. Invenções popularmente conhecidas, como o controle remoto e o *mouse*, há muito tempo surgiram para substituir métodos antigos de interação; são raros os casos de usuários que preferem usar somente o teclado para interagir com um computador, ou ao contrário de usar o controle remoto, sempre que precisarem mudar de canal, caminhar até a televisão.

No entanto, mesmo atingindo um alto grau de aceitação, sendo usados praticamente no mundo inteiro, estes dispositivos não têm suas vagas garantidas. Existe uma tendência de, para cada aplicação, se encontrar uma interface específica aprimorada e da mesma forma que um dia estas opções de interação (*mouse* e controle remoto) superaram outras, em um momento posterior elas podem ser superadas. Todavia, para que tal inversão aconteça faz-se necessário o surgimento de novas interfaces, que se destaquem em relação às anteriores em pelo menos alguns conceitos como conforto, eficácia e intuitividade.

É difícil prever se uma nova interface se estabelecerá de forma consolidada, ou determinar se uma tecnologia vai ser bem aceita pelos usuários. No entanto, se essa interface preza por ser mais intuitiva que suas antecessoras, então ela já está alguns pontos na frente. Entende-se como intuitividade o grau de simplicidade e facilidade com que a interface pode ser utilizada [Bærentsen 2001].

Dentro deste conceito é possível perceber uma tendência nas formas de interação das novas tecnologias na atualidade. Aos poucos, a idéia de apertar botões vem dando espaço para outras abordagens até então pouco exploradas. Um exemplo que ilustra bem essa diminuição do uso dos botões é uma das métricas usadas para ter-se noção da usabilidade de determinada interface; quanto menor a quantidade de cliques ou botões que o usuário deve pressionar para chegar ao seu objetivo, melhor [Bao *et al*. 2006]. Outro exemplo são os serviços de teleatendimento, que já fazem uso de reconhecimento de padrões para interpretar comandos de voz do usuário, substituindo o apertar de uma tecla pela pronúncia de uma palavra ou frase; esta troca não é necessariamente mais veloz, mas com certeza mais intuitiva para o usuário.

Pode-se dizer que essas novas abordagens tendem a explorar os movimentos naturais do corpo humano. Por exemplo, as mesas digitalizadoras (*drawing tablets* ou *graphics*

tablets) fazem uso da metáfora 'papel e lápis', propondo ao usuário uma maneira de interação bem similar a atividades culturalmente consolidadas que são os atos de escrever e desenhar. Uma idéia semelhante está presente nas interfaces multitoque, que sugerem a substituição do mouse por movimentos com os dedos (Figura 3), ampliando o campo das possibilidades de interação. Neste caso, o que antes se resumia a cliques e uma coordenada 2D na tela (com o uso do *mouse*), agora se expande para a interpretação de diversos gestos: desde os mais simples, como apontar um caminho na tela ou arrastar um item no espaço planar, até os mais complexos, usando combinações de movimentos com vários dedos, levando em conta ainda níveis diferentes de pressão de cada dedo sobre a interface multitoque [Miller 2009].

Figura 3. Interface multitoque proposta como alternativa para a combinação teclado-*mouse* [Miller 2009].

Existe uma constante em todos esses aperfeiçoamentos de interfaces citados acima; eles migram para formas de interação mais próximas ao ser humano, ou seja, mais intuitivas. O foco está mudando de forma que ao invés do usuário interpretar o dispositivo para poder interagir com determinado sistema, o dispositivo passa a interpretar o usuário, fazendo com que seus movimentos naturais façam sentido para a aplicação. Desta forma, o conceito de interação natural ganha sentido.

Neste mesmo contexto surgem dispositivos capazes de detectar movimentos, por exemplo através de sensores como os acelerômetros. Com esses aparelhos dá-se início ao rastreamento de movimentos corporais, sendo exemplos populares do uso dessas tecnologias o iPhone [Apple 2007] e o Nintendo Wii (que além de acelerômetros faz uso de um sensor infra-vermelho) [Nintendo 2006]. Com o Wiimote (*joystick* do Nintendo Wii) é possível obter informações sobre uma série de movimentos do braço e do pulso de uma pessoa, que podem ser usadas para melhorar a experiência do usuário em seus jogos, como mostrado na Figura 4. Seguindo a mesma linha, neste ano surgiu a promessa do Project Natal da Microsoft [Microsoft 2009], no qual um sensor específico é capaz de detectar uma série de movimentos corporais, sem que este esteja usando quaisquer adereços ligados ao *console*.

Dentro deste fluxo de mudanças, tecnologias que são capazes de rastrear o corpo do usuário (ou parte dele) ganham visibilidade. Nos dias de hoje já é possível encontrar uma série de técnicas que são capazes de retornar informações sobre a posição e orientação do corpo de determinado usuário. Dentro desta vasta gama de técnicas, é

possível restringir um grupo particular: técnicas de rastreamento que dispensam *hardwares* específicos (diferentemente do Nintendo Wii e do Project Natal), ou seja, métodos de rastreamento que possam ser executados com o auxílio apenas de tecnologias acessíveis e genéricas (câmeras *web*, por exemplo). Eles podem ser chamados de técnicas de rastreamento corporal monocular, pois usam como dispositivos genéricos de auxílio para o rastreamento câmeras comuns, com uma só lente, que são periféricos consideravelmente populares na atualidade.

Figura 4. Modalidade de boxe do jogo Wii Sports do *console* Nintendo Wii. Os movimentos dos braços do jogador são rastreados, interpretados e usados como entrada para controlar o personagem boxeador do jogo.

2.2. Rastreamento de modelos

Considerando um escopo maior (incluindo técnicas de caráter não-corporal), pode-se perceber um notável desenvolvimento da área de rastreamento monocular na última década. Os métodos que rastreiam marcadores específicos na cena (Figura 5) [Kato *et al*. 1999], exigindo objetos intrusivos, que não são encontrados naturalmente no nosso mundo, aos poucos vêm dando lugar a técnicas capazes de rastrear objetos contidos naturalmente na cena (representados por modelos 3D), por exemplo, utilizando informações de arestas ou de texturas (Figura 6) [Lima *et al*. 2009] [Lima *et al.* 2009]. Em outras palavras, objetos mais complexos do que os marcadores e contextualizados no mundo real, passam para a categoria de rastreáveis. Diante destes progressos, o conceito de rastreamento de objetos passa também a ser aplicado ao rastreamento corporal, de forma geral considerando o corpo humano como um modelo 3D a ser rastreado. No entanto, existe uma série de complicações neste tipo de abordagem que exigem cuidados específicos, como por exemplo, a filtragem de cor de pele, a existência de articulações no modelo e como consequência disto a ocorrência de auto-oclusão, *etc*. Logo, não se pode dizer que o rastreamento corporal é uma simples restrição ou combinação das técnicas de rastreamento monocular baseadas em

modelos; no entanto, é certo que existe uma sinergia proveniente desta proximidade e que isto contribui para o desenvolvimento de ambas.

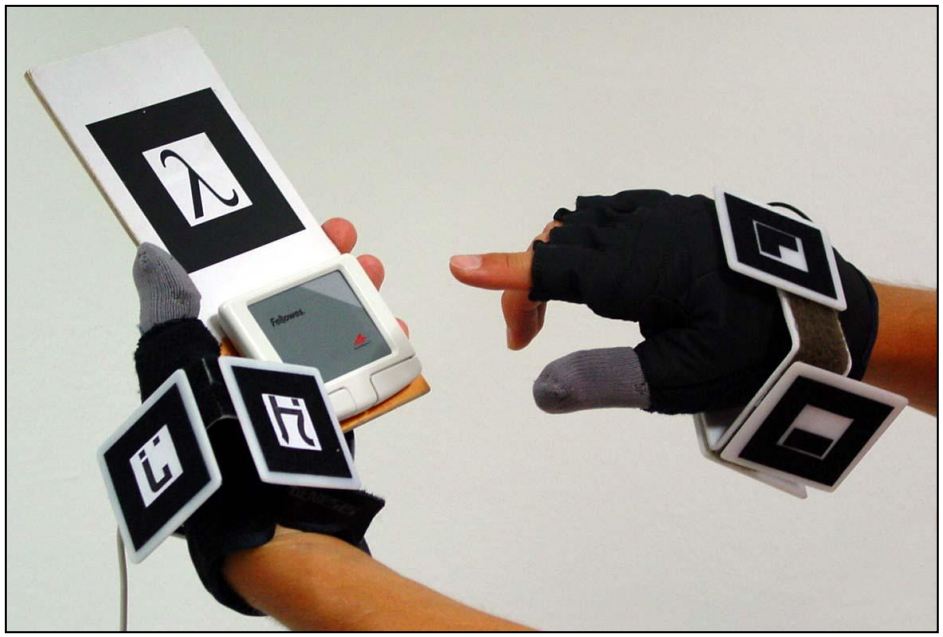

Figura 5. Exemplo do uso de marcadores em uma aplicação para rastrear as mãos do usuário e um dispositivo móvel de interesse.

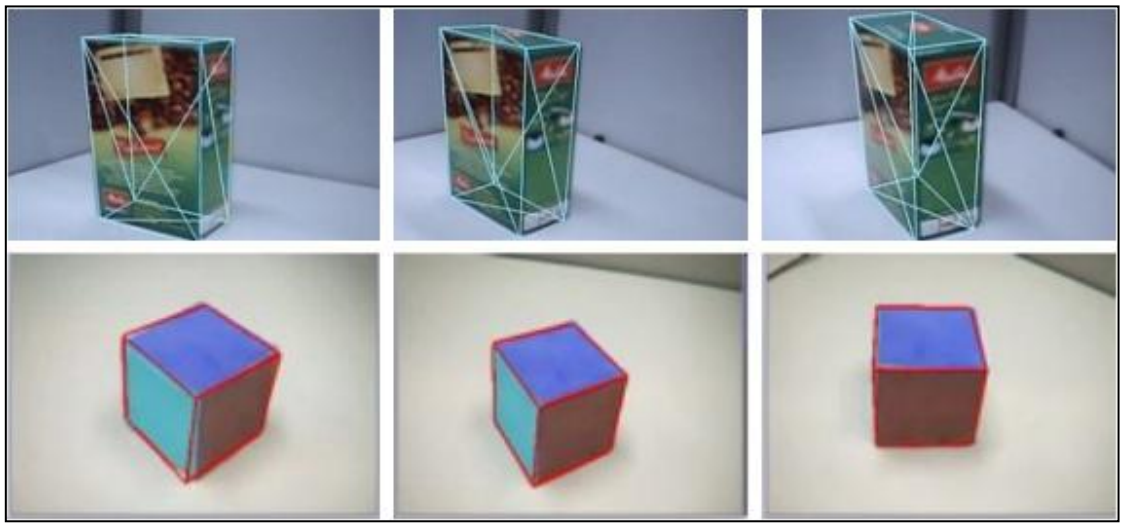

Figura 6. Exemplo do uso de técnicas que rastreiam objetos reais a partir de informações de textura (acima) e de arestas (abaixo). Um *wireframe* é desenhado sobre o objeto para demonstrar que este foi rastreado com sucesso.

Diante deste contexto, surgem ferramentas que possibilitam a implementação de métodos de rastreamento corporal monocular, ainda que de forma geral incipientes. Esse grupo específico de técnicas se torna relevante a partir do momento que abre um leque de possibilidades de interação para uma quantidade consideravelmente maior de aplicações. Em sua maioria essas técnicas se agrupam em três formas de rastreamento: rastreamento do corpo de forma simplificada, da face (e/ou cabeça) e, por fim, das mãos (Figura 7). O fato da maioria das técnicas se enquadrarem nesses três grupos é facilmente compreendido quando se pensa nas partes do corpo mais comunicativas de um ser humano. No entanto, a grande maioria destas técnicas,

apesar de se encontrarem no estado da arte da área, dificilmente atingem satisfatoriamente requisitos primordiais para a área de interação como robustez, precisão e execução em tempo real.

2.3. Rastreamento corporal

Em [Stenger *et al*. 2004] é mostrada uma técnica de rastreamento de modelos aplicada especificamente para a detecção de mãos que exemplifica bem o problema mencionado anteriormente. Para que esta técnica atingisse tal objetivo, foi necessário realizar uma série de adaptações voltadas especificamente para o rastreamento de mãos como, por exemplo, a filtragem de cor de pele e a criação de uma representação da mão através de componentes conexos (palma, falanges proximais, falanges médias, etc.). Em [Thayananthan *et al*. 2008] é possível ver uma abordagem similar aplicada ao rastreamento do corpo inteiro de forma simplificada. Apesar de em [Stenger *et al*. 2004, Thayananthan *et al*. 2008] ter sido demonstrada a possibilidade da aplicação de técnicas de rastreamento 3D do corpo humano, é relevante perceber que esses resultados não aparecem sem ônus. Nestes casos, problemas como excesso do uso de memória, frequentes tremidas dos modelos rastreados ou alto tempo de execução sinalizam dificuldades na aplicabilidade destas técnicas.

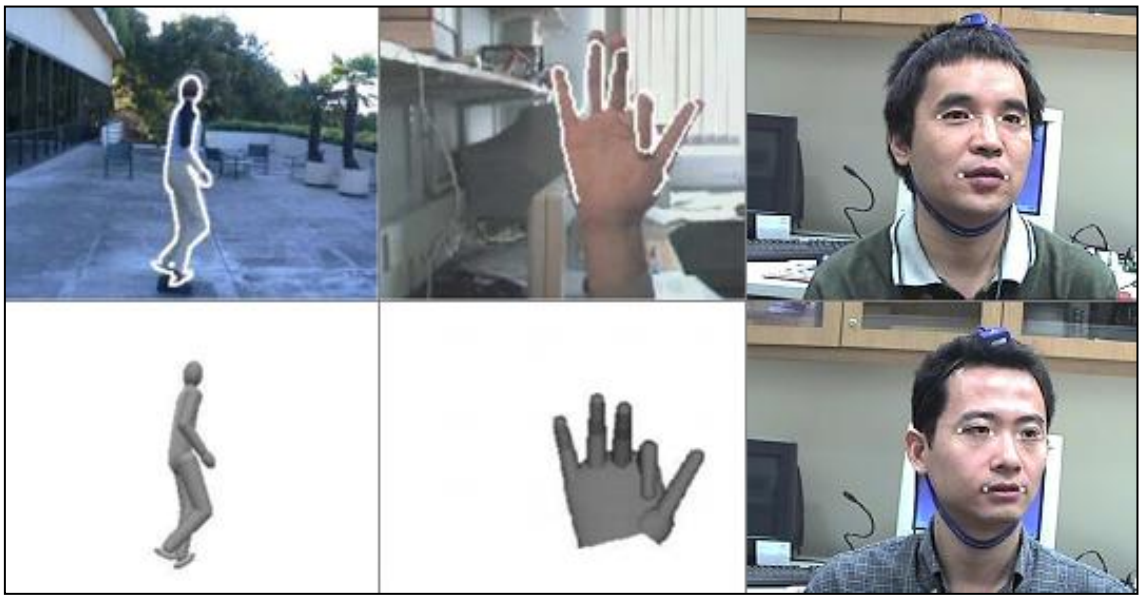

Figura 7. Rastreamento monocular do corpo humano [Thayananthan *et al*. 2008] (esquerda), especificamente de mãos [Stenger 2004] (centro) e de faces (direita) [Wang *et al*. 2007].

Também existe a possibilidade de aplicar técnicas de rastreamento de modelos 2D ao rastreamento corporal. Na Figura 8 podem ser vistos alguns resultados deste tipo de rastreamento; na imagem, é possível perceber que o corpo humano é considerado como um conjunto conexo de modelos 2D [Zhang *et al.* 2008] (parte superior da figura). Em [Para *et al.* 2008] são apresentados resultados que demonstram uma técnica robusta a auto-oclusão (parte inferior da Figura 8), característica relevante para o rastreamento corporal levando em conta que em aplicações de interação é praticamente impossível garantir que um membro (do corpo do usuário) não oblitere a visão dos demais. Usualmente, as técnicas que abordam duas dimensões possuem métodos de rastreamento mais simples e em geral conseguem bons resultados em

relação à velocidade e precisão. É importante ressaltar que estes são benefícios interessantes em contraste com a deficiência de não ser possível conseguir informações de profundidade com precisão (somente possível com o uso da terceira dimensão), e que dependendo da aplicação, estas técnicas podem ser mais adequadas do que aquelas que fazem uso de modelos 3D.

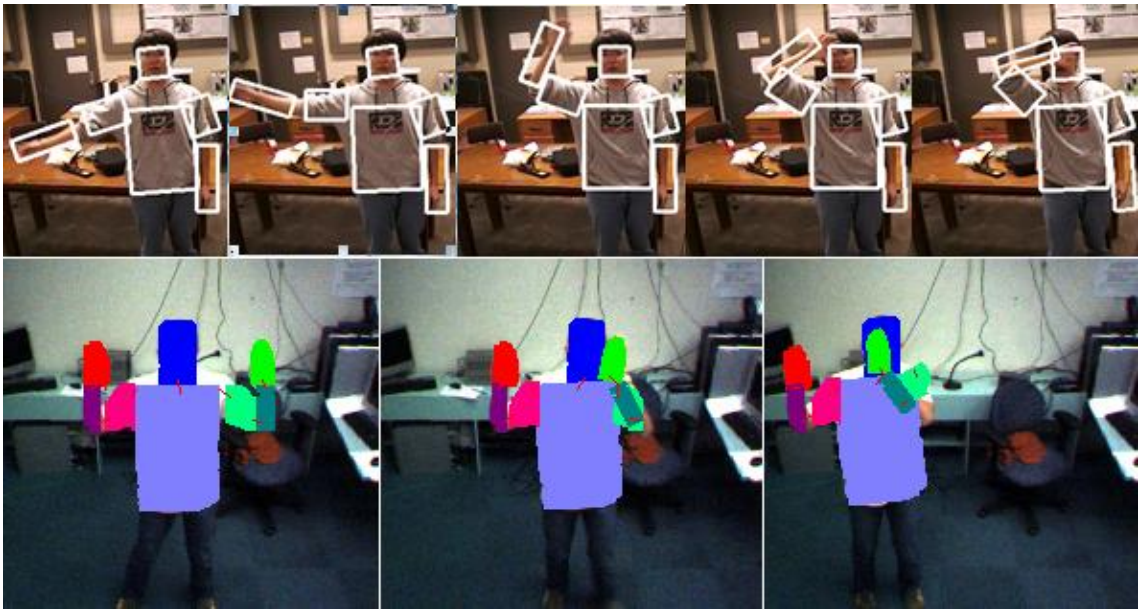

Figura 8. Resultados de [Zhang *et al.* 2008] acima e [Para *et al.* 2008] abaixo; técnicas de rastreamento 2D aplicadas ao rastreamento do corpo humano.

2.4. Rastreamento de mãos

Um dos focos do presente trabalho é fazer uso da interação corporal nos jogos musicais, mais especificamente na prática de *Air Guitar*. Dentre os tipos de técnicas de rastreamento foi escolhida a de rastreamento de mãos. Essa opção se deu basicamente por caráter eliminatório; o rastreamento de face não teria muita utilidade na prática de *Air Guitar*, já a idéia de rastrear o corpo por completo é uma opção válida, no entanto tende a ser consideravelmente mais complexa e como o rastreamento das mãos por si só é suficiente para a simulação de uma guitarra virtual, ele foi escolhido como foco do trabalho.

Existe uma vasta gama de trabalhos sobre rastreamento de mãos, e na Figura 9 podemos ver alguns exemplos de resultados. Entretanto, como o objetivo deste trabalho é rastrear mãos que se movem com uma velocidade considerável, alguns problemas foram encontrados. Primeiramente, boa parte das técnicas no estado da arte não possui uma taxa de atualização que possa ser considerada de tempo real, o que prejudica diretamente a jogabilidade, causando atrasos na resposta para o usuário. Outro problema é a robustez destas técnicas em relação a movimentos rápidos, que além de mudar o posicionamento das mãos na imagem capturada pela câmera, comumente causa um efeito borrado, que deixa a imagem menos nítida.

Para suprir esses problemas é necessário recorrer a métodos que dispensam detalhes, como detecção de arestas, por exemplo (por conta do efeito borrado). Por esse motivo, passou-se a investigar técnicas que abordavam outras abordagens, capazes de

superar problemas como os citados acima em detrimento de outros tópicos que não prejudicassem de forma relevante a experiência do usuário. Neste sentido, foram encontradas técnicas que fazem uso de luvas de cor de destaque no cenário para rastrear as mãos do jogador, ou seja, o usuário veste luvas e através de métodos de rastreamento de cores aplicados na imagem capturada se torna possível conhecer a posição das mãos rastreadas. As grandes vantagens desse tipo de rastreamento são a velocidade da taxa de atualização e a robustez a movimentos rápidos. Por outro lado, existem alguns pontos negativos, como o fato de o usuário ter que vestir luvas, pouca robustez a mudanças de iluminação, que não comprometem necessariamente a jogabilidade, mas merecem ser citados. Em [Wang *et al*. 2009] e [Dorner 1994] podemos ver exemplos do uso de luvas coloridas para rastrear mãos (Figura 10). Especificamente, para a implementação do *framework* proposto neste trabalho foi feita a opção por uma técnica simplificada, de autoria própria, que também faz uso de luvas de cor de destaque no cenário e que atende de forma sucinta aos requisitos do trabalho proposto, evitando a complexidade, por hora desnecessária, presente nos trabalhos citados anteriormente.

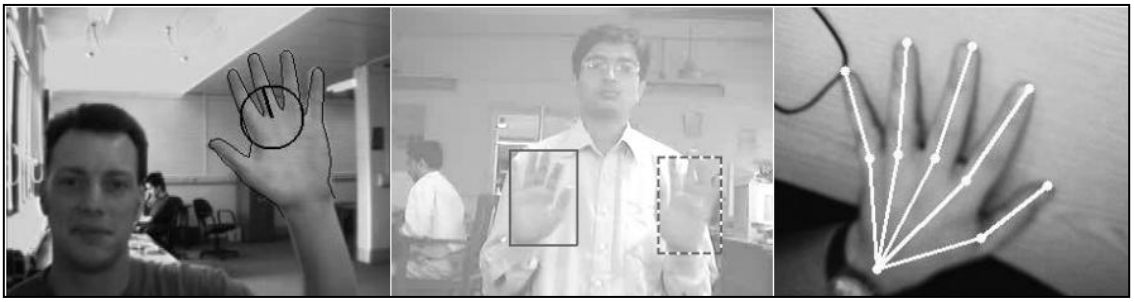

Figura 9. Resultados de três técnicas de rastreamento de mãos distintas descritas em [Heap *et al*. 1995], [Mammen *et al*. 2004] e [Stefanov *et al*. 2005], respectivamente.

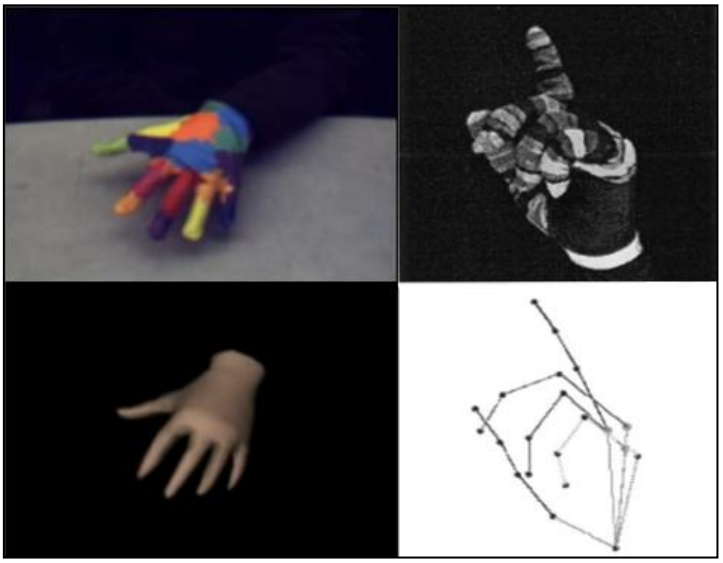

Figura 10. Técnicas que fazem uso de luvas coloridas para rastrear mãos. Resultados de [Wang *et al*. 2009] na esquerda e [Dorner 1994] na direita.

2.5. Trabalhos similares

Por fim, é preciso citar alguns trabalhos de identidade bem similar ao proposto. Em [Camurri *et al*. 1999], foi proposto um sistema que reconhece algumas intenções do

usuário que está executando uma performance de dança e usa estas informações como entrada para algumas tarefas, por exemplo a geração de sons. Ainda mais próximo da idéia do *framework*, existe o trabalho chamado de Virtual Air Guitar [Mäki *et al*. 2005] (Figura 11) que de fato, é um sistema com praticamente as mesmas funcionalidades disponíveis no *framework*, oferecendo resultados semelhantes através de métodos também similares. No entanto, o projeto Virtual Air Guitar foi concebido como uma solução proprietária, e não existe nenhum tipo de código relacionado ao trabalho, tampouco um aplicativo de demonstração disponível para a comunidade de desenvolvedores e usuários. O mesmo ocorre com o projeto Virtual Slide Guitar [Pakarinen *et al*. 2008], trabalho similar que difere basicamente pelo método de captura, usando sensores infra-vermelho. Neste caso também não existe código-fonte disponível nem aplicações de teste acessíveis.

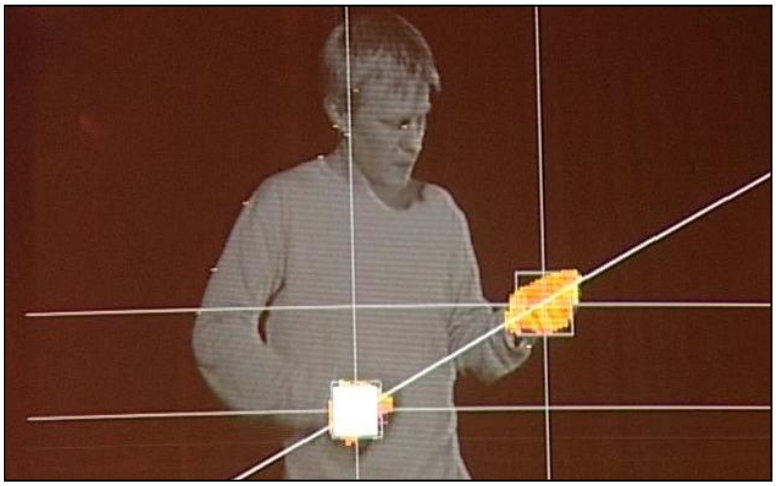

Figura 11. Interface visual do projeto Virtual Air Guitar [Mäki *et al*. 2005].

Desta forma, uma das maiores contribuições do *framework* proposto neste trabalho é o fato dele ser de código aberto. A ferramenta resultante faz uso de um simples, porém robusto algoritmo de rastreamento de cores (o que favorece o desempenho obtido) e é capaz de interpretar movimentos como se uma guitarra virtual estivesse sendo tocada. Além do mais, uma aplicação demonstrativa é provida para os desenvolvedores, podendo ser modificada e usada como ponto de partida para cenários mais complexos.

3. O *framework*

O *framework* proposto consiste em uma ferramenta que habilita o usuário a usar suas mãos como um dispositivo de entrada para jogos e outras aplicações de entretenimento [Lucas *et al*. 2009]. Mais especificamente, este *framework* é focado em transformar a performance de *Air Guitar* em uma forma de interação. A idéia é fornecer aos desenvolvedores de jogos uma ferramenta base para construir a interação com uma "guitarra imaginária", de forma que seja possível ajustar o uso do *framework* de acordo com as necessidades do projeto a ser desenvolvido.

A ferramenta é modularizada no intuito de ser facilmente usada, alterada e atualizada. Um esboço de sua arquitetura é apresentado na Figura 12, ilustrando um *pipeline* que contém as principais etapas envolvidas no funcionamento de uma aplicação genérica de *Air Guitar*: aquisição da imagem de entrada, processamento da imagem, simulação de uma guitarra virtual, geração de som e renderização. É importante ressaltar que este *framework* faz uso de outras bibliotecas que dão suporte para cada fase do processamento. A DSVideoLib [Pintaric 2005] foi usada para capturar a imagem proveniente da câmera *web*, o FMOD Ex [Technologies 2005] foi responsável por gerar os sons de saída e, por fim, todos os resultados visuais foram renderizados com OpenGL [Graphics 1992]. Todo o código foi escrito em inglês usando a linguagem C++. A escolha por este idioma se deu principalmente devido à sua abrangência; levando em conta que a proposta do trabalho é ser de código aberto é interessante que o seu código-fonte possa ser lido por um público amplo. Cada etapa do *pipeline* está mais bem detalhada nas próximas subseções.

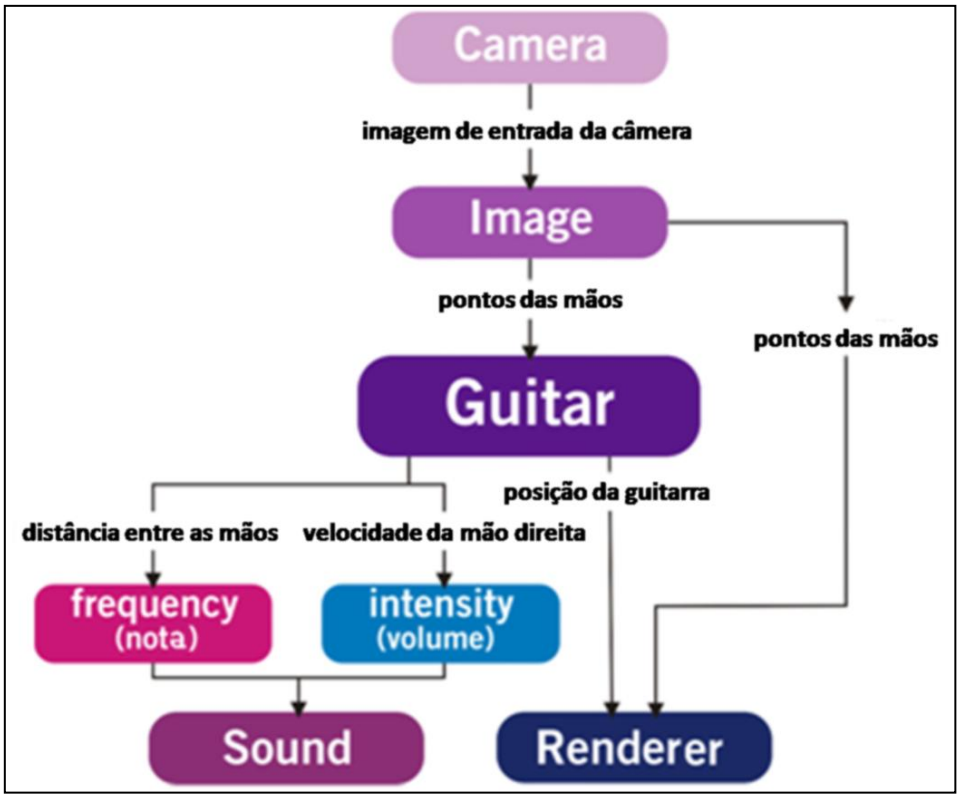

Figura 12. *Pipeline* da arquitetura do *framework*.

3.1. Camera

Rastrear as mãos do usuário é uma tarefa que pode ser alcançada usando-se muitas técnicas distintas. *Haptic gloves*, *wiimotes* [Nitendo 2006] ou rastreamento de mãos baseado em modelos são alternativas conhecidas para realizar tal tarefa. O *framework* proposto adota uma abordagem de rastreamento de cores, pois esta demonstrou ser a melhor em custo-benefício para adquirir conhecimento sobre as posições das mãos do usuário em tempo real. Esse tipo de rastreamento requer, além do PC propriamente dito, uma câmera *web* comum e um par de luvas com uma cor de contraste em relação ao ambiente, que será vestido pelo usuário.

Este método de rastreamento apresenta diversas vantagens que serão listadas a seguir. Primeiramente, o custo do equipamento requerido é consideravelmente menor quando comparado, por exemplo, ao preço das *haptic gloves* ou até dos *wiimotes*. A carga de processamento demandada pelo algoritmo é baixa, o que possibilita sua execução em tempo real. Ele também é robusto a oclusões parciais, movimentos rápidos e a efeitos de borrado (*blur*), além de ser capaz de recuperar-se imediatamente após falhas de rastreamento (geralmente ocasionadas por oclusões totais).

O módulo Camera contém toda implementação referente à configuração da câmera *web* e à captura de imagem. Ele possui funções que proporcionam acesso a diferentes informações da câmera, como por exemplo a aquisição dos dados do último *frame* capturado (para o uso em futuras renderizações). Este módulo se comunica com a câmera *web* que está conectada ao computador usando a biblioteca DSVideoLib.

3.2. Image

Através das funções do módulo Camera obtém-se as informações necessárias do *frame* atual para que o rastreamento de cor seja inicializado. O papel do módulo Image é encontrar os dois grupos de *pixels* que representam ambas as mãos do jogador (assumindo que este já tenha devidamente vestido as luvas). Esta informação é usada na estimativa da posição das mãos na tela. A fase de processamento responsável por esta tarefa procura por todos os *pixels* dentro do *frame* atual que se encontram em um intervalo pré-definido de valores de cor e então organiza os grupos encontrados como está descrito a seguir.

A busca começa no *pixel* superior esquerdo da imagem e continua, um a um, por todos os *pixels* subsequentes. O primeiro passo é a tentativa de criar um grupo de cores baseado na cor buscada ao longo da imagem. No intuito de reconhecer se determinado *pixel* se encontra no intervalo de valores de cor especificado, a relação entre os seus componentes RGB (vermelho, verde e azul) é analisada. Por exemplo, no caso em que a aplicação procura por luvas amarelas, a busca leva em conta que as componentes vermelha e verde devem apresentar valores similares e o valor da componente azul deve ser consideravelmente menor que o das duas anteriores. Essa abordagem funciona relativamente bem, dado que a relação entre as componentes RGB é robusta à maioria das mudanças de iluminação.

Um algoritmo de *labeling* [Suzuki *et al*. 2003] é um procedimento que fornece uma única identificação para cada objeto (um grupo conexo de componentes) em uma imagem. Este tipo de algoritmo é usado para qualquer procedimento de análise subsequente e para distinguir e referenciar objetos. *Labeling* é uma parte indispensável de praticamente todas as aplicações de reconhecimentos de padrões e visão computacional. Neste trabalho, um algoritmo específico de *labeling* (de concepção própria) foi implementado com o objetivo de localizar as regiões correspondentes às luvas de cor de destaque. A implementação utilizada está descrita a seguir.

Inicialmente, um mapa da imagem é criado no intuito de armazenar os *pixels* que já foram visitados, e os que foram marcados como *pixels* válidos (ou seja, que se encontram no intervalo de cor desejado). Para cada *pixel,* a verificação é feita para checar se sua cor corresponde ao intervalo de cor usado na busca. No caso em que não há correspondência, o *pixel* é marcado como visitado no mapa da imagem e a busca continua a partir do *pixel* adjacente. Se ocorrer uma correspondência, então um objeto que representa um grupo de *pixels* é criado e a busca é reiniciada usando o *pixel* atual como origem. Este novo início considera um esquema de vizinhança com quatro direções (vertical e horizontal: para cima, para baixo, à esquerda e à direita) e um raio de tamanho configurável n (o valor padrão para n é 2). Esse valor garante que mesmo quando um *pixel* é separado do *pixel* de origem por uma pequena distância, de acordo com o raio de busca n, ele pode ser incluído no grupo de *pixels* correspondente.

O algoritmo de *labeling* desenvolvido adota uma abordagem gulosa, em que cada vizinho válido (um *pixel* nas proximidades que está no intervalo de cor especificado) é adicionado ao grupo do *pixel* inicial e uma nova busca é iniciada usando este *pixel* como ponto de partida. É importante frisar que a busca por grupo de *pixels* sempre marca os *pixels* visitados no mapa da imagem e tais *pixels* não são considerados em buscas subsequentes. Devido à baixa resolução das imagens usadas (320 x 240 *pixels*) é garantido que o tempo de processamento seja aceitável para aplicações em tempo real. Quando a busca por *pixels* do grupo instanciado termina (ou seja, não existem mais vizinhos válidos), o algoritmo continua a visitar toda a imagem ignorando os *pixels* marcados e criando novos grupos de *pixels*, até que a imagem tenha sido visitada por completo.

Na sequência, depois que todos os grupos de *pixels* estiverem construídos, os dois com a maior quantidade de *pixels* são selecionados. Se por acaso, estes dois grupos tenham tamanhos menores do que um valor pré-determinado, o processamento pára e o *frame* atual é considerado como uma falha de rastreamento. Em outras palavras, se os dois maiores grupos da cor de interesse encontrados são relativamente pequenos, então provavelmente eles representam algum ruído da captura ou componente de fundo da cena. Se este não é o caso, estes dois maiores grupos irão representar as mãos do jogador como mostrado na Figura 13. Esta é uma forma eficiente de encontrar as mãos do usuário, considerando que as luvas coloridas contrastam claramente com o ambiente ao fundo.

Caso existam outros elementos que correspondam ao requisito da cor (além das luvas), eles provavelmente devem ser representados por grupos de cor menores, dado que as mãos do jogador estão localizadas próximas à câmera e consequentemente

ocupam uma área maior na tela. Por razões de simplificação, somente o centro de cada mão é considerado no processamento posterior. O centro de um grupo de cor (também conhecido como centro geométrico, ou simplesmente centróide) é obtido pela soma das coordenadas (horizontais e verticais) de todos os *pixels* relacionados àquele grupo e então a quantia resultante é dividida pelo número de *pixels* do grupo.

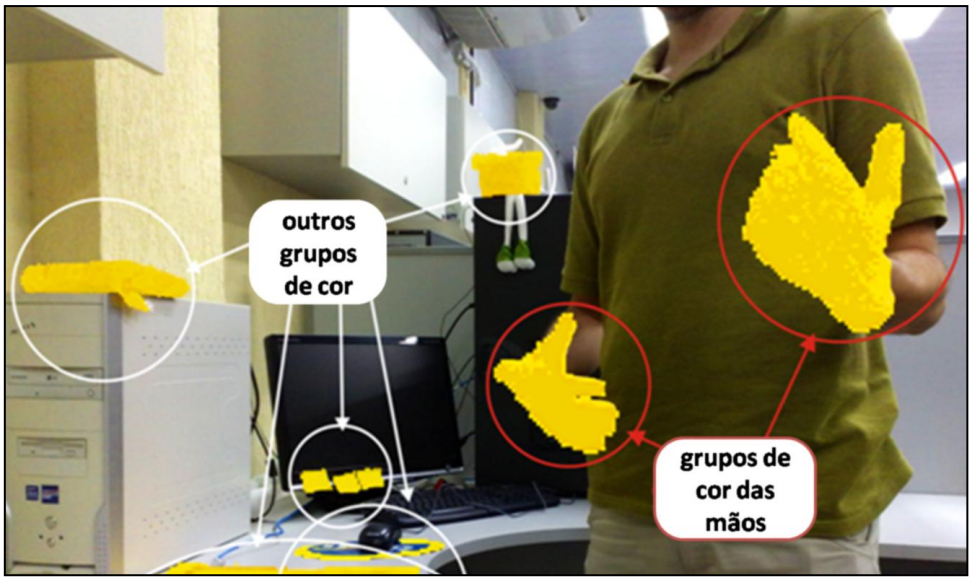

Figura 13. Grupos de cor compostos por conjuntos de *pixels* amarelos. Os grupos maiores irão, usualmente, representar as mãos do jogador.

Uma vez que ambos os centros são calculados, àquele localizado mais à esquerda na tela corresponderá à mão esquerda e analogamente, o outro corresponderá à direita. Considerando que quando se está tocando uma guitarra o guitarrista raramente inverte o lado de suas mãos, essa abordagem apresenta um bom resultado. Além dessa funcionalidade, o módulo Image proporciona aos desenvolvedores funções para acessar os grupos de cor, de forma que seja possível entender e alterar alguns comportamentos do *framework*.

3.3. Guitar

Considerando que os pontos das mãos foram corretamente obtidos (ou seja, não ocorreu uma falha de rastreamento), é iniciada a simulação da guitarra virtual. A tarefa do módulo Guitar é detectar as intenções do jogador quando está executando movimentos similares ao ato de tocar uma guitarra real. A idéia é interpretar automaticamente seus movimentos e dar retornos visuais e sonoros de forma coerente. O modelo da guitarra disponível no *framework* corresponde a uma versão simplificada de uma guitarra comum; a guitarra virtual possui menos trastes e uma única corda. Essa simplificação é fundamental no sentido de tornar possível a manipulação do instrumento virtual pelo jogador. O uso de mais cordas tornaria a interação impraticável, dado que os movimentos com as mãos teriam que ser muito mais precisos (como acontece com uma guitarra real, com o adicional de que não existe retorno tátil). Já o número de trastes é completamente configurável, no entanto é altamente recomendado o uso de poucos (de 2 a 10 trastes) ao invés do número correspondente ao de uma guitarra real (entre 22/23 na maioria dos casos), devido ao mesmo problema descrito anteriormente (precisão da interação).

O processo de simulação da guitarra tem como resultado o posicionamento correto da guitarra virtual baseado na posição das mãos, esta fase do processamento está descrita a seguir. Assumindo que a mão esquerda está posicionada sobre o braço e a direita sobre o corpo da guitarra virtual (esta é a configuração para destros, se necessário ela pode ser invertida), os seguintes passos são executados em sequência:

1. Os dois pontos em vermelho mostrados na Figura 14 são usados como pontos de referência, encontrados baseados nas posições das mãos. A partir deles é possível fazer uma estimativa das posições dos pontos do braço e do corpo da guitarra (estes últimos serão os pontos de referência para o posicionamento da guitarra). Ambos os pontos, do braço e do corpo, possuem velocidades de movimento, que são definidas antes da aplicação ser executada. Essas velocidades são usadas para fazer com que a guitarra siga as mãos do usuário, como se o jogador estivesse segurando-a. Mais especificamente, o ponto do corpo seguirá a mão direita e o ponto do braço a esquerda. Dado que os movimentos da mão esquerda seguem o eixo da guitarra e que os movimentos da mão direita são perpendiculares ao mesmo eixo, diferentes velocidades são aplicadas aos pontos de referência. Por conta da alta amplitude perpendicular do movimento da mão direita, o corpo da guitarra apresenta uma baixa velocidade de movimento, diferentemente do braço. Esta opção é o que possibilita que o usuário efetive a palhetada na guitarra virtual, pois quando este levanta a mão direita e o corpo da guitarra segue sua mão suavemente dando-lhe tempo para descer a mesma mão, cruzando assim a corda da guitarra.
2. Sempre que o *frame* atual representar o primeiro *frame* rastreado com sucesso (ou seja, todos os *frames* anteriores foram considerados como falhas de rastreamento, ou este é o primeiro *frame* capturado pela câmera), os pontos de referência da guitarra terão exatamente a mesma posição dos pontos das mãos. Se este não é o caso, ambos os pontos do braço e do corpo da guitarra irão seguir respectivamente os pontos das mãos esquerda e direita, como ilustrado na Figura 14.
3. O movimento da guitarra é descrito pelo seguinte algoritmo. A posição dos pontos de referência é atualizada baseando-se numa porcentagem fixa do vetor de distância entre um ponto de referência e o seu ponto de mão associado. Tal percentual corresponde à velocidade de movimento do braço ou do corpo da guitarra. Se a distância encontrada é menor do que um valor configurável chamado de “limiar de tremidas” (*jitter treshold*), então os pontos de referência mantêm as suas posições. Essa abordagem garante um deslocamento suave para os pontos do braço e do corpo da guitarra, o que favorece a visualização e a experiência de interação.
4. A interação do jogador é definida a seguir. Sempre que ele cruzar a “linha da corda” (eixo da guitarra), o *framework* captura a intenção de tocar. O fato de o corpo da guitarra se movimentar mais lentamente do que o braço ajuda o usuário, dando-lhe tempo para tocar a corda virtual além de prevenir a execução de sons não-intencionais. Por outro lado, a velocidade do braço da guitarra não necessariamente precisa ser lenta, inclusive é recomendável que

seja rápida para que o jogador tenha noção instantânea da posição que sua mão esquerda ocupa no braço da guitarra.

5. O *framework* faz uso de alguns outros mecanismos para garantir que a interação do jogador com o instrumento virtual seja capturada corretamente. O "limiar de tremidas" é uma destas. Por conta dele, a linha da corda pára de seguir as mãos do jogador todas as vezes em que estas estão muito próximas da linha. Desta forma, a linha da corda não cruzará o ponto da mão direita por si só. Outra política importante nesse sentido é limitar os movimentos que representam a intenção de tocar aos deslocamentos de alta intensidade. Esta intensidade é medida através da distância entre a posição do ponto da mão direita no *frame* anterior e no atual. Finalmente, uma única direção de cruzamento é levada em consideração, para cima ou para baixo. Se por um acaso ambas fossem utilizadas, eventualmente o som seria tocado mesmo que não fosse a intenção do usuário.
6. Simultaneamente à captura da intenção do jogador, dois valores são armazenados para futuro uso no módulo Sound. Tais valores correspondem à intensidade capturada e à distância entre os pontos das mãos esquerda e direita no momento em que o jogador cruzou a linha da corda da guitarra virtual.

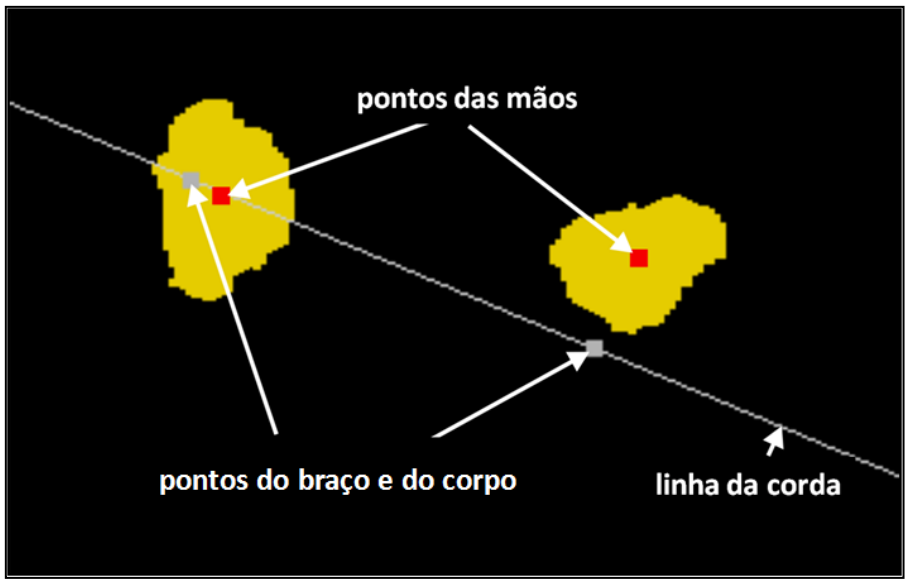

Figura 14. Pontos da guitarra (braço e corpo) seguindo os pontos da mão. É importante perceber que o ponto do braço está bem mais próximo ao seu ponto correspondente do que o ponto do corpo. Isto ocorre devido à velocidade de movimento de cada parte da guitarra.

3.4. Sound

Após coletar ambos os valores, intensidade e distância, como descrito anteriormente, o processamento de som é iniciado. O módulo Sound deve estar configurado antes de ser executado para funcionar corretamente. Primeiramente, o número de trastes deve ser especificado. Com base nesta informação, o braço da guitarra virtual é dividido em diferentes espaços ou casas (separadas pelos trastes) que possuem distintos sons associados. Quanto mais próximo a casa se encontra do corpo da guitarra, maior é a frequência do seu som correspondente.

Cada som gerado pela guitarra virtual está relacionado a uma amostra base previamente escolhida (geralmente uma única nota), armazenada em qualquer formato suportado pelo FMOD Ex (.wav, .mp3, *etc*.). Esta amostra base é tocada sempre que o jogador tocar a guitarra, o que significa que o valor da intensidade foi maior do que o mínimo necessário (como explicado na subseção anterior). A amostra base é tocada de forma que sua frequência é alterada de acordo com a casa em que a mão esquerda do jogador está posicionada. Em outras palavras, o módulo Sound leva em consideração a informação da frequência padrão da amostra base de som (arquivo de som) e incrementa esta frequência sempre que necessário, usando a casa em que está a mão esquerda do usuário como referência. A primeira casa da guitarra virtual (a mais distante do corpo da guitarra) irá corresponder ao som base, enquanto as próximas terão sua frequência incrementada como está descrito a seguir.

O módulo Sound usa a escala igualmente temperada de doze tons como base para realizar os incrementos de frequência. Esta divisão foi escolhida por conta do seu método de segmentação, o qual é largamente utilizado no atual panorama musical. Praticamente todos os instrumentos baseados em trastes usam esta divisão, e ela representa a divisão que mais se aproxima da entonação justa (outra segmentação que apresenta uma divisão não igualitária, porém com intervalos perfeitamente consonantes) sem fazer uso de muitas notas adicionais. A diferença entre a escala temperada e o caso de referência (entonação justa) nunca é maior do que 1%, como pode ser visto na Tabela 1. A entonação justa pode também ser usada, mas ao mesmo tempo em que esta apresenta intervalos perfeitos, em outras combinações ela não soa tão bem. Outras escalas temperadas também podem ser usadas, dado que o número de divisões por oitava (atualmente doze) pode ser facilmente alterado em tempo real (por exemplo, alguns músicos indianos usam trinta e uma divisões).

Tabela 1. Relação entre a entonação justa e a escala igualmente temperada. É possível perceber que a diferença entre os incrementos correspondentes nunca é maior que 1%.

Incrementos	Entonação Justa	Igualmente Temp.	Diferença
0	1/1 = 1.000	$2^{0/12}$ = 1.000	0.0%
1	16/15 = 1.067	$2^{1/12}$ = 1.059	0.7%
2	9/8 = 1.125	$2^{2/12}$ = 1.122	0.2%
3	6/5 = 1.200	$2^{3/12}$ = 1.189	0.9%
4	5/4 = 1.250	$2^{4/12}$ = 1.260	0.8%
5	4/3 = 1.333	$2^{5/12}$ = 1.335	0.1%
6	7/5 = 1.400	$2^{6/12}$ = 1.414	1.0%
7	3/2 = 1.500	$2^{7/12}$ = 1.498	0.1%
8	8/5 = 1.600	$2^{8/12}$ = 1.587	0.8%
9	5/3 = 1.667	$2^{9/12}$ = 1.682	0.9%
10	9/5 = 1.800	$2^{10/12}$ = 1.782	1.0%
11	15/8 = 1.875	$2^{11/12}$ = 1.888	0.7%
12	2/1 = 2.000	$2^{12/12}$ = 2.000	0.0%

Desta forma, a frequência inicial da amostra base é alterada multiplicando o seu valor por $2^{1/12}$ (aproximadamente 1.0595) tantas vezes quanto indicado pelo número de casas deslocadas na direção do corpo da guitarra. O som base é considerado como matriz para todos os outros sons, mesmo caso este não seja uma nota conhecida,

simplesmente preservando a relação entre eles; ou seja, a entonação absoluta considerando o A4 como 440Hz não é levado em conta, dado que o objetivo é garantir que as diferentes casas toquem sons coerentes, diferenciando uns dos outros.

Alguns melhoramentos podem ser adotados com o objetivo de fazer com que a interação se torne mais atrativa. Para evitar notas indesejadas, é possível excluir alguns sons, dessa forma limitando o tom a ser executado. Qualquer um dos doze sons e seus correspondentes oitavados (sons com o dobro da frequência do anterior) podem ser ativados/desativados em qualquer momento. Desta forma, caso o jogador deseje tocar, por exemplo, uma escala pentatônica usando o som base como matriz, ele deve simplesmente desativar as notas indesejadas como mostrado na Figura 15. A interface da aplicação de demonstração desenvolvida para testar o *framework* mostra no topo da tela todas as notas ativadas, da primeira à décima segunda. Para habilitar ou desabilitar qualquer uma delas, o usuário deve pressionar um dos botões F_n do teclado, onde n varia de 1 a 12. Essa funcionalidade também torna possível restringir as notas a um conjunto de sons necessários para tocar uma música específica, como detalhado posteriormente no capítulo Estudo de Caso. Quando uma nota é desativada, o traste que tocaria aquele som passa a tocar o som da próxima nota habilitada. Desta forma, todos os trastes permanecem válidos; o que difere são as notas tocadas.

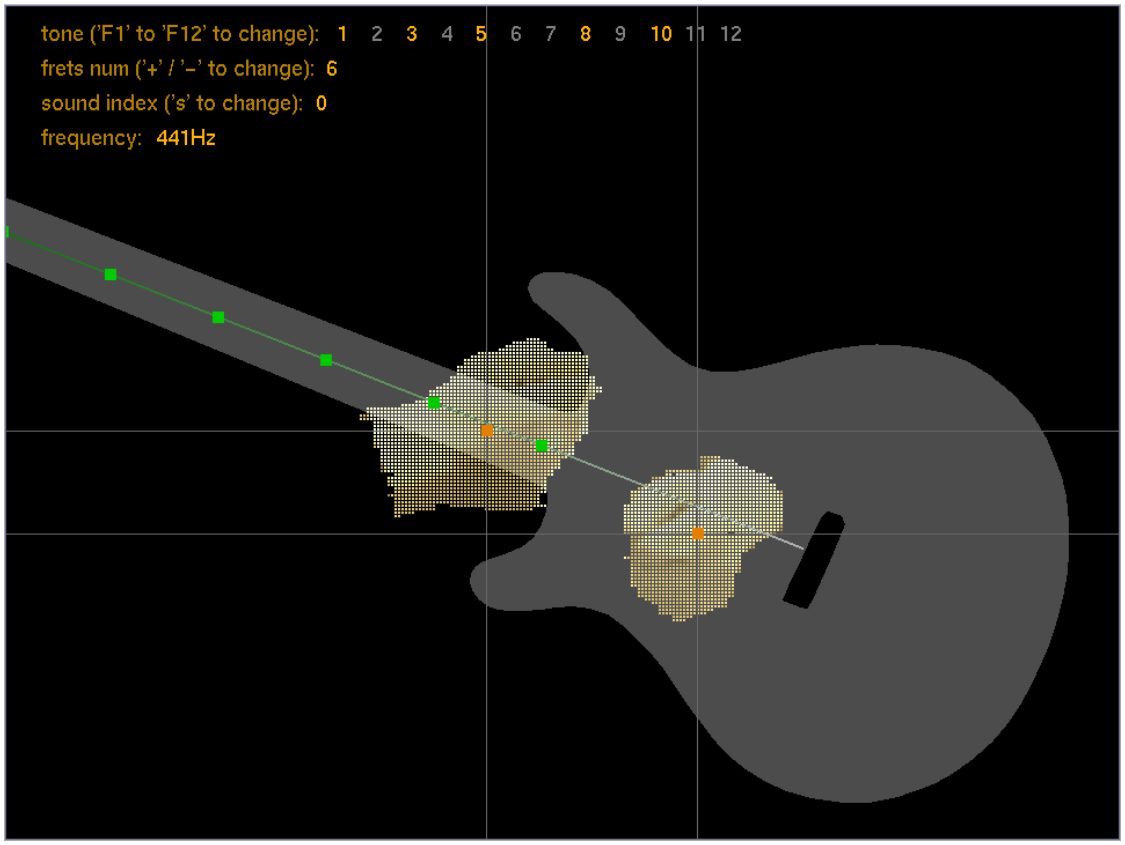

Figura 15. Interface da aplicação exemplo. No topo da imagem está definido o tom da guitarra virtual; somente as notas 1, 3, 5, 8 e 10 estão ativadas, determinando uma escala pentatônica usando o som base como primeira nota. Outras configurações podem ser modificadas durante a execução como o número de trastes (quadrados verdes) e o som base utilizado.

Outra funcionalidade implementada corresponde ao volume do som tocado, o qual é baseado no valor de intensidade recebido como entrada pelo módulo Sound. Quanto

mais rápido é o movimento da mão direita ao cruzar a corda, maior será o volume do som a ser executado. O envoltório do som tocado (o que determina o volume do som durante uma quantia de tempo específica) também pode ser alterado proporcionando um crescente efeito de sustentação da nota tocada. O efeito de *slide* também foi implementado, o qual faz com que o som tocado seja modificado sempre que o jogador desliza a mão esquerda, mesmo sem movimentar a outra mão, de maneira similar a técnica de *slide* aplicada em guitarras reais. Finalmente, mais canais de sons podem ser adicionados com a intenção de criar combinações mais complexas, aprimorando assim a experiência do usuário, por exemplo através da simulação de acordes. O módulo Sound, juntamente com a biblioteca FMOD Ex, provê funções capazes de gerenciar todos os parâmetros citados nesta subseção a qualquer momento, em tempo real.

3.5. Renderer

Este módulo é responsável por retornar as informações visuais para o jogador, ajudando-o a alcançar os seus objetivos e manipular a aplicação. Mesmo sendo inteiramente possível jogar sem tais informações visuais, a interação se torna mais interessante com a visão da guitarra virtual (e suas divisões de trastes) de acordo com as posições das luvas. Além disso, outras informações podem ser mostradas, como indicações de textos para as notas ativadas/desativadas, a frequência da última nota tocada, e outras mais como pode ser visto na Figura 15.

O módulo Renderer também provê aos desenvolvedores algumas funções para a visualização de detalhes do funcionamento do *framework*, como por exemplo renderizar os pontos da guitarra (do braço e do corpo). Estas funções são úteis para a fase de testes de uma aplicação, dado que uma das atividades que mais consomem tempo dos desenvolvedores é a busca por erros e a manutenção dos mesmos, de forma que dicas visuais sempre são bem-vindas. Esse módulo faz uso da biblioteca OpenGL [Silicon 1992] (e GLUT [Silicon 1996]) para executar tais funções. Se necessário, o motor de renderização pode ser alterado através da construção de outro módulo Renderer (por exemplo, usando o OGRE3D [Ogre 2005]), similar ao fornecido pelo *framework*, sem modificar os demais módulos.

4. Estudo de caso

Nesta seção está descrito o estudo de caso, baseado em uma simples aplicação de exemplo que usa as funcionalidades do *framework*. A interface utilizada para esta aplicação pode ser vista na Figura 15. O processo de avaliação do *framework* é descrito a seguir.

4.1. Método de avaliação

Neste trabalho, o *framework* proposto foi avaliado usando uma implementação de demonstração de um jogo simplificado de *Air Guitar*. Os testes foram realizados usando um processador AMD Athlon X2 4800+, com um 1GB de memória RAM e equipado com uma placa de vídeo NVidia 8800 GTX. Foi cronometrado o tempo necessário para que fosse executada corretamente uma sequência de notas. Para assegurar que os resultados são consistentes, houve um esforço no sentido de expandir o número de usuários de teste ao máximo. Um total de 23 pessoas foi submetido aos testes realizados, variando a idade (de 17 até 47 anos), o grau de experiência com jogos e também o conhecimento musical dos mesmos.

Após se habituarem à aplicação demonstrativa (jogando livremente durante alguns minutos), foi pedido a eles que executassem uma versão da introdução da música *Smoke on the Water* (de *Deep Purple*); o trecho em questão é composto principalmente por 4 notas diferentes, como ilustrado na Figura 16.

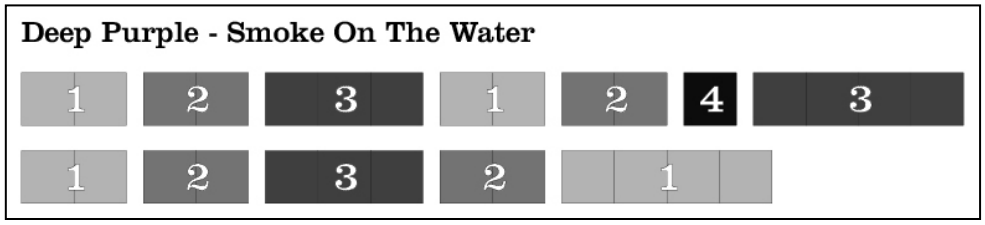

Figura 16. Representação da sequência inicial da música "*Smoke on the Water*". A largura (quantidade de quadrados) de cada número representa a duração de tempo da nota.

Para a execução dos testes, a guitarra virtual foi previamente configurada especificamente para permitir que a sequência fosse executada com facilidade. Uma amostra de som da nota A3 tocada por uma guitarra foi capturada para ser usada como som base, tornando-se a primeira nota da introdução (diferentemente do acorde baseado na nota B2 que é executado na versão original). Somente quatro notas foram habilitadas, sendo estas o som base (A3 como a nota 1) e as notas 4, 6 e 7 (derivadas do som base). O número de trastes foi configurado para quatro, cada um representando uma das notas ativadas, desta forma, facilitando a tarefa dos jogadores.

O ambiente de teste foi organizado como está descrito a seguir. Uma câmera *web* (A4Tech) foi posicionada a 1,30 metros do chão e a cerca de metro e meio de distância à frente do usuário. Devido a características da câmera, esse espaçamento demonstrou ser uma boa escolha para capturar de maneira confortável as mãos do usuário, permitindo a este uma boa amplitude nos seus movimentos. A visualização da aplicação foi projetada em uma parede, a dois metros e meio de distância do usuário,

desta forma, aprimorando a visão do mesmo e a imersão deste na aplicação. O método de avaliação é descrito em mais detalhes a seguir:

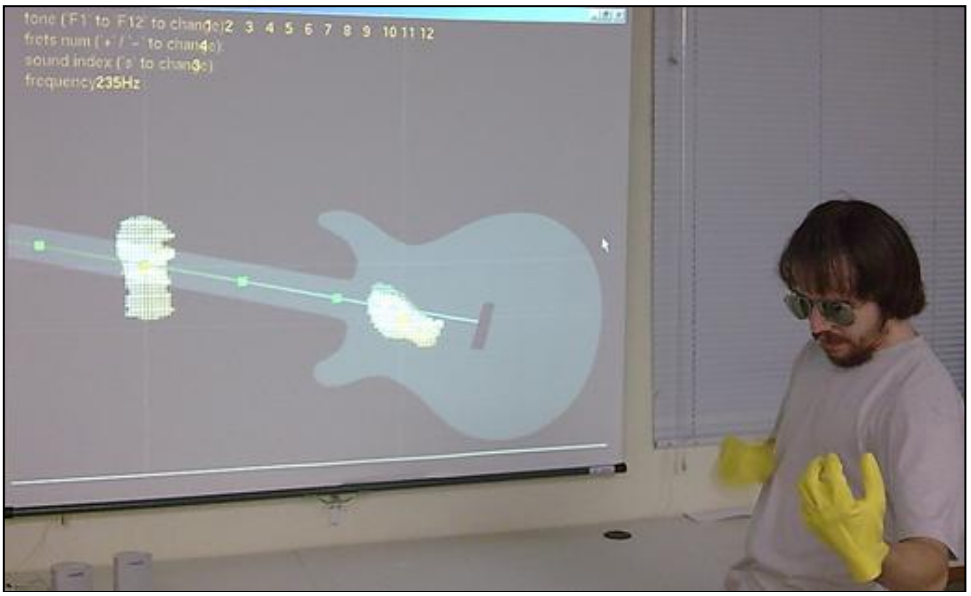

Figura 17. Aplicação exemplo sendo controlada por um par de luvas amarelas.

- Preparação: os usuários colocam as luvas amarelas e se posicionam em frente à câmera. Logo após, eles são instruídos sobre como posicionar corretamente o corpo e as mãos com o objetivo de fazer com que a guitarra virtual, bem como os grupos de cor das mãos, possam ser visualizados por inteiro na tela, como visto na Figura 17.
- Familiaridade com a aplicação: antes de executar o teste em si, os usuários tiveram o tempo de 3 minutos para entender a interface e avaliar a sua performance tocando notas de forma aleatória.
- Teste: depois de compreender como a guitarra virtual funciona, os usuários receberam uma partitura simplificada das notas que deveriam ser tocadas (como mostrado na Figura 16). Eles foram informados da correspondência entre os números escritos na partitura e as notas na guitarra, que variam de acordo com a posição da mão esquerda. A quantia total de tempo considerada foi aquela usada para executar corretamente toda a sequência, desde o momento em que o jogador toca a primeira nota até a última. No caso de falhas na execução, o usuário foi instruído a reiniciar a sequência desde o começo, e a contagem do tempo não era paralisada e tampouco reiniciada.

Depois de realizar o teste, os usuários foram orientados a responder, de forma livre, um pequeno questionário sobre a experiência que tiveram com o aplicativo. Foi perguntado a eles se sentiram alguma dificuldade no posicionamento das mãos, se os retornos, visual e sonoro, estavam coerentes com seus movimentos e também sobre as impressões que eles tiveram em relação à experiência.

4.2. Resultados

Os testes com a aplicação demonstrativa do *framework* demonstraram que o aplicativo é eficaz, intuitivo, e atrativo, além disso, não aconteceram quaisquer

sinalizações por parte dos usuários em relação incômodos ou desconfortos. Todos os 23 usuários que se submeteram aos testes foram capazes de executar corretamente a sequência. O tempo médio levado para tal performance foi de 33.6 segundos, variando de 11 a 80 segundos (respectivamente, menor e maior tempo).

A Figura 18 apresenta a idade dos usuários e o tempo gasto por eles realizando o teste, indicando uma densidade maior no intervalo entre 20 e 30 anos, o qual é um potencial alvo para um produto usando a tecnologia propiciada pelo *framework*. A maior parte dos usuários com idade dentro deste intervalo gastou entre 11 e 40 segundos. Este tempo foi considerado como bastante satisfatório, levando em conta que estas pessoas nunca haviam interagido com a aplicação antes.

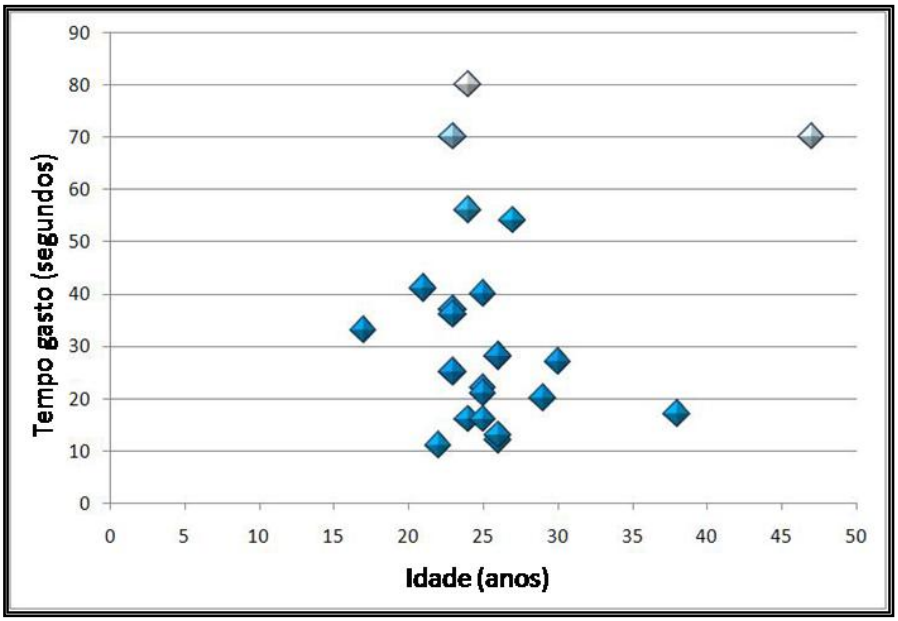

Figura 18. Tempo gasto *versus* idade do usuário.

Depois de questionados sobre o retorno da guitarra virtual de acordo com o movimento de suas mãos, 100% dos usuários relataram que não sentiram qualquer atraso ou interpretação equivocada por parte da aplicação. Esta capacidade de fornecer o retorno (tanto sonoro quanto visual) em tempo real é um dos principais atributos do *framework*, devido ao fato de que qualquer atraso, mesmo que pequeno, comprometeria a usabilidade da guitarra virtual, tornando difícil para o usuário movimentar suas mãos de forma espontânea e intuitiva.

Alguns dos usuários sentiram dificuldades em relação ao posicionamento das mãos, em parte devido ao completo desconhecimento da interface, em parte devido à posição da câmera, que estava fixada na mesma altura em todos os testes. Variações do tamanho dos usuários não foram levadas em conta, e como consequência alguns tiveram que posicionar as mãos muito acima, muito abaixo, muito abertas ou muito fechadas. Entretanto, a maior parte deles conseguiu superar este problema inicial encontrando uma forma adequada de posicionar as mãos para executar o trecho musical usado como teste.

Todos os indivíduos que testaram o *framework* forneceram resultados positivos em relação à experiência ao tocar a guitarra virtual. Eles demonstraram estar motivados

ao ver a aplicação das capacidades do *framework* em um cenário real. Muitas das respostas ao questionário sugeriram o desenvolvimento de um jogo baseado no aplicativo demonstrativo. À parte disso, eles apontaram algumas melhorias e adaptações, tais como: adição de mais cordas à guitarra virtual, no intuito de prover uma simulação mais realística; aplicação da tecnologia para outros instrumentos, como baixo elétrico e bateria; possibilidade de mudar algumas configurações usando as próprias luvas, ao invés do teclado; implementação de outros tipos de interação com a guitarra, como por exemplo o *bend* (movimentando a mão esquerda lentamente no sentido vertical para tencionar a corda virtual aumentando a frequência do som tocado).

4.3. Análise da experiência

Os testes descritos anteriormente foram executados usando um pequeno conjunto das funcionalidades do *framework*. No entanto, foi provado que é possível tocar músicas completas, se usadas todas as possibilidades de notas e trastes. O *framework* possui uma série de possibilidades de uso, tais como desenvolvimento de jogos (como demonstrado na Figura 19), ou uma aplicação com foco na educação musical, por exemplo, ilustrando alguns conceitos como notas, tons e ritmo para os usuários.

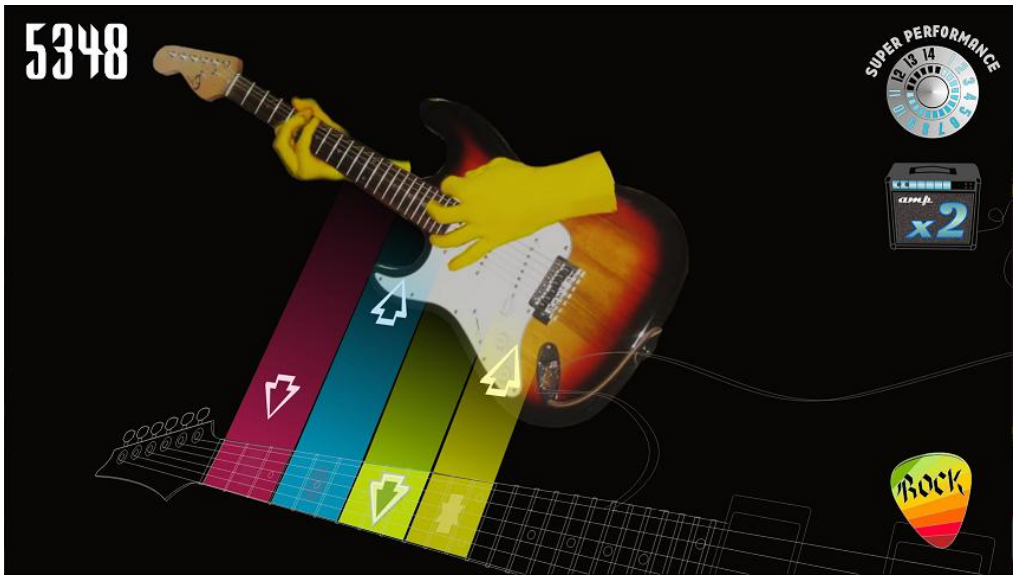

Figura 19. Esboço de um possível jogo usando a tecnologia do *framework*.

Devido a essas funcionalidades, é possível perceber que o *framework* é capaz de prover a ambos, desenvolvedores e usuários, os três requisitos previamente listados: interação em tempo real, controle da captura de movimentos de *Air Guitar* e resposta sonora realística. A satisfação dos usuários que testaram a ferramenta demonstra que executar um tipo de performance familiar incrementa a usabilidade de um aplicativo. É possível obter uma aplicação simples, porém funcional usando o *framework*, enquanto funcionalidades mais complexas podem ser adicionadas tendo como base a aplicação demonstrativa.

Levando em conta um jogo como um possível exemplo de aplicação, algumas funcionalidades podem ser adicionadas para fazer com que este se torne mais interessante, tais como: a possibilidade do jogador escolher entre diversos modelos de

guitarra (como ilustrado na Figura 20); um sistema monetário de recompensas, baseado na forma como as músicas foram executadas; possibilidade de jogar novas músicas, progressivamente destravadas ao longo do jogo; navegação pelos *menus* do jogo usando simplesmente as luvas, ao invés do teclado e do *mouse*; suporte a efeitos de guitarra como o *vibrato* (ilustrado na Figura 21); um *design* mais atrativo na interface do jogo, mostrando informações como pontuação e qualidade da performance do jogador (além da sequência a ser executada, claro).

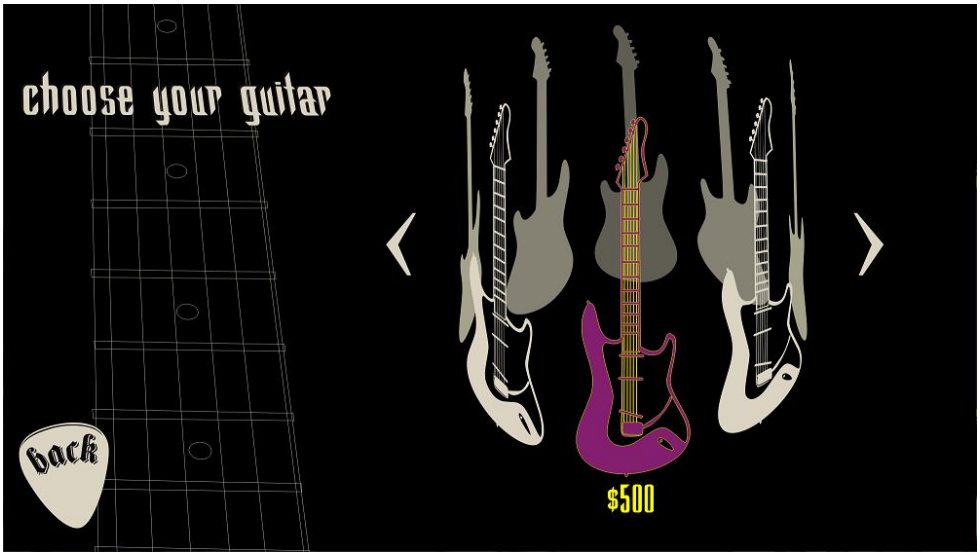

Figura 20. Tela de seleção da guitarra que será usada na performance.

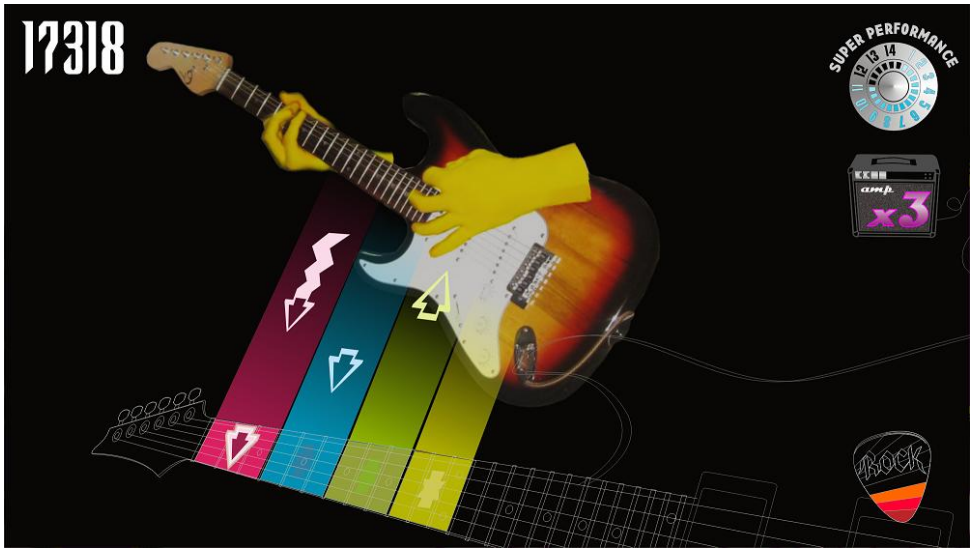

Figura 21. Tela ilustrando o efeito de *vibrato* durante a execução de um jogo.

5. Conclusões e trabalhos futuros

Neste trabalho, um *framework* de código aberto para aplicações baseadas em *Air Guitar* foi detalhado, demonstrando que com um par de luvas comuns e uma câmera *web* é possível reconhecer a intenção do usuário de tocar uma guitarra virtual. Também foi descrito um estudo de caso, no qual uma aplicação exemplo foi usada para validar o *framework*. Esta aplicação foi testada e os resultados se mostraram bem sucedidos: todos os indivíduos que testaram a plataforma foram capazes de executar uma música simples em uma pequena quantia de tempo e mais importante ainda, todos eles gostaram e se sentiram satisfeitos com a experiência.

Este trabalho também resultou em um artigo completo publicado no Simpósio Brasileiro de Jogos e Entretenimento Digital – SBGames em sua edição de 2009 [Lucas *et al.* 2009]. Além disso, como reconhecimento do esforço de toda uma equipe envolvida e da relevância do trabalho, o artigo recebeu o prêmio de melhor artigo da trilha de computação no mesmo evento. Ainda dentro do cenário de premiações, o mesmo artigo recebeu um convite para ser submetido em uma versão expandida para uma revista científica de abrangência internacional (*ACM Computers in Entertainment*) ainda neste ano de 2009.

Um resultado de publicação adicional foi a publicação de um artigo completo no Simpósio de Realidade Virtual e Aumentada – SVR em sua edição de 2009 [Lima *et al.* 2009], cujo conteúdo é relacionado com técnicas de rastreamento 3D baseadas em modelos. Durante o SVR2009 também foi ministrado um minicurso sobre o tema, e o material didático preparado para o mesmo foi publicado como um capítulo de livro [Lima *et al.* 2009].

Um vídeo demonstrativo do *framework* de *Air Guitar* pode ser visto no endereço http://www.youtube.com/watch?v=9saKS6V12X0, e o código fonte da última versão está disponível para *download* em http://sourceforg.net/projects/airguitarframew.

Como trabalhos futuros, um número de novas funcionalidades pode ser adicionado ao *framework*. Alguns efeitos como a simulação de *bends* e *vibratos* pode ser implementada para tornar a interação do jogador ainda mais interessante. Um modo multijogador possivelmente também estará presente nas próximas versões, com a simples restrição de que os jogadores devem usar luvas de cores distintas.

Bibliografia

[Apple 2007] Apple. Junho, 2007. *iPhone*. http://www.apple.com/iphone/

[Bærentsen 2001] Bærentsen, K. B. Janeiro, 2001. *Intuitive user interfaces*. Scandinavian Journal of Information Systems, 12, 1-2, 29-60. http://portal.acm.org/citation.cfm?id=372680

[Bao *et al*. 2006] Bao, X., Herlocker, J. L., and Dietterich, T. G. Fevereiro, 2006. *Fewer clicks and less frustration: reducing the cost of reaching the right fold*er. In Proceedings of the 11th international Conference on intelligent User interfaces, 178-185. http://portal.acm.org/citation.cfm?id=1111449.1111490#

[Camurri *et al*. 1999] Camurri, A., Ricchetti, M., Trocca, R. 1999. *Eyesweb - toward gesture and affect recognition in dance/music interactive systems*. In ICMCS '99: Proceedings of the IEEE International Conference on Multimedia Computing and Systems, IEEE Computer Society, Washington, DC, USA, 9643.

[Dorner 1994] Dorner, B. 1994. *Chasing the colour glove : visual hand tracking*. http://ir.lib.sfu.ca/bitstream/1892/6545/1/b14873412.pdf

[Engadget 2009] Engadget. Junho, 2009. *Sony announces new PS3 motion controller*. http://www.engadget.com/2009/06/02/sony-announces-new-ps3-motion-controller/

[FreeStyleGames 2009] FreeStyleGames. Outubro, 2009. *DJ Hero*. http://www.djhero.com/

[Games 2007] Armor Games. 2007. *Adagio*. http://www.newgrounds.com/portal/view/388085

[Harmonix 2001] Harmonix. Novembro, 2001. *Frequency*. http://www.harmonixmusic.com/#games

[Harmonix *et al*. 2007] Harmonix, MTV Games. Novembro, 2007. *Rockband*. http://www.rockband.com/

[Heap *et al*. 1995] Heap , T., Samaria, F. Junho, 1995. Real-Time Hand Tracking and Gesture Recognition Using Smart Snakes. http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.17.6931

[Hoysniemi 2006] Hoysniemi, J. Abril, 2006. *International survey on the Dance Dance Revolution game*. Computer in Entertainment (CIE), 4, 2, 8. DOI= http://doi.acm.org/10.1145/1129006.1129019

[Kato *et al*. 1999] Kato, H., Billinghurst, M. Outubro, 1999. *Marker Tracking and HMD Calibration for a video-based Augmented Reality Conferencing System*. In Proceedings of the 2nd International Workshop on Augmented Reality (IWAR 99), San Francisco, USA, 85. http://www.hitl.washington.edu/artoolkit/Papers/IWAR99.kato.pdf

[Konami *et al*. 1998] Konami, Konami Digital Enterteinment. 1998. *Guitarfreaks*. http://en.wikipedia.org/wiki/GuitarFreaks

[Lima *et al*. 2009] Lima, J. P. S. M., Simões, F. P. M., Figueiredo, L. S., Teichrieb, V., Kelner, J., Santos, I.H. F. 2009.*Model Based 3D Tracking Techniques for Markerless Augmented Reality*. In: Symposium on Virtual and Augmented Reality, 2009, Porto Alegre. Proceedings of the Symposium on Virtual and Augmented Reality.

[Lima *et al*. 2009] Lima, J.P., Simões, F.P., Figueiredo, L., Teichrieb, V. and Kelner, J. 2009. *Online Monocular Markerless 3D Tracking for Augmented Reality*. Abordagens Práticas de Realidade Virtual e Aumentada: SVR2009 – Livro dos Minicursos, Sociedade Brasileira de Computação.

[Lucas *et al*. 2009] Figueiredo, L. S., Teixeira, J. M. X. N., Cavalcanti, A. S., Teichrieb, V., Kelner, J. Outubro, 2009. *An open-source framework for air guitar games*. n: VIII Simpósio Brasileiro de Jogos e Entretenimento Digital (SBGames), 2009, Rio de Janeiro. http://www.sbgames.org/papers/sbgames09/computing/full/cp9_09.pdf

[Mäki *et al*. 2005] Mäki-Patola, T., Laitinen, J., Kanerva, A., And Takala, T. 2005. *Experiments with virtual reality instruments*. In NIME '05: Proceedings of the 2005 conference on New interfaces for musical expression, National University of Singapore, Singapore, 11–16. http://portal.acm.org/citation.cfm?id=1085939.1085946#

[Mammen *et al*. 2004] Mammen, J. P., Chaudhuri, S., Agrawal, T. 2004. *Simultaneous Tracking of Both Hands by Estimation of Erroneous Observations*. http://citeseerx.ksu.edu.sa/viewdoc/summary?doi=10.1.1.3.1199

[Microsoft 2009] Microsoft Corporation. 2009. *Project Natal*. http://www.xbox.com/en-US/live/projectnatal/

[Miller 2009] Miller, C. R. 2009. *10/GUI*. http://10gui.com/

[Nintendo 2006] Nintendo. Novembro, 2006. *Wii*. http://www.nintendo.com/wii

[Ogre 2005] Ogre Team, The. Fevereiro, 2005. *Object-oriented graphics rendering engine (OGRE3D)*. http://www.ogre3d.org/

[Pakarinen *et al*. 2008] Pakarinen, J., Puputti, T., Välimäki, V. 2008. *Virtual slide guitar*. Comput. Music J, 32, 3, 42–54. DOI= http://dx.doi.org/10.1162/comj.2008.32.3.42

[Para *et al*. 2008] Para, E., Bernier, O., and Achard, C. Julho, 2008. *2D Articulated Body Tracking with Self-occultations Handling*. In Proceedings of the 5th international Conference on Articulated Motion and Deformable Objects. F. J. Perales and R. B. Fisher, Eds. Lecture Notes In Computer Science, vol. 5098. Springer-Verlag, Berlin, Heidelberg, 156-165. DOI= ,http://dx.doi.org/10.1007/978-3-540-70517-8_16

[RedOctane 2005] RedOctane. Novembro, 2005. *Guitarhero*. http://hub.guitarhero.com/index uk.html

[Silicon 1992] Silicon Graphics. 1992. Open graphics library (opengl). http://www.opengl.org/

[Silicon 1996] Silicon Graphics. 1996. *The opengl utility toolkit (GLUT)*. http://www.opengl.org/resources/libraries/glut/

[Stefanov *et al*. 2005] Stefanov, N., Galata, A., Hubbold, R. Junho, 2005. *Real-time Hand Tracking With Variable-Length Markov Models of Behaviour*. In Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (Cvpr'05) - Workshops - Volume 3. CVPR. IEEE Computer Society, Washington, DC, 73. DOI= http://dx.doi.org/10.1109/CVPR.2005.518

[Stenger *et al*. 2004] B. Stenger, A. Thayananthan, P. H. S. Torr, R. Cipolla. Maio, 2004. *Hand Pose Estimation Using Hierarchical Detection*. Intl. Workshop on HCI, 105-116, Prague, Czech Republic. http://mi.eng.cam.ac.uk/reports/svr-ftp/stenger_hci04.pdf

[Suzuki *et al*. 2003] Suzuki, K., Horiba, I., Sugie, N. Janeiro, 2003. *Linear-time connected-component labeling based on sequential local operations*. Computer Vision and Image Understanding, 89, 1, 1–23. http://portal.acm.org/citation.cfm?id=780781

[Thayananthan *et al*. 2008] A. Thayananthan, R. Navaratnam, B. Stenger, P. H. S. Torr, R. Cipolla. Julho, 2008. *Pose Estimation and Tracking Using Multivariate* Regression. Pattern Recognition Letters, 29, 9, 1302-1310. DOI= http://dx.doi.org/10.1016/j.patrec.2008.02.004

[Turoque 2006] Turoque, B. 2006. *To Air is Human: One Man's Quest to Become the World's Greatest Air Guitarist*. Riverhead Trade. http://bjornturoque.com/home.htm

[Valli 2005] Valli, A. 2005. *Notes on Natural Interaction*. http://www.citeulike.org/user/eckel/article/4324923

[Voodoo 2006] Unreal Voodoo. Agosto, 2006. *Frets on Fire*. http://fretsonfire.sourceforge.net/

[Wang *et al*. 2007] Wang, J., Sung, E. 2007. *EM enhancement of 3D head pose estimated by point at infinity*. Image Vision Computing, 25, 12, 1864-1874. DOI= http://dx.doi.org/10.1016/j.imavis.2005.12.017

[Wang *et al*. 2009] Wang, R. Y., Popović, J. Agosto, 2009. *Real-time hand-tracking with a color glove*. In ACM SIGGRAPH 2009 Papers. H. Hoppe, Ed. SIGGRAPH '09. ACM, New York, NY, 1-8. DOI= http://doi.acm.org/10.1145/1576246.1531369

[Wixon 2007] Wixon, D. Maio, 2007. *Guitar Hero: the inspirational story of an "overnight" success*. Interactions, 14, 3, 16-17. DOI= http://doi.acm.org/10.1145/1242421.1242435

[Zhang *et al*. 2008] Zhang, L., Chen, J., Zeng, Z., Ji, Q. Dezembro, 2008. *2D and 3D upper body tracking with one framework*. 19th International Conference on Pattern Recognition (ICPR 2008), 1-4. http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=4761484&isnumber=4760915

Guia do Usuário

A seguir é apresentado o guia do usuário para interessados tanto no desenvolvimento quanto na utilização da aplicação demonstrativa do *framework* de *Air Guitar*. O *framework* é uma ferramenta de código aberto que está disponível para *download* na *web*, e tem uso livre para atividades acadêmicas e de forma geral, com objetivos não-financeiros. A seguir está descrita uma série de etapas que facilitam o entendimento no primeiro uso do *framework*. Caso exista a necessidade de obter-se um conhecimento mais profundo sobre o funcionamento da ferramenta recomenda-se que o usuário busque como fontes de estudo o artigo publicado no SBGAMES 2009, "*An Open Source Framework for Air Guitar Games*" (http://www.sbgames.org/papers/sbgames09/computing/full/cp9_09.pdf), ou o presente Trabalho de Graduação "Um *framework* de código aberto para jogos de *Air Guitar*", defendido no Centro de Informática da Universidade Federal de Pernambuco (CIn – UFPE).

1. Fazendo o *download* do pacote do *framework*.

Acesse o endereço http://sourceforge.net/projects/airguitarframew/ e clique no botão verde "*download now!*" como indicado na figura abaixo.

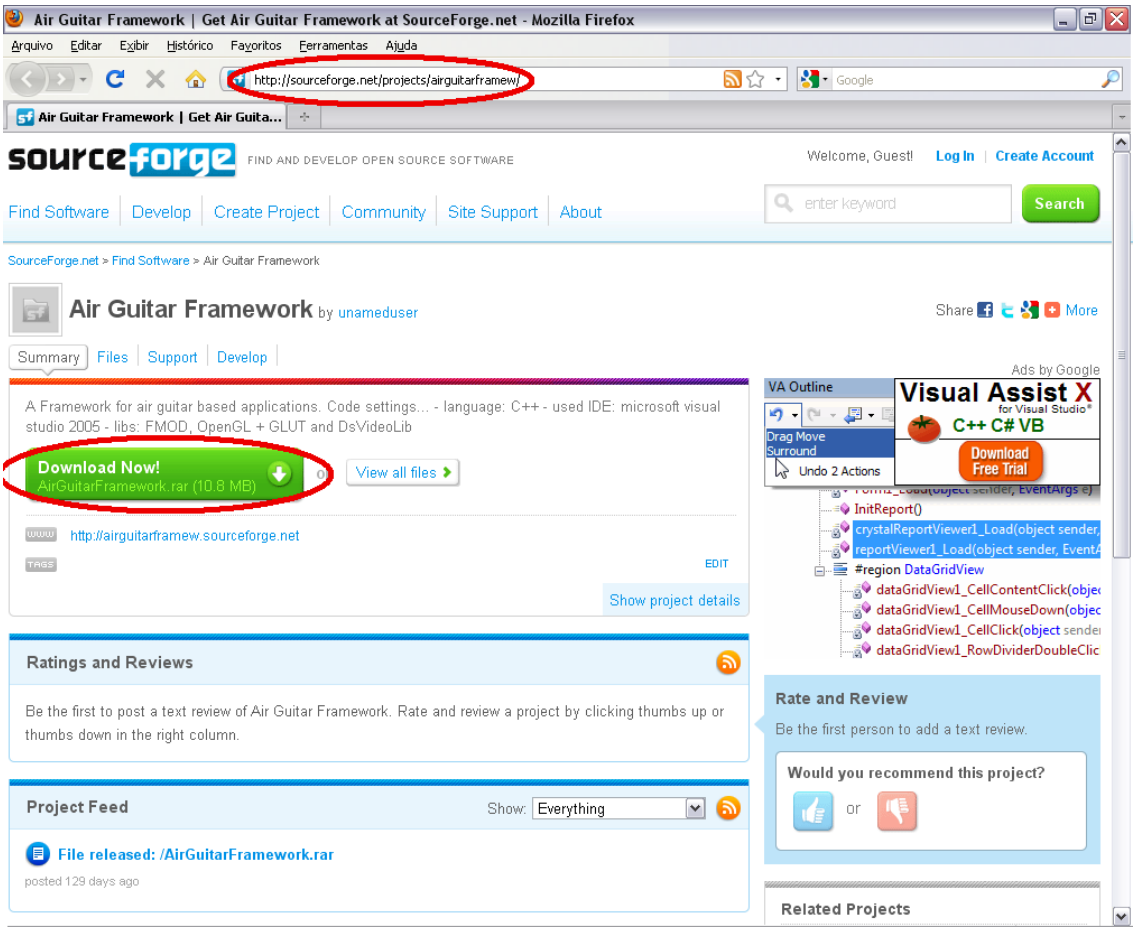

Figura 1. Repositório no SourceForge.net contendo o código do framework e a aplicação exemplo.

2. Analisando o conteúdo do arquivo “AirGuitarFramework.rar”.

Extraia os arquivos contidos no AirGuitarFramework.rar para uma pasta de sua opção. Todos os arquivos disponíveis estão na pasta AirGuitarPro (na figura abaixo esses arquivos podem ser visualizados).

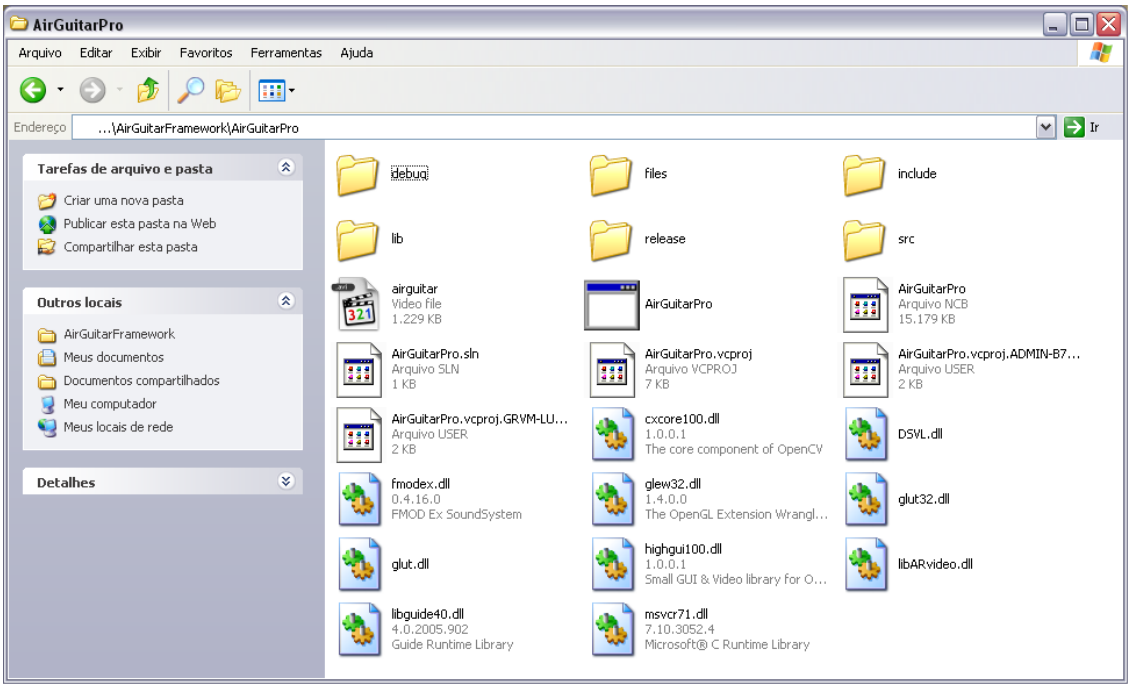

Figura 2. Arquivos contidos na pasta AirGuitarPro.

Nesta pasta existe um arquivo executável (AirGuitarPro.exe) que executa a aplicação exemplo do *framework*. Este executável pode ser configurado para usar como entrada tanto a câmera *web* quanto um vídeo. *A priori* esta configuração usa um vídeo como entrada; desta forma, permitindo que qualquer um possa executar a aplicação exemplo mesmo que não possua uma câmera ligada ao PC. O vídeo utilizado como entrada é o arquivo “airguitar.avi” que está na mesma pasta (AirGuitarPro). Este vídeo é uma sequência de gravação com 29 segundos simulando a interação de um usuário da aplicação exemplo. Para mudar esta configuração, ou seja, usar a sua câmera *web* como entrada, abra o arquivo “.\AirGuitarPro\files\camera\camera.xml”. Neste arquivo estará contido o código em ‘xml’ que indica qual fonte está sendo usada como entrada, como pode ser visto na figura a seguir.

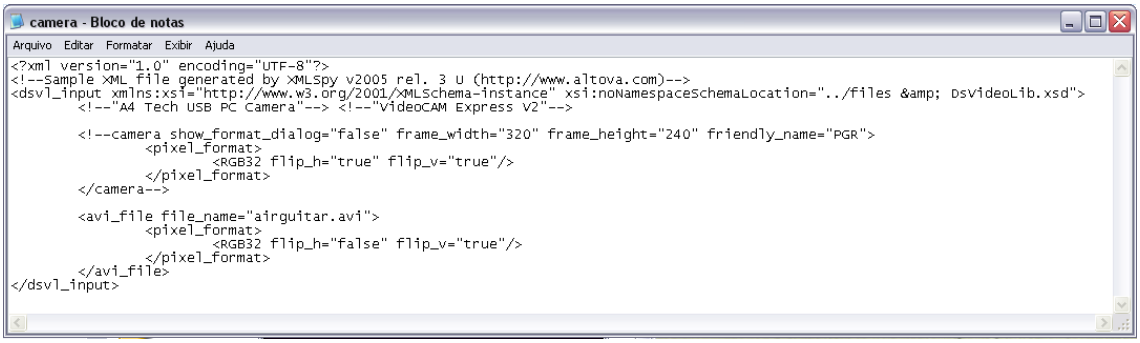

Figura 3. Arquivo “camera.xml”, contendo as configurações sobre a fonte de entrada para a aplicação exemplo.

Perceba que o trecho referente à câmera está comentado pelas *tags* ‘<!--‘ e ‘-->’. Para trocar a fonte usada basta retirar estas *tags* da parte de câmera (deixando somente as *tags* ‘<’ e ‘>’) e adicioná-las no trecho que começa e termina com ‘avi_file’ (este é o trecho que indica que será usada a entrada a partir de um vídeo (arquivo ‘.avi’).

3. Executando a aplicação exemplo “AirGuitarPro.exe”.

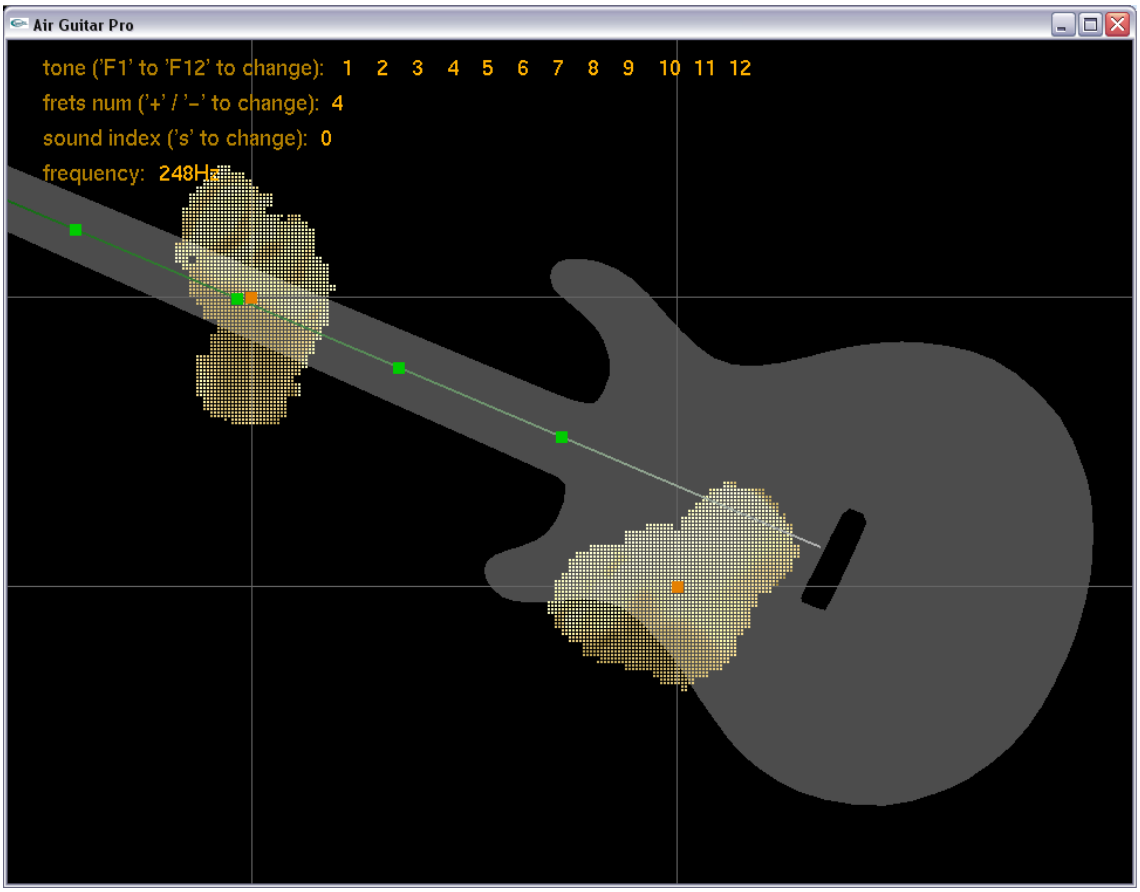

Figura 4. Interface da aplicação exemplo.

Na figura acima é possível ver a interface da aplicação exemplo. Esta aplicação possui algumas funcionalidades que podem ser alteradas em tempo real, que são:

O tom (tone) que pode ser alterado pressionando as teclas f1, f2, f3, ..., f12 do teclado, desta forma desativando e ativando os sons que podem ser tocados.

O número de trastes (frets num) que define a quantidade de divisões da guitarra virtual e pode ser alterado pressionado os botões ‘+’ para adicionar trastes e ‘-‘ para removê-los.

O índice do som a ser utilizado (sound index) pressionando a tecla ‘s’.

O índice do som irá indicar o arquivo de som que será reproduzido ao tocar a guitarra virtual. Este índice é usado com base no arquivo “.\AirGuitarPro\files\guitar\sounds.txt” que contém um lista de endereços (com base

na pasta ".\AirGuitarPro") para amostras de som. No topo deste arquivo existe um número indicando o índice máximo que pode ser usado, desta forma, a quantidade de endereços listados abaixo no mesmo arquivo tem que ser maior ou igual a este índice máximo e assim, quando o usuário selecionar o índice "0" na aplicação ele estará usando o primeiro som listado no arquivo "sounds.txt" como som base para sua guitarra virtual. O índice '1' indicará o próximo na lista e assim sucessivamente. Por este motivo não podem existir menos arquivos na lista do que o índice máximo, pois desta forma o aplicativo poderia tentar carregar um arquivo inexistente durante a execução. O formato do arquivo de som pode ser visto na figura abaixo.

Figura 5. Arquivo "sounds.txt".

4. Abrindo o *framework*.

Todo o framework de Air Guitar está escrito na linguagem C++ e foi codificado usando a ferramenta Microsoft Visual Studio 2005 (VS 2005) usando como Sistema Operacional (SO) o Windows XP. Existe uma solução do VS 2005 na pasta "AirGuitarPro"; basta abri-la e você poderá ver todo o framework em detalhes. O arquivo "main.cpp" contém a lógica da aplicação exemplo e é nele que está o método de entrada da aplicação, "main()". Além deste arquivo existem vários outros, todos divididos em seus respectivos repositórios, separados de acordo com os módulos e suas funções.

Para aqueles que usam outro ambiente de desenvolvimento, você precisará adicionar os arquivos ".cpp" e ".h" devidamente em uma solução/projeto correspondente ao seu ambiente de desenvolvimento. Além disso, lembre-se de copiar as bibliotecas dinâmicas ".dll" para a pasta correta no seu novo projeto e também de referenciar corretamente através da ferramenta que você usa todas as bibliotecas estáticas ".lib".

Caso você possua outro SO que não seja compatível com o Windows XP então provavelmente você terá um trabalho adicional de procurar por versões para o seu SO das mesmas bibliotecas usadas no framework (DSVideoLib, OpenGL, GLUT e FMOD Ex). Caso não existam tais versões então será necessário procurar por bibliotecas compatíveis que forneçam as mesmas funcionalidades e integrá-las ao framework.

Espero que aproveitem!