Luciano Rodrigues Lucio Neto

# Desenvolvimento de um *drum pad* usando visão artificial

Brasil

4 de junho de 2019

Luciano Rodrigues Lucio Neto

Trabalho de Conclusão de Curso Submetido à Coordenação do Curso de Engenharia de Computação e Automação do Centro de Tecnologia da Universidade Federal do Rio Grande do Norte, como parte dos requisitos necessários para a obtenção do grau de Engenheiro de Computação.

Universidade Federal do Rio Grande do Norte - UFRN

Coordenação do Curso de Engenharia de Computação e Automação - DCA

Graduação em Engenharia de Computação

Orientador: Agostinho de Medeiros Brito Júnior

Brasil

4 de junho de 2019

**Resumo**

Apresenta o desenvolvimento de um *drum pad* usando visão artificial capaz de controlar sintetizadores musicais virtuais criando uma sequência de notas musicais em repetição. Instrumentos assim são muito usados por músicos amadores que precisam criar acompanhamentos de bateria ou baixo para suas composições e não dispõem de músicos auxiliares para fazê-lo. A ferramenta criada permite que, usando apenas uma webcam, uma folha de papel e software livre, um músico amador seja capaz de criar efeitos semelhantes aos de um drum pad físico desenhando ou sobrepondo pequenas fichas coloridas na folha de papel.

***Palavras-chaves:*** Drum pad, sequenciador, controlador midi, OpenCV, Visão artificial

**Abstract**

*Introduces the development of a drum pad utilizing artificial vision capabçe of controling virtual synthesizers creating a sequence of musical notes on loop. Instruments like these are commonly used by amateur musicians artist who need to generate drums or bass support to their compositions and couldn't find other musicians to support it. The developed tool allows that with only a webcam, one paper sheet and open source software, an amateur musician artist can be able to replicate simmilar effects of a physical drum pad while drawing or overlapping small tokens over the paper sheet.* ***Keywords:*** Drum pad, sequencer, midi controller, openCv, artificial vision

# Lista de Figuras

# Sumário

# Capítulo 1

# Introdução

Um *drum pad* é um periférico utilizados por diferentes segmentos da sociedade de diferentes formas. Sua principal função é reproduzir sons escolhidos pelo usuário ao apertar de botões de sua interface e, por isso, é comumente utilizado principalmente pela comunidade de músicos sendo eles profissionais ou amadores. Utilizado também como uma forma de se programar uma sequência de sons que será reproduzida em um intervalo de tempo específico, um *drum pad* serve como acompanhamento musical com diversas aplicações sejam elas para guiar um músico, acompanhar o ritmo com alguma percussão ou até reproduzir detalhes específicos que o usuário pode não conseguir no momento correto.

Apesar de ser um periférico de fácil utilização e de rápida adaptabilidade, ainda depende da capacidade do usuário em reproduzir as notas no tempo correto e principalmente da capacidade que o usuário tem em investir num equipamento do tipo, variando de cerca de 50 dólares, podendo chegar a mais de 600 dólares, dependendo do seu tamanho, sua funções, caracterísitcas, inovações e acabamento.

A proposta do projeto apresentado neste trabalho de conclusão de curso é de facilitar a programação de uma sequência de sons que um usuário venha querer utilizar para qualquer propósito que seja, principalmente os descritos anteriormente. Com apenas uma webcam de cerca de 60 dólares, uma folha de papel e um programa de computador que utiliza conceitos de visão artificial, o usuário é capaz de programar, de forma interativa, uma sequência de sons provenientes de um sintetizador midi, que será reproduzida em um tempo determinado sem depender, por exemplo, de suas próprias habilidades rítmicas, a partir de marcações feitas na

folha.

No capítulo um será apresentado o modelo utilizado como referência para se utilizar o software apresentado neste trabalho, no capítulo seguinte o algoritmo que o software utiliza para reproduzir sons de acordo com o modelo. O capítulo seguinte explica conceitos básicos do Protocolo MIDI utilizados no desenvolvimento do software. O capítulo quatro apresenta os marcadores utilizados no modelo dde folha de papel, elementos cruciais para o funcionamento do software„ enquanto o capítulo seguinte explica como, a partir desses marcadores, o software aplica uma transformada de perspectiva da matriz da imagem caputrada pela câmera. No capítulo sete é explicado como se utilizar o software. No próximo capítulo estão conceitos da biblioteca RtMidi, utilizada para haver comunicação midi a partir do protocolo apresentado no capítulo três, enquanto os próximos dois capítulos discorrem, respectivamente, sobre resultados do software e conclusões tiradas ao final do desenvolvimento do projeto.

# Capítulo 2

# Modelo para Interpretação

Como muitas aplicações que envolvam conceitos de visão artificial, é necessário um ambiente controlado de onde se possa extrair as informações necessárias para o funcionamento do programa de computador. Para isso, foi necessária a criação de um *layout* com elementos específicos que façam com que o ambiente seja bem interpretado pelo programa.

IMAGEM DO LAYOUT

As principais características do *layout* para que haja a interação com o programa de computador são os marcadores nos cantos da folha e o código de barras. Tendo em vista que, nesse modelo, sempre haverão 13 notas distintas que podem ser reproduzidas, o código de barras é utilizado para guardar a informação de quantas notas o programa pode reproduzir e sua fórmula de compasso, sem haver a necessidade do usuário especificá-las já que não faria sentido o modelo ter, por exemplo, uma área quadriculada de 13x8 e o usuário configurar que o essa área tem uma distribuição diferente da real ou uma fórmula de compasso que não corresponda ao modelo. Além do código de barras, são utilizados marcadores nos cantos da folha que, ao serem interpretados, mostram a imagem final que o programa utilizará como fonte de interpretação.

# Capítulo 3

# Algoritmo

Partindo da imagem capturada da câmera do usuário, o programa procura os marcadores dos cantos da folha, o código de barras e os interpreta guardando as informações necessárias. Essas verificações acontecem a cada frame capturado pela câmera para manter a integridade com a imagem de tempo real. Com pontos capturados a partir dos marcadores, é selecionada a área de interesse da folha que o programa deverá interpretar ao sinal do usuário.

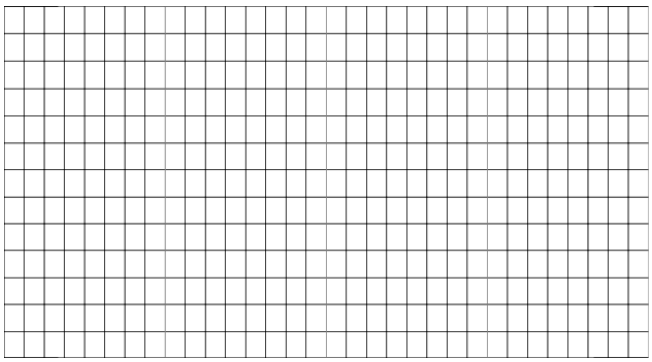
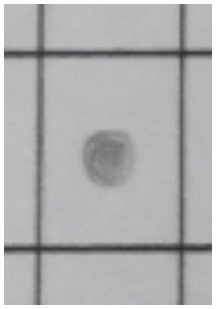


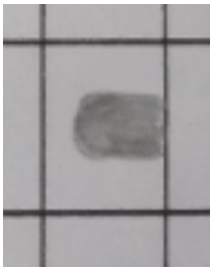
Figura 3.1: Área central de uma folha

Com a informação de quantas colunas existem na área quadriculada, adquirida do código de barras presente na folha, a próxima etapa executada pelo algoritmo é de verficiar, em cada um dos retângulos da área de interesse, se há dois tipos de marcações diferentes: uma representa um som que só tocará naquele momento e a outra representa a continuidade do som, permitindo que ela seja reproduzido durante o tempo desejado pelo usuário. Foram escolhidas duas marcações

diferentes para o software identificar quando deve ocorrer cada uma das duas situações. O que identifica se a nota será ou não mantida, é haver ou não pixels marcados do meio da marcação original padrão até o final do quadrado identificado pelo programa.



(a) Marcação de nota



(b) Marcação de nota continuada

A partir das informações que o programa consegue da etapa anterior, ele envia mensagens MIDI, utilizando o protocolo MIDI, para se comunicar com uma fonte de áudio selecionada pelo usuário, a qual será responsável pela reprodução dos sons em sí. Isso tudo com um intervalo de tempo definido entre cada nota, devido à cálculos a partir da fórmula do compasso daquele modelo, extraída do código de barras da folha, e definida ao gerar o pdf a partir de um segundo programa criado para esse propósito.

Devido à falta de dinamicidade uma vez que é impossível alterar quantos retângulos existem na folha de papel após impressa, não foi possível utilizar a fórmula de compasso convencional na teoria musical, então foi definida uma convenção de que, para o software, existem dois valores: um que determina quantas batidas aquela folha comportará e outro que determina quantas notas serão tocadas em cada batida.

O cálculo feitos para se obter o tempo de coluna da folha é um simples cálculo utilizando a fórmula do compasso convencionada e o valor de batidas por minuto passado como parâmetro, pelo usuário, ao executar o software:

$$tempo = \frac{60}{bpm \times y} \tag{3.1}$$

Onde y representa o número de sons que serão reproduzidos a cada batida, ou seja, o denominador da fórmula do compasso convencionada.

A ordem de disposição das notas foi escolhida da maneira como mostrada na imagem (REFERENCIA A IMAGEM) pois é o mais próximo de como uma "pista"MIDI é representada

em softwares profissionais de gravação de músicas, denominados DAW (*Digital Audio Workstation*),quando o usuário utiliza recursos MIDI, como se pode ver na imagem abaixo:

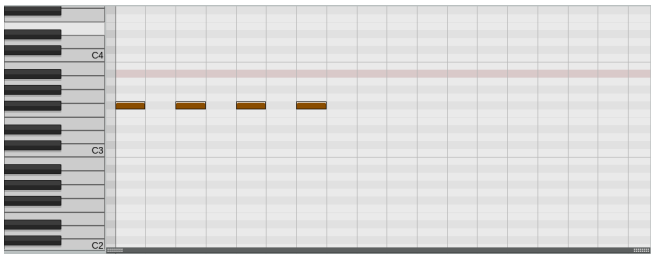
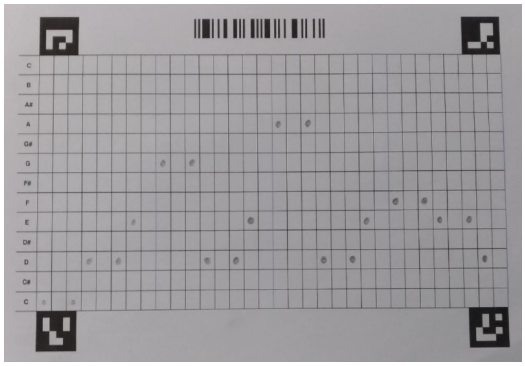
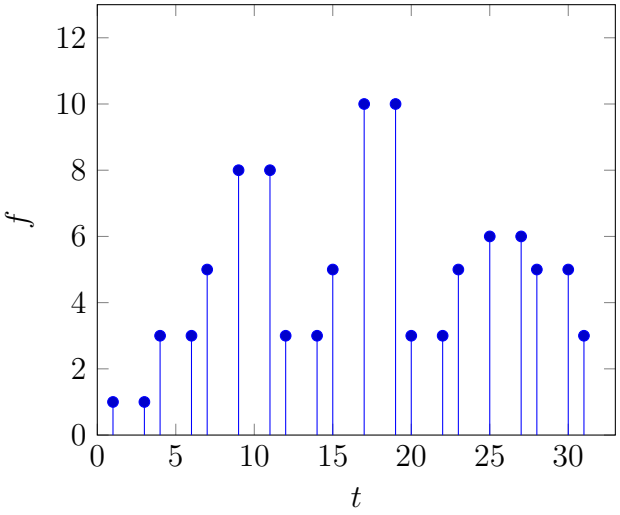



Figura 3.3: *Midi Item* na DAW Reaper

Também podemos interpretar o layout como um gráfico num simples plano cartesiano. O gráfico representa frequênciaXtempo, fazendo com que a posição da marcação represente uma frequência num intervalo de tempo. Essa interpretação pode ser representada da seguinte maneira:



(a) Folha com marcações




(b) Gráfico discreto

# Capítulo 4

# Protocolo MIDI

O Protocolo MIDI é o principal responsável pela reprodução dos sons de sintetizadores e sequenciadores MIDI e, sendo um protocolo, é universal, ou seja, independe do fabricante do sintetizador ou sequenciador uma vez seguido de maneira correta. Apesar de bastante extenso e complexo, serão explicadas aqui apenas conceitos que foram utilizados no desenvolvimento do projeto.

Esse protocolo consiste na troca de "mensagens MIDI"entre o software ou hardware controlador, que controla os sons, e do software ou hardware sintetizador, que sintetiza os sons. É utilizado para transmissão de informação em tempo real entre controlador e sintetizador a fim de gerar sons quando o usuário desejar.

Essas mensagens baseam-se em um conjunto de um ou mais *bytes*, onde o primeiro *byte* se classifica como *STATUS byte*, o qual define o tipo da mensagem, e é geralmente seguido por outros *bytes* chamados *DATA bytes*, os quais dão as características desejadas para a mensagem.

Existem diversos tipos de mensagens, mostradas na tabela (REFERENCIA TABELA) e de forma expandida na tabela (REFERENCIA TABELA), extraídas diretamente da *The MIDI Association* (www.midi.org), principal portal com informações do protocolo MIDI, porém, para o desenvolvimento do projeto descrito nesta tese de conclusão de curso, foram utilizadas bascicamente as mensagens chamadas *Note ON* e *Note OFF*, que especificam quando uma nota deve ser ativada e desativada. Outras mensagens utilizadas, que são mensagens de configuração do canal MIDI, são as mensagens *Control Change*, enviada para o sintetizador, no caso deste software, apenas na inicialização do programa, a qual pode designar diversas ações diferentes

para o sintetizador, porém, no software apresentado neste trabalho, ela é utilizada apenas para controle de volume e a mensagem *Program Change*, a qual seleciona o identificador do instrumento que o sintetizador irá simular, enviada, neste software, assim como a Control Change, apenas na sua inicialização.

As mensagens de ativação e desativação do som são compostas por um conjunto de 3 *bytes* os quais são representados da seguinte forma:

- A mensagem *Note ON*:
  - *Status byte*: 1001 nnnn
  - *Data byte*: 0kkk kkkk
  - *Data byte*: 0vvv vvvv

Onde nnnn representa o canal que esta mensagem controlará, variando entre 16 canais distintos, kkkkkkk representa o *pitch* que será reproduzido e vvvvvvv representa a dinâmica com a qual a nota será reproduzida que, em notação musical, são representadas da seguinte maneira:

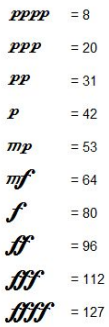



Figura 4.1: Dinâmica em notação musical e seus respectivos valores para a mensagem *Note On*

Essa velocidade de reprodução, dependendo do tipo de instrumento selecionado, também varia a dinâmica que o som é reproduzido como quando uma nota de um piano é pressionada de forma brusca ou mais suave.

- A mensagem *Note OFF*:
  - *Status byte*: 1000 nnnn
  - *Data byte*: 0kkk kkkk

  - *Data byte*: 0vvv vvvv

Para as mensagens *Note OFF*, os *bits* nnnn e kkkkkkk representam o mesmo da mensagem de ativação da nota sendo diferente apenas no terceiro *byte*, o vvvvvvv, que representa a velocidade de *release* do som, ou seja, a suavização até ele ser desligado.

A mensagem de alteração de valores dos controladores é representada da seguinte maneira:

- A mensagem *Control Change*:
  - *Status byte*: 1000 nnnn
  - *Data byte*: 0ccc cccc
  - *Data byte*: 0vvv vvvv

Onde nnnn representa o canal para o qual a mensagem será enviada, da mesma forma das mensagens de ativação e desativação do som, ccccccc representa o controlador que terá o seu valor alterado e vvvvvvv representa o valor que será atribuído para o controlador. Os controladores controlados a partir dessas mensagens podem variar entre 128 controladores destintos sendo eles os seguintes:

Figura 4.2: Controladores controlados a partir da mensagem *Control Change*

Com essa mensagem, é possível variar, em cada canal, desde variáveis de volume, até variáveis de modulação e efeitos que caracterização como o som será reproduzido pelo sintetizador.

A última mensagem utilizada no software, a *Program Change*, é representada da seguinte maneira:

- A mensagem *Control Change*:
  - *Status byte*: 1100 nnnn
  - *Data byte*: 0ppp pppp

Onde nnnn representa o número do canal para o qual a mensagem será enviada e ppppppp representa o identificador do instrumento que será selecionado.

# Capítulo 5

# Marcadores ArUco

A funcionalidade do software projetado nesta tese depende intrinsecamente da AOI (*area of interest*) do *layout* e, para se adquirir essa área de interesse da imagem inicial, uma das soluções mais comuns é a utilização de marcadores fiduciais quadrados binários. O principal benefício em se utilizar marcadores do tipo é de se obter, a partir dos quatro cantos de um único marcador, a *camera pose*, ou seja, tanto a orientação quanto a posição da câmera no ambiente.

No desenvolvimento deste projeto foi escolhido o módulo Aruco da biblioteca de visão artificial OpenCv para geração e identificação desses marcadores.O módulo Aruco é baseado na biblioteca ArUco, uma popular biblioteca utilizada para detecção de marcadores fiduciais quadrados, desenvolvida por Rafael Muñoz e Sergio Garrido, em aplicações de realidade aumentada.

A partir do módulo Aruco, pode-se gerar marcadores ArUco que consistem em marcadores quadrados compostos por uma larga borda da cor preta e uma matriz binária intríseca ao marcador, que determina seu identificador. As bordas existem para facilitar a detecção do marcador em uma imagem e a codificação da matriz binária para permitir a detecção desse marcador com intuíto de serem aplicadas outras técnicas de visão artifial. Abaixo temos alguns exemplos de marcadores ArUco:

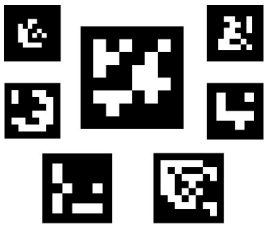


Figura 5.1: Exemplos de marcadores ArUco

Com uma detecção rápida e precisa desses marcadores, feita pela biblioteca OpenCv, foram escolhidos quatro identificadores que simbolizam os cantos da folha modelo e o software, ao decodificar os quatro marcadores, dependendo do seu identificador, são selecionados 4 pontos, um de cada marcador, a fim de adquirir a área de interesse da folha para o software.

Com a área de interesse adquirida, é aplicada uma transformação de perspectiva nessa nova imagem a fim de desprezar distorções causadas pelo posicionamento da câmera, uma vez que esta nova imagem que será analizada pelo software precisa ser paralela à folha, o que é possível a partir de diferentes ângulos, isso se os quatro marcadores forem visíveis, graças a transformação de perspectiva, fazendo o usuário manusear o software de forma mais confortável.

Outra forma de identificação que foi testada no projeto foi por cores em círculos coloridos posicionados no canto da folha. Devido à diversas variáveis de ambiente, essas cores poderiam variar de usuário para usuário, local para local, câmera para câmera entre outros, o que tornava necessária a marcação das cores ser feita manualmente pelo usuário com o propósito de haver um cálculo da média da cores dos círculos para sua identificação e, devido a essa inconsistência ao utilizar cores, a ideia foi descartada. O modelo anterior pode ser visto abaixo:

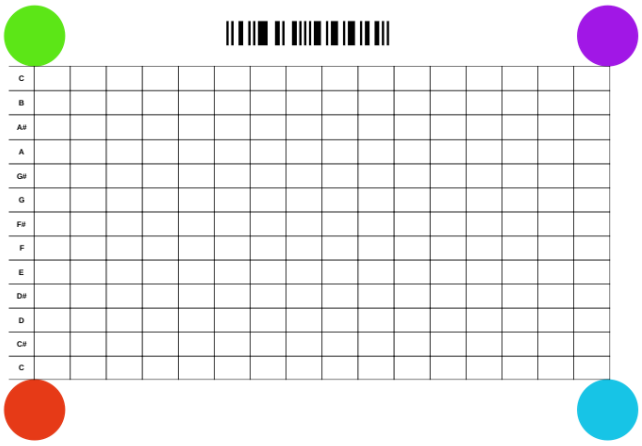



Figura 5.2: Modelo com círculos

# Capítulo 6

# Transformação de Perspectiva

Para o software interpretar a imagem de forma correta, é necessário que ela seja paralela à câmera, tornando desconfortável a sua utilização. Uma maneira de solucionar esse problema foi utilizar uma transformada de perspectiva, desprezando os problemas gerados para imagens capturadas de posições que não sejam paralelas à câmera que está capturando o ambiente.

Graças à biblioteca OpenCv, essa manipulação é de rápida e fácil aplicação, precisando-se apenas de quatro pontos de referência na imagem original e uma nova matriz imagem resultado.

A partir dos quatro pontos da imagem original, aplica-se a função getPerspectiveTransform, que a partir da fórmula mostrada abaixo, gera uma matriz de transformação de perspectiva 3x3:

$$\begin{bmatrix} t_i x'_i \\ t_i y'_i \\ t_i \end{bmatrix} = MapMatrix \times \begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix} \tag{6.1}$$

$$dst(i) = (x'_i, y'_i), src(i) = (x_i, y_i) \tag{6.2}$$

Onde dst representa a matriz de destino e src a matriz de origem.

Após obtida a matriz de tranformação 3x3, aplica-se a função warpPerspective, disponibilizada pelo OpenCv, que aplica a transformada de perspectiva na imagem original utilizando uma combinação de métodos de interpolação do OpenCv ($INTER_LINEAR$ ou $INTER_NEAREST$) passada como parâmetro. Para se obter a imagem transformada, o seguinte cálculo é aplicado:

$$dst(x,y) = src\left(\frac{M_{11}x + M_{12}y + M_{13}}{M_{31}x + M_{32}y + M_{33}}, \frac{M_{21}x + M_{22}y + M_{23}}{M_{31}x + M_{32}y + M_{33}}\right) \tag{6.3}$$

Onde M é a matriz de transformação 3x3 obtida a partir da função getPerspectiveTransform, dst é a matriz ou imagem de destino e src é a matriz ou imagem de origem.

Com essa transformação é possível desconsiderar as linhas que formam os retângulos na folha de papel, economizando processamento ao utilizar linhas virtuais calculadas de forma linear de acordo com as informações extraídas do código de barras da folha.

O resultado das opreções pode ser observado na imagem abaixo:

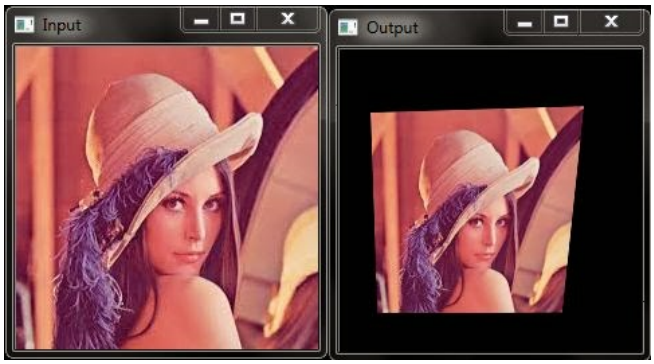



Figura 6.1: Resultado de uma transformação de perspectiva em uma imagem

# Capítulo 7

# Como Utilizar o Software

Ao se iniciar o software, é necessário passar três parâmetros iniciais na sua execução. O primeiro é o identificador da entrada de vídeo do usuário para a cena ser capturada com a câmetra correta caso haja mais de uma conectada.

O segundo é a velocidade de batidas por minuto (bpm) para o tempo de referência para reprodução das notas reproduzidas pelo software. Ao gerar o modelo da folha, o usuário deve passar os valores que ditaram a regra de velocidade de cada intervalo de tempo baseado no bpm informado pelo usuário. Devido à limitações de espaço, o máximo de retângulos que podem ser gerados para o bom funcionamento do software é de 46 retângulos. Um exemplo que faria com que o software apresentasse inconsistência pode ser encontrado abaixo:

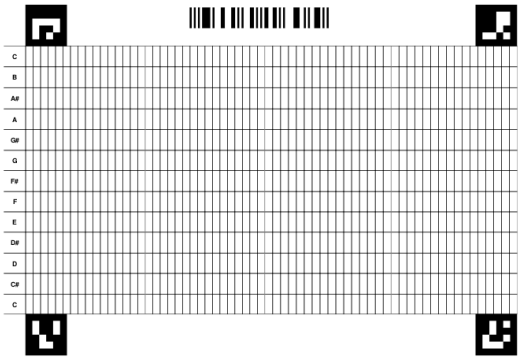


Figura 7.1: Folha com 4 batidas por folha e 16 notas por batida

Podemos ver que, dependendo do que o usuário definisse, a espaço livre para as marcações seria muito pequeno, prejudicando o funcionamento do software.

O terceiro é a oitava que será reproduzida no software. A oitava, na teoria musical, representa uma nota com o dobro ou metade de sua frequência. É quando a nota será, apesar de ter uma frequência maior ou menor que a nota de referência, a mesma e se chama oitava devido ao fato de, na representação mais comum das notas (dó, ré, mi, fá, sol, lá, sí), ela seria a oitava nota, que seria a mesma da primeira porém com uma frequência dobrada, ou seja: dó (oitava x), ré, mí, fá, sol, lá, sí e dó (oitava x+1), onde o último só e a oitava nota em relação ao primeiro.

Com esses paramêtros definidos, o software inicia-se e lista todas as entradas midi disponíveis no momento para o usuário escolher a que o software enviará as mensagens do protocolo. Os parâmetros iniciais informados pelo usuário ao iniciar o software são apenas para configuraão inicial do ambiente, porém, ao iniciar-se o software de maneira correta, são editáveis a partir do clique de teclas específicas informadas pelo programa.

Após essas configurações iniciais, é preciso se cumprir, assim como outros softwares que utilizam conceitos de visão artificial, certas condições para o funcionamento correto do software projetado neste trabalho. Primeiramente precisa-se de condições de ambiente favoráveis à captura das imagens da câmera como iluminação da cena, principalmente de forma que a folha não fique muito escura ou com sombreados escuros, pois o filtro de branco utilizado na área de interesse pode identificar essas sombras como marcações, o prejudica no funcionamento do software. Outra condição é uma angulação da câmera de forma que se consiga visualizar os quatro marcadores ArUco e o código de barras presentes na folha. Marcações serão mostradas no vídeo quando os cinco elementos forem identificados de forma correta. Abaixo temos a imagem capturada pela câmera com as devidas marcações indicando que as instruções iniciais foram seguidas como o desejado:

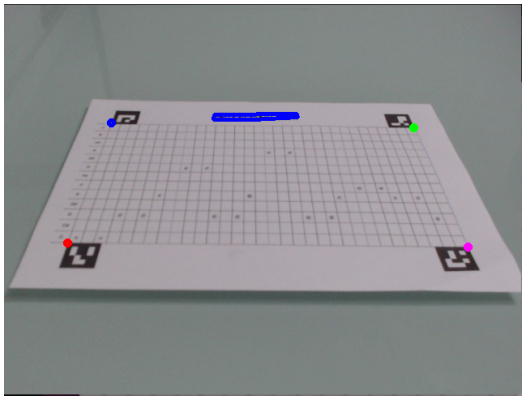


Figura 7.2: Imagem capturada pela câmera com marcações de reconhecimento

A partir desse ponto, pode-se utilizar a barra de espaço para o software tocar ou parar a sequência identificada na folha. Essa sequência pode ser livremente alterada pelo usuário diretamente na folha de papel e reproduzida novamente quando se desejar, bastando apenas que as condições já explicadas sejam sempre cumpridas antes de se informar ao software que ele deverá reproduzir a sequência da folha.

As marcações devem ser feitas sempre o mais centralizadas possível no retângulo que indica o espaço de tempo da sequência, sendo possível representar sons que vão continuar soando ou que vão parar no próximo intervalo de tempo com duas marcações distintas já mostradas no capítulo que destrincha o algoritmo utilizado.

# Capítulo 8

# Biblioteca RtMidi

Como já visto no capítulo (capítulo do protocolo midi), o sintetizador escolhido pelo usuário precisa receber as mensagens enviadas pelo software e foi com auxílio da biblioteca RtMidi para a linguagem de programação C++ que foi possível abstrair todo o fluxo de troca de mensagens em simples funções disponibilizadas pela biblioteca.

Para a utilização do protocolo MIDI e para a comunicação entre sintetizadores, foi escolhida a biblioteca RtMidi para a linguagem de programação C++. Toda a transmissão e criação das mensagens foi abstraída durante o desenvolvimento do projeto graças às funcionalidades da biblioteca e foi por esse motivo, além de ser Open Source, que ela foi escolhida para fazer parte do desenvolvimento do software.

A biblioteca consiste em duas classes chamadas RtMidiOut e RtMidiIn. A classe RtMidiOut proporciona, para o desenvolvedor, funcionalidades simples para a imediata troca de mensagens MIDI entre dispositivos virtuais ou não conectados através de uma conexão MIDI. Com essas funcionalidades foi possível ocorrer o envio de mensagens para o sintetizador selecionado pelo usuário como porta de entrada ou cliente para escrita da conexão midi com apenas o uso, após configurada essa entrada, de uma função responsável exclusivamente pelo envio dessas mensagens. A outra classe, RtMidiIn, apesar de não ter sido utilizada no software apresentado neste trabalho, utiliza uma função interna de callback ou thread para receber mensagens MIDI que são enviadas a partir de uma porta de saída MIDI selecionável e as lendo, transformando no formato utilizado pela biblioteca, ou repassando-a imediatamente para uma função de callback especificada pelo usuário, o que tornaria possível, por exemplo, o software reproduzir, num

sintetizador digital, o que foi enviado a partir de um controlador físico conectado à máquina do usuário.

Para o testes do software, foram utilizados 3 softwares de terceiros, sendo eles o QSynth, sintetizador virtual MIDI, o VMPK, outro sintetizador virtual MIDI mas também controlador, e o QJackCtl, um software que permite o usuário administrar as conexões MIDI entre saídas e entradas MIDI disponíveis na máquina.

Abaixo pode-se visualizar como se dá a comunicação dos softwares controlador e sintetizador:

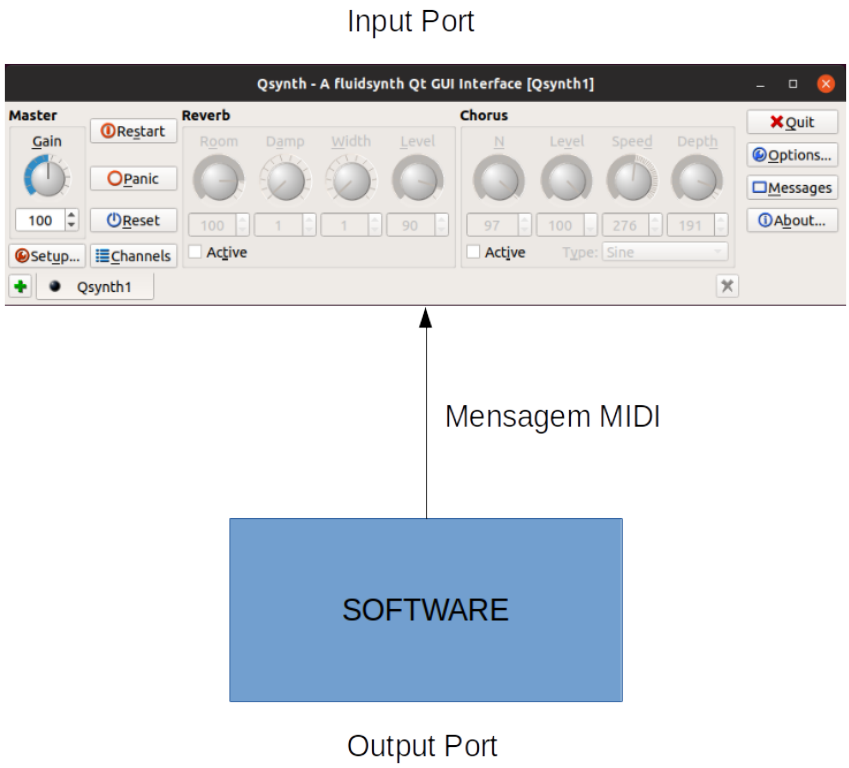



Figura 8.1: Diagrama de fluxo entrada e saída MIDI

Onde as portas de saída são os controladores e as de entrada são os sintetizadores.

# Capítulo 9

# Resultados

Quando em ambiente bem iluminada, ângulação da câmera corretamente ajustada e marcações devidamente adicionadas à folha, o software apresenta ótimos resultados em conjunto com softwares sintetizadores que seguem o Protocolo MIDI, pecando apenas na temporização das notas devido ao algoritmo utilizado.

Também é possível utilizar o software como um controlador midi em uma DAW, porém, da forma como ele está no momento da escrita desde documento, ainda apresenta inconsistências quando utilizada dessa forma. A DAW utilizada para se obterem os resultados chama-se Reaper e é gratuita.

Pode-se considerar, de certa forma, cada folha de papel como um compasso de uma partitura musical, o torna possível a composição não de músicas completas, mas sim de trechos e ideias de músicas, e o usuário pode guarda-lás assim como se faz com partituras, tabalturas e gravações, porém com a vantagem de alterar o tempo e o instrumento a cada vez que reproduzir a informação contida na folha de papel.

# Capítulo 10

# Conclusões

O software desenvolvido consiste num algoritmo criado especificamente para o que o software pretende entregar para o usuário. Utilizando de conceitos de visão artificial e de comunicação MIDI para interpretar informaçoes e elementos, a partir de uma folha A4 impressa com um layout espefícico e preenchida de acordo com o que o usuário pretende que seja reproduzido, e enviando mensagens MIDI para um segundo software sintetizador escolhido elo usuário, que será o responsável por reproduzir as notas desejadas e encontradas na folha de papel.

Durante o desenvolvimento, foram precisos armazenar informações no modelo da folha de papel e se identificar pontos distintos para haver uma transformada de perspectiva na imagem original capturada pela câmera e, para esses dois quesitos, foram utilizados como solução um código de barras para o armazenamento das informações e marcadores ArUco para o software determinar corretamente a área de maior interesse da folha.

Devido a reprodução e repetição do algoritmo não otimizado de maneira correta, o software acaba tendo problemas com relação à temporização de cada nota, para a qual foi convencionado um substituto da fórmula do compasso da teoria musical para ser aplicado um cálculo de tempo que cada nota deve ser mantida ativada. Uma solução seria a aplicação de um vetor de mensagens MIDI que seria preenchido no momento que o software interpretasse folha de papel da primeira vez e com isso, ao invés do software procurar informação em cada retângulo da folha a todo momento, o que gera *overhead* considerável, só aconteceria a leitura desse vetor onde foram armazenadas as mensagens MIDI, tornando o programa mais veloz e provavelmente solucionando o problema da temporização de cada som e adicionando também a possibilidade

de serem criados diferentes vetores a partir de diferentes folhas de papel, tornando possível o usuário reproduzir mais de uma folha de papel em sequência.

O software utiliza interfaces para o usuário disponibilizadas pela biblioteca de visão aritificial OpenCv, porém essas interfaces não são modernas ou intuitivas, prejudicando a experiência do usuário final. Outra adição ao programa seria criar uma interface visual intuitiva e moderna com as opções de se selecionar a oitava, velocidade, volume, instrumento que o sintetizador deverá simular, criar um novo pdf a partir de parâmetros definidos do usuário diretamente da aplicação e não utilizando um segundo programa e uma visualização, no caso da adição da funcionalidade dos vetores apresentada no parágrafo anterior, de cada folha de papel armazenada e uma linha do tempo de qual folha está sendo reproduzida no momento caso o usuário tenha escolhido reproduzir mais de uma folha em sequência.