Projeto de Graduação

DEPARTAMENTO DE ENGENHARIA ELÉTRICA

07 de dezembro de 2022

Controle Robótico Baseado em Processamento de Áudio Musical

Otto Vinicius da Silva Rodrigues

PUC RIO

Projeto de Graduação

DEPARTAMENTO
DE ENGENHARIA
ELÉTRICA

Controle Robótico Baseado em Processamento de Áudio Musical

Aluno(s): Otto Vinicius da Silva Rodrigues

Orientador(es): Jan Krueger Siqueira

Trabalho apresentado com requisito parcial à conclusão do curso de Engenharia de Controle e Automação na Pontifícia Universidade Católica do Rio de Janeiro, Rio de Janeiro, Brasil.

Agradecimentos

Há doze anos, para ser mais específico, dia 09 de janeiro de 2010 eu estava dando uma entrevista para o Jornal O Globo pois havia ganhado minha primeira medalha na Olimpíada Brasileira de Astronomia e Aeronáutica, sim ganharia a segunda no ano seguinte. Tendo quinze anos, todo o futuro acadêmico não passava de sonho mas mesmo assim me perguntaram sobre meu desejo de profissão e eu respondi o seguinte:

"Ainda tenho um bom tempo para escolher uma carreira, mas já sei que trabalharei com tecnologia. Acho robótica fascinante."

Hoje, dia 07 de dezembro de 2022 concluo minha graduação em Engenharia de Controle e Automação projetando um robô. Porém não é apenas qualquer robô, sua principal característica é interagir com música, minha paixão mais antiga. Sei que não posso conversar com meu eu do passado e mostrar o que iríamos acabar fazendo mas tenho certeza que ele ficaria tão orgulhoso e incrédulo quanto eu estou agora. Sei disso porque no fundo ainda sou o mesmo Otto de doze anos atrás.

Então além de agradecer a mim mesmo da década passada gostaria também queria guardar um pedaço desse documento para mencionar o apoio que eu recebi do Jan, meu orientador. Apoio esse que foi além da área técnica, me deixando livre para tornar meu sonho realidade.

Resumo

O escopo do projeto é criar um robô, através de um microcontrolador Arduino, que tenha reações motoras a partir de sinais de áudio. Por via de regra será escolhida uma música através de um player. Consequentemente esta música passará por um processo de decomposição onde será isolada a voz e a percussão. A partir destes dois canais será possível gerar todos os momentos de canto e o ritmo, fazendo com que o robô simule uma dança e cante simultaneamente. Todos os processos envolvendo software serão desenvolvidos em Python e terá uma interface gráfica de um player, onde será possível escolher as músicas e visualizar os dados do processamento.

O robô será modelado no formato humanoide, tendo dois braços, duas pernas, tronco e cabeça. No que se diz respeito ao movimento, seis servos motores alocados em dupla (pernas, braço e mandíbula) garantirão a atividade. Além da simulação de canto e dança, os olhos de LED indicarão diferentes sentimentos de acordo com a voz.

Palavras-chave: Robô; Python; Arduino

Robot Control Based on Musical Audio Processing

Abstract

The scope of the project is to create a robot, using an Arduino microcontroller, that has motor reactions from audio signals. As a rule, a song will be chosen from a player. Consequently, this music will go through a decomposition process where the voice and percussion will be isolated. From these two channels it will be possible to generate all the singing moments and the rhythm, making the robot simulate a dance and sing simultaneously. All the processes involving software will be developed in Python and will have a graphic interface of a player, where it will be possible to choose the songs and visualize the processing data.

The robot will be modeled in humanoid format, having two arms, two legs, a trunk, and a head. As far as movement is concerned, six motor servos allocated in pairs (legs, arm, and jaw) will guarantee the activity. In addition to the singing and dancing simulation, LED eyes will indicate different feelings according to the voice.

Keywords: Robot; Python; Arduino

Sumário

1. Software

Para o desenvolvimento do projeto foi utilizado o **Python** como linguagem de programação, uma vez que é de fácil entendimento, extremamente versátil com bibliotecas eficientes, notadamente utilizada para, entre outras tarefas, desenvolvimento de software, criação de jogos, ambientes e automação em todos os níveis, desde processos e residencial até as aplicáveis em inteligência artificial, que é o caso.

Em linhas gerais a aplicabilidade deste software, no projeto, tem como primeiro objetivo ser um **player** que possibilitasse a importação e manipulação de músicas. Em conjunto com a manipulação, o Python também daria espaço para todo o desenvolvimento do processamento dos sinais vindo do áudio. E não menos importante, a programação nessa fase também tem o papel de trocar informação via serial os dados do Python vindo da decomposição do áudio com o Arduino.

Em resumo, o software é o segmento decisivo para todo o projeto, não é coincidência que mais tempo foi alocado para esse fragmento. Além disso, programar um software mais organizado e otimizado possível trará lucros no futuro, já que toda programação do lado do hardware e a intercomunicação será exponencialmente mais fluida e direta.

Para alcançar uma estrutura mais coerente e descomplicada, será seccionado em subtópicos todo o processo de desenvolvimento. Começando com o player, dando segmento para o processamento e a intercomunicação.

a. Player

O player tem como o principal objetivo ser o canal de interatividade entre o usuário e o código, sendo ele o resultado do projeto, o **robô.** Para isso, é recomendável que cumpra alguns requisitos: Ter uma interface intuitiva e de fácil compreensão onde seja possível acessar às músicas do computador, realizar a sua reprodução de maneira eficiente e por fim ser possível visualizar e analisar todos os dados vindo do processamento dos sinais simultaneamente.

Para isso foi usada, majoritariamente, uma biblioteca do Python chamada Tkinter que é focada em criar interfaces gráficas, ou seja, a GUI (Graphical User Interface). A partir da GUI foi feita toda parte visual do player, como botões, textos, o slider da música e os medidores de dados.

Um obstáculo percebido foi que existem limitações no Tkinter, precisamente na manipulação de áudio. Portanto para superar esse problema foi usado outro pacote, o **PyGame**. Este pacote possui ferramentas específicas para manipulação de músicas, o *pygame.mixer*, onde foi possível dar play, pause, e gerar a duração da faixa e conseguir desenvolver o player por completo.

b. Processamento

Tendo a música em mãos já é possível embarcar na fase do processamento musical. Porém, antes de programar é preciso planejar e organizar o que queremos extrair da música.

No nível mais baixo de complexibilidade temos que fazer o robô dançar, e para isso é de extrema necessidade ouvir e compreender a batida da música, já que é ela que dita o ritmo. Ao mesmo tempo, para ser possível que o robô cante teremos que processar a voz, passo esse de maior complexibilidade e detalhes.

A voz terá dupla funcionalidade no robô: Primeiro, fazer o movimento do 'cantar', e para isso teremos que saber quando existe voz ou não. Dando continuidade, teremos que nos ater ao ato de 'falar'. Nesse passo, será necessário que a boca faça o movimento sincronizado com o artista, de abrir e fechar. Esse movimento será dividido em posições diferentes sendo a primeira apenas um pouco aberta, voz baixa, e a última posição totalmente esticada quando houver gritos.

Outro ponto importante são os olhos do robô. Mais uma vez precisaremos sincronizar com o sinal vindo da voz, ou seja, será o mesmo start do movimento de cantar. Nesse momento a complexidade está em "combinar" voz, movimento corporal e "sentimento".

A ideia foi mostrar a diversidade de "emoções" através do "olhar" do robô. Quando não houver voz na música, ou seja, quando apenas houver a parte instrumental os olhos terão uma expressão "neutra". Contudo, logo que a parte cantada começa, o olhar se torna mais "agressivo" em consonância com a proposta do ritmo da música em questão.

Partindo para a parte prática do processamento, serão necessárias duas bibliotecas para possibilitar todos os passos do planejamento: Spleeter e Librosa.

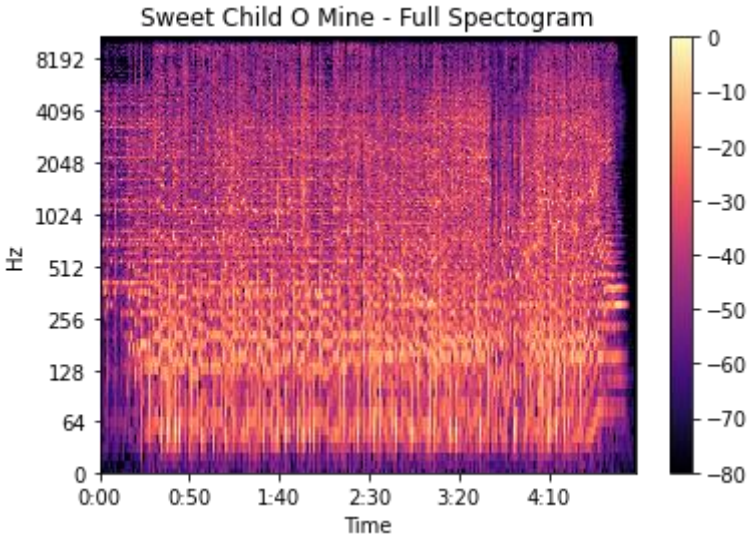
Spleeter é uma biblioteca do Deezer em Python que usa redes neurais pré treinadas para separação em canais de uma música, a partir do Tensorflow. Para ser mais técnico o Spleeter é formado a partir de uma CNN (Rede Neural Convolucional) que em sua totalidade possui 12 camadas.

Para cada camada é usado uma máscara, conceito esse que tenta isolar um certo elemento do conteúdo total, nesse caso é em busca do isolamento de voz, percussão etc. Porém esse conceito também é amplamente utilizado em visão computacional e segmentação de imagens.

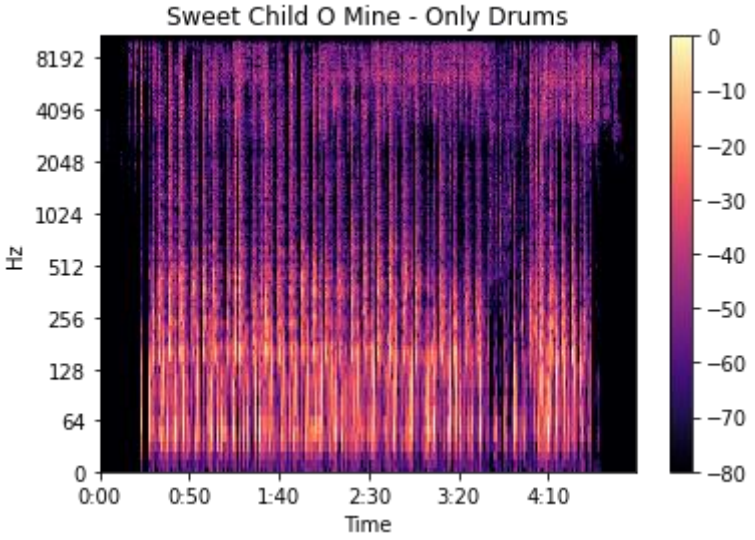
Em termos gerais, caso tenhamos uma música, o Spleeter é capaz de separar de dois em até cinco canais os elementos do áudio. Isso significa que teríamos a voz, a bateria, piano, baixo e harmonia isolados na hipótese de separação dos cinco canais. Caso quiséssemos apenas dois canais seria a voz e a harmonia. No caso deste projeto só foi necessário ter a voz e a batida separadamente.

O Librosa por sua vez é um pacote especializado em processamento de áudio e será vital para todo o projeto dar certo. Nele é possível extrair diversas informações desde o mais simples como duração, tempo e batida de uma música, até funções mais complexas como separação harmônica e percussiva. É importante ressaltar que esta biblioteca é super útil para visualização dos sinais, que será visto mais a frente. Abaixo será detalhado o processo visto até aqui:

Suponhamos que a música escolhida seja por exemplo Sweet Child O' Mine, do Guns N' Roses, que possui aproximadamente cinco minutos de duração. A partir do Librosa, com uma função que gera a Transformada de Fourier em tempo curto conseguimos chegar no seguinte espectrograma da música:



Como foi mencionado anteriormente usaremos o Spleeter para isolar tanto a voz quanto a bateria. Começando com a bateria temos o seguinte resultado:

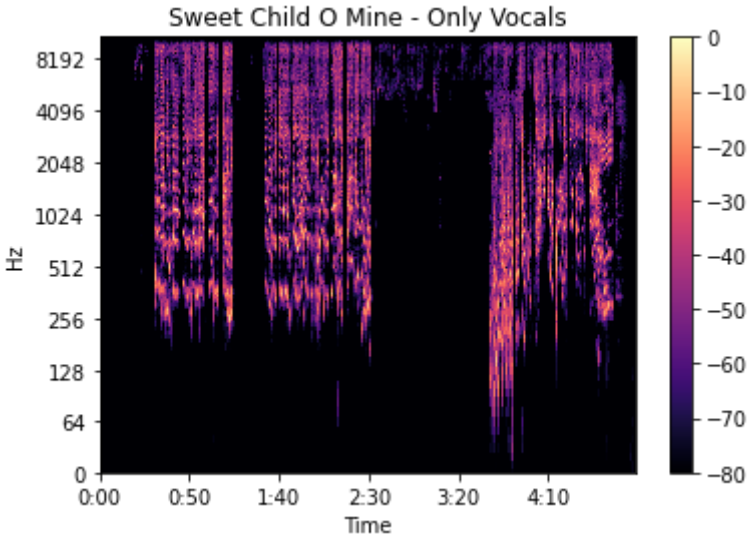


Apenas com essa imagem é possível compreender que a bateria faz parte do segmento vertical do espectro do áudio, sendo composto por linhas extremamente finas de potências altas (snare ou caixa), numa frequência praticamente linear, tirando o início e o final da música, obviamente.

Com isso conseguimos entender qual seria o processo caso quiséssemos o instante de tempo de cada 'beat'. Primeiro veremos qual o tempo da música (bpm), e se os instantes de maior força se repetem em um intervalo consistentemente próximo do bpm da música, assim teremos o tempo exato de cada batida. E já existe uma função do Librosa que faz exatamente esse processo, chamada de *beat_tracker*. O retorno desta função quando passamos este específico áudio e o bpm da música é o seguinte: [16.253968253968253, 17.182766439909297, …, 272.39328798185943]

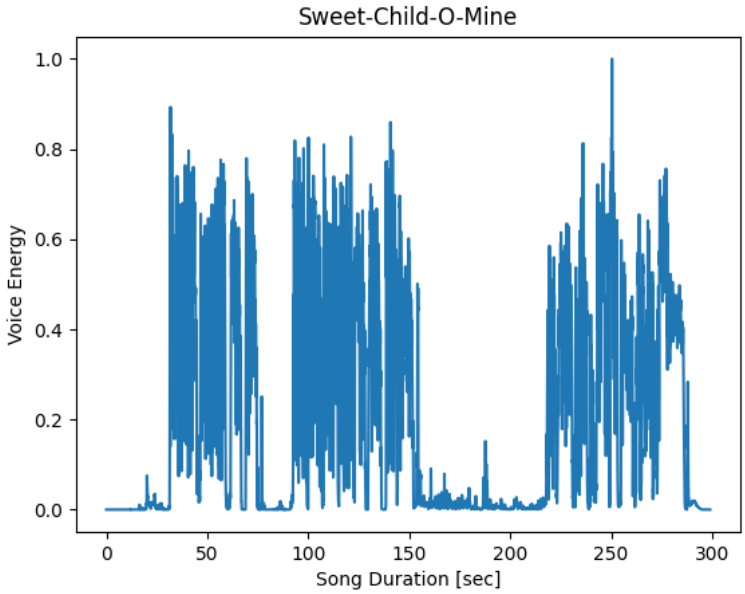
Onde cada elemento do vetor é o segundo exato onde está cada batida, em ordem. A partir deste vetor fica nítido o que precisa para fazer o robô dançar: Para cada momento indicado no vetor, o pé do robô mexe em um determinado ângulo.

Para a voz o processo se inicia praticamente igual. Com a ajuda do Spleeter é gerado o seguinte espectrograma:



Nessa ocasião podemos ver algo parecido com 'ondas' que se mantêm na potência horizontalmente, e faz sentido já que o timbre vocal não varia muito durante a música. Entretanto, não é possível fazer a mesma análise do processamento de batidas para a voz, já que o padrão é horizontal e não vertical, a favor do tempo.

Portanto o áudio será quebrado em pequenas janelas, e em cada uma dessa janela vai ser tirada a energia. E assim no final teremos um gráfico de energia da voz em função do tempo. Lembrando que para isso teremos que tirar a raiz quadrada média de cada janela, ou seja, rms (root mean square), que já existe pronto no Librosa novamente. O plot da energia fica desta maneira:

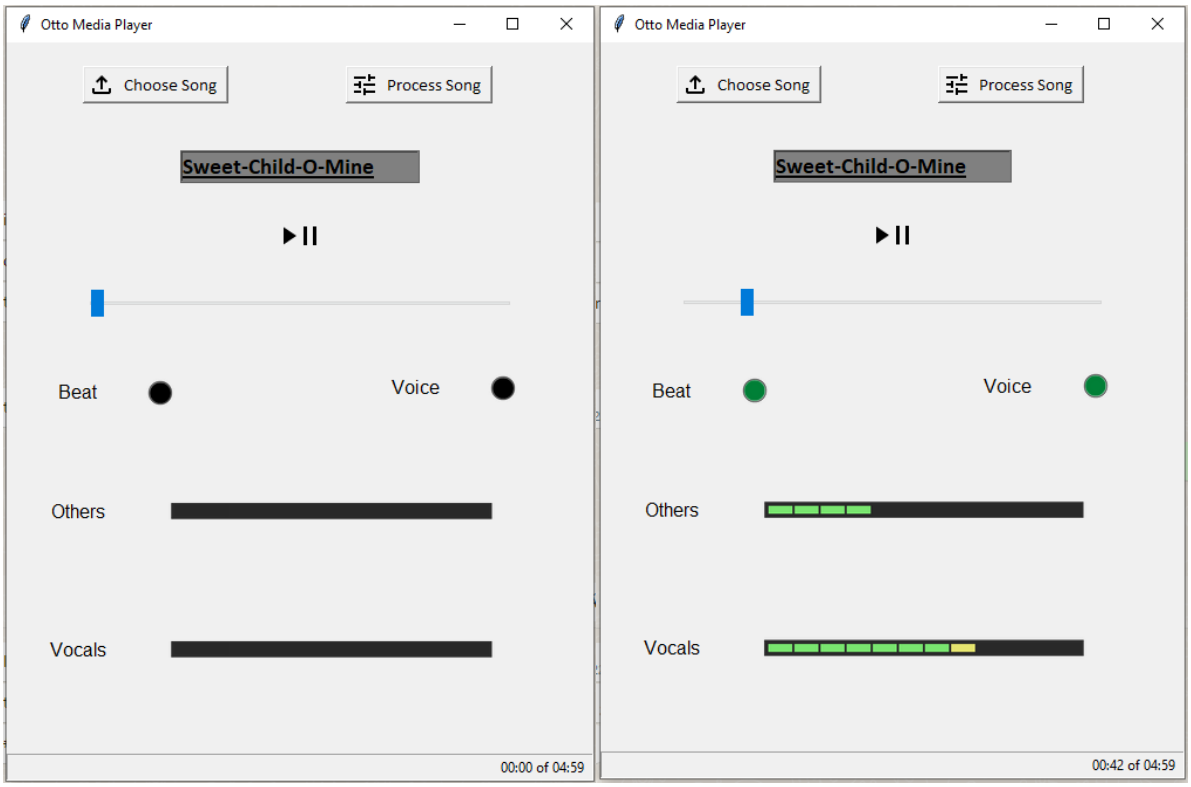


É visível a semelhança da energia em relação ao tempo nas duas imagens, porém nesta fica mais clara a compreensão. Tendo o resultado do rms, que é um vetor, podemos normalizar os valores entre zero e um. Então se em um determinado instante o valor do vetor for zero, não existe voz, caso for 1 a voz está com energia máxima.

Dessa forma já fica possível controlar a boca e o microfone. Para o microfone fica simples: Zero é sem voz, do resto existe voz e pode levantar o microfone.

No caso da boca não fica muito mais complexo, é preciso apenas mapear o valor da energia com os ângulos do servo. Por exemplo, se o servo começa em noventa graus e o ângulo máximo for cento e vinte, uma função afim simples resolve o problema. Lembrando que o olho é definido do mesmo jeito que o microfone, quando tem voz é gerado um olho na matriz de led 'mais expressivo', caso contrário o olho fica 'normal'.

Por fim foi necessário por todas estas informações no player, para ser possível inferir e analisar todo o processamento durante a música. Abaixo está sendo mostrado dois momentos diferentes da reprodução:



Na primeira imagem, a música não havia começado a tocar e isto faz com que todas as informações visuais sejam nulas. Já na próxima, no quadragésimo segundo segundo da música é possível ver como cada aspecto funciona: *Beat* e *Voice* são parâmetros binários, como foi dito anteriormente, ou é zero ou um; tem ou não tem. Já o *Others* e *Vocals* têm a visualização em formato de 'velocímetro', onde é possível ver a energia gerada naquele momento pela voz e pela harmonia. Vale a pena destacar que o *Others* foi posto no player para passos futuros, como definir momentos de solos.

Nesse interim, a dinâmica se torna a seguinte:

- Beat – Mexe servo dos pés
- Voice – Mexe servo do microfone + Muda emoção dos olhos
- Vocals – A partir da força do velocímetro a boca abre num determinado ângulo

Todo código pode ser encontrado no link: https://github.com/OttoRod/TCC

2. Hardware

Como mencionado no tópico de software, a partir do momento que tenhamos uma programação coerente e concisa, a parte de hardware se tornaria mais simples e descomplicada. Hipótese essa, que se provará verdadeira nos tópicos que virão.

Este segmento da proposta, de ter um robô que tenha reações motoras, se faz mais evidente já que será discutido não só a concepção do próprio robô, ou seja, o que foi pensado no momento de criar os modelos da peça, montagem e impressão; como também o lado do microcontrolador, notadamente, o Arduino.

Isto significa discutir quais componentes seriam necessários para compreender o escopo do projeto, assim como a integração dos mesmos componentes com o robô e o circuito gerado para fazer funcionar por completo a ideia inicial.

Antes de tudo, para se ter um robô é preciso saber primeiro como a parte visual será feita, isto é, o design de cada peça e do conjunto montado, e para isso será relevante abordar as referências.

Quando se fala em robô a primeira imagem que surgiu no campo das ideias foi a configuração clássica, amplamente presente em filmes, desenhos e cultura pop, como por exemplo a Rosie, robô doméstica dos Jetsons, ou então o porteiro do Castelo Rá Tim Bum ou o robô maluco que persegue o Pica Pau.

Em todos esses casos o robô tem características humanoides (cabeça, tronco e membros) e ao mesmo tempo possuem também traços de montagem e uma certa robustez, como parafusos a mostra e peças em sua maioria inspiradas em formas geométrica simples (cilindros e paralelepípedeos). Logo, foram essas características que nortearam o design do robô.

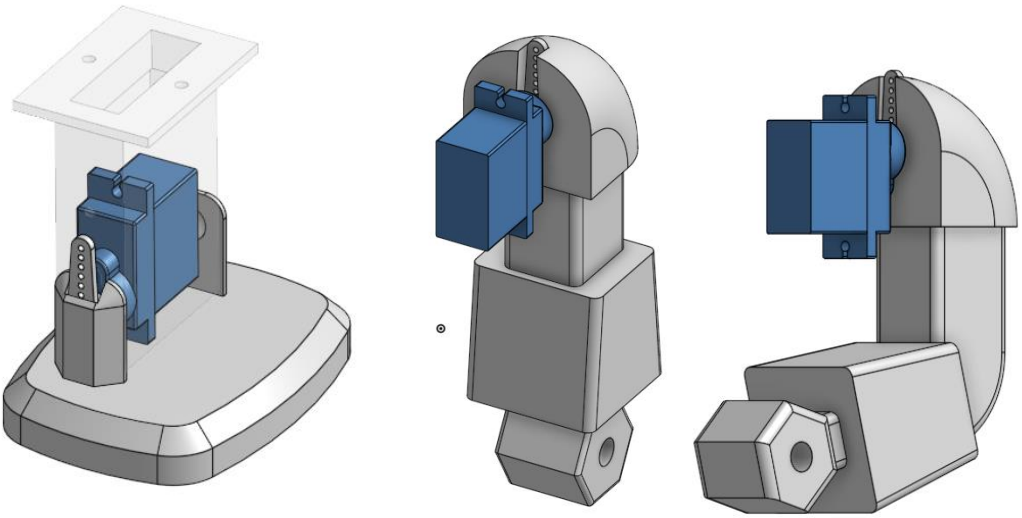
Outro ponto que também influenciou em como o robô se apresentaria foram os componentes. Aproveitando o adendo, serão listados os componentes usados neste projeto e como o design foi afetado em cada um deles:

a. Micro Servos SG90

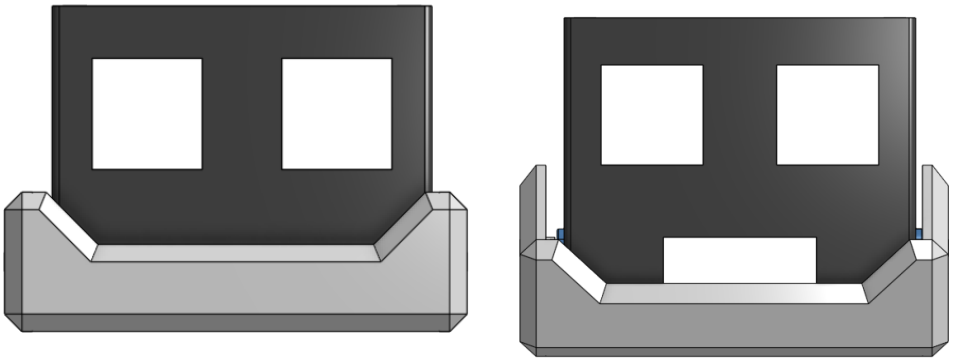
Nesse projeto em específico os servos eram responsáveis por toda movimentação do robô, então foram usadas seis unidades em pares, distribuídas da seguinte forma: Dois para o movimento da mandíbula, um para cada braço e por fim, um para cada perna.

Cada membro citado acima tem sua particularidade: Os membros inferiores do robô, por exemplo, são composto de duas peças: o pé e a perna. Essa configuração foi necessária para que o robô consiga realizar, com maior efetividade, o movimento de dança. É relevante dizer que o modo com que a perna se move teve inspiração em outro robô já existente, da marca Otto DIY.

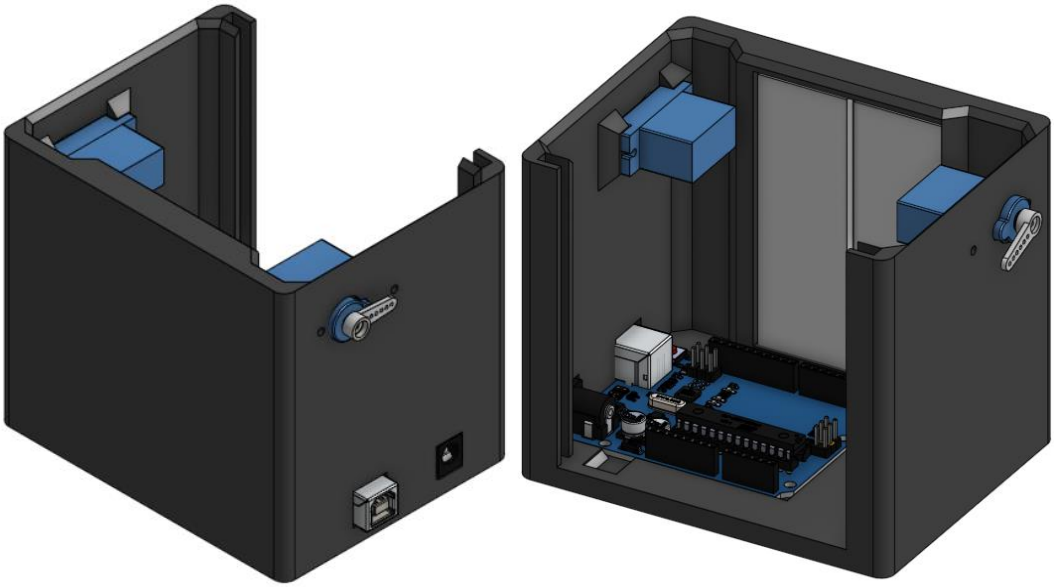
Os braços, por sua vez, são peças únicas, porém o direito é diferente do esquerdo. Para entendermos melhor vale a pena lembrar que o robô precisa cantar, e para isso um braço tem que se mover até a boca. Para alcançar esse movimento de maneira coerente, o braço que for segurar o microfone deverá ser desenhado de forma angular. Por outro lado, o braço que estiver livre deverá ser reto.



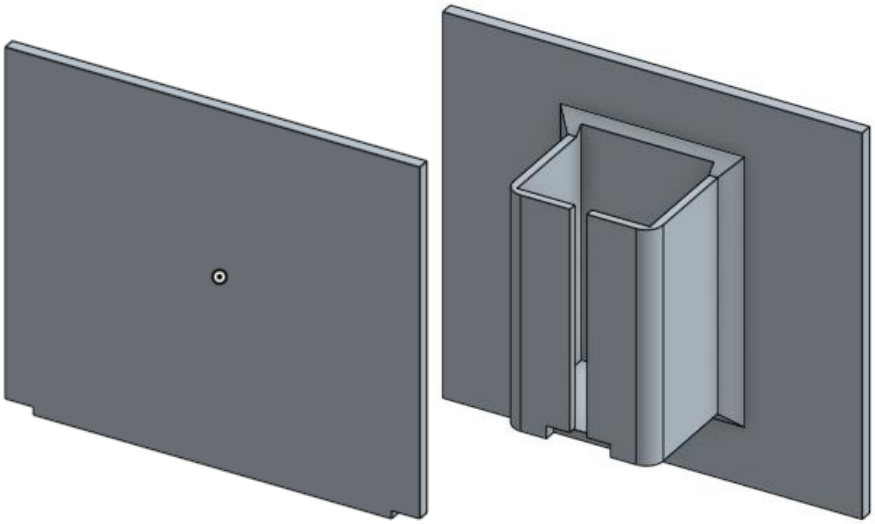
Para a cabeça, foram necessários dois servos, um para cada lado da mandíbula. Um ponto importante é que o design da boca foi um ponto de discussão, já que poderia ser apresentado de diversas formas. O modelo que melhor traduziu a proposta do projeto foi ter a boca sobreposta ao maxilar, como o robô do Pica Pau. Outro desenho que fizemos da boca foi com ela quadrada, na proporção de uma boca 'normal', mais inspirado no robô do Castelo Rá Tim Bum.



Para concluir, no que se trata de servo motores, é importante falar do corpo do robô. Sem dúvidas a peça mais complicada do projeto, já que é a principal, isto é, peça que liga cabeça e membros. Além de ter as conexões com os braços, pernas e crânio, o corpo também abriga o microcontrolador, Arduino. Além disso, é necessário conter uma placa protoboard para se fazer possível conectar todos os fios. Falando em fio, o corpo foi desenhado e elaborado de forma a ter um tamanho apropriado para caber todos os cabos com folga, possibilitando um alcance razoável para manutenção e montagem por parte do engenheiro.



Outro ponto da montagem que se faz relevante neste momento é que tanto o corpo como a cabeça possuem tampas deslizantes na parte de trás para permitir acesso ao circuito e componentes. Porém a tampa do corpo possui uma espécie de "mochila" para poder acoplar uma pilha de 9V, que serve como fonte externa para o Arduino.

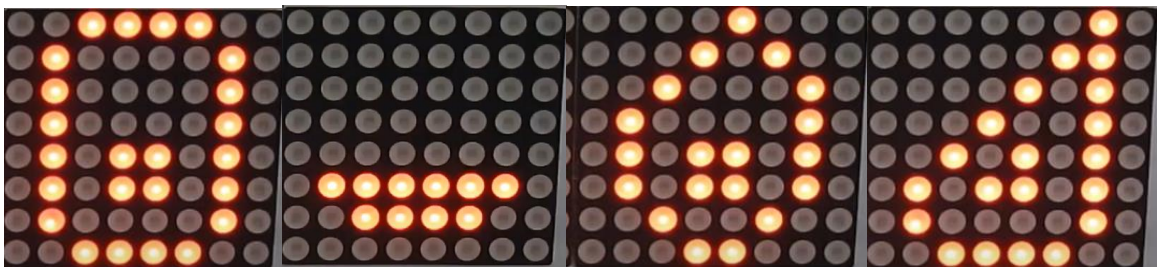


Por fim, falta tratar do último componente do projeto: As matrizes de LED para os olhos:

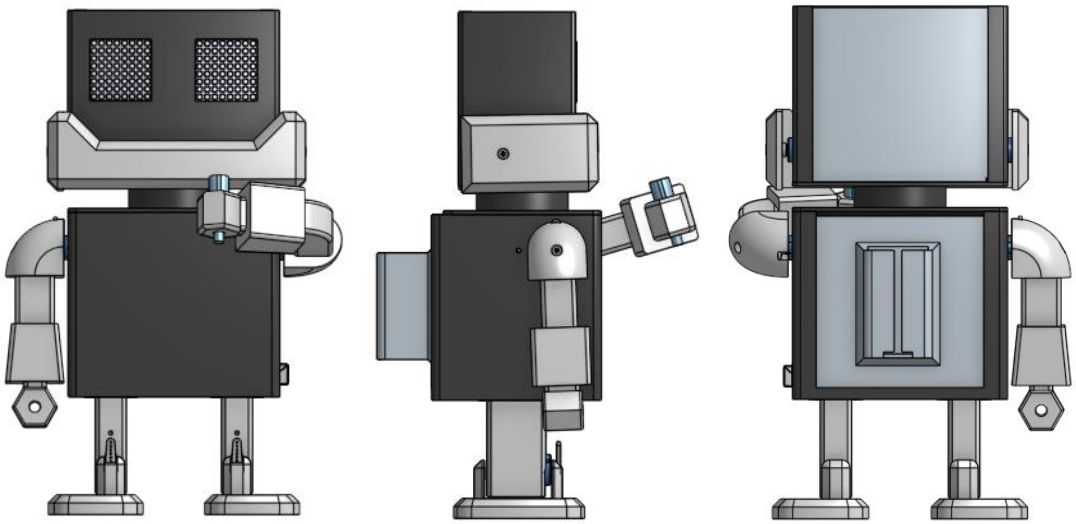
b. Módulos de Matriz LED MAX7219

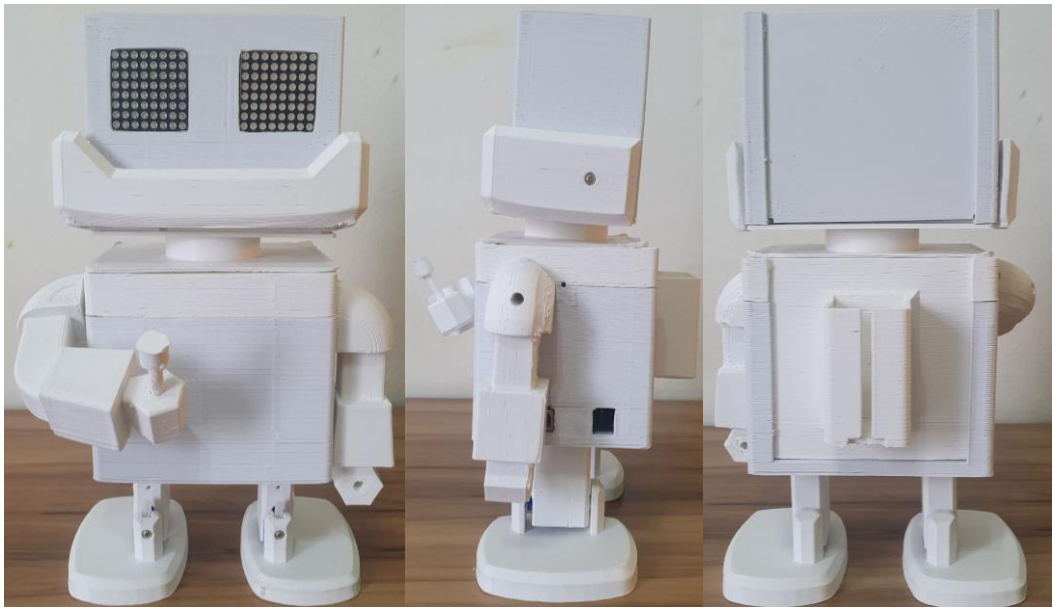
Como o nome já indica, o componente é formado a partir de uma matriz de leds, nesse caso cada módulo possui uma matriz 8x8 totalizando 64 pontos de luz. Contudo, no projeto serão usados dois módulos de LED em conjunto, um para cada olho.

O funcionamento do MAX7219 é bastante simples, de acordo com um mapa de bits é escolhido qual led é aceso ou não, formando assim o desenho que quiser. No caso do robô foram feitas quatro configurações diferentes: Neutro, fechado, zangado e malvado:



Assim sendo temos o robô completo montado:

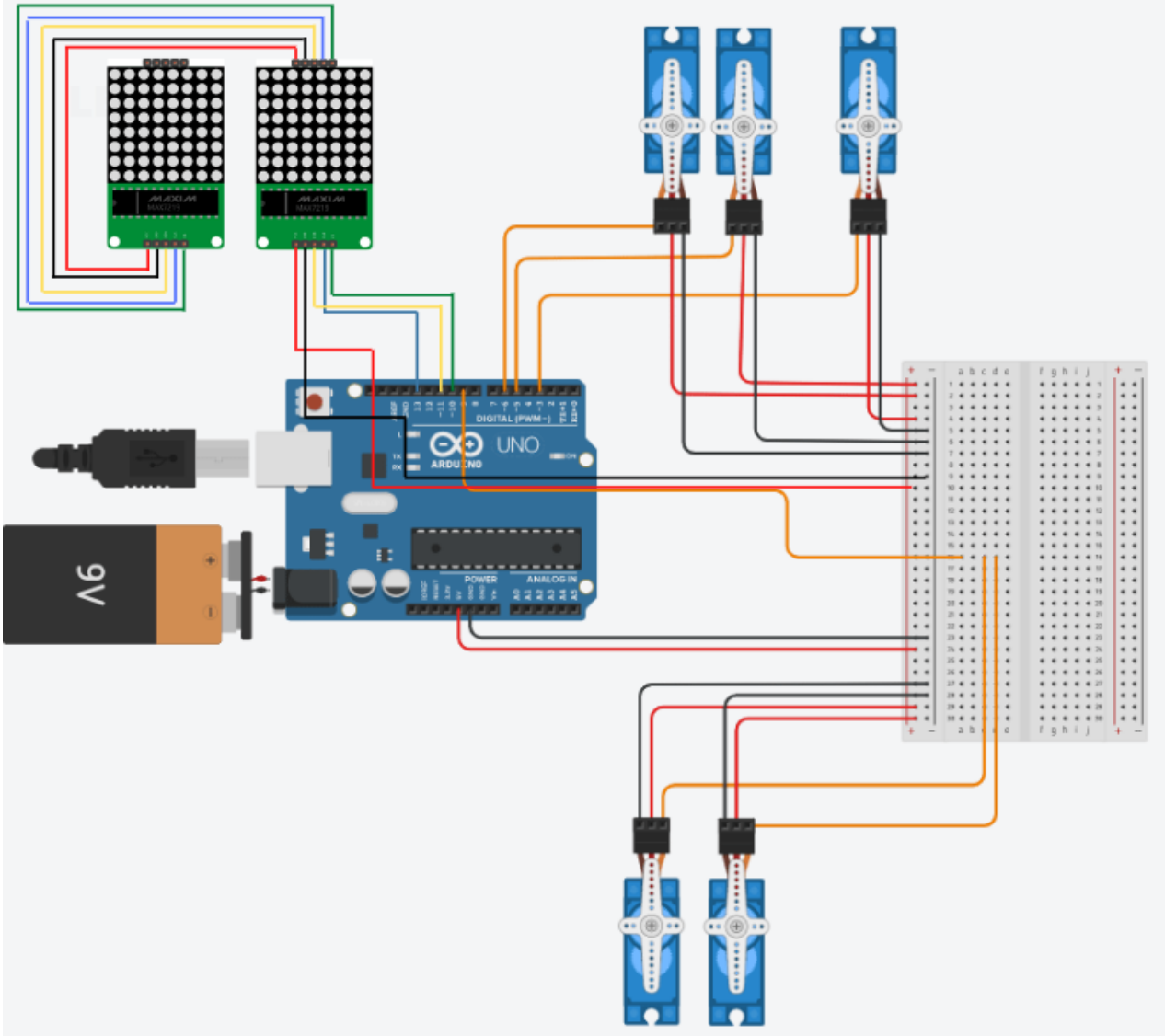




Acesso aos modelos 3D do robô:

https://cad.onshape.com/documents/f61283b1fc311db5923d375d/w/8dd203b56d9e28b43159c713/e/19a4a87245c6292f1db63eb1?renderMode=0&uiState=63914b47f55cc77e1bc6e852

Agora vamos falar do circuito do Arduino, que a partir de todos os componentes listados acima e com a ajuda da protoboard ficou da seguinte maneira:



O primeiro conjunto de servos acima do Arduino são referentes à boca, o servo "solitário", ao lado direito do conjunto da boca, é referente ao microfone e os dois de baixo são responsáveis pelo movimento do pé. É bom salientar que como os pés mexem no mesmo sentido e os servos estão na mesma direção, o sinal dos dois é conectado em apenas um pino de PWM do Arduino.

As matrizes de LED estão no superior direito da imagem e torna-se necessário descrevê-las: São duas matrizes ligadas uma na outra diretamente, com cinco saídas: GND, VCC, CS, CLK e DIN, nesse caso, o terra e o vcc (pino de 5V) entram na protoboard; o pino de clock não precisa ser PWM então ele entra no pino de número 13. Os dois pinos que sobraram de PWM ficam para o sinal dos leds e o dado de cada led. Sendo assim temos a seguinte definição no Arduino:

```
LedControl lc = LedControl(11, 13, 10, 2); // Pins: DIN,CLK,CS
servo_microphone.attach(3);
servo_mouth1.attach(5);
servo_mouth2.attach(6);
servo_foot.attach(9);
```

Desta maneira, se encontra concluída a etapa de discussões acerca dos integrantes do projeto, sendo software e hardware.

Por fim, passaremos a expor, na Conclusão, alguns problemas do projeto, passos futuros e conclusão.

3. Conclusão

Levando em consideração todas as propostas e ideias que foram discutidas desde o início do período letivo de 2022.2, acerca do projeto, é justo dizer que ele teve êxito.

De acordo com o escopo inicial de fazer um robô que respondesse a estímulos sonoros através da dança e canto, ambas as partes de software e hardware se complementaram e cumpriram com o combinado. Foi um longo caminho onde muitas críticas e revisionismo aconteceu para que fosse possível chegar ao sucesso.

Alguns obstáculos e limitações valem a pena serem discutidos: Primeiramente, o robô só performa de forma ótima em uma faixa de bpm. Quanto mais rápido a música for mais truncada será a resposta motora dos servos do pé. Isso acaba restringindo os ritmos em que a dança fica completamente padronizada. Por exemplo, um rock com andamento quatro por quatro é mais eficiente do que uma salsa ou um jazz onde tem andamentos não tão convencionais. Em contrapartida a voz passa em todos os testes.

Ao mesmo tempo, foi notado um comportamento de bounce esporádicos enquanto a música é reproduzida. Uma explicação é que a serial manda mais informações do que o tempo necessário para realizá-las fisicamente. Mas novamente não compromete em nada o desempenho do robô.

É importante relatar que se algo foi insuficiente, ainda que bem pouco, foi no acabamento das peças, que precisaram ser lixadas para se adequar aos encaixes. Diante desse pequeno contratempo da impressão 3D o design tem algumas imperfeições de impressão, na maioria das vezes quando a peça tem uma parede muito fina ou formato circular, porém, esse ponto é meramente estético.

Este projeto em questão tem o potencial quase infinito de melhorias e novas funcionalidades. Uma que enriquecerá a experiência é fazer com que o robô "sinta" a emoção que a música passa, por exemplo uma música feliz ou triste, e representar através dos seus olhos, a partir de diferentes desenhos da matriz de LED. Já existe uma API do Spotify que faz essa separação.

Outro ponto é conseguir trazer a interface para o robô, deixando de ter a interação com o python. E isso pode ser feito com um visor touch no próprio corpo do robô, onde seja possível escolher as músicas e ver os medidores.

A inserção de mais servos também é possível, desde que aumente o Arduino uno para o Mega, por exemplo, e isso traz uma vasta gama de novos movimentos. Ter a cabeça balançando no ritmo da música assim como a perna será algo muito interessante, bem como abrir a possibilidade da rotação do próprio tronco.

Outra possível melhoria seria identificar momentos de solo ou de instrumental, sem voz. E com isso fazer com que o robô contemple ou reaja de maneira diferente nesses momentos.

Por fim o projeto atingiu seus objetivos, onde um robô autônomo pode realizar várias ações programadas e desejadas sem a ajuda humana.

Com certo grau de complexidade, ainda que relativamente baixo, foi possível trazer o lúdico para o ambiente humano, seja através da "dança", da música, e até da interação "emocional" do olhar do "robozinho", assim batizado por algumas crianças que aprovaram e se divertiram com o resultado do trabalho.

Link do Robô pronto e funcionando:

https://www.youtube.com/watch?v=xYBDxhtpJ6g

4. Referências Bibliográficas

LIBROSA. librosa, 2022. Audio and music processing in Python. Disponível em: https://librosa.org/. Acesso em: 19 nov. 2022.

SPLEETER. Spleeter, 2022. Deezer source separation library including pretrained models. Disponível em: https://github.com/deezer/spleeter. Acesso em: 19 nov. 2022.

TKINTER. Docs. Python, 2022. Graphical User Interfaces with Tk. Disponível em: https://docs.python.org/3/library/tk.html. Acesso em: 19 nov. 2022.

PYGAME. Pygame, 2022. Disponível em: https://pygame.org/. Acesso em: 19 nov. 2022.

OTTO DIY. Otto DIY, 2022. Build, code and design your own robot. Disponível em: https://www.ottodiy.com/. Acesso em: 19 nov. 2022.