



Instituto Politécnico do Cávado e do Ave
Escola Superior de Tecnologia

**Mestrado em Engenharia Eletrónica e de
Computadores**

Kustic: Braço robótico guitarrista

Robótica Avançada

Grupo I
André Moreira nº23914
Joaquin Dillen nº15463
Nuno Fernandes nº15464

julho 2022

Resumo

A robótica é uma área tecnológica que abrange diversas áreas tais como, a mecânica, a eletrônica e a computação. Na sociedade de hoje, o uso de robôs como máquinas industriais e também para desempenhar outras tarefas específicas, têm sido cada vez mais exponencial.

O robô utilizado para este trabalho foi a KUKA KR 6 R900, e tinha como objetivo tocar músicas numa guitarra elétrica. Para isso foi desenvolvida uma ferramenta, que foi posteriormente impressa numa impressora 3D. Foram também desenvolvidos algoritmos capazes de calcular as coordenadas exatas dos trastes e cordas, para que o robô fosse capaz de se deslocar eficientemente. Por último foi criada uma interface gráfica para que o utilizador pudesse interagir com o robô e para que pudesse escolher as músicas que queria ouvir.

Palavras-Chave (Tema): Braço robótico, Guitarra

Palavras-Chave (Tecnologias): KUKA, OrangeEdit, Python, PyQt5

Índice

<i>Resumo</i>	<i>iii</i>
<i>Índice</i>	<i>v</i>
<i>Índice de Figuras</i>	<i>vii</i>
<i>Índice de Tabelas</i>	<i>ix</i>
<i>Notação e Glossário</i>	<i>xi</i>
1 Introdução	13
1.1 Enquadramento	13
1.2 Tecnologias utilizadas	14
1.3 Contributos deste trabalho	14
1.4 Organização do relatório	14
2 Descrição técnica	15
2.1 Ferramenta	16
2.2 Criação das músicas	19
2.3 Aquisição das coordenadas	23
2.3.1 Coordenadas do traste.....	23
2.3.2 Coordenadas da corda.....	26
2.4 Comunicação	28
2.5 Interface	30
2.6 Braço robótico	32
2.7 Simulação	36
3 Conclusões	37
<i>Bibliografia</i>	<i>39</i>
Anexo 1 – Esquemático da ferramenta	41

Índice de Figuras

<i>Figura 1 – Composição de uma guitarra elétrica.</i>	15
<i>Figura 2 – Ferramenta desenvolvida.</i>	16
<i>Figura 3 – Esquemático da ferramenta Kustic.</i>	17
<i>Figura 4 – Estrutura completa da Ferramenta.</i>	18
<i>Figura 5 – Designação das cordas de uma guitarra.</i>	19
<i>Figura 6 – Trastes de guitarra.</i>	20
<i>Figura 7 – Tablatura de guitarra.</i>	20
<i>Figura 8 – Dicionário Python de cordas de guitarra.</i>	21
<i>Figura 9 – Dicionário Python de trastes de guitarra.</i>	21
<i>Figura 10 – Dicionário Python de tempos das figuras musicais.</i>	22
<i>Figura 11 – Representação de uma nota individual.</i>	22
<i>Figura 12 – Exemplo de uma música.</i>	22
<i>Figura 13 – Definição do eixo de coordenadas da base.</i>	23
<i>Figura 14 – Cálculo da coordenada X.</i>	25
<i>Figura 15 – Cálculo da coordenada Y.</i>	27
<i>Figura 16 – Exemplo de ligação, leitura e escrita de variáveis do robô através da livreria OpenShowVar.</i>	28
<i>Figura 17 – Envio das coordenadas referentes às notas que iriam ser tocadas.</i>	28
<i>Figura 18 – Ecrã principal da GUI.</i>	30
<i>Figura 19 – Ecrã da seleção de músicas da GUI.</i>	31
<i>Figura 20 – Atribuição da ferramenta e da base a serem utilizadas pelo robô.</i>	32
<i>Figura 21 – Inicializações de variáveis, estruturas e interrupções.</i>	32
<i>Figura 22 – Função de emergência executava nas interrupções.</i>	33
<i>Figura 23 – Função para se deslocar para o ponto HOME_KUSTIC.</i>	33
<i>Figura 24 – Espera da ordem de arranque e reset/set de variáveis de estado.</i>	34
<i>Figura 25 – Ciclo de movimentos do robô para as coordenadas enviadas.</i>	34
<i>Figura 26 – Movimento para tocar uma corda.</i>	35
<i>Figura 27 – Movimento para tocar uma corda aberta.</i>	35

<i>Figura 28 – Espera e elevação da ferramenta.</i>	35
<i>Figura 29 – Ambiente de Simulação.</i>	36

Índice de Tabelas

<i>Tabela 1 – Ilustração de Tempos.</i>	21
---	----

Notação e Glossário

GUI Interface de utilizador gráfica

PTP Ponto a ponto

1 Introdução

A elaboração deste trabalho prático seguiu-se nos conteúdos lecionados ao longo da cadeira de Robótica Avançada do 1º ano do Mestrado em Engenharia Eletrónica e de Computadores. O projeto consiste no desenvolvimento de uma aplicação ao critério do aluno que envolva a operação de um robô industrial, neste caso, o robô industrial KR 6 R900 da KUKA.

1.1 Enquadramento

Para o projeto foram exploradas diferentes ideias, sendo no final definida a implementação de um sistema que permite ao robô industrial reproduzir músicas operando uma guitarra elétrica.

O processo de desenvolvimento passou primeiro por uma abordagem teórica para identificar as necessidades para o sucesso da implementação. Depois foi feita a abordagem técnica, onde foi realizado o desenvolvimento das ferramentas necessárias para conseguir habilitar o robô industrial a interagir com o instrumento de uma forma eficaz, assim como conceber os algoritmos que iriam definir a forma do robô reproduzir as músicas ao operar o instrumento.

O robô era capaz de reproduzir músicas monofónicas e receber informação do utilizador através uma interface gráfica que permitia a seleção de músicas e a indicação para serem reproduzidas.

1.2 Tecnologias utilizadas

O robô utilizado para a realização deste trabalho, foi a KUKA KR 6 R900. Este robô tem seis graus de liberdade, consegue manusear uma carga máxima de 6 kg e um alcance máximo de 901.5 mm.

Para a abordagem teórica foi utilizado o software de simulação Kuka.SIM, onde foram testados os novos conceitos. Para a elaboração do código do braço robótico foi utilizado o WorkVisual dentro da consola do robô industrial. O desenvolvimento da ferramenta foi feito no software Fusion 360 e as impressões foram feitas por uma impressora Creality CR10S.

Os restantes algoritmos foram feitos na linguagem de programação Python, no IDE PyCharm. Para fazer a comunicação entre o computador e o robô foi utilizada a livreria OpenShowVar, que funciona como um cliente TCP. Para a interface gráfica de utilizador (GUI) foi utilizada a livreria PyQt5, que permite usufruir das capacidades do QT Designer em código Python.

1.3 Contributos deste trabalho

O presente trabalho consegue expor uma implementação recreativa que permite expressar as capacidades técnicas de um robô industrial de uma forma atrativa para o público e também expõe as capacidades fora do âmbito industrial que dispõem os equipamentos deste estilo.

1.4 Organização do relatório

O presente trabalho consta de uma descrição técnica onde são abordados todos os aspetos que fizeram possível a implementação do trabalho descrito passando desde uma breve descrição geral até aspetos mais característicos como o desenvolvimento da ferramenta, a criação e adaptação das músicas para a interpretação do robô, a definição dos sistemas de equações para definição das respetivas coordenadas e respetivas validações por simulação.

2 Descrição técnica

Como descrito no capítulo anterior, o objetivo do presente projeto foi desenvolver um algoritmo que permitisse que o robô industrial conseguisse reproduzir músicas com fins recreativos.

Para esse objetivo foi utilizada uma guitarra elétrica Stratocaster, da marca Squier by Fender, e foi desenvolvida uma ferramenta que tornava possível a interação entre o robô e a guitarra.

Para tornar o sistema mais interativo foi desenvolvida uma interface gráfica na linguagem Python utilizando ferramentas de comunicação wireless para a transmissão de dados entre o robô e o computador.

Para atingir com sucesso a implementação foi necessário um nível de compreensão sobre a natureza do instrumento e o seu comportamento rigoroso, desde a sua composição até o seu comportamento a nível de interação.

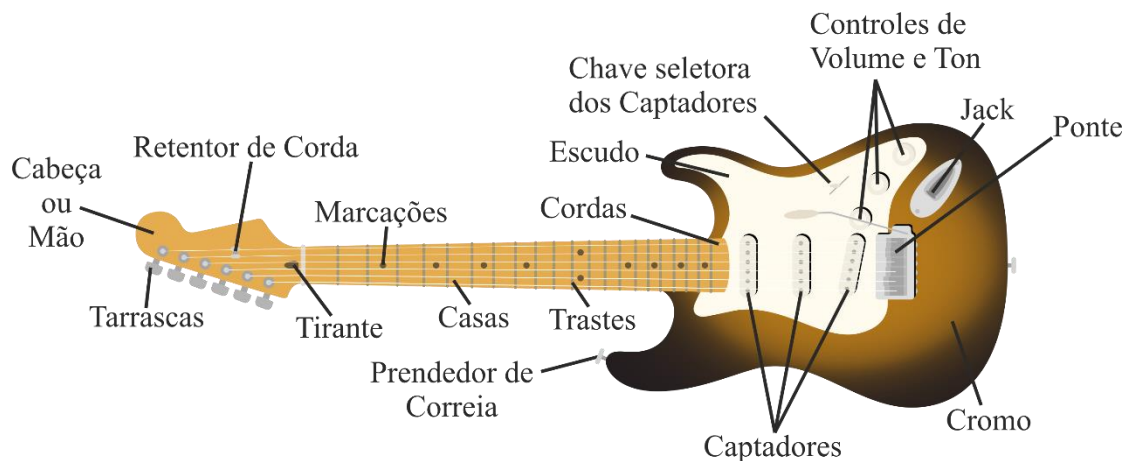


Figura 1 – Composição de uma guitarra elétrica.

Na Figura 1 é possível observar a terminologia técnica da composição de uma guitarra, a qual será referida ao longo do trabalho para uma melhor compreensão dos distintos aspetos que fazem parte da elaboração do projeto.

2.1 Ferramenta

Para o desenvolvimento da ferramenta de interação com a guitarra foi necessária a análise e estudo do processo de interação com o instrumento, para o correto desenvolvimento de uma ferramenta capaz de atingir o nível de funcionalidade necessário para a operação em questão.

A interação com a guitarra consta numa perspectiva muito técnica e abstrata da colocação de pressão no ponto da corda pretendido para uma referida nota, assim como também a indução da vibração da própria corda para ser obtido o sinal ou a frequência pretendida pelos captadores.

Para este efeito e tentando dividir as duas operações foi desenvolvida uma ferramenta capaz de fazer estas duas interações separadamente.



Figura 2 – Ferramenta desenvolvida.

Como se pode observar na Figura 2 a ferramenta desenvolvida tem uma constituição particular, a principal secção é a do “dedo” que é composto por uma estrutura em PLA e uma ponta construída utilizando TPU com o objetivo de dar uma sensação mais parecida com o dedo humano, ao ser um material elástico e flexível consegue ter uma melhor interação com o instrumento.

Em laranja temos uma secção amovível que consta de uma unha habitualmente utilizada para colocar as cordas a vibrar após a seleção das notas nos trastes já foi efetuada. Esta secção foi construída de forma amovível pois existia um conhecimento prévio da possível não necessidade desta implementação para a reprodução de música de parte do robô industrial.

Por motivos de segurança e para uma melhor performance da ferramenta para com o instrumento o “dedo” dispõe de uma mola interna, que permite certo movimento do próprio e caso seja exercida uma força elevada na guitarra, é possível evitar a danificação do instrumento.

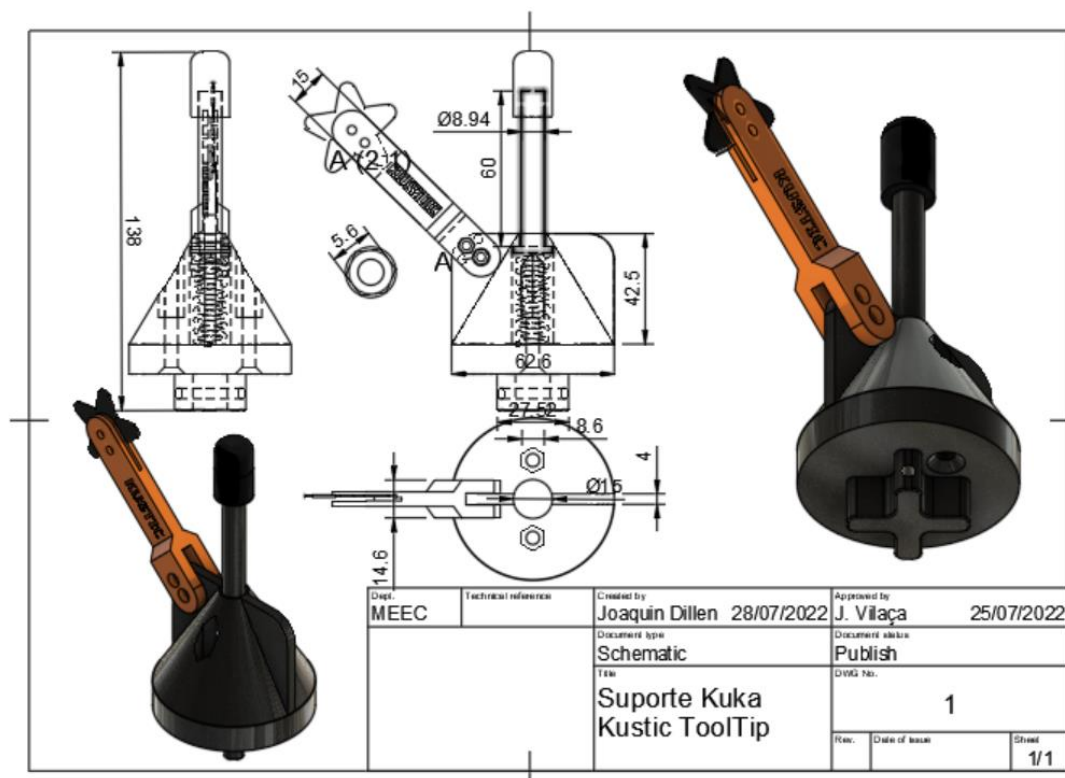


Figura 3 – Esquemático da ferramenta Kustic.



Figura 4 – Estrutura completa da Ferramenta.

Na Figura 4 é possível observar a estrutura completa da ferramenta assim como a mola interna. Todas as secções que fazem parte da ferramenta com a exceção da palheta, da mola e da ponta que é de TPU, foram construídas em PLA, recorrendo a uma impressora 3D.

Para juntar a secção inferior da ferramenta com a secção superior foi preciso utilizar um par de parafusos e porcas M6. Para a fixação da palheta e do “braço” laranja foram usados um total de 4 parafusos e porcas M3. Na ferramenta foram desenhados diferentes buracos para ser possível ajustar o ângulo de incidência da palheta nas cordas.

2.2 Criação das músicas

Para criar as músicas foi necessário fazer um estudo prévio dos componentes que constituem uma tablatura de guitarra. As tablaturas de guitarra partilham semelhanças com a notação da pauta musical, mostrando quais as notas a tocar, quanto tempo as tocar, e quais as técnicas a utilizar. Mas quando comparada com a notação de música padrão, a tablatura da guitarra oferece a vantagem de mostrar onde tocar as notas na guitarra, especialmente, porque a guitarra tem várias maneiras diferentes de tocar as mesmas notas.

A tablatura da guitarra é a representação visual das notas de uma canção e consiste em seis linhas horizontais, com cada linha a representar as seis cordas da guitarra. Ao olhar para a tablatura da guitarra de cima para baixo, a linha superior representa a corda E alto (a corda mais fina) seguida por linhas que representam as cordas B, G, D, A e E baixo (a corda mais grossa).

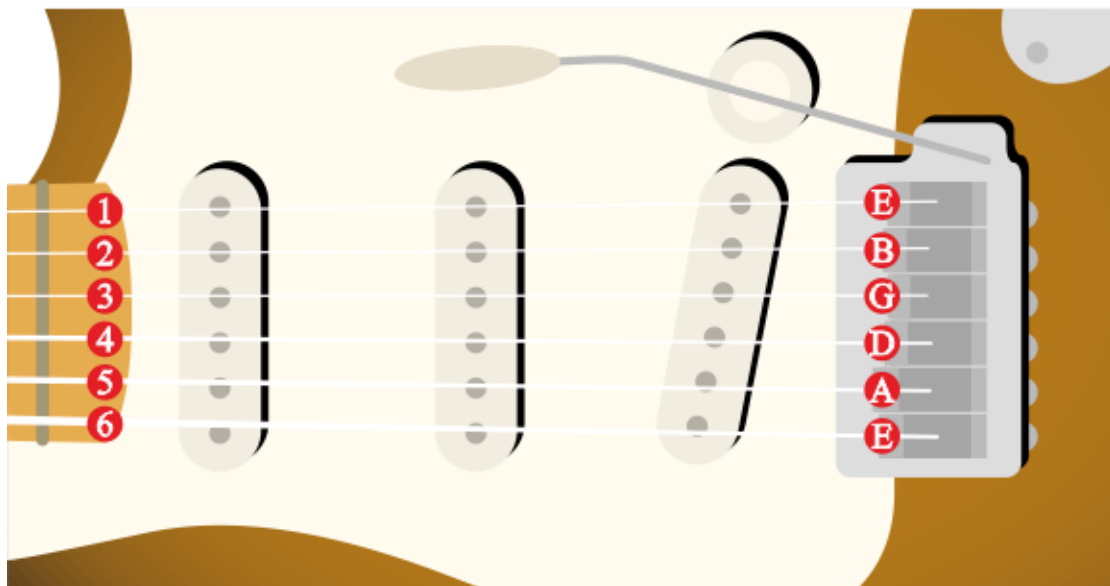


Figura 5 – Designação das cordas de uma guitarra.

Em cada linha de tabulação existem também números. Estes números representam os trastes da guitarra, que são as faixas de metal encontradas na escala. Os trastes são numerados de 0 a 24, e começam na pestana (a peça mais próxima da cabeça), e correm ao longo de todo o comprimento da escala da guitarra.

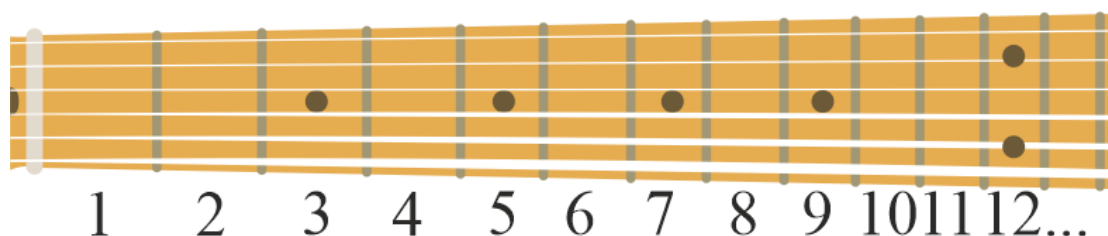


Figura 6 – Trastes de guitarra.

Por exemplo, se a corda tiver um 0, isso significa que tem de se tocar essa corda “aberta”, ou sem usar a mão no traste. Se a corda tiver um 1, então isso significa que deve ser tocada usando o primeiro traste.

Sendo que só tínhamos um robô disponível e este só podia tocar uma nota de cada vez, foram utilizadas tablaturas que fossem possíveis de ser tocados com um só dedo (Figura 7).

```

E- | ----- | |
B- | ----- | |
G- | ----- | |
D- | ----- | |
A- | -7---7-10--7--5--3---2--- | |
E- | ----- | |

```

Figura 7 – Tablatura de guitarra.

A última componente presente na tablatura é o tempo que as notas devem ser tocadas, que é representado pelo espaço entre os números.

Todos estes componentes são facilmente convertidos para código Python através do uso de dicionários, sendo que todos eles têm um nome e um valor associado. No caso das cordas, a chave era a sua letra e a sua numeração, pois existem duas cordas com a mesma letra (E baixo e E alto), e o seu valor era a numeração (Figura 8).

```
# guitar string dictionary
guitar_strings = {'E1': 1, 'B2': 2, 'G3': 3, 'D4': 4, 'A5': 5, 'E6': 6}
```

Figura 8 – Dicionário Python de cordas de guitarra.








Da mesma maneira, criamos também um dicionário para os trastes, sendo a sua chave o nome do traste e o seu valor o seu número correspondente (Figura 9).

```
# guitar fret dictionary
guitar_frets = {'fret0': 0, 'fret1': 1, 'fret2': 2, 'fret3': 3, 'fret4': 4, 'fret5': 5, 'fret6': 6,
                'fret7': 7, 'fret8': 8, 'fret9': 9, 'fret10': 10, 'fret11': 11, 'fret12': 12,
                'fret13': 13, 'fret14': 14, 'fret15': 15, 'fret16': 16, 'fret17': 17, 'fret18': 18,
                'fret19': 19, 'fret20': 20, 'fret21': 21, 'fret22': 22, 'fret23': 23, 'fret24': 24}
```

Figura 9 – Dicionário Python de trastes de guitarra.

Os valores da nota musical são determinados em referência ao comprimento de uma nota inteira (Semibreve). As outras notas são nomeadas em comparação: uma mínima que é metade do comprimento de uma nota inteira, uma semínima é um quarto do comprimento, etc. Todos as notas musicais podem ser vistas na Tabela 1.

Tabela 1 – Ilustração de Tempos.

Nome	Símbolos	Tempos
<i>Semibreve</i>		4
<i>Mínima</i>		2
<i>Semínima</i>		1
<i>Colcheia</i>		$\frac{1}{2}$
<i>Semicolcheia</i>		$\frac{1}{4}$
<i>Fusa</i>		$\frac{1}{8}$
<i>Semifusa</i>		$\frac{1}{16}$

Tal como foi feito para as cordas e para os trastes, criamos um dicionário para as notas musicais e os seus tempos (Figura 10).

```
# tempo duration dictionary
tempos = {'semibreve': 4, 'minim': 2, 'crotchet': 1, 'quaver': 1/2,
          'semiquaver': 1/4, 'demisemiquaver': 1/8, 'hemidemisemiquaver': 1/16}
```

Figura 10 – Dicionário Python de tempos das figuras musicais.

Para criar uma nota é dado o frete, a corda e o tempo que essa nota devia de ser tocada, que por sua vez é multiplicado pelo ritmo, para que a música possa ser tocada mais rápida ou mais lenta, mantendo a sua consistência.

```
[guitar_frets['fret8'], guitar_strings['A5'], tempos['minim'] * beat]
```

Figura 11 – Representação de uma nota individual.

Sendo que uma música é um conjunto de notas, foi natural o uso de listas para a criação das mesmas (Figura 12).

```
# Survivor - Eye of the tiger
survivor_eye_of_the_tiger = [[guitar_frets['fret3'], guitar_strings['A5'], tempos['minim'] * beat],
                             [guitar_frets['fret3'], guitar_strings['A5'], tempos['crotchet'] * beat],
                             [guitar_frets['fret1'], guitar_strings['A5'], tempos['crotchet'] * beat],
                             [guitar_frets['fret3'], guitar_strings['A5'], tempos['minim'] * beat],
                             [guitar_frets['fret3'], guitar_strings['A5'], tempos['crotchet'] * beat],
                             [guitar_frets['fret1'], guitar_strings['A5'], tempos['crotchet'] * beat],
                             [guitar_frets['fret3'], guitar_strings['A5'], tempos['minim'] * beat],
                             [guitar_frets['fret3'], guitar_strings['A5'], tempos['crotchet'] * beat],
                             [guitar_frets['fret3'], guitar_strings['E6'], tempos['crotchet'] * beat],
                             [guitar_frets['fret4'], guitar_strings['E6'], tempos['semibreve'] * beat]]
```

Figura 12 – Exemplo de uma música.

Estes dicionários tornam a inserção de músicas mais fácil, pois conseguimos ter a certeza de que o valor que estamos a inserir é igual em todas as músicas e se fosse necessário alterar o valor de um componente, apenas o teríamos de alterar num só sítio.

2.3 Aquisição das coordenadas

Numa tablatura de guitarra, são indicados o traste e a corda a tocar para cada nota, sendo a combinação dessas informações o ponto (X, Y) onde temos de colocar o dedo.

Estas coordenadas são definidas em relação à uma base, a base foi definida considerando a origem a primeira nota, isto é, a nota do traste 1 e da corda 6, como se pode observar na imagem a continuação.

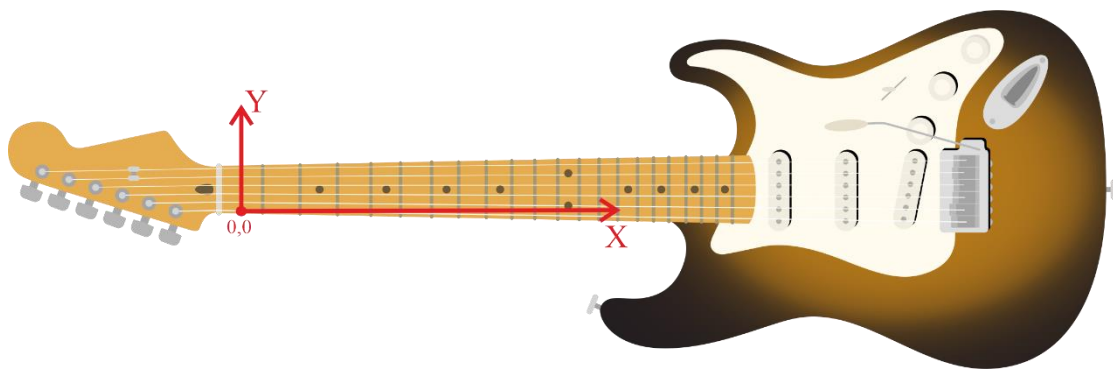


Figura 13 – Definição do eixo de coordenadas da base.

Para fazer com que o robô toque uma nota, é necessário converter o número do traste e a corda para valores em milímetros. Nos seguintes subcapítulos são explicados os procedimentos utilizados para obter esses valores.

2.3.1 Coordenadas do traste

Os luthiers utilizam uma fórmula matemática, intitulada de “Regra de 18”, para calcular a colocação dos trastes. Ao contrário de muitos predecessores históricos, a Regra de 18 baseia-se no conceito de temperamento igual, provavelmente antes de o termo existir no vernáculo. Mas, mais importante ainda é que, a Regra de 18 inclui o seu próprio fator de compensação no que diz respeito à posição da ponte.

Ao contrário do que o nome indica, 17,817, é o valor mais comumente utilizado atualmente, como o fator com o qual se divide o comprimento da corda. O resultado é considerado por muitos como sendo mais exato por colocar o 12º traste no ponto médio exato do comprimento da corda. Uma vez que o 12º traste é a oitava acima da corda aberta, faz todo o sentido que esteja no meio do comprimento da corda vibratória.

Para calcular o comprimento de um traste (CT) basta aplicar a seguinte fórmula, onde primeiro se subtrai o comprimento acumulativo (CA), de todos os trastes até ao traste desejado, ao comprimento total da escala (CE), desde a pestana da guitarra até à ponte da guitarra, e de seguida divide-se por 17,812.

$$CT = \frac{CE - CA}{17,812}$$

Seguem-se 3 exemplos de como aplicar a fórmula, assumindo que o comprimento da escala é de 65 cm.

- Traste 1:

$$CT1 = \frac{65 - 0}{17,812} = 3,65 \text{ cm}$$

- Traste 2:

$$CT2 = \frac{65 - 3,65}{17,812} = 3,44 \text{ cm}$$

- Traste 3:

$$CT3 = \frac{65 - (3,65 + 3,44)}{17,812} = 3,25 \text{ cm}$$

Utilizando esta fórmula e fazendo algumas alterações é possível calcular o valor da coordenada em X. Para isso é necessário somar os comprimentos de todos os trastes anteriores ao desejado e depois subtrair do comprimento acumulativo, metade do comprimento do traste pretendido, para o robô se posicionar no centro do traste, e metade do comprimento do primeiro traste, porque no nosso caso, o ponto (0, 0) era no centro do primeiro traste e na sexta corda (E baixo).

$$x = CA - \frac{CT}{2} \frac{CE}{17,812}$$

Na Figura 14 é apresentada a implementação em código Python do método explicado anteriormente.

```
def get_fret_coordinates(fret_number):  
    # length in mm from the nut of the guitar to the bridge of the guitar  
    scale_length = 650  
    # accumulative length of all frets  
    accumulative_length = 0  
  
    # calculate coordinates of fret using the rule of 18 (17.817)  
    for i, fret in enumerate(guitar_frets, start=1):  
        fret_length = (scale_length - accumulative_length) / 17.817  
        # add fret length to accumulative length  
        accumulative_length += fret_length  
  
        # reached selected fret  
        if i == fret_number:  
            # x = total length - half the length of the last fret - half the length of the first fret  
            x_coordinate = accumulative_length - (fret_length / 2) - (scale_length / 17.817 / 2)  
            return round(x_coordinate, 2)
```

Figura 14 – Cálculo da coordenada X.

2.3.2 Coordenadas da corda

Além das coordenadas em X as cordas têm uma componente em Y a qual não é 100% paralela à orientação definida para a base de operações do robô como pode ser observado na Figura 13. Nesse sentido foi preciso definir equações que conseguissem ajustar os valores em Y para a posição do robô ser constante e precisa e manter a ferramenta sempre encima da corda de forma adequada para tocar guitarra.

Para definir qual é a equação que representa as coordenadas em Y em cada corda foi medida a distância entre as duas cordas, que se encontram no limite (corda 6 e 1) tanto na pestana como na ponte da guitarra (Figura 1). De seguida foram alteradas essas distâncias tomando em consideração o eixo de coordenadas definido para a base da aplicação e com esses dois pontos para cada uma das cordas foi possível determinar a equação da reta que define a altura em Y para cada uma das cordas ao longo dos trastes.

A equação utilizada foi uma equação da reta simples pois é o comportamento normal de uma corda de guitarra.

$$y = mx + b$$

Segue-se um exemplo da utilização da fórmula de calculo das coordenadas da corda. Para a corda 6 da guitarra foram extraídos dois pontos, sendo que, o ponto do traste 1, corda 6 é a origem de coordenadas para a base definida. O ponto (0, 0) serve como ponto de referência para o cálculo da equação, e o segundo ponto, é o ponto (X = 551, Y = -1,2). Para calcularmos o valor de m utilizamos a fórmula conhecida.

$$\frac{X_2 - X_1}{Y_2 - Y_1} = \frac{551 - 0}{-1.2 - 0} = -0,0022$$

De seguida substituímos os valores na equação da reta para determinarmos o valor de b .

$$b = y - mx = -1.2 - (-0,0022) * 551 = 0.0122$$

Obtendo assim a nossa equação da coordenada Y da corda 6

$$y_{(x)} = 0.0022 * x + 0,0122$$

Na qual é inserido o valor da coordenada em X ou a distância em X do traste, para depois calcularmos a altura (coordenada em Y) da corda.

Estas coordenadas, são calculadas dentro do código Python, do lado do computador, com o código que se pode observar a continuação:

```
def get_string_coordinates(string_number, x_coordinate):  
    # string E1  
    if string_number == 1:  
        y_coordinate = 0.0323 * x_coordinate + 34.5027  
    # string B2  
    elif string_number == 2:  
        y_coordinate = 0.0232 * x_coordinate + 28.5168  
    # string G3  
    elif string_number == 3:  
        y_coordinate = 0.0196 * x_coordinate + 21.0004  
    # string D4  
    elif string_number == 4:  
        y_coordinate = 0.0105 * x_coordinate + 14.5145  
    # string A5  
    elif string_number == 5:  
        y_coordinate = 0.0042 * x_coordinate + 7.4858  
    # string E6  
    else:  
        y_coordinate = 0.0022 * x_coordinate + 0.0122  
  
    return round(y_coordinate, 2)
```

Figura 15 – Cálculo da coordenada Y.

2.4 Comunicação

Depois de obtidas todas as coordenadas era necessário enviá-las para o robô. Para esse efeito foi utilizada a livreria desenvolvida em Python, chamada OpenShowVar. Esta livreria é um cliente TCP que serve de interface para os robôs KUKA. OpenShowVar implementa um protocolo fácil, para leitura e escrita de variáveis utilizadas dentro de um programa de movimento do robô.

```
# connect to kuka
client = openshowvar("192.168.10.254", 7000)

# read variable
ov = client.read('$OV_PRO', debug=True)
# write variable
client.write('KUKA_VAR', 'TRUE', debug=True)

# close connection with kuka
client.close()
```

Figura 16 – Exemplo de ligação, leitura e escrita de variáveis do robô através da livreria OpenShowVar.

Uma vez estabelecida a ligação entre o computador e o robô, e depois de selecionada a música e calculadas as coordenadas, estas eram enviadas para a estrutura definida no programa do robô, sendo apenas alterados os valores de X, Y e do ritmo.

```
# write points to kuka
for i, point in enumerate(points, start=1):
    # write guitar fret variable
    client.write(f'POINTS[{i}].POINT.X', f'{point[0]}', debug=False)
    # write guitar string variable
    client.write(f'POINTS[{i}].POINT.Y', f'{point[1]}', debug=False)
    # write tempo variable
    client.write(f'POINTS[{i}].TEMPO', f'{point[2]}', debug=False)

# give start order to kuka
client.write('KUSTIC_START', 'TRUE', debug=False)
```

Figura 17 – Envio das coordenadas referentes às notas que iriam ser tocadas.

Para além de mandar as coordenadas para o robô, era também lido o seu estado atual, para saber se este estava a tocar ou se estava à espera de receber a ordem de arranque. Com essa informação sabíamos qual a ordem a enviar (ordem de arranque ou ordem de paragem), quando o utilizador premia o botão de reprodução.

Foi também adicionado um botão para ativar a emergência através da interface gráfica, que quando premido alterava o valor de uma variável no robô sendo que no seu código essa variável estava ligada a uma interrupção, que parava o seu movimento.

2.5 Interface

Com a lógica do lado do computador completa, foi necessário desenvolver uma GUI para que um utilizador conseguisse escolher qual a música que o robô iria tocar, e transmitir esses dados de uma forma simples e intuitiva. Para criar a GUI, foi utilizada a livreria PyQt5, que torna possível o uso das capacidades e funcionalidades do Qt em código Python.

Qt é um conjunto de bibliotecas C++ multiplataforma que implementam APIs de alto nível para aceder a muitos aspetos dos modernos sistemas desktop e móveis. Estes incluem serviços de localização e posicionamento, multimédia, conectividade NFC e Bluetooth, um navegador Web baseado em Chromium, bem como o desenvolvimento de interfaces de utilizador tradicionais.

PyQt5 é um conjunto abrangente de ligações Python para Qt v5. É implementado como mais de 35 módulos de extensão e permite que Python seja utilizado como uma linguagem de desenvolvimento de aplicações alternativa a C++ em todas as plataformas suportadas, incluindo iOS e Android.

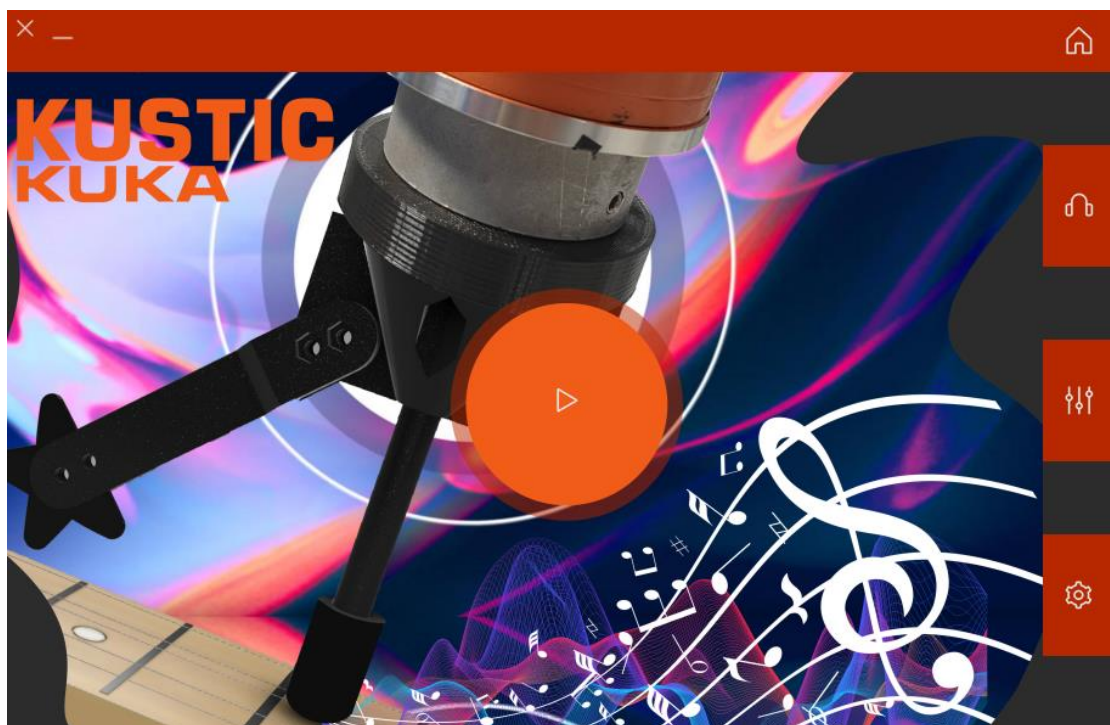


Figura 18 – Ecrã principal da GUI.

A maior parte, dos elementos gráficos da GUI foram desenvolvidos no Qt Designer, que é a ferramenta Qt para conceber e construir GUIs com Qt Widgets. É possível compor e personalizar as suas janelas ou diálogos, e testá-los usando diferentes estilos e resoluções.

A GUI é composta por três ecrãs: o ecrã principal, que acolhe o utilizador (Figura 18); o ecrã de definições que tem informação sobre o programa e os seus criadores; e por último, e o mais relevante, o ecrã da seleção de músicas (Figura 19). O utilizador pode saltar diretamente para o ecrã que deseja, utilizando o menu lateral, presente em todos os ecrãs e composto por quatro botões: “Home”, “Músicas”, “Creativo” e “Definições”.



Figura 19 – Ecrã da seleção de músicas da GUI.

A seleção de músicas é feita através de uma lista onde é possível ver a banda, se for o caso, e o título da música. Uma vez selecionada a música, o utilizador tem apenas de pressionar o botão de reprodução (com o ícone da nota musical) ou então, se tiver um teclado disponível, a tecla “ENTER”.

No momento que esse botão é pressionado, o programa faz todos os cálculos necessários para obter as coordenadas da música selecionada e envia esses dados para o robô. Caso o utilizador queira parar a música, apenas tem de premir o botão de reprodução novamente, e este dará a ordem de paragem ao robô.

2.6 Braço robótico

A programação do braço robótico foi feita na linguagem KRL, tanto no IDE OrangeEdit como diretamente na consola do robô. O programa está dividido em três partes sendo estas: Inicializações, deslocação para as coordenadas, e tocar as cordas.

A primeira tarefa a ser executada pelo robô era a atribuição da ferramenta e da base que criamos e calibramos às respetivas variáveis globais do robô, para que deste modo não houvesse dúvidas de qual ferramenta e/ou base estaríamos a utilizar.

```
ROBOT TOOL
$TOOL = TOOL_DATA[2]
$BASE = BASE_DATA[1]
```

Figura 20 – Atribuição da ferramenta e da base a serem utilizadas pelo robô.

As inicializações eram a segunda tarefa feita pelo robô, e certificavam que todos os valores das variáveis estavam bem definidos e reinicializados, e que todos os pontos da estrutura tinham o valor de X e Y a zero para serem capazes de receber novas coordenadas e não serem influenciados pelas antigas. Era também atribuído um valor a Z, A, B e C, para que a altura da ferramenta e a sua posição fossem idênticas em todos os pontos da estrutura. Para além de alterar o valor dos pontos, era também alterado o valor do tempo que as notas deviam de ser tocadas.

```
INITIALIZATIONS
;RESET VARIABLES
KUSTIC_EMERGENCY2 = TRUE
KUSTIC_START = FALSE
KUSTIC_CALIBRATE = FALSE

;RESET STRUCS
FOR I=1 TO MUSIC_SIZE
  POINTS[I].POINT = {X 0.0, Y 0.0, Z 35.0, A -110, B 0.0, C 169}
  POINTS[I].TEMPO = 0
ENDFOR

;INTERRUPTS
INTERRUPT DECL 1 WHEN $IN[1]==FALSE DO EMERGENCY()
INTERRUPT DECL 2 WHEN KUSTIC_EMERGENCY2==FALSE DO EMERGENCY()
INTERRUPT ON

;ADVANCE
$ADVANCE = 3
```

Figura 21 – Inicializações de variáveis, estruturas e interrupções.

Era aqui, que as interrupções eram declaradas e ativadas, sendo que no nosso código tínhamos apenas duas, que executavam a mesma função e serviam como paragem de emergência. Quando o botão que estava fisicamente ligado ao robô ou o botão de emergência da GUI fossem pressionados, estas interrupções iriam ser ativadas e o robô iria parar de imediato qualquer movimento que estivesse a ser feito.

```
DEF EMERGENCY ()

BRAKE
WHILE KUSTIC_EMERGENCY==FALSE
;WAIT FOR EMERGENCY TO END
ENDWHILE

END
```

Figura 22 – Função de emergência executava nas interrupções.

Depois de feitas as inicializações o robô executava a função HOME que o deslocava-se para o ponto *HOME_KUSTIC*, através de um movimento ponto a ponto (PTP), onde aguardava a ordem de arranque. O ponto *HOME_KUSTIC* estava definido localmente no ficheiro de declarações (.dat).

Neste movimento era limitada a velocidade do movimento PTP a 10% da velocidade de funcionamento do robô, visto que era um movimento longo e poderia causar a deterioração da calibração, devido à base onde o robô se encontrava, e porque só era realizado no início e no fim do código tornando o seu impacto, em termos de tempo e eficiência, negligenciável.

```
DEF HOME ()

PTP HOME_KUSTIC Vel=10 % PDAT19

KUSTIC_PLAYING_STATUS = FALSE

END
```

Figura 23 – Função para se deslocar para o ponto HOME_KUSTIC.

Uma vez no ponto HOME o robô envia uma mensagem para a consola a informar o utilizador do seu estado e fica à espera de receber a ordem de arranque. Quando essa ordem é dada, é feito o *reset* às variáveis de início e paragem, e *set* à variável do estado, para que o computador possa ser informado, que o robô está a executar o código.

```

;WAIT FOR START FROM INTERFACE
$LOOP_MSG[] = " WAITING FOR START "
WAIT FOR (KUSTIC_START)

;RESET START AND STOP
KUSTIC_START = FALSE
KUSTIC_STOP = FALSE
;SET STATUS BITS
KUSTIC_PLAYING_STATUS = TRUE

```

Figura 24 – Espera da ordem de arranque e reset/set de variáveis de estado.

O código principal está dentro de um ciclo *for* que vai de 1 até ao número máximo definido na variável `MUSIC_SIZE`, que no nosso caso é igual a 30. Neste ciclo é feita a verificação de duas variáveis, a `KUSTIC_STOP` e a valor do `TEMPO` da coordenada. Quando o utilizador dava a ordem de paragem, era alterado o valor da variável `KUSTIC_STOP` e era parado o ciclo. Da mesma forma, se a música chegasse ao fim, e o próximo ponto não fosse preciso de se tocar (valor do `TEMPO` igual a 0), o era também parado o ciclo.

A cada ponto, expeto o primeiro ou o último, era calculado o ponto intermédio, entre o ponto atual e o seguinte ponto, para que desta forma o movimento fosse mais fluído e eficaz. Depois o robô deslocava-se para o próximo ponto através de um movimento PTP e era chamada a função “STRUM”, responsável por tocar a corda.

```

;GO TROUGH ALL NOTES OF THE SONG
FOR I=1 TO MUSIC_SIZE

;IF STOP WAS PRESSED OR LAST NOTE WAS PLAYED EXIT LOOP
IF ((KUSTIC_STOP) OR (POINTS[I].TEMPO==0)) THEN
  EXIT
ENDIF

;IF THE CURRENT POINT ISN'T THE FIRST OR LAST
IF ((I > 1) AND (I < MUSIC_SIZE)) THEN
  ;CALCULATE THE POINT IN BETWEEN THE CURRENT POINT AND THE NEXT POINT
  AUX = POINTS[I].POINT
  AUX.X = (POINTS[I].POINT.X + POINTS[I+1].POINT.X) / 2
  LIN AUX CONT
ENDIF

;GO TO NOTE
PTP POINTS[I].POINT

;STRUM GUITAR STRING
STRUM()

ENDFOR

```

Figura 25 – Ciclo de movimentos do robô para as coordenadas enviadas.

A função STRUM, está dividida em duas partes. Isto é necessário porque existe a possibilidade de ter de tocar uma corda “aberta”, ou seja, a corda não pode ser pressionada o que requer diferentes tipos de movimento. Para tocar uma corda normal, o movimento consiste em baixar a ferramenta até à corda e de seguida incliná-la e rodá-la e desta maneira tocar a corda.

```
;PRESSED STRING
IF POINTS[I].POINT.X >= 0 THEN

    ;LOWER TOOL TO STRING
    LIN {Z -23} C_DIS

    ;ROTATE TOOL TO STRUM STRING
    LIN {A -118 ,C 170} C_DIS
```

Figura 26 – Movimento para tocar uma corda.

Para tocar uma corda aberta, o processo era semelhante, sendo que a diferença era reside na distância que a ferramenta baixava, sendo esta mais curta, e na sua inclinação, que tinha de ser mais acentuada.

```
;OPEN STRING
ELSE

    ;LOWER TOOL
    LIN {Z -12} C_DIS
    ;TILT TOOL WHILE NOT PRESSING STRING
    LIN {C 160} C_DIS

    ;ROTATE TOOL TO STRUM STRING
    LIN {A -118, C 165} C_DIS
ENDIF
```

Figura 27 – Movimento para tocar uma corda aberta.

De seguida era feita uma pausa no programa de acorda com o *tempo* da nota musical tocada e era levantada a ferramenta para que o processo pudesse ser repetido novamente.

```
;WAIT TO KEEP UP WITH THE TEMPO OF THE MUSIC
WAIT SEC POINTS[I].TEMPO

;LIFT TOOL
LIN{Z 10} C DIS
```

Figura 28 – Espera e elevação da ferramenta.

2.7 Simulação

Para acompanhar o processo de implementação a nível prático foi utilizado o software de simulação KUKA.Sim e foi reconstruído o ambiente do trabalho prático. O objetivo desta implementação era testar e validar conceitos e ter acesso de forma rápida e eficiente no sistema.

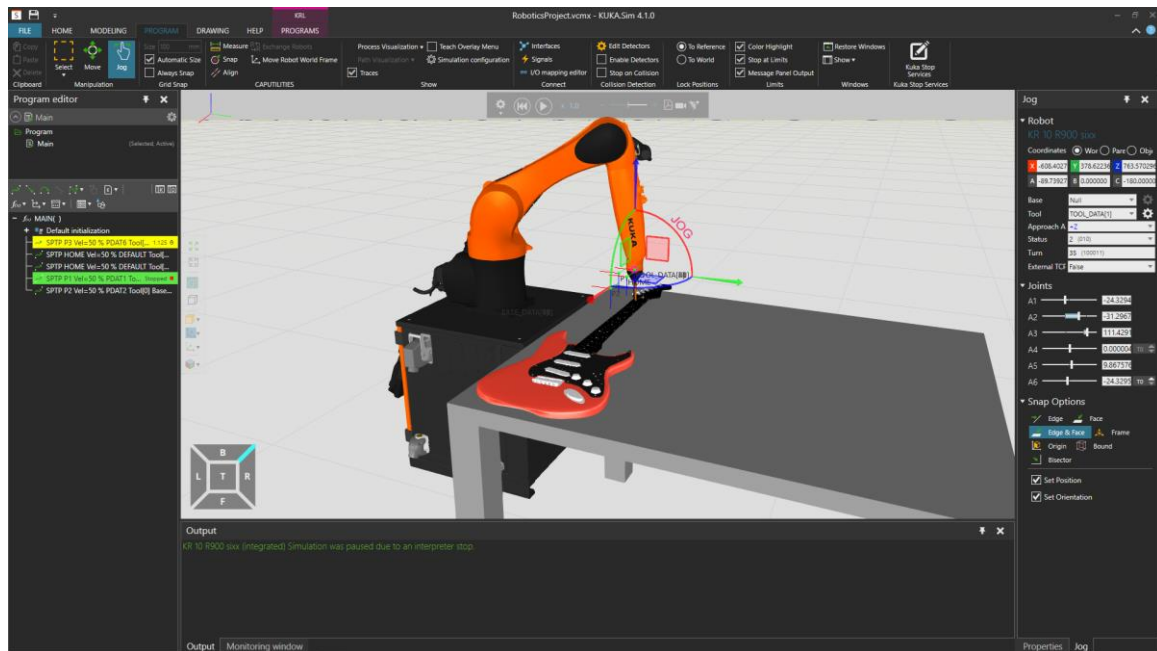


Figura 29 – Ambiente de Simulação.

No ambiente de simulação como se pode observar na Figura 29 foi possível escrever código e simular o comportamento da solução, com a ferramenta instalada para conseguir fazer a validação dos movimentos mais específicos, como por exemplo, os movimentos associados com a interação com a corda.

3 Conclusões

O presente trabalho faz uma adaptação dentro do ambiente da robótica para desenvolver capacidades humanas dentro das máquinas industriais. Com este projeto foi possível aplicar conceitos, técnicas e estratégias adquiridas ao longo do semestre, mesmo à medida que iam sendo lecionadas.

Existiu uma grande adversidade com a implementação dos movimentos do robô ao tocar a guitarra, devido à sua velocidade, e características técnicas como a rugosidade e pouca plasticidade do material TPU, o que fazia com que a reprodução de sons não fosse tão bem controlada. Estes problemas foram minimizados, mas não corrigidos por completo, sendo que ainda existem músicas que o robô não consegue tocar de uma maneira rigorosa.


A comunicação, cálculos e interface gráfica, desempenham bem o seu papel, sendo possível interagir com o robô de uma forma fácil e intuitiva. Para além destes aspetos o código do robô comportasse como bem e de forma espectável, mesmo com o problema referido anteriormente.

Para trabalho futuro, seria de enorme interesse colocar alguma componente de visão por computador, de forma a tornar todo o processo mais dinâmico. Seria também interessante continuar o desenvolvimento da interface gráfica, de forma que os utilizadores pudessem construir as suas próprias músicas, dando assim mais espaço à criatividade.

A nível de reprodução exitosa de músicas ainda é preciso fazer uma avaliação intensiva da melhor maneira de otimizar a ferramenta e a própria interação do robô com a guitarra com o objetivo de conseguir padronizar quais são as alterações necessárias da implementação para converter composições ou tablaturas standard em uma sequência que o robô consiga representar de maneira exitosa e seja facilmente identificável.

Essa avaliação passa por um estudo dos materiais adequados para conseguir representar de forma eficiente a textura da pele humana, com o objetivo de ter maior controlo sobre as cordas, ao mesmo tempo uma calibração muito específica para conseguir evitar a reprodução de notas desnecessárias por parte do robô devido à esta falta de eficiência dos materiais implementados na ferramenta

Bibliografia

- [1]  100 Classic Riffs! Only ONE Finger Needed! PDF with all TABs in description, (29 de dezembro de 2017). Acedido: 29 de julho de 2022. [Em linha Video]. Disponível em: <https://www.youtube.com/watch?v=zuD5N0eXfwA>
- [2] «0000293617_en.pdf». Acedido: 29 de julho de 2022. [Em linha]. Disponível em: https://www.kuka.com/-/media/kuka-downloads/imported/6b77eacafe542d3b736af377562ecaa/0000293617_en.pdf?rev=5b0ab04d25af46b981902f4c9b855573&hash=D3E50FB41CE09AAD958E92C2044C793F
- [3] «Explorando os Símbolos da Partitura - Aprenda Piano», *Aprenda Teclado*, 9 de junho de 2017. <https://www.aprendateclado.com/simbolos-da-partitura/> (acedido 29 de julho de 2022).
- [4] Dr_K, «Geometric Fret Layout--the Rule of 18», *Instructables*. <https://www.instructables.com/Geometric-Fret-Layout-the-Rule-of-18/> (acedido 29 de julho de 2022).
- [5] «KSS_82_SI_en.pdf». Acedido: 29 de julho de 2022. [Em linha]. Disponível em: https://elearning1.ipca.pt/2122/pluginfile.php/473970/mod_resource/content/0/KSS_82_SI_en.pdf
- [6] linuxsand, «linuxsand/py_openshowvar». 8 de julho de 2022. Acedido: 29 de julho de 2022. [Em linha]. Disponível em: https://github.com/linuxsand/py_openshowvar
- [7] GuitarQuarter, «Nomes das cordas da guitarra: notas e números», *Guitar Quarter*, 1 de maio de 2021. <https://guitarquarter.com/pt/guitarras/nomes-das-cordas-da-guitarra-notas-e-numeros/> (acedido 29 de julho de 2022).
- [8] Guitarriego, «Partes da guitarra elétrica e a importância de cada», *Guitarriego*, 11 de março de 2022. <https://guitarriego.com/pt/guias-pt/partes-da-guitarra-eletrica/> (acedido 29 de julho de 2022).
- [9] H. Jie, «py-openshowvar: A Python port of KUKA VarProxy client (OpenShowVar).» Acedido: 29 de julho de 2022. [Em linha]. Disponível em: https://github.com/linuxsand/py_openshowvar
- [10] «Qt Designer Manual». <https://doc.qt.io/qt-6/qt designer-manual.html> (acedido 29 de julho de 2022).
- [11] «Quarter Note vs. Crotchet: Rhythm notation in American English and British English». <https://musiker.com.au/blogs/blog-musiker/posts/crotchet-vs-quarter-note> (acedido 29 de julho de 2022).
- [12] «Reading Guitar Tabs for Beginners», *School of Rock*, 2 de abril de 2019. <https://www.schoolofrock.com/resources/guitar/reading-guitar-tabs-for-beginners> (acedido 29 de julho de 2022).
- [13] «Rule of 18 vs Rule of 17.817 – Guild of American Luthiers». https://luth.org/2021_0351300-buckland-ro18-direct/ (acedido 29 de julho de 2022).

[14] M. Dagani, «The Rule of 18 - It's Something to Fret About», *The Institute for Arts Integration and STEAM*, 1 de maio de 2017. <https://artsintegration.com/2017/05/01/rule-18-nothing-fret/> (acedido 29 de julho de 2022).

Anexo 1 – Esquemático da ferramenta

