Revista brasileira de iniciação científica
ISSN: 2359-232X

EDITOR DE COREOGRAFIAS PARA ROBÔ HUMANOIDE

COREOGRAPHY EDITOR FOR HUMANOID ROBOT

EDITOR DE COREOGRAFÍAS PARA ROBOT HUMANOIDE

Luiza Amador Pozzobon[1]
Frederico Menine Schaf[2]
Rodrigo da Silva Guerra[3]

Resumo: A necessidade de comunicação é algo quase intrínseco de robôs humanoides por serem utilizados tipicamente em situações sociais e interativas. Nesse sentido, é necessária a existência de softwares para edição de seus movimentos e falas. Assim, este projeto teve como objetivo o desenvolvimento de uma ferramenta *web* atraente e intuitiva para a criação de cenários de interação com um robô humanoide, também chamadas de coreografias, de forma que mesmo leigos possam programar suas ações. O *website* funciona totalmente independente do robô físico, controlando um avatar, podendo, ainda, estar conectado a primeiro e controlá-lo *online*.
Palavras-chave: Website. Robô. Controle. Interação. Coreografias.

Abstract: The need for communication is something almost intrinsic to humanoid robots because of their typical usage in social and interactive situations. In this sense, the existence of softwares for editing its movements and speeches is necessary. Thus, this project aimed at developing an attractive and intuitive web tool to create scenarios of interaction with a humanoid robot, also called choreographies, so that even lay people can program their actions. The website works completely independent of the physical robot, controlling an avatar, but can also connect to the first and control it online.
Keywords: Website. Robot. Control. Interaction. Choreographies.

Resumen: La necesidad de comunicación es algo casi intrínseco de robots humanoides pues han sido utilizados típicamente en situaciones sociales e interactivas. En este sentido, la existencia de softwares para la edición de sus movimientos y conversaciones es necesaria. De esa manera, este proyecto tuvo como objetivo el desarrollo de una herramienta web atractiva e intuitiva para la creación de escenarios de interacción con un robot humanoide, también denominadas coreografías, de manera que incluso laicos puedan programar sus acciones. El sitio web funciona totalmente independiente del robot físico, y puede aún conectarse a él y controlarlo en línea.
Palabras-clave: Website. Robot. Control. Interacción. Coreografía.

Envio: 20/04/2019 **Revisão: 22/04/2019** **Aceite: 05/07/2019**

[1] Graduanda em Engenharia de Controle e Automação. Universidade Federal de Santa Maria.
E-mail: luiza.pozzobon@gmail.com.
[2] Professor do Departamento de Processamento de Energia Elétrica. Universidade Federal de Santa Maria.
E-mail: frederico.schaf@ufsm.br.
[3] Professor do Departamento de Processamento de Energia Elétrica. Universidade Federal de Santa Maria.
E-mail: rodrigo.guerra@ufsm.br.

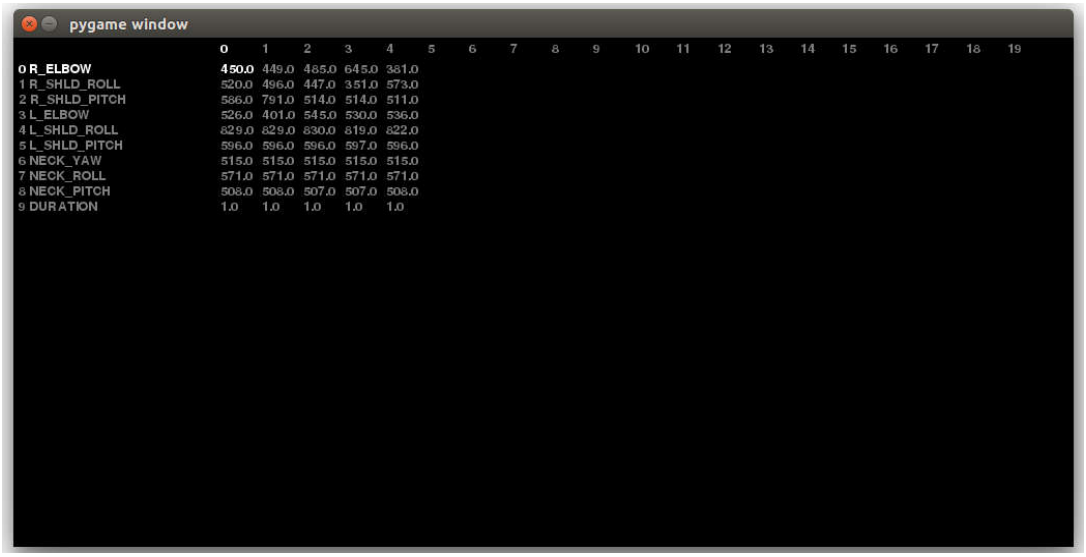
Revista brasileira de iniciação científica
ISSN: 2359-232X

Introdução

Robôs humanoides, muito utilizados em situações sociais e interativas (Shamsuddin et al. 2012; Alemi et al. 2016; Scassellati 2003), necessitam não só de aparência semelhante a dos seres humanos, mas também de capacidades de comunicação e de movimentação (Breazeal 2003). Tais características garantem a empatia e a conexão emocional com a pessoa com a qual o robô se comunica, em alguns casos incentivando a interação social (Robins et al., 2005).

Com essa necessidade de comunicação intrínseca desse tipo de máquina, a empresa *Qiron Robotics*, incubada na Incubadora Tecnológica de Santa Maria, fabricante do robô humanoide Beo, já possuía um *software* de edição de movimentos próprio, visualizado na Figura 1. Entretanto, esse era pouco amigável para novos usuários, tanto por funcionar primariamente no terminal de comandos do sistema operacional, quanto por operar por comandos do teclado. Além disso, não só a criação de arquivos de áudio com a voz característica do Beo, mas também a integração dos movimentos criados com as falas e outras funcionalidades dele só era possível através de comandos diretos na linguagem de programação Python. A programação de um cenário de integração de movimentos, falas e expressão dos olhos do robô é observado na Figura 2.

Figura 1 – Editor de movimentos da empresa Qiron Robotics.



Fonte: *Qiron Robotics*.

Revista brasileira de iniciação científica
ISSN: 2359-232X

Figura 2 – Programação de um pequeno cenário para o robô humanoide Beo, que reúne arquivos de movimento, falas e expressão dos olhos para transmitir uma mensagem de agradecimento.

```
def agradecer():
    """
    Função que utiliza áudio, olhos e movimentos para expressar agradecimento.

    """
    audio.play_mp3("audio/cenario/obrigado_alimentar.mp3") # muito obrigado por me alimentar
    eyes.set_expression('default')
    movements.play_motion_file("moves/078eu.dat")
    sleep(3)
```

Fonte: *Qiron Robotics*.

Tendo em vista as propriedades apresentadas, é desejável que a programação destes movimentos e falas seja intuitiva e de fácil aprendizagem, de modo que mesmo leigos consigam criar cenários de interação com o robô em poucos minutos. Optou-se que essa programação fosse baseada em aplicações *web*, devido à alta escalabilidade, aplicabilidade, possibilidades de popularização de seu uso e capacidade de adaptação para diversos dispositivos, como celulares, computadores e *tablets*. Embora existam exemplos de *websites* que auxiliam a criação de novos movimentos ou tarefas para robôs industriais (Papadopoulos et al. 2017), para os robôs humanoides e sociais não foram encontradas ferramentas semelhantes baseadas na *web* com interface intuitiva para leigos. Nesse caso, outras soluções foram estudadas, como a geração de movimentos a partir de animação gráfica (Balit, Vaufreydaz e Reignier 2016), a criação online com códigos em Python (Casan et al. 2015) ou a ferramenta *Choreographe* para o famoso robô humanoide Nao, que funciona *offline* e suporta, também, programação em Python (Pot et al. 2009).

Desse modo, com este projeto, buscou-se o desenvolvimento de um *website* atraente que reunisse ferramentas de criação de falas, de movimentos e de coreografias para o robô humanoide Beo da empresa *Qiron Robotics*. O conceito de comunicação para o Beo tem como base a criação de coreografias para os cenários de interação. Essas coreografias podem ser compostas por falas, movimentos de braços, cabeça ou rodas, expressão dos olhos e tempo de inatividade. Além disso, o *website* possui uma tela de controle remoto para utilização com o conceito de "Mágico de Oz" (Hüttenrauch et al. 2006), onde um humano controla o robô em tempo real a partir dos arquivos criados no ambiente de simulação.

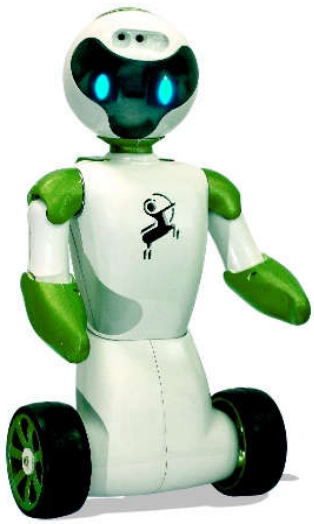
Revista brasileira de iniciação científica
ISSN: 2359-232X

Metodologia

A fim de atingir o objetivo final do controle virtual do robô humanoide Beo, esse projeto é constituído por três partes fundamentais: (1) o robô a ser controlado; (2) o *website* responsável pelo seu controle, criação e edição de coreografias; e (3) as ferramentas responsáveis pela interligação das duas tecnologias. Visto que o robô já havia sido desenvolvido pela empresa *Qiron Robotics*, o foco do trabalho reside nos dois últimos tópicos, o que não exclui a necessidade de intervenções e desenvolvimento no primeiro. Dessa forma, a seguir serão expostas tanto as principais funcionalidades do *site* editor de coreografias, quanto de como ocorre a conectividade do mesmo com o Beo.

O Beo, evidenciado na Figura 3, é um robô humanoide de aproximadamente 42 cm de altura com nove graus de liberdade, sendo três em cada braço e três na cabeça. Além disso, possui duas rodas, um sensor de toque em sua cabeça, câmera, microfone e alto-falante. Seu *software* foi desenvolvido na linguagem *Python* e possui como "cérebro" o microprocessador *Raspberry Pi 3.*

Figura 3 – O robô humanoide Beo.



Fonte: *Qiron Robotics*.

Revista brasileira de iniciação científica
ISSN: 2359-232X

O *website*, por sua vez, foi desenvolvido a partir do MEAN *Stack*, uma pilha de *softwares JavaScript* constituída pelo banco de dados MongoDB, pelo ExpressJS, responsável pela conectividade, pelo Angular[4], um *framework* para desenvolvimento *front-end*, e pelo NodeJS para desenvolvimento *back-end*. O *site* construído possui cinco telas principais de criação ou edição, sendo elas: (1) a de movimentos, (2) a de fala, (3) a de coreografias, (4) a de controle total e (5) a de customização dos botões. As coreografias podem ser compostas por movimentos, falas e outras coreografias criadas previamente, além de permitir o controle das rodas, da animação dos olhos e do tempo de inatividade do robô.

A organização dos códigos base do *site* seguiu o modelo de *design* baseado em componentes, de acordo com as funcionalidades do *framework front-end* utilizado. Dessa forma, todas as telas principais citadas são componentes, que, na maioria dos casos, possuem outros componentes dentro de si. Essa peça, por sua vez, é formada por três arquivos distintos: um na linguagem de programação *TypeScript*, um em HTML e outro em CSS.

Os dois últimos arquivos citados são as linguagens de marcação padrão do desenvolvimento *web*, que fornecem as peças básicas para a apresentação da informação nos *sites*, como caixas de texto, botões e a personalização desses. O primeiro, por sua vez, é onde os dados apresentados nos dois últimos podem ser manejados e tratados de forma mais completa. Ou seja, o arquivo *TypeScript* tem conexão direta ao arquivo HTML, tornando possível a ligação bidirecional dos dados. Com isso, quando os usuários interagem com a interface *web* (na parte que eles entram em contato direto, o HTML), também alteram as variáveis criadas no arquivo *TypeScript*.

Conforme comentado, um componente pode ser formado por diversos outros. Com o intuito de facilitar a comunicação entre essas peças, existem também arquivos chamados de "serviços". Esses fornecem funcionalidades não necessariamente ligadas à visualização do *website*, tendo seu uso atribuído principalmente ao tratamento e à organização dos dados obtidos pelos componentes.

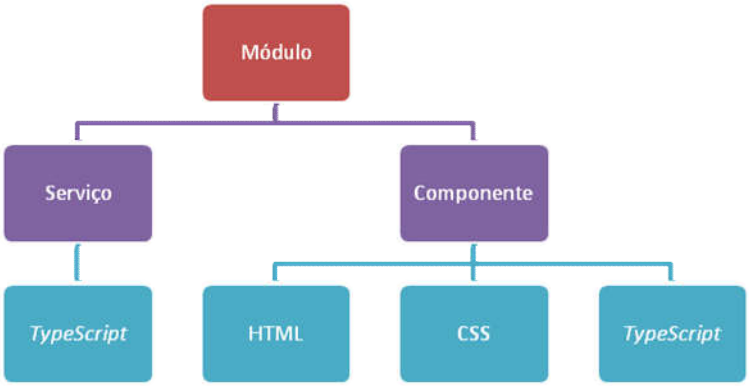
O Angular permite, ainda, a criação de módulos. Módulos tem o intuito da formação de unidades funcionais através do uso conjunto de componentes e serviços para a viabilização de uma determinada aplicação. Assim como os serviços, o uso de módulos aumenta ainda

[4] https://angular.io/docs

Revista brasileira de iniciação científica
ISSN: 2359-232X

mais a eficiência e a possibilidade de reutilização do código em aplicações complexas. Salienta-se que serviços de quaisquer módulos podem ser utilizados dentro de outros módulos, permitindo códigos não só reutilizáveis, mas também eficientes. Essa regra, entretanto, não se aplica aos componentes, que podem apenas fazer parte de uma relação pai-filho com outros componentes. Na Figura 4 observam-se os tipos de arquivos presentes em cada um dos elementos de um módulo.

Figura 4 – Elementos presentes em um módulo e os arquivos atrelados a esses.



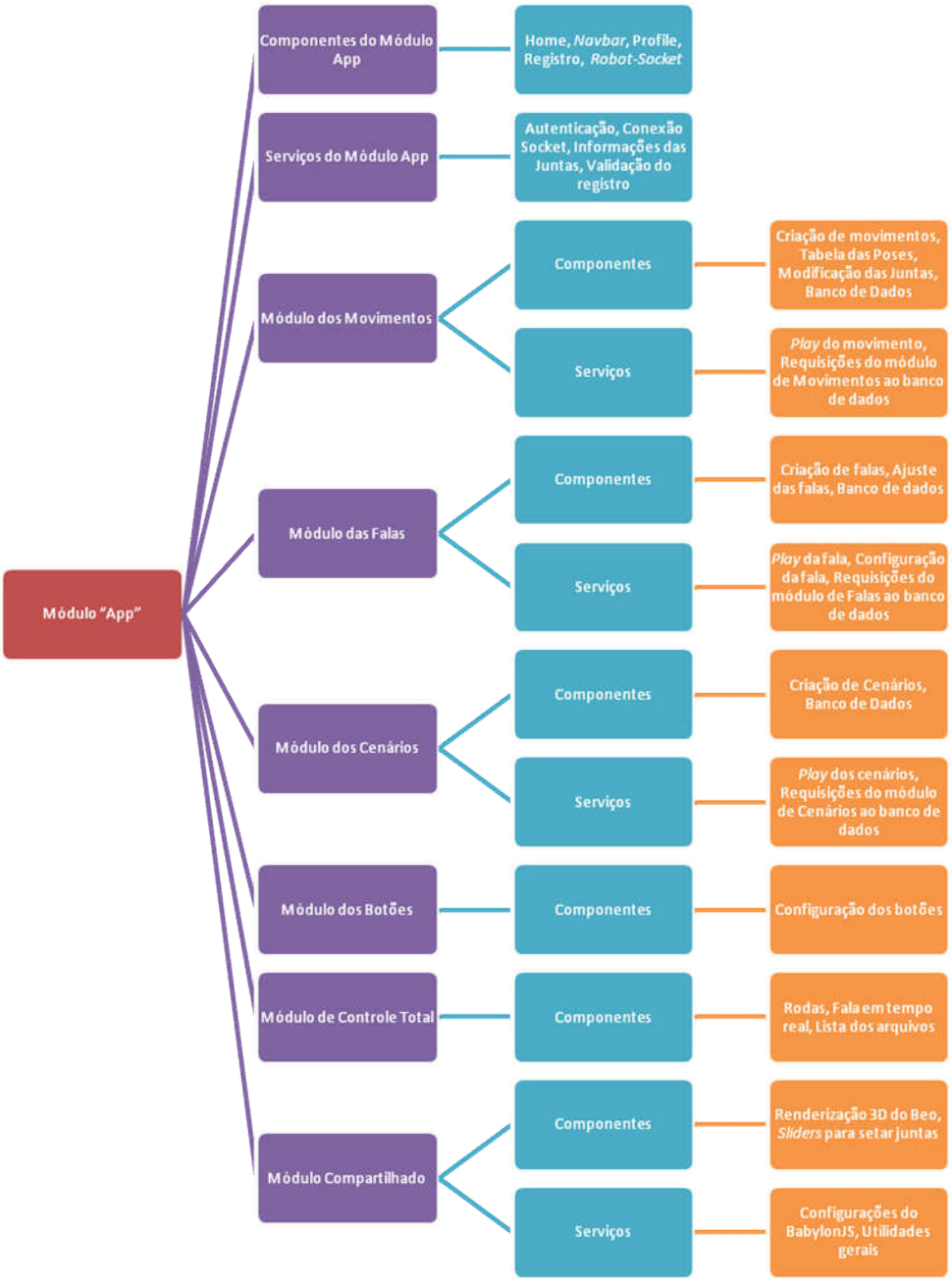
Fonte: autora.

Além disso, com a separação do código em módulos, pode-se fazer uso da ferramenta de *lazy-loading* – ou carregamento preguiçoso, em tradução literal – que é o carregamento dos da ferramenta citada de acordo com a demanda do usuário. Em vista disso, nem todos os módulos são inicializados com o *site*, o que aumenta sua velocidade de carregamento inicial.

O *website* tratado por esse artigo, portanto, faz uso de todas as ferramentas citadas acima: componentes, serviços, módulos e o *lazy-loading*. Cada grande funcionalidade está em um módulo, que, por sua vez, possui uma tela principal composta, no geral, por diversos componentes. Dessa forma, existem sete módulos no *site*, sendo um deles o módulo principal do aplicativo, com as funcionalidades básicas como tela de *login*, registro e serviços de conectividade. Os outros módulos são: o de movimento, o de fala, o de cenários, o de botões, o de controle total e o módulo compartilhado. A estrutura do *site*, composta pelos sete módulos e seus respectivos componentes e serviços, é visualizada na Figura 5.

Revista brasileira de iniciação científica
ISSN: 2359-232X

Figura 5 – Diagrama estrutural do *website*. O módulo principal, App, invoca os outros seis módulos. Cada módulo possui componentes e serviços próprios. Os serviços podem ser utilizados por qualquer módulo, mas os componentes só podem ser utilizados dentro de seu módulo de origem. A invocação do "Módulo Compartilhado" por outros módulos possibilita a reutilização de componentes.



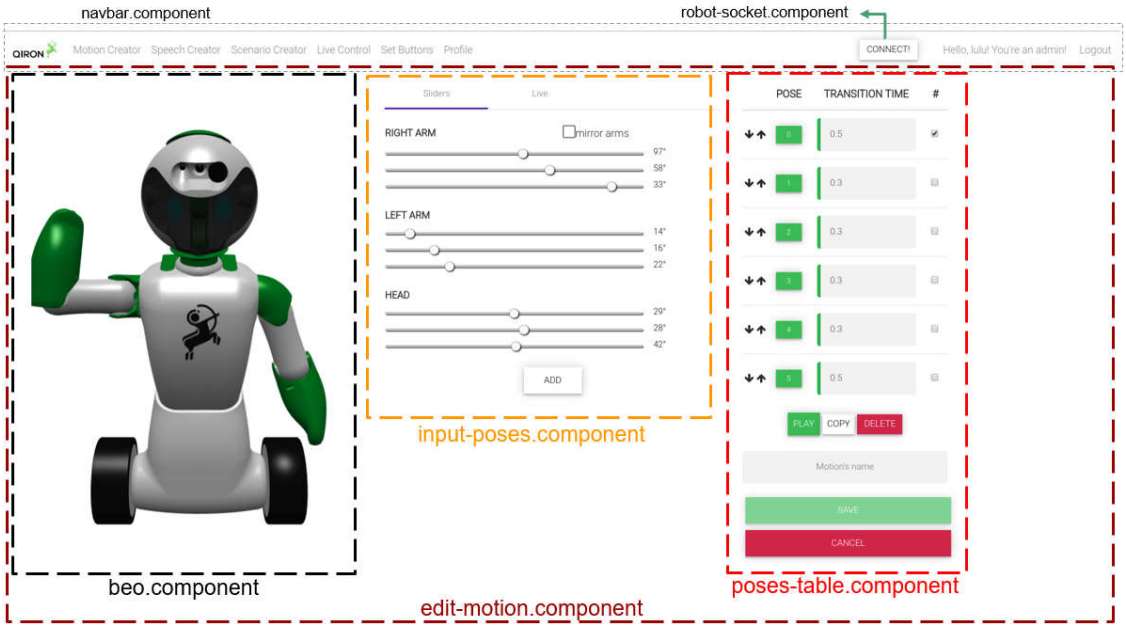
Fonte: autora.

Revista brasileira de iniciação científica
ISSN: 2359-232X

Os módulos de fala, movimento e cenários tem a mesma estrutura geral: uma região para acesso aos arquivos criados anteriormente (que foram salvos no banco de dados), e outra para a criação de novos arquivos. Para os módulos de movimento e cenários, essas duas regiões estão dispostas em telas distintas, enquanto que, para o de fala, tanto os arquivos existentes quanto a parte de criação de novas falas estão na mesma tela.

A tela de criação de movimentos é a que possui o maior número de componentes do *site*. Para essa tela, faz-se uso de quatro componentes, além daquelas de uso global (barra de navegação e seção de conectividade), sendo um apenas para acomodação e organização dos outros. Na Figura 6 observa-se a disposição desses na tela.

Figura 6 – Disposição de diferentes componentes em uma única tela. No caso, a tela de edição ou criação de movimentos.



Fonte: autora.

Por outro lado, a tela de criação de cenários é uma das mais complexas. Nela, todos os arquivos já criados pelos usuários são reunidos, além da adição de controle de outras funcionalidades do robô, como as rodas e os olhos. Assim, esse componente faz uso de serviços de outros três módulos, de movimentos, falas e a renderização 3D do robô. Com isso,

Revista brasileira de iniciação científica
ISSN: 2359-232X

é possível não só observar a movimentação programada para o robô na animação presente na tela, mas também escutar a fala gerada anteriormente, conforme a programação do usuário.

Todo tipo de movimentação realizado no robô real é passível de reprodução na renderização 3D presente nas telas, que foi modelado no *software Blender* e exportado para a *web* com auxílio do *framework BabylonJS*. O usuário pode movimentar não só as nove juntas, correspondentes aos braços e cabeça, mas também as rodas. Além disso, também é possível observar a expressão dos olhos desejada nas telas de cenários e de controle total. A movimentação de cada uma das juntas citadas anteriormente é obtida através da Equação 1, para as do braço direito, através da Equação 2 para as do braço esquerdo e pela Equação 3 para a cabeça. Sendo que "*Pmin"* é a posição mínima, "*Pmax"*, a posição máxima e "*Patual*" a posição atual da junta de acordo com a padronização adotada pela *Qiron Robotics* para os motores de movimentação do Beo. Salienta-se que algumas das equações possuem sinais variáveis para juntas diferentes, conforme explicitado.

As rodas, por sua vez, são movimentadas através da rotação em seu eixo. Todas as possibilidades de expressão dos olhos estão na renderização, e a modificação da expressão atual na renderização é obtida através da ativação da expressão desejada e desativação de todas as outras.

$$Posição\ da\ junta\ 3D\ (Braço\ Direito) = \pm \frac{360\ (Pmin - Patual)}{1024} \frac{\pi}{180} \quad (1)$$

$$Posição\ da\ junta\ 3D\ (Braço\ Esquerdo) = \frac{360\ (Pmax - Patual)}{1024} \frac{\pi}{180} \quad (2)$$

$$Posição\ da\ junta\ 3D\ (Cabeça) = \frac{360\ \left(Pmax - \frac{(Pmax - Pmin)}{2} \pm Patual\right)}{1024} \frac{\pi}{180} \quad (3)$$

Embora o *website* possibilite a criação de movimentos, falas e cenários sozinho devido à existência de uma renderização 3D do Beo na maioria das telas, julgou-se fundamental existir a interação do mesmo com o *hardware*, ou seja, com o Beo físico. Essa ligação não só permite seu uso em uma maior gama de momentos, mas também intensifica a experiência de

Revista brasileira de iniciação científica
ISSN: 2359-232X

criação, ao viabilizar o teste *online* e ao vivo no robô de todos os arquivos gerados. Para isso, tanto os robôs físicos, quanto as instâncias do *website* foram designadas como clientes, que se conectam a um servidor TCP, responsável pelo gerenciamento da conexão e transmissão dos dados.

A fim de garantir a robustez da interação do *website* com o robô, cada um desses pode estar conectado a apenas uma instância do *site*. As mensagens são transmitidas em formato JSON, com quatro campos: origem, destino, ação e dados. No campo de ação, está a função a ser executada pelo *website* ou pelo robô e, nos dados, quaisquer informações adicionais que essa ação necessite ou a resposta a uma requisição prévia. Tanto o cliente por parte do robô, quanto o servidor foram desenvolvidos em *Python*, enquanto o cliente, pela parte do *website*, em *JavaScript*, como ilustrado na Figura 7.

Figura 7 – Diagrama de conexão do projeto.

Fonte: autora.

O *website* conta, também, com um sistema de usuários: os comuns e os administradores. Enquanto que usuários comuns tem controle total sobre seus arquivos

Revista brasileira de iniciação científica
ISSN: 2359-232X

gerados, podendo excluí-los e editá-los quando desejarem, administradores possuem não só prioridade de conectividade aos robôs físicos, mas também são capazes de tanto excluir como editar arquivos de outros usuários. Se um usuário está conectado a um robô e um administrador solicita conexão ao mesmo, ambos recebem um aviso na tela sobre essa disputa. Se o administrador requisitar novamente a conexão, então o usuário é desconectado do robô para dar espaço ao primeiro. O manejo da conectividade aos robôs é realizado no servidor *socket* mencionado anteriormente. Na Tabela 1 observa-se os casos de uso das classes de usuários para conexão ao robô, e na Tabela 2 para a criação ou edição de arquivos no *website*.

Tabela 1 – Prioridade de conexão dos tipos de usuário ao robô físico Beo.

Conectado ao Robô	**Requisição de conexão**	**O que acontece**
Usuário 1	Usuário 2	Usuário 1 mantém a conexão.
Usuário	Administrador	Administrador pode assumir a conexão.
Administrador	Usuário	Administrador mantém a conexão.
Administrador 1	Administrador 2	Administrador 2 pode assumir a conexão.

Fonte: autora.

Tabela 2 – Possibilidades de edição dos arquivos do site de acordo com os tipos de usuário.

Criou o arquivo	**Requisição de edição**	**O que acontece**
Usuário 1	Usuário 2	Usuário 2 não consegue alterar o arquivo. Pode renomeá-lo e salvar como seu próprio.
Usuário	Administrador	Administrador pode editar o arquivo.
Administrador	Usuário	Usuário 1 não consegue alterar o arquivo. Pode renomeá-lo e salvar como seu próprio.
Administrador 1	Administrador 2	Administrador 2 pode editar o arquivo.

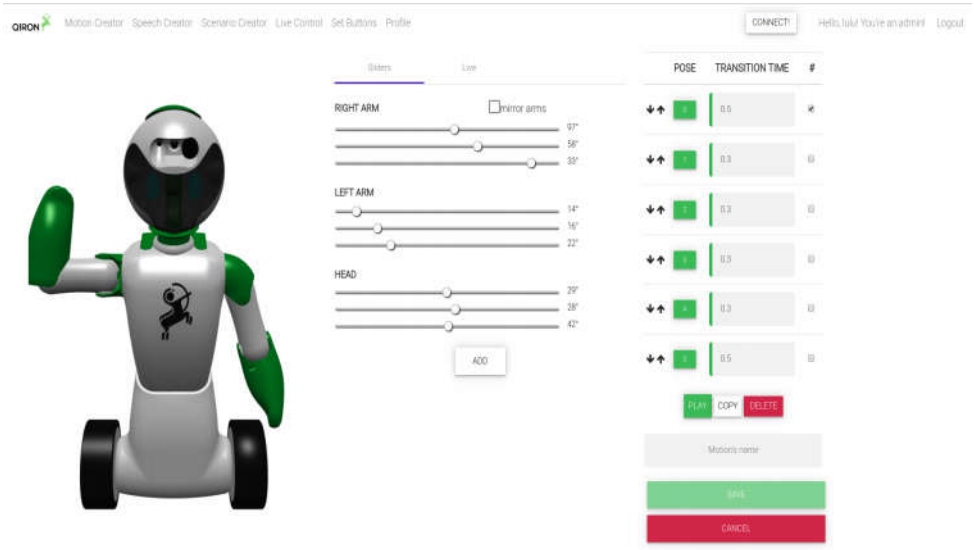
Revista brasileira de iniciação científica
ISSN: 2359-232X

Fonte: autora.

Resultados e Discussão

Um dos elementos principais das coreografias são os movimentos, que determinam a capacidade de expressão física do robô humanoide utilizado. Cada movimento é composto por poses, e cada uma tem a sua duração, que é, naturalmente, o tempo necessário para ir da pose anterior à pose atual. Em todas as poses é possível alterar o posicionamento das nove juntas do robô, três para cada braço e três para a cabeça, que variam, no geral, de 0 a 90º ou 180º, com exceção das juntas da cabeça. Além disso, em qualquer momento da criação do movimento, o usuário pode visualizar não só a execução completa do mesmo em uma renderização 3D do Beo na tela (vide Figura 8), mas também cada uma das poses já adicionadas. A fim de garantir maior facilidade de uso e experimentação, existem também botões de cópia, deleção ou alteração da ordem das poses.

Figura 8 – Tela de criação de movimentos. À esquerda a simulação 3D do Beo, ao centro os botões deslizantes para ajustar a posição das juntas, e à direita as poses que compõem o movimento.



Fonte: autora.

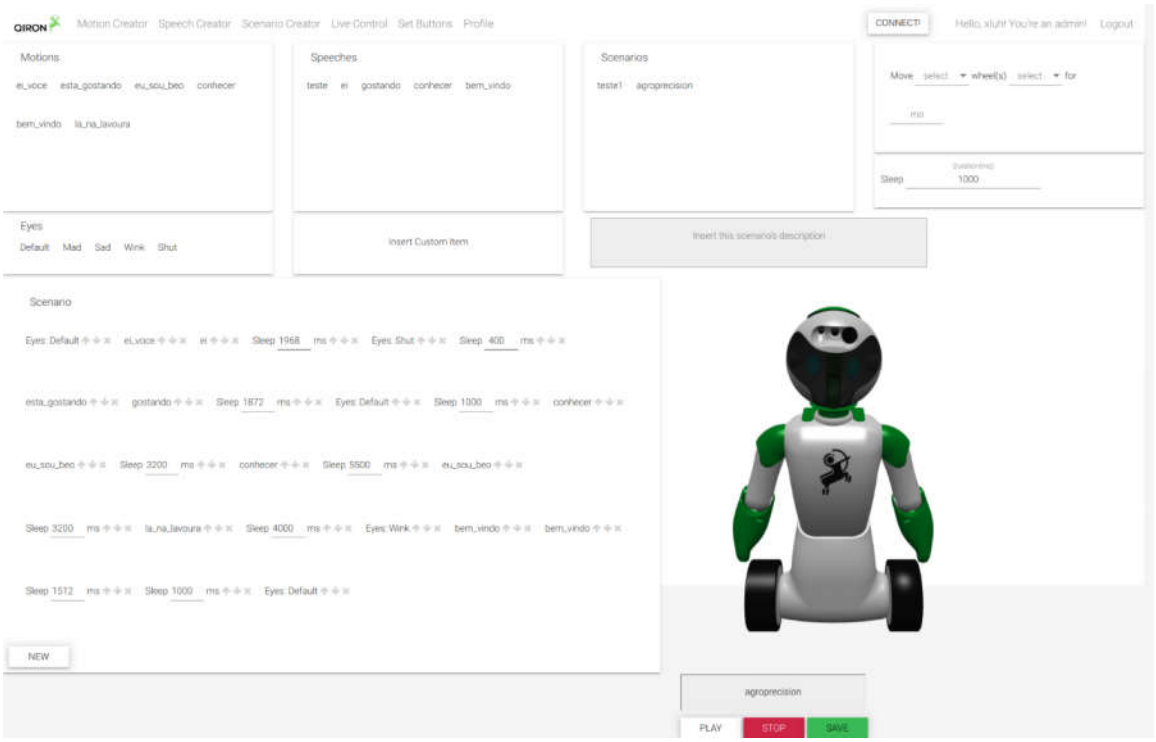
O segundo elemento fundamental para a criação de coreografias são as falas, que garantem a expressão verbal do robô. No geral, são combinadas a movimentos durante as

Revista brasileira de iniciação científica
ISSN: 2359-232X

coreografias. As mesmas são geradas a partir do *Google Text-to-Speech* (gTTS), podendo ser criadas em todos os idiomas suportados pela ferramenta. Elas são produzidas automaticamente com a voz padrão do Beo, porém o usuário pode alterar suas velocidade e tonalidade de acordo com seu gosto.

Com arquivos de movimentos e falas existentes, o usuário pode proceder para a tela de criação de coreografias, que em sua primeira versão conta com interface planejada a partir do conceito de arrastar e soltar, como o ilustrado na Figura 9. Nessa tela estão presentes todos os movimentos, falas e outras coreografias já criadas, além das ferramentas de movimentação de rodas e expressão dos olhos citadas anteriormente. Basta, portanto, arrastar os itens desejados para uma grande caixa presente na tela. A execução do cenário é realizada de acordo com a ordem de leitura dos itens soltos nessa caixa, que podem ser reordenados conforme desejados. Assim como na tela de movimentos, é possível visualizar na renderização 3D do Beo a execução da coreografia em criação.

Figura 9 – Tela de criação de coreografias.



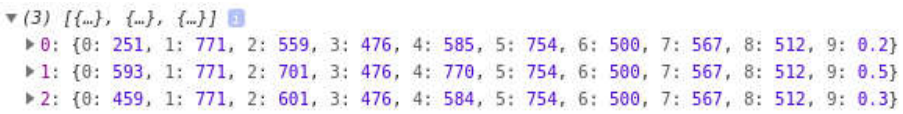
Fonte: autora.

Revista brasileira de iniciação científica
ISSN: 2359-232X

Os arquivos gerados tanto pela tela de criação de movimentos, quanto pela de cenários podem ser, então, baixados em formato JSON pelo usuário nas respectivas telas dos bancos de dados dos arquivos. Se o usuário estiver conectado a um robô no momento do *download* desses arquivos, eles são automaticamente baixados ao robô também.

O arquivo de movimento é uma lista onde cada elemento é um dicionário correspondente a uma pose, conforme apresentado na Figura 10. Cada um dos dicionários, por sua vez, é composto por dez itens: nove são os valores dos posicionamentos dos motores do robô, numerados de zero a oito, e o último, de valor nove, é o tempo de duração da pose. O padrão de organização e escrita desse arquivo é o escolhido pela empresa *Qiron Robotics*, de forma que o uso no robô a partir da geração pelo *site* é direto.

Figura 10 – Lista de poses que compõem um movimento.



Fonte: autora.

Os cenários gerados pela tela da Figura 9 possuem como fundamento a ordenação dos itens. Ou seja, o cenário é executado sequencialmente de acordo com a ordem disposta na tela. Assim, o arquivo gerado por essa tela também é uma lista, onde cada elemento pode ser algum dos itens de movimento, fala, rodas, olhos ou tempo de inatividade. Esse tipo de organização traz alguns problemas, como não só a falta de controle sobre o momento de execução dos itens (que leva a dificuldade de localização e correção de problemas na coreografia), mas também a própria desorganização visual para o usuário.

Além disso, fazia-se imprescindível para funcionamento dos cenários a utilização de tempos de inatividade para o robô, com o intuito de impedir que movimentos (ou falas) se sobrepusessem a outros. Por exemplo, ao colocar um arquivo de movimento seguido por um de fala, esses executavam ao mesmo tempo, o que não é um problema por si só, e, na maioria das vezes, é desejado. Entretanto, o próximo conjunto de movimento e fala também seria executado concomitantemente ao primeiro. Por isso, o usuário precisaria informar o tempo de espera para a execução do próximo item da lista.

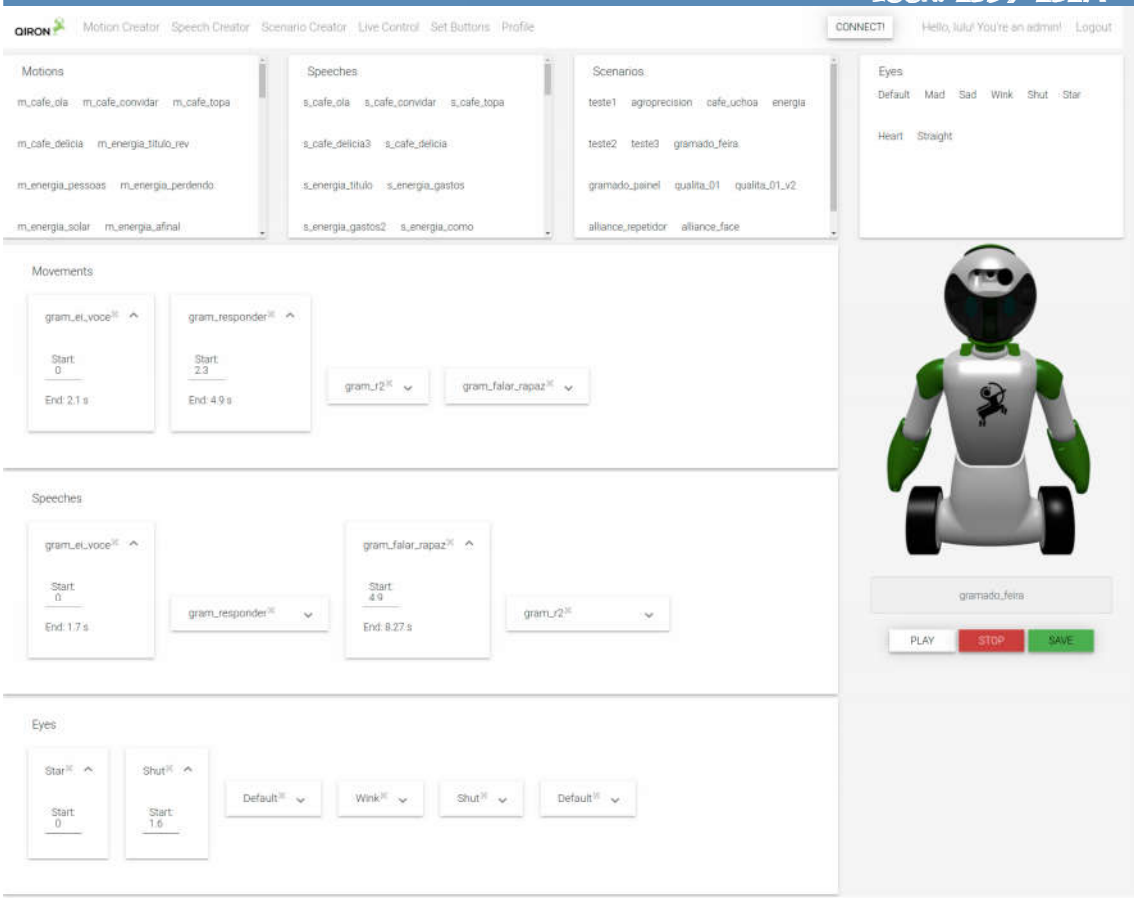
Revista brasileira de iniciação científica
ISSN: 2359-232X

Conforme recente demanda da *Qiron Robotics*, houve uma reformulação da tela de cenários, que pode ser vista na Figura 11. Agora, ao invés da execução ordenada dos itens arrastados para a caixa maior, os usuários definem um tempo de início para esses. Assim, não é mais necessária a utilização de tempos de inatividade para o robô, visto que é possível programar diretamente quando determinado item será executado. Esse novo conceito simplifica, ainda, a modificação ou correção dos cenários gerados pela facilidade de identificação do problema, que está atrelado ao início da execução dos itens, que é conhecido, e não à ordem da lista.

Nesse viés, houve também a reformulação do arquivo gerado pela tela de coreografias, cuja nova versão é observada na Figura 12. O arquivo gerado é, ainda, em formato JSON, mas é um dicionário composto por três campos: falas, movimentos e cenários. No campo de movimento estão todos os movimentos que compõem o cenário, de formato igual ao gerado pela tela de criação desse *site*, e seus respectivos IDs naquele cenário. Já para o campo de fala estão os textos das falas e seus IDs. O campo de cenário, por sua vez, possui uma lista com todos seus itens, onde cada um tem seu tipo (movimento, olho ou fala) e seu tempo de início. Os itens de movimento ou fala tem o ID para busca no campo principal desses, enquanto que os olhos possuem o estilo de olho do robô em texto (sendo eles, em livre tradução da autora: normal, piscada, fechados, retos, estrela, coração, triste ou bravo). Salienta-se que nem todas as animações dos olhos estão implementadas na renderização 3D e que, também, a inserção de cenários dentro de cenários ou movimentação de rodas ainda não são possíveis na nova tela.

Figura 11 – Nova tela de criação de cenários. O usuário insere os itens com movimentos de arrastar e soltar, que caem na caixa respectiva ao seu tipo: movimento, fala ou olhos. Ao clicar sobre cada item, ele se expande e o usuário pode digitar o tempo para início desse. Ao lado, a renderização 3D executa o cenário completo.

Revista brasileira de iniciação científica
ISSN: 2359-232X



Fonte: autora.

Revista brasileira de iniciação científica
ISSN: 2359-232X

Figura 12 – Arquivo de cenário JSON gerado pela nova tela de cenários. O novo arquivo se baseia em tempos de início para cada tarefa. As informações dos itens de movimentos e falar são armazenados em dicionários próprios, sendo invocados pelo dicionário de cenários pelo seu ID.


```
{
  "movements": [
    {
      "id": 1,
      "movement": "Acenar (Braço Direito)",
      "poses": [
        {
          "0": 373,
          "1": 577,
          "2": 513,
          "3": 513,
          "4": 504,
          "5": 818,
          "6": 595,
          "7": 499,
          "8": 554,
          "9": 1.5
        },
        { ...
        },
        { ...
        }
      ]
    }
  ],
  "speeches": [
    {
      "id": 1,
      "speech": "Olá",
      "language": "pt-BR",
      "text": "Olá, tudo bem? Meu nome é Beo."
    }
  ],
  "scenario": {
    "scenario": "Aceno com comprimento",
    "tasks": [
      {
        "eye": "default",
        "start": 0
      }
      {
        "motion": 1,
        "start": 0
      }
      {
        "speech": 1,
        "start": 1
      }
    ]
  }
}
```


Fonte: autora.

Revista brasileira de iniciação científica
ISSN: 2359-232X

Na tela de botões o usuário pode personalizar o que é executado por cada botão do robô. Assim, ali estão campos dos três botões da mochila e o sensor de toque da cabeça, seguidos de uma lista com todos os cenários já criados no *site*. O usuário pode personalizar eles individualmente, ou selecionar os cenários e modifica-los todos de uma vez só.

Por fim, existe a tela de controle total, cujo objetivo é controlar um robô Beo em tempo real a partir do *website* – técnica conhecida como "Mágico de Oz". Na primeira tela estão presentes todos os cenários, movimentos e falas criados, além da renderização do Beo, uma caixa de texto para geração em tempo real de novas falas, opções de expressões dos olhos e botões deslizantes para mudança de cada uma das nove juntas do robô. Além disso, é possível controlar as rodas através das teclas de setas do teclado. Em todas as telas descritas, é possível conectar-se pela rede a um Beo através de *sockets* de rede e todas as ações realizadas na configuração do *website* são executadas no robô.

Conclusão

O objetivo do projeto, criação do editor de coreografias *online* e intuitivo foi alcançado com êxito. Acredita-se na importância da iniciativa, visto que não foram encontradas ferramentas com características semelhantes, especialmente nos quesitos de conectividade e facilidade de uso combinados. Salienta-se a possibilidade de utilizar a ferramenta em qualquer dispositivo com navegador *web* e acesso à Internet. É possível utilizá-la para controlar um robô Beo em qualquer parte do mundo, devido a robustez da conexão cliente-servidor utilizada, além de permitir, também, a conexão através de redes locais.

Apesar da interface das telas do *website* ter caráter agradável e de clara usabilidade, espera-se, no futuro, deixá-las ainda mais acessíveis para uso leigo, especialmente a tela específica do editor de coreografias. Além disso, espera-se adicionar a ela todas as funcionalidades remanescentes, conforme comentado no texto. Vale destacar que o *website* e suas versões futuras não só serão incorporados aos produtos da empresa *Qiron Robotics*, mas também projetos de pesquisa, iniciação científica ou mestrado que o utilizarão para estudar interações humano-robô estão em curso na Universidade Federal de Santa Maria.

Revista brasileira de iniciação científica
ISSN: 2359-232X

Referências

ALEMI, Minoo et al. Clinical application of a humanoid robot in pediatric cancer interventions. **International Journal of Social Robotics**, v. 8, n. 5, p. 743-759, 2016.

BALIT, Etienne; VAUFREYDAZ, Dominique; REIGNIER, Patrick. Integrating animation artists into the animation design of social robots: An open-source robot animation software. In: **The Eleventh ACM/IEEE International Conference on Human Robot Interaction**. IEEE Press, 2016. p. 417-418.

BREAZEAL, Cynthia. Emotion and sociable humanoid robots. **International journal of human-computer studies**, v. 59, n. 1-2, p. 119-155, 2003.

CASAN, Gustavo A. et al. Ros-based online robot programming for remote education and training. In: **2015 IEEE International Conference on Robotics and Automation (ICRA)**. IEEE, 2015. p. 6101-6106.

HÜTTENRAUCH, Helge et al. Investigating spatial relationships in human-robot interaction. In: **2006 IEEE/RSJ International Conference on Intelligent Robots and Systems**. IEEE, 2006. p. 5052-5059.

PAPADOPOULOS, Christos et al. An Advanced Human-Robot Interaction Interface for Teaching Collaborative Robots New Assembly Tasks. In: **International Conference on Interactive Collaborative Robotics**. Springer, Cham, 2017. p. 180-190.

POT, Emmanuel et al. Choregraphe: a graphical tool for humanoid robot programming. In: **RO-MAN 2009-The 18th IEEE International Symposium on Robot and Human Interactive Communication**. IEEE, 2009. p. 46-51.

ROBINS, Ben et al. Robotic assistants in therapy and education of children with autism: can a small humanoid robot help encourage social interaction skills?. **Universal Access in the Information Society**, v. 4, n. 2, p. 105-120, 2005.

SCASSELLATI, Brian. Investigating models of social development using a humanoid robot. In: **Proceedings of the International Joint Conference on Neural Networks, 2003.**IEEE, 2003. p. 2704-2709.

SHAMSUDDIN, Syamimi et al. Humanoid robot NAO interacting with autistic children of moderately impaired intelligence to augment communication skills. **Procedia Engineering**, v. 41, p. 1533-1538, 2012.