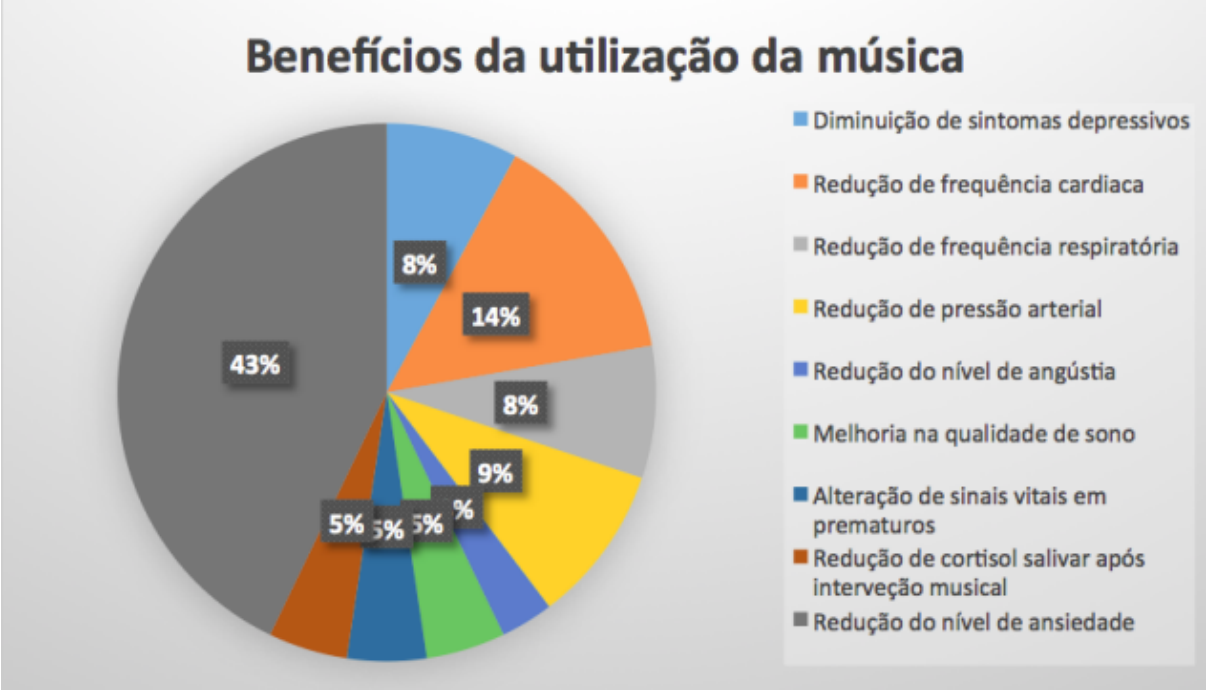
RELATÓRIO DO TRABALHO PRÁTICO
GCC 128 - INTELIGÊNCIA ARTIFICIAL

Gabrielly Corrêa
Ricardo Santos
Thiago Henrique

INTRODUÇÃO

O robô “Eva” foi desenvolvido com a premissa de ajudar os pais no entretenimento de seus filhos por meio de músicas, uma vez que foi comprovado pela neurociência que a música é capaz de regular a vontade de comer, o sono e o estado de ânimo, além de produzir a sensação de satisfação, felicidade, relaxamento e bem-estar.

Com isso, planejamos nosso robô de forma que tenha a habilidade de ouvir o usuário, o qual poderá pedir uma música em específico para ser tocada, uma playlist ou então todas as músicas baixadas. A partir disso, o robô começará a reproduzir o que foi pedido, caso seja encontrado.



https://www.iar.unicamp.br/wp-content/uploads/2021/07/V06_ED11_A02_EfeitosUtilizMusicHosp.pdf

SOBRE AS CLASSES

control
Em resumo, o código runRobot() simplesmente importa o módulo sense_music e chama a função speech_to_text() desse módulo. O que exatamente a função speech_to_text() faz depende da implementação dentro do módulo sense_music, que não está presente no código fornecido. Para entender completamente o que esse código faz, você precisaria examinar o conteúdo do módulo sense_music.

think_music
Ela define funções para buscar e reproduzir músicas, reproduzir playlists, controlar o estado da reprodução (pausar, continuar, parar, pular), e lidar com comandos de voz para interagir com o programa. A variável global pastaPlaylistAtual é usada para rastrear qual playlist está sendo reproduzida no momento.

act_music
Esta classe oferece funções básicas para controlar a reprodução de música, incluindo reproduzir, pausar, parar, continuar, alterar o volume e pular para a próxima música em uma lista de reprodução. Ela utiliza a biblioteca pygame para gerenciar a reprodução de áudio.

act_talk
A principal finalidade dessa classe é permitir que você envie mensagens de texto para serem reproduzidas como fala em um sistema de saída de áudio, como um assistente de voz. Ela usa o gTTS para a conversão de texto em fala e o pygame para a reprodução do áudio resultante.

sense_music
Em resumo, essa classe cria uma interface para interagir com um assistente virtual por meio de comandos de voz. Ela permite ao usuário controlar a reprodução de música e executar outras ações com base no reconhecimento de fala. O assistente virtual "Beto" responde aos comandos de voz do usuário e interage com as funções relacionadas à música ou outras ações definidas no arquivo JSON.

main
Ele inicializa o Pygame, executa a interação por voz (provavelmente com a função menuSentir), e depois encerra o Pygame quando a interação é concluída. A função falarMensagens é usada para fornecer mensagens de áudio ao usuário.

BIBLIOTECAS UTILIZADAS

Pygame

A biblioteca Pygame é uma biblioteca de código aberto para Python que é amplamente utilizada para desenvolver jogos e aplicações multimídia interativas. Ela fornece uma série de recursos e funcionalidades que tornam mais fácil a criação de jogos 2D e aplicações gráficas. No nosso projeto nós utilizamos o recurso de controle de áudio que permitiu a reprodução dos sons e música, além de controlar a reprodução de áudio de maneira simples.

Google Text-to-Speech

A biblioteca "gTTS" (Google Text-to-Speech) é uma biblioteca Python que permite converter texto em fala usando a tecnologia de síntese de fala do Google. Nós utilizamos o gTTS, para criar os arquivos de áudio a partir de texto, o que é útil em uma variedade de aplicativos, como assistentes virtuais, aplicativos de acessibilidade, geração de áudio para vídeos e consequentemente no nosso projeto.

Speech Recognition

A biblioteca "speech_recognition" é uma biblioteca Python que fornece uma interface simples para trabalhar com reconhecimento de fala (speech recognition). Ela permite que você integre facilmente recursos de reconhecimento de voz em seus programas Python, tornando possível converter fala em texto. Nós utilizamos para capturar áudio do microfone em tempo real e convertê-lo em texto.

OS

A biblioteca os é uma biblioteca padrão do Python que fornece uma interface para interagir com o sistema operacional subjacente no qual o Python está sendo executado. Ela permite que você execute várias tarefas relacionadas ao sistema, como navegar pelo sistema de arquivos, manipular caminhos de arquivos, acessar variáveis de ambiente, executar comandos do sistema, criar diretórios e muito mais. É uma biblioteca fundamental para realizar operações de baixo nível relacionadas ao sistema operacional.

Beepy

A biblioteca beepy é um módulo da linguagem Python que permite aos usuários reproduzir facilmente sons de notificação no Linux, OSX e Windows. Funciona apenas no Python 3 e requer a instalação do pacote **simpleaudio**. Ela foi usada no projeto para emitir os “apitos” toda vez que o usuário tentar falar com o robô, com a finalidade de que o usuário saiba que o robô está ouvindo.

DIFICULDADES ENCONTRADAS

- A princípio, o objetivo do projeto era além de fazer as interações por comandos de voz, fazer a integração com um robô, disponibilizado para a disciplina do Departamento de Ciência da Computação, que dançaria de acordo com os comandos apresentados. Porém, a nossa principal dificuldade foi fazer essa conexão do nosso código com o robô, porque solicitava conexão bluetooth mas dava erro de porta que não soubemos solucionar.

- Dificuldade de realizar a comunicação com o robô enquanto havia música tocando, pois inicialmente tivemos a ideia da função “act” conectar com a “sense”, porém aparecia erro de “loop” entre arquivos. Isso foi solucionado ao designar uma thread para reproduzir as músicas.

- Dificuldade ao importar funções de outras pastas, a solução foi fazer uma pesquisa e criar um arquivo __init__.py.

- Dificuldade em encontrar uma API para fornecer uma lista de músicas para serem reproduzidas no projeto, isso foi solucionado criando nossa própria “api” de músicas por meio de uma pasta no projeto contendo músicas e playlist.