Victor Luis Pires Kusumota

Desenvolvimento e Integração de um Sistema de Servidor web Baseado com Arquitetura em Nuvem para Robótica Educacional

São Carlos - SP

2019

Victor Luis Pires Kusumota

Desenvolvimento e Integração de um Sistema de Servidor web Baseado com Arquitetura em Nuvem para Robótica Educacional

Trabalho de Conclusão de Curso apresentado à Universidade Federal de São Carlos, como parte dos requisitos para obtenção do título de bacharel em Engenharia Elétrica

Universidade Federal de São Carlos – UFSCar

Centro de Ciências Exatas e de Tecnologia – CCET

Departamento de Engenharia Elétrica - DEE

Orientador: Prof. Dr. Rafael Vidal Aroca

Coorientador: Prof. Dr. Felipe Nascimento Martins

São Carlos - SP

2019

Victor Luis Pires Kusumota

Desenvolvimento e Integração de um Sistema de Servidor web Baseado com Arquitetura em Nuvem para Robótica Educacional

Trabalho de Conclusão de Curso apresentado à Universidade Federal de São Carlos, como parte dos requisitos para obtenção do título de bacharel em Engenharia Elétrica

São Carlos - SP, 05 de julho de 2019:

Prof. Dr. Rafael Vidal Aroca
Orientador

Prof. Dr. Osmar Ogashawara
Professor Convidado

Prof. Dr. André Carmona Hernandes
Professor Convidado

São Carlos - SP
2019

Dedico este trabalho aos meus pais, Mauricio e Florinda, que sempre me apoiaram e me motivaram durante toda minha jornada de estudos.

Agradecimentos

Agradeço, primeiramente, à toda minha família por sempre estarem ao meu lado, me apoiando e incentivando a dedicação aos meus estudos.

Agradeço aos meus orientadores Rafael Vidal Aroca e Felipe Nascimento Martins pela amizade, ensinamentos, suporte e orientação durante todo o processo de desenvolvimento deste trabalho. Agradeço também à Fundação de Amparo à Pesquisa do Estado de São Paulo (FAPESP) pelo suporte financiamento.

À Universidade Federal de São Carlos (UFSCar) pela oportunidade de participar do programa de bacharelado em Engenharia Elétrica.

Por fim, à todos os meus colegas de turma pela amizade e ajuda nos estudos durante todo o programa de graduação.

"Mesmo desacreditado e ignorado por todos, não posso desistir,
pois para mim, vencer é nunca desistir."
(Albert Einstein)

Resumo

A robótica é uma ferramenta atrativa e motivadora para alunos de qualquer nível. Dessa forma, o uso de um robô na educação pode ajudar a despertar o interesse de alunos acerca de diversos assuntos. Este trabalho apresenta o desenvolvimento e a implementação de uma plataforma educacional para um robô móvel chamado Cozmo, com diversos recursos, incluindo um servidor web para interface de usuário, visão computacional, reconhecimento de voz, controle de rastreamento de trajetória, serviços em nuvem, entre outros. Funções para fins educacionais foram implementadas no sistema desenvolvido, incluindo operações matemáticas, ortografia, direção e um módulo de perguntas-respostas para os professores elaborarem roteiros didáticos (*scripts*) com flexibilidade. No projeto proposto, uma ferramenta de reconhecimento de voz em nuvem foi implementada para tornar o sistema interativo com o robô mais realista e natural. Além disso, um sistema de visão computacional em nuvem foi utilizado para efetuar o reconhecimento de objetos em fotos tiradas pela câmera de robô. Funções foram criadas com o objetivo de controlar as emoções e os motores de Cozmo para criar roteiros mais criativos e sofisticados. Para executar os *scripts*, foi desenvolvido um algoritmo interpretador que traduz as funções criadas no projeto para a linguagem de programação do Cozmo. Para validar este trabalho, o projeto foi apresentado à alguns professores do ensino fundamental (turmas com alunos entre 4 e 12 anos). Os resultados obtidos são relatados neste trabalho e indicam que o sistema proposto pode ser uma ferramenta educacional útil. Por fim, com o objetivo de tornar o projeto mais acessível, foi desenvolvido um sistema mais simplificado que utiliza um Raspberry PI para dar os comandos ao Cozmo, substituindo assim a necessidade da utilização de um computador.

Palavras-chave: Cozmo. servidor web. robótica educacional.

Abstract

Robotics is an attractive and motivating tool for students of any level. Using robots in education can help to arouse students' interest in many subjects. This work presents the development and implementation of an educational platform for a mobile robot called Cozmo, with several features, including web server for user interface, computer vision, voice recognition, robot trajectory tracking control, among others. Functions for Educational purposes were implemented in the developed system, including mathematical, spelling, directions, and questions functions that gives more flexibility for the teachers to create their own scripts. A cloud voice recognition tool was implemented to improve the interactive system between Cozmo and the users. Also, a cloud computing vision system was used to perform object recognition using Cozmo's camera to be applied on educational games. Other functions were created to control Cozmo's motors and expressions to create sophisticated scripts. To run the scripts, an interpreter algorithm was developed to translate the created functions into Cozmo's programming language. To validate this work, the project was presented to several elementary school teachers (classes with students between 4 and 12) and the results are reported in this work, indicating that the proposed system can be a useful educational tool. Lastly, in order to reduce the project cost, a simpler system was developed that uses a Raspberry PI to give commands to Cozmo, a computer is no longer necessary.

Keywords: Cozmo. web server. educational robotics.

Lista de ilustrações

Lista de tabelas

Lista de abreviaturas e siglas

HTTP	*Hypertext Transfer Protocol*
HTML	*Hypertext Markup Language*
API	*Application Programming Interface*
JVM	*Java Virtual Machine*
MVC	*Model View COntroller*
SDK	*Software Development Kit*

Sumário

1 Introdução

O uso de robótica vem se tornando cada vez mais presente em todos os níveis da educação, incluindo o ensino fundamental, médio, graduação e pós-graduação. Hoje sabe-se que a robótica, como ferramenta multi-disciplinar, é uma solução ideal para integração curricular, oferecendo aos estudantes a possibilidade de relacionar conceitos aprendidos em várias disciplinas (MANSEUR, 1997) promovendo também, de forma natural, o incentivo ao trabalho em equipe (TUR; PFEIFFER, 2006). Além disso, a maioria das pessoas aprendem mais facilmente quando a execução de tarefas e atividades práticas estão envolvidas no processo de aprendizagem (CONRAD, 2005), de forma que os robôs podem ser usados como ferramentas pedagógicas que oferecem uma experiência do tipo "aprender fazendo" (ALVES et al., 2011a; AHLGREN, 2002).

Contudo, ainda existem alguns problemas para a adoção mais abrangente de tecnologias de robótica na educação, destacando-se os custos e a dificuldade de uso. Com relação aos custos, vários autores (ALVES et al., 2011a; ALVES et al., 2011b) alertam que os custos altos de robôs podem impedir seu uso em salas de aula. Esse problema é ainda mais grave no Brasil, onde muitas escolas e alunos sofrem dificuldades financeiras, não podendo adquirir facilmente robôs ou kits de robótica. De fato, é comum kits disponíveis no Brasil custarem milhares de reais.

O desenvolvimento deste projeto é motivado pelo fato de que não há literatura correlata em robótica educacional que dê liberdade e flexibilidade aos professores para desenvolver suas próprias propostas e roteiros didáticos para robôs interagirem com os alunos. Os trabalhos existentes consistem em ações previamente definidas e programadas de forma fixa. Neste novo desenvolvimento, os roteiros didáticos (*scripts*) criados pelos professores no sistema web são automaticamente disponibilizados para robôs conectados a esta plataforma, permitindo que diferentes professores implementem diferentes atividades.

Dessa forma, o trabalho em questão visa o desenvolvimento de uma plataforma aberta para colaboração de professores e dar autonomia para que os profissionais da área de educação criem seus *scripts* para uso da robótica em sala de aula em diferentes situações.

No final do projeto, para diminuir o custo do sistema e torná-lo mais acessível, foi desenvolvido uma segunda versão mais simplificada que utiliza um Raspberry PI 2 para dar comandos ao robô, substituindo assim a utilização de um computador (que se enquadram nas categorias de *desktops* e *notebooks*) para esta função.

1.1 Objetivos gerais

Os objetivos que permeiam este projeto são todos voltados a tornar a robótica educacional mais presente em salas de aula, oferecendo liberdade aos professores para construir seus roteiros didáticos de interação entre robôs e alunos. Deste modo, este trabalho possui como objetivo principal o desenvolvimento de um sistema com servidor web dinâmico e um algoritmo interpretador de texto (comandos) para executar funções básicas no Cozmo.

1.2 Objetivos específicos

- Revisão bibliográfica e estudos sobre robótica educacional;
- Revisão das funcionalidades e plataforma de programação do Cozmo;
- Estudos sobre a linguagem HTML;
- Desenvolvimento de ambiente web para elaboração de roteiros;
- Programação de um protocolo de comunicação entre o servidor web, o *smartphone* e o Cozmo;
- Implementação de um sistema de reconhecimento de voz em nuvem e um sistema de visão computacional em nuvem no Cozmo;
- Desenvolvimento de jogos e funções educacionais para serem utilizados nos roteiros didáticos;
- Desenvolvimento de um interpretador dos roteiros para executar comandos no Cozmo;
- Substituição do computador por um Raspberry PI;
- Validação do projeto através de testes com professores do ensino fundamental

1.3 Organização do Trabalho

Este trabalho está organizado da seguinte forma: o capítulo 2 descreve algumas pesquisas sobre trabalhos relacionados, incluindo aspectos de uso em nuvem e de servidores web na área de robótica educacional. O capítulo 3 aborda todos os conceitos teóricos, materiais e métodos utilizados no desenvolvimento do projeto. O capítulo 4 descreve o sistema proposto. O capítulo 5 apresenta os resultados obtidos durante os testes realizados. Por fim, no capítulo 6 é apresentado as conclusões deste projeto.

2 Revisão Bibliográfica

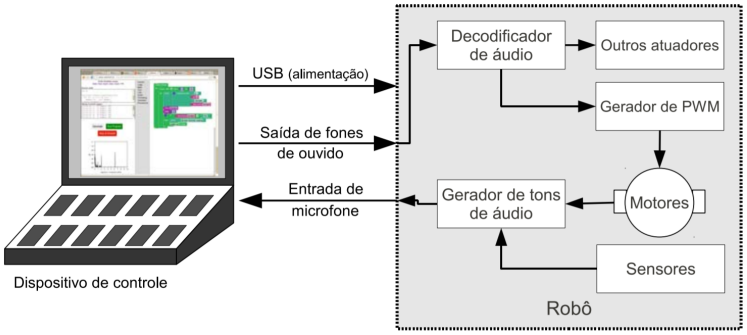
Foram pesquisados alguns trabalhos relacionados com serviços em nuvem, utilização de *smartphones* na área de robótica e utilização de um ambiente web para programações básicas.

Jeonghye Han, da Universidade Nacional de Educação de Cheongju, definiu em seu trabalho que existem dois tipos de robôs educacionais: robôs *hands-on* e robôs de serviços educacionais (HAN, 2012). Em sua pesquisa, Han considera que os robôs *hands-on* são usados para ciências, tecnologia, engenharia e educação matemática para aumentar a criatividade e o interesse das pessoas por esses campos. Para os robôs de serviços educacionais, Han argumenta que por causa de sua aparência amigável e movimentos físicos, os interesses dos alunos aumentam quando os robôs interagem com eles, tornando o aprendizado mais divertido (HAN, 2012).

Além disso, em (JEONG et al., 2014), Gu-Min Jeong e seus colegas argumentam que quando a interação entre o robô e a criança é natural, o aprendizado das crianças apresenta melhores resultados. Assim, o sistema robótico tem que ser sofisticado e inteligente o suficiente para poder interagir naturalmente com as crianças. Por causa disso, os custos dos sistemas aumentam, o que atualmente é um problema para usá-los como uma ferramenta de aprendizado (JEONG et al., 2014; MARTINS; GOMES; SANTOS, 2015).

Sendo assim, uma pesquisa realizada por Rafael Aroca na Universidade Federal do Rio Grande do Norte (UFRN) apresentou formas de desenvolver robôs de baixo custo que podem ser utilizados em ambientes educacionais (AROCA, 2012). Nesta pesquisa, foi desenvolvido um robô denominado N-Bot que utiliza um canal de áudio como interface de controle. As frequências desses áudios são utilizadas tanto para transmitir comandos para os atuadores quanto para receber os sinais capturados dos sensores do N-Bot. Esses sinais podem ser interpretados por um computador de controle, um *smartphone* ou um *tablet*, que deve ser acessado e programado remotamente através de redes locais, Internet ou serviços de robótica na nuvem. Além disso, também é possível utilizar uma página da web onde os alunos podem criar programas para geração e recepção de áudio a fim de controlar os movimentos dos robôs. Também, um sistema de reconhecimento de imagem do N-bot foi desenvolvido utilizando a câmera do dispositivo móvel instalada no robô e por meio de uma programação da web, é possível visualizar em um navegador web as imagens capturadas pela câmera, em tempo real. A Figura 1 ilustra uma visão geral do sistema proposto.

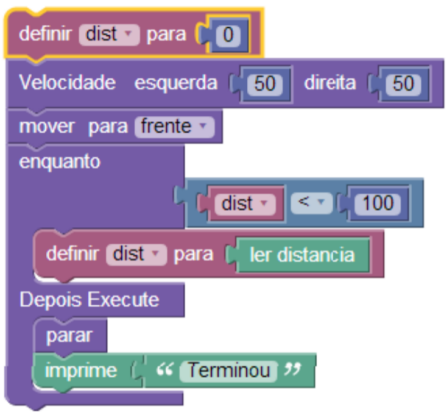
Figura 1 – Visão geral do sistema proposto da pesquisa realizado por Aroca na Universidade Federal do Rio Grande do Norte.



Fonte: (AROCA, 2012)

A robótica educacional também pode ser utilizada como uma ferramenta para facilitar o aprendizado de lógica e linguagens de programação. A literatura indica que estas ferramentas podem auxiliar jovens a raciocinar de forma sistemática e desenvolver o pensamento computacional, e que este resultado pode ser potencializado com aplicação da metodologia de aprendizado baseado em problemas (REIS; SARMENTO; ZARAMELLA, 2014). Deste modo, uma pesquisa realizada na Universidade Tecnológica Federal do Paraná (UTFPR) propõe o desenvolvimento de uma plataforma constituída por um ambiente web de programação em blocos (Figura 2), e um *smartphone* como uma unidade controladora de uma base robótica. No trabalho em questão, foi utilizado o modelo de programação *Model View Controller* (MVC) (REIS; SARMENTO; ZARAMELLA, 2014). No padrão de projeto de software MVC, o *Controller* recebe requisições do usuário, o *Model* aplica as regras de negócio e faz acesso ao banco de dados (caso seja necessário) utilizando os dados inseridos pelo usuário na requisição, e o *View* corresponde a uma interface gráfica para o usuário, como por exemplo uma página HTML (REIS; SARMENTO; ZARAMELLA, 2014).

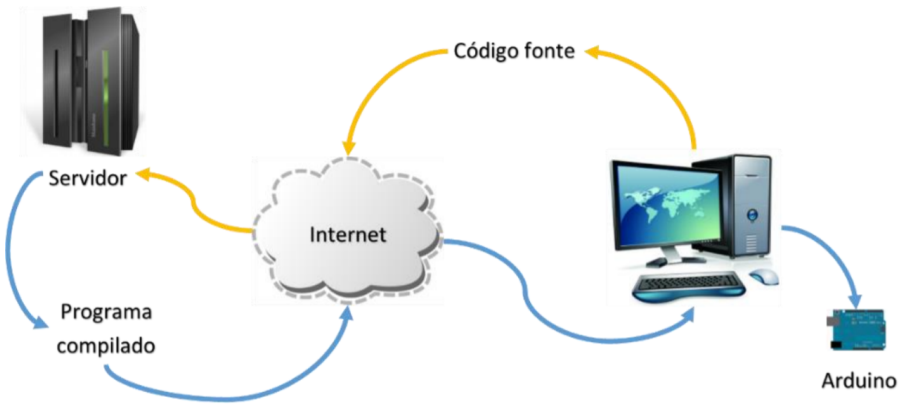
Figura 2 – Exemplo de programação em blocos.



Fonte: (REIS; SARMENTO; ZARAMELLA, 2014)

Um outro trabalho desenvolvido pelos pesquisadores Torres, Aroca e Burlamaqui propõe o desenvolvimento de um ambiente de programação multiplataforma baseado em web que permite que os usuários escolham entre a programação escrita tradicional e o método didático de programação web (TORRES; AROCA; BURLAMAQUI, 2014). O objetivo desse trabalho é facilitar a maneira como estudantes aprendem a programar sistemas embarcados utilizando hardware aberto e de baixo custo, como o Arduino Uno, além de métodos modernos e diferentes abordagens de programação (TORRES; AROCA; BURLAMAQUI, 2014). Neste projeto, foi utilizado o sistema CodeRhino. Tal sistema possui um editor de código com destaque de sintaxe, identação, contagem de linhas, editor de programação visual em blocos e módulo de envio de código. O módulo de envio é um Java Applet executada em um navegador, que realiza o *download* do código escrito pelo usuário (compilado em um servidor Linux) e envia o programa para o Arduino conectado na máquina, conforme é ilustrado na Figura 3. Todo o processo é transparente ao usuário e acontece no navegador web de sua preferência, desde que a Máquina Virtual Java (JVM) e seus respectivos plug-ins estejam instalados e configurados corretamente, tanto no sistema operacional como no browser (TORRES; AROCA; BURLAMAQUI, 2014).

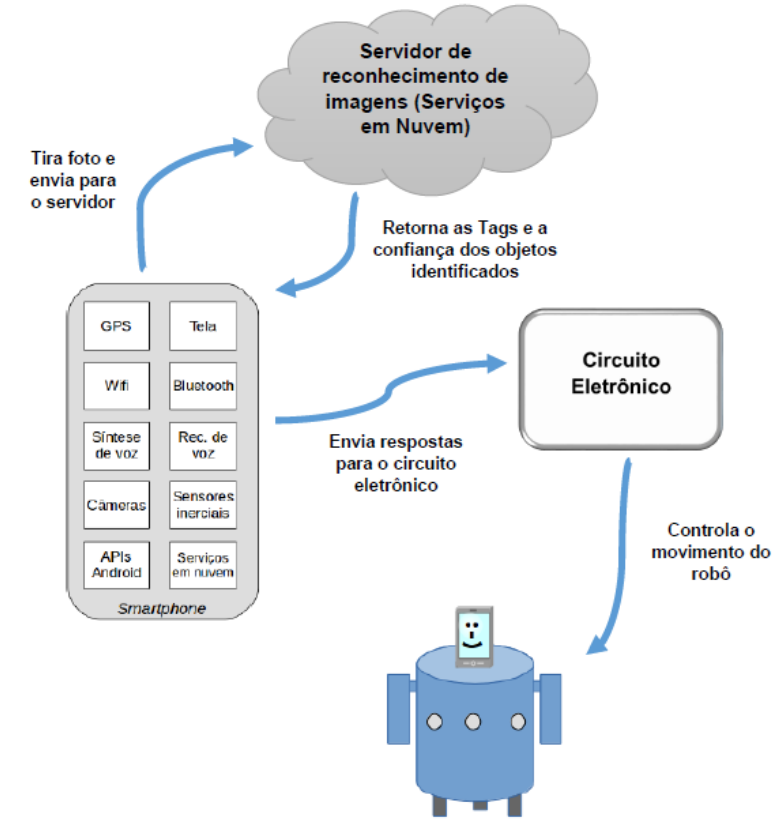
Figura 3 – Diagrama de alto nível do sistema proposto por Torres, Aroca e Burlamaqui. A ordem do fluxo é do amarelo para o azul.



Fonte: (TORRES; AROCA; BURLAMAQUI, 2014)

Em (KUSUMOTA et al., 2018), um trabalho desenvolvido pelo mesmo autor deste projeto propõe o desenvolvimento de um robô de baixo custo que utiliza um *smartphone* como sua unidade de controle para interagir com usuários. O robô desenvolvido utiliza recursos da plataforma Android e ferramentas de processamento em nuvem para reconhecimento e síntese de fala, reconhecimento de padrões visuais vistos pela câmera do *smartphone*, assim como uma simulação de personalidade simplificada através de padrões mostrados na tela do *smartphone*. O robô foi programado com algoritmos de interação sobre assuntos educacionais como português e matemática (KUSUMOTA et al., 2018). A Figura 4 ilustra o esquemática da visão geral do sistema.

Figura 4 – Visão geral do sistema proposto em (KUSUMOTA et al., 2018)



Fonte: Autor

Resumidamente, o sistema de reconhecimento de imagem funciona da seguinte forma: primeiro, o *smartphone* com o sistema operacional Android tira fotos automaticamente e as envia para um servidor de serviços em nuvem para executar o reconhecimento de imagens. O servidor retorna as respostas e, em seguida, o sistema Android envia comandos ao microcontrolador Arduino para controlar os movimentos do robô.

3 Fundamentação Teórica, Materias e Métodos

Este capítulo tem como objetivo introduzir os conceitos teóricos, materiais e métodos relacionados com o projeto.

3.1 Cozmo

Cozmo é um robô inteligente desenvolvido pela Anki Company, projetado para interagir com pessoas. Este robô possui quatro motores, cinquenta engrenagens, uma câmera VGA de 30 fps com software de reconhecimento facial, uma tela de resolução 128x64, um alto-falante para comunicação e um braço. Por meio do alto-faltante, o robô pode pronunciar palavras e textos em 4 idiomas diferentes: inglês, japonês, alemão e francês. A Figura 5 ilustra uma imagem de Cozmo.

Figura 5 – O robô Cozmo.

Fonte: (PINTEREST, 2019)

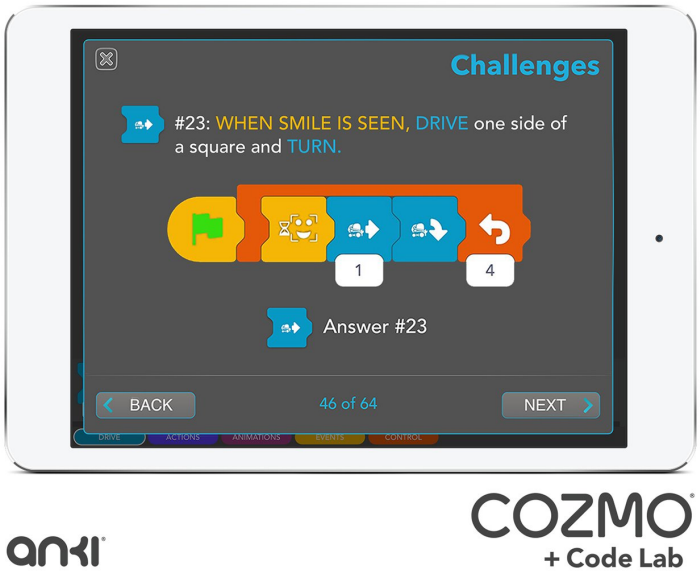
Anki disponibiliza um aplicativo para *tablets* e *smartphones* denominado “Cozmo”, que pode ser baixado na Google Play Store. Este aplicativo é utilizado para dar funcionalidade ao Cozmo, pois é necessário executá-lo para ligar o robô. Neste aplicativo, existem duas ferramentas que tornam possível programar Cozmo: Code Lab e Cozmo SDK.

3.1.1 Code Lab

O Code Lab é uma ferramenta de programação que utiliza uma interface gráfica e permite ao usuário manipular e conectar objetos visuais uns aos outros. Com os blocos conectados, a ferramenta gera um algoritmo que realiza a interpretação desses blocos para

a linguagem de programação de Cozmo (ANKI, 2017b). A Figura 6 ilustra um exemplo de programa desenvolvido no Code Lab.

Figura 6 – Exemplo de um algoritmo desenvolvido no Code Lab.



Fonte: (T3N, 2019)

O Code Lab é uma ferramenta prática para programar pois a conexão dos blocos é intuitiva. No entanto, existem algumas limitações de programação para gerar novas funções para o robô, uma vez que é possível utilizar apenas os blocos disponíveis nesta ferramenta. Para desenvolver algoritmos mais sofisticados, é necessário utilizar o Cozmo SDK.

3.1.2 Cozmo SDK

O Cozomo SDK é uma plataforma de código aberto baseada na linguagem Python. Com um conjunto abrangente de funções de baixo e alto nível, e acesso total aos dados dos sensores de Cozmo, o SDK é tão simples ou eficiente quanto necessário. Um dos principais pontos fortes da plataforma é a versatilidade do hardware e do software. O emparelhamento de um robô altamente expressivo e interativo com uma linguagem de programação fácil de utilizar o torna perfeito para diversos tipos de aplicação (ANKI, 2017a).

Para utilizar o Cozmo SDK, é necessário inicializar o aplicativo "Cozmo" e conectar o *smartphone* à rede Wi-fi criada pelo próprio robô. Em seguida, o aplicativo fornece uma função denominada "*enable* SDK", que quando executada, possibilita a transferência de algoritmos escritos na linguagem Python para serem executados por Cozmo. Para transferir os algoritmos, é necessário conectar o *smartphone* ao computador via porta

USB e executar o arquivo do algoritmo em Python (extensão ".py"). A Figura 7 ilustra o módulo esquemático de comunicação entre o Cozmo, o *smartphone* e o computador.

Figura 7 – Módulo de comunicação entre o Cozmo, o *smartphone* e o computador.

Fonte: Autor

O processo de instalação do módulo SDK depende dos dispositivos a serem utilizados. Neste trabalho, utilizou-se um computador com o Windows 10 e um *smartphone* com a versão 8.0.0 do Android. Os arquivos para download e todas as informações necessárias para a instalação no Windows podem ser encontrados em (ANKI, 2017c).

3.2 Servidor web

Conforme citado em (DOCS, 2019), o conceito de servidor web pode estar relacionado com hardware, software ou ambos. Referente ao hardware, um servidor web é um computador que hospeda os arquivos de sites, como por exemplo os páginas HTML. Quando um usuário conecta no servidor, os arquivos são transferidos para este usuário (Cliente). Referente ao software, o servidor web é responsável por controlar o acesso dos arquivos pelos usuários através de protocolos de comunicação. Um exemplo de protocolo é o HTTP (Hypertext Transfer Protocol), utilizado para visualizar as páginas web.

3.2.1 Protocolo HTTP

Para realizar a transferência de arquivos entre um servidor e um cliente, é necessário estabelecer algum tipo de protocolo de comunicação. Conforme citado em (DOCS, 2019), o protocolo HTTP é responsável por transferir arquivos de hipertexto (por exemplo HTML) entre um servidor e um cliente. Utilizando este protocolo, é possível processar as requisições

recebidas em uma páginas web. Algumas características do protocolo HTTP citadas em (DOCS, 2019) são:

- Somente Clientes fazem requisições HTTP para somente servidores;
- Servidores somente e sempre respondem as requisições HTTP;
- Para requisições de arquivos via HTTP, o cliente deve fornecer a URL (endereço virtual) do arquivo.

3.2.2 Página HTML

O HTML é a linguagem básica da internet e foi criada para ser de fácil entendimento por seres humanos e máquinas, como por exemplo o Google ou outros sistemas que percorrem a internet capturando informações (EIS, 2011). Os elementos HTML são os blocos de construção de páginas HTML representados por tags (comandos de formatação da linguagem). Porém, os navegadores não exibem as tags HTML, mas usam-nas para renderizar o conteúdo da página. A Figura 8 ilustra um exemplo de uma estrutura HTML.

Figura 8 – Exemplo de estrutura HTML.

```
1  <!DOCTYPE html>
2  <html lang="pt-br">
3  <head>
4      <meta charset="UTF-8"/>
5      <title>Título da página</title>
6  </head>
7  <body>
8      --Conteúdo da página--
9  </body>
10 </html>
```

Fonte: Autor

Todos os documentos HTML devem começar com uma declaração de tipo de documento, o `DOCTYPE`. A declaração `<!DOCTYPE>` é uma instrução para o navegador web que indica o tipo de documento em que a página está escrita, como por exemplo os documentos html ou xml. Em seguida, o atributo `HTML lang` é utilizado para declarar o idioma de uma página web ou uma determinada parte. O elemento `head` define o cabeçalho de um documento HTML que traz informações sobre o documento que está sendo aberto, como por exemplo o título, estilo do documento (especificação de como os elementos HTML devem ser renderizados em um navegador) e os metadados, que define propriedades da página, como codificação de caracteres, descrição da página, autor e etc. Logo depois da tag de fechamento `head`, inicia-se o elemento `body`. Este elemento define o conteúdo principal de um documento HTML, como texto, hiperlinks, imagens, tabelas, listas, etc.

Em uma página HTML é possível implementar diversos tipos de elementos e funções. A Tabela 1 ilustra alguns elementos com suas respectivas descrições.

Tabela 1 – Alguns elementos que podem ser utilizados na linguagem HTML.

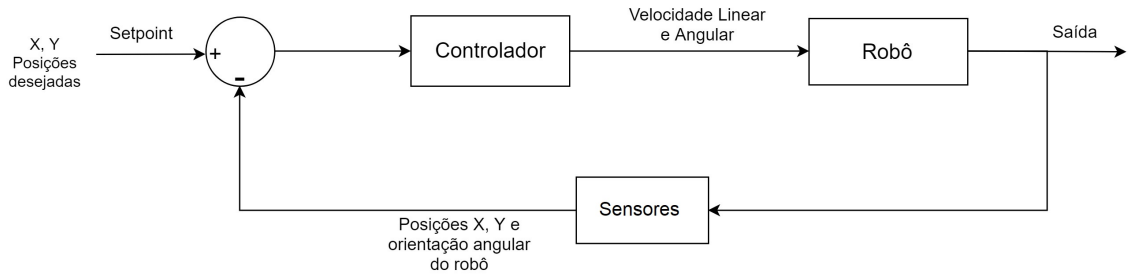
Elementos	**Descrição**
`<form>`	Representa um formulário, consistindo em controles, que podem ser enviados para um servidor para processamento
`<fieldset>`	Representa um conjunto de controles
`<legenda>`	Representa uma legenda para o `<fieldset>`
`<label>`	Representa a legenda de um controle de formulário
`<input>`	Representa um campo de dados digitado que permite ao usuário editar os dados
`<button>`	Representa um botão
`<select>`	Representa um controle que permite a seleção entre um conjunto de opções
`<datalist>`	Representa um conjunto de opções predefinidas para outros controles
`<option>`	Representa uma opção em um elemento `<select>`, ou uma sugestão de um elemento `<datalist>`.
`<textarea>`	Representa um controle de edição de texto multilinha

Fonte: (DOCS, 2019)

3.3 Controlador de Trajetória

Conforme citado em (BORGES; LIMA; DEEP, 2003) os controladores de trajetória para robôs móveis atuam sobre o sistema de propulsão para fazer o veiculo seguir uma determinada trajetória de referência. Tais trajetórias podem ser definidas de duas maneiras: curvas livres (como por exemplo a curva de Bézier) ou estruturas geométricas (retas, segmentos de arco) (BORGES; LIMA; DEEP, 2003). O controlador utilizado neste trabalho foi apresentado por Felipe Martins e seus colegas em (MARTINS et al., 2008). O controlador proposto utiliza as coordenadas (x,y) da trajetória desejada como *Setpoint* para controlar as velocidades angulares e lineares do robô. Com isto, é necessário utilizar um encoder para medir as coordenadas reais e orientação angular do robô para serem utilizadas como variáveis de processo do controlador. A Figura 9 ilustra o diagrama de controle do controlador de trajetória descrito.

Figura 9 – Diagrama de controle de trajetória do robô móvel



Fonte: Autor

Para implementar o controlador, primeiramente deve-se inicializar as coordenas x,y e orientação angular (φ) do robô como 0. Em seguida, calcula-se a velocidade linear e angular usando a multiplicação de matrizes da equação (3.1).

$$\begin{bmatrix} v \\ \omega \end{bmatrix} = \begin{bmatrix} cos(\varphi) & sin(\varphi) \\ \frac{-1}{\alpha} sin(\varphi) & \frac{1}{\alpha} cos(\varphi) \end{bmatrix} \begin{bmatrix} dxd + k_x x_{err} \\ dyd + k_y y_{err} \end{bmatrix} \tag{3.1}$$

Onde:

- v e ω são as velocidades linear e angular, respectivamente;
- φ é a orientação angular do robô;
- α é a distância entre o centro do eixo do robô e o ponto a ser controlado;
- dxd e dyd são as velocidades nos eixos x e y do robô, respectivamente;
- k_x e k_y são os ganhos do controlador do eixo x e do eixo y, respectivamente;
- x_{err} e y_{err} são a diferença entre as coordenadas x e y reais e desejadas do robô, respectivamente.

Para aplicar os valores de velocidade linear e angular retornados pelo controlador, é necessário converter esses valores em velocidades do motor esquerdo e direito. As Equações 3.2 e 3.3 representam as funções utilizadas para calcular as velocidades de tais motores.

$$v_{direito} = v + \frac{d}{2} * \omega \tag{3.2}$$

$$v_{direito} = v - \frac{d}{2} * \omega \tag{3.3}$$

Onde d é a distância entre as rodas esquerda e direita.

Após aplicar as velocidade nos motores, é necessário medir os novos valores reais das posições x, y e a orientação do robô (φ). A biblioteca de Cozmo tem uma função que retorna esses valores medidos por seus encoders. Essas funções são:

- `robot.pose.position.x_y_z` → retorna as posições x,y e z de Cozmo;
- `robot.pose.rotation.angle_z.radians` → retorna a orientação φ de Cozmo

Ao realizar a medição de tais valores, o algoritmo do controlador é iniciado novamente e as velocidades linear e angular são recalculadas por meio da Equação 3.1. Esse processo é executado em *loop* até o robô alcançar o final da trajetória.

3.4 Raspberry PI

Como citado em (PI, 2019b), o Raspberry PI é um pequeno computador de baixo custo desenvolvido pela Raspberry PI Foundation (uma instituição britânica educacional) que possui entradas para a conexão de dispositivos como mouse, teclado e monitor. O Raspberry PI possibilita realizar diversas funções, como navegar na internet, controlar saídas lógicas, reproduzir vídeos, entre outras. Além disso, é afirmado em (PI, 2019b) que "o Raspberry Pi tem a capacidade de interagir com o mundo exterior e tem sido usado em uma ampla gama de projetos de fabricantes digitais, de máquinas de música e detectores de pais a estações meteorológicas e tuítes de *birdhouses* com câmeras infravermelhas. Queremos ver o Raspberry Pi sendo usado por crianças de todo o mundo para aprender a programar e entender como os computadores funcionam". A Figura 10 ilustra uma foto do Raspberry PI 2.

Figura 10 – Fotografia de um Raspberry PI 2.

Fonte: (PI, 2019a)

4 Sistema Proposto

Para o desenvolvimento do sistema proposto, primeiramente foi necessário estabelecer um protocolo de comunicação entre o Cozmo, o computador e o *smartphone*. O protocolo de comunicação foi implementado conforme descrito na seção 3.1.2 e representado pelo esquema da Figura 7. Em seguida, foi implementado um algoritmo de comunicação por voz e um algoritmo de visão computacional para criar novas funções e aprimorar o sistema de interação de Cozmo. Para controlar as funções utilizadas, uma página web com um design simples e intuitivo foi desenvolvida. Por meio da integração de todas estas ferramentas, diversas funções e jogos educativos foram criados e são discutidos nesta seção. Por fim, foi desenvolvido uma versão simplificada do sistema proposto utilizando um Raspberry PI para substituir o computador.

4.1 Sistema de Comunicação por Voz

Esta seção descreve o sistema de comunicação por voz desenvolvido para o Cozmo receber comandos de fala e executar funções pré-programadas. Este recurso foi implementado com o objetivo de tornar o sistema de interação com o robô mais sofisticado e realista.

Devido ao Cozmo não possuir um microfone embutido, foi necessário utilizar o microfone do computador para capturar os sons do ambiente. Um algoritmo foi programado para ativar tal microfone, gravar os sons e efetuar um reconhecimento de fala.

Foi realizado um estudo para verificar e analisar possíveis ferramentas da linguagem de programação Python que realizam a conversão das falas em textos. As ferramentas analisadas foram:

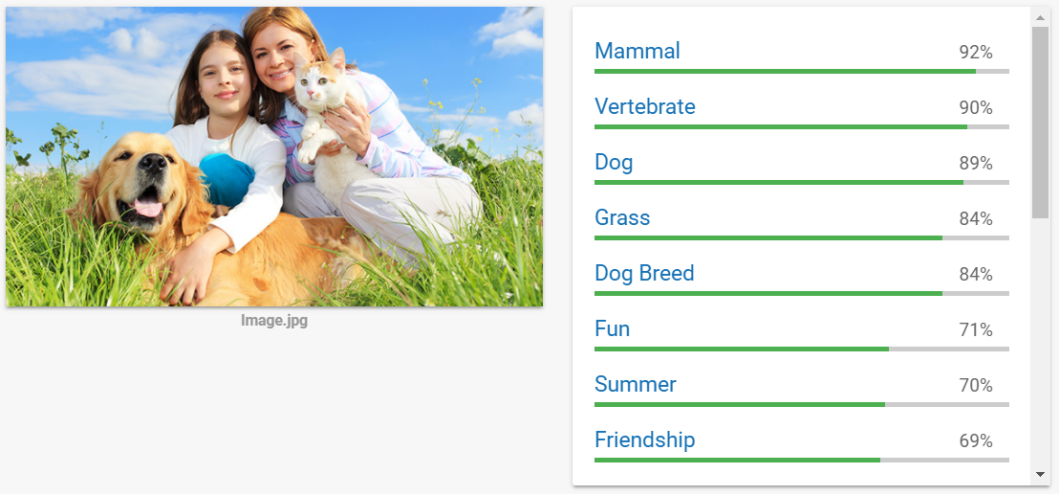
- CMU Sphinks;
- Google Cloud Sp;
- Wit.ai;
- Microsoft Bing Voice Recognition;
- Houdify API;
- IBM Speech to Text;
- Snowby Hotword Detection

Ao verificar a documentação e especificações de cada ferramenta, a que apresentou as melhores características e facilidade de implementação foi a Google Cloud Speech API. Esta ferramenta possui uma disponibilidade de mais de 110 idiomas para realizar o reconhecimento de fala e a capacidade de retornar os resultados em tempo real. Durante a execução do Google Cloud Speech, os sons capturados pelo microfone do computador são enviados para um servidor em nuvem, onde o reconhecimento de fala é realizado, e os resultados são retornados para serem utilizados no algoritmo. Todas as informação e procedimento de implementação para esta API podem ser encontrados em: https://cloud.google.com/speech/.

4.2 Sistema de Visão Computacional

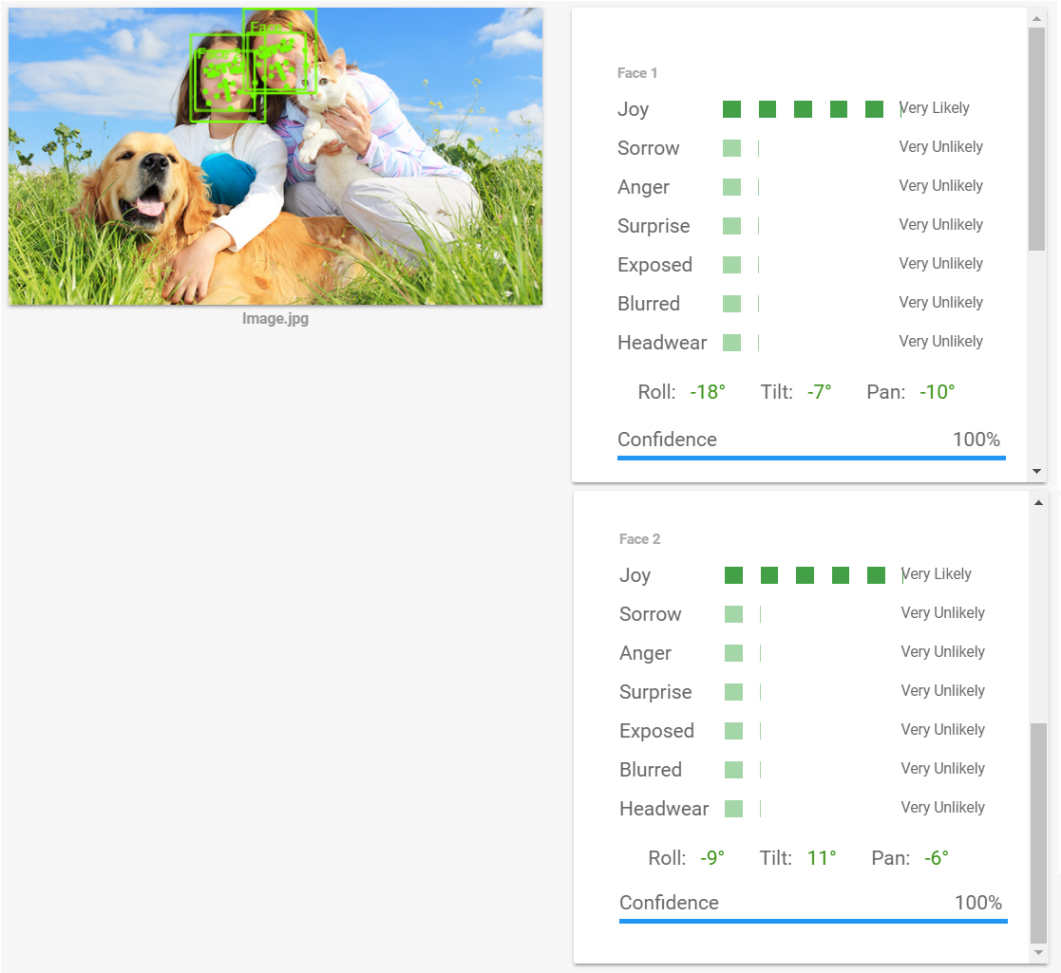
Para o sistema de visão computacional, utilizou-se o Google Cloud Vision API. Este serviço recebe imagens por meio de requisições web e retorna um conjunto de características textuais de tal imagem. O sistema pode reconhecer qualquer tipo de imagem e identificar objetos predominantes, como carros, pessoas, rostos, emoções (triste, feliz, zangado, etc), textos, locais e logotipos. As Figuras 11 e 12 ilustram exemplos do Google Cloud Vision utilizado para reconhecer objetos e o emoções, respectivamente. A fotografia utilizada está disponível em: https://www.tesh.com/articles/are-dog-people-and-cat peopledifferent/.

Figura 11 – Exemplo de reconhecimento de objetos utilizando a API do Google Cloud Vision.



Fonte: Autor

Figura 12 – Exemplo de reconhecimento de emoções utilizando a API do Google Cloud Vision.



Fonte: Autor

Utilizando a câmera VGA de 30 fps do robô, um algoritmo foi desenvolvido para tirar fotos e efetuar o sistema de reconhecimento nas imagens. Foram implementados três tipos de reconhecimentos: Emoções, textos e objetos.

Para o reconhecimento de emoções, verificou-se a possibilidade de identificar 4 tipos: alegria, tristeza, raiva e surpresa. O sistema de identificação de tais emoções foi testado com o autor deste projeto e os resultados são apresentados na seção 5. Devido a baixa resolução da câmera de Cozmo, o reconhecimento de emoções apresentou resultados insatisfatórios e não foi implementado neste projeto projeto.

O sistema de reconhecimento de objetos detecta conjuntos amplos de categorias dentro de uma imagem. Este recurso fornece a possibilidade de identificar cores, formas, características de objetos, pessoas, animais e diversas outras informações visuais. Este sistema foi implementado durante um jogo de orientação, discutido na seção 4.4.5.

A ferramenta de reconhecimento de textos fornece os resultados de todas as letras,

palavras e números contidas em uma imagem, com suas respectivas posições x e y em pixel. A Figura 13 ilustra um exemplo dos resultados obtidos por esta ferramenta para reconhecer os textos de uma imagem disponibilizada em: https://cloud.google.com/vision/docs/ocr. A implementação deste recurso é discutida na seção 4.4.4.

Figura 13 – Exemplo de reconhecimento de textos utilizando a API do Google Cloud Vision.


```
"textAnnotations": [
  {
    "locale": "en",
    "description": "ABBEY\nROAD NW8\nCITY OF WESTMINSTER\n",
    "boundingPoly": {
      "vertices": [
        {
          "x": 45,
          "y": 43
        },
        {
          "x": 269,
          "y": 43
        },
        {
          "x": 269,
          "y": 178
        },
        {
          "x": 45,
          "y": 178
        }
      ]
    }
  }
]
```

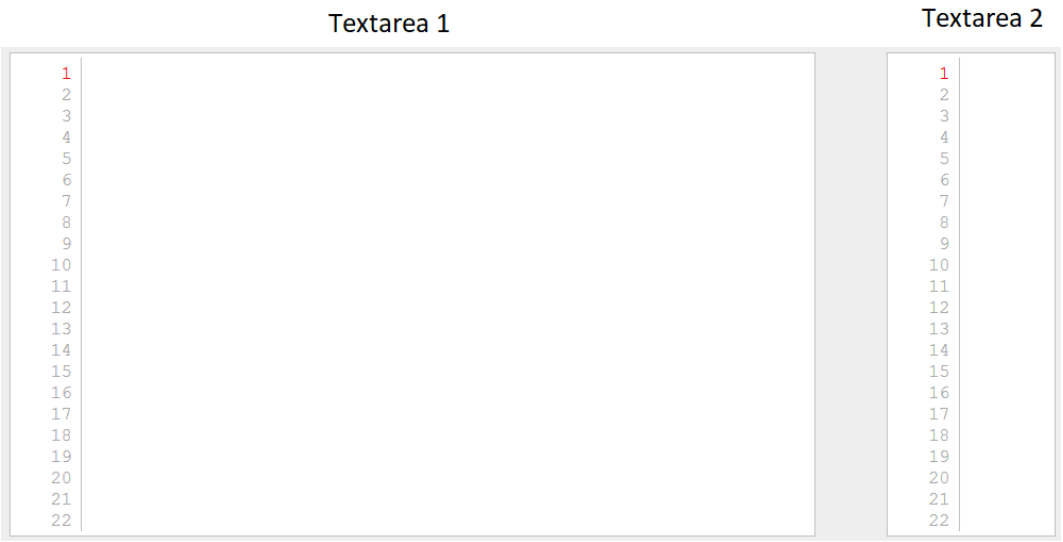

Fonte: Autor

4.3 Página web

Com o objetivo de implementar uma interface de controle do Cozmo para elaboração de *scripts*, foi necessário desenvolver uma página da web utilizando a linguagem de marcação de hipertexto HTML. Para isso, foram utilizados alguns elementos citados na Tabela 1.

Para os usuários elaborarem os *scripts*, foi necessário utilizar o elemento de caixa de texto `<textarea>`. Os textos digitados nesta caixa são enviados para o servidor e decodificados por um algoritmo interpretador de textos desenvolvido. Além disso, outra `<textarea>` foi utilizada para informar o usuário se existem erros no *script* escrito. A estrutura das textareas descritas é ilustrada na Figura 14, onde a Textarea 1 representa a caixa de texto onde o código é escrito e a Textarea 2 representa o verificador dos códigos.

Figura 14 – Estrutura das Textareas para elaboração de *scripts*.



Fonte: Autor

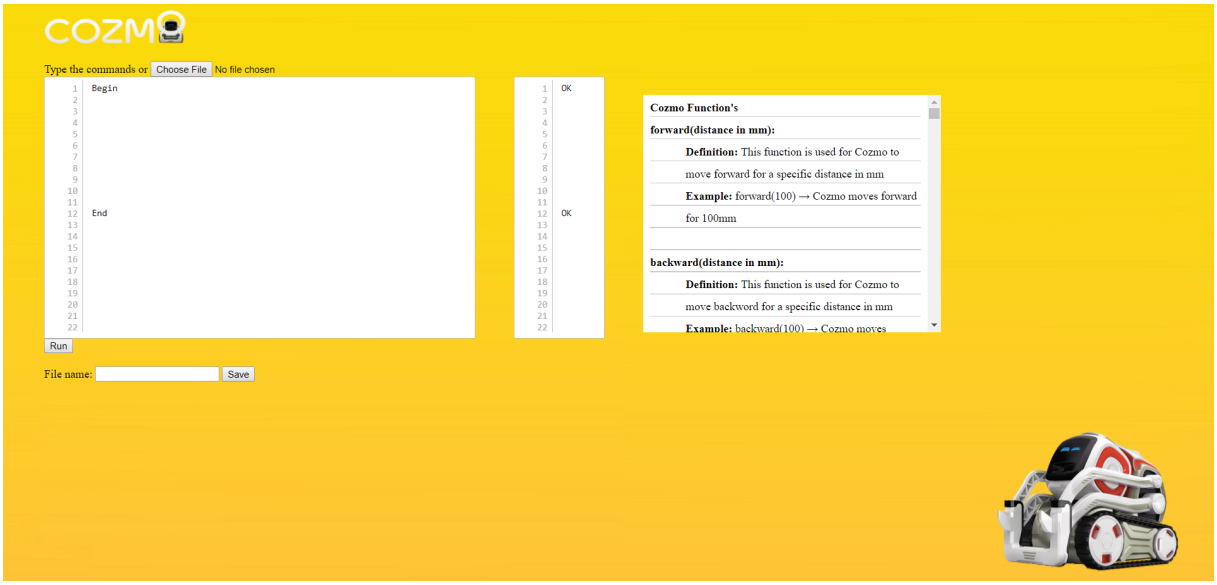
O algoritmo desenvolvido em Python varre o *script* e indica linha por linha se erros foram encontrados. No caso da detecção de erros no código, é escrito *ERROR* na linha da Textarea 2 que corresponde à mesma linha da Textarea 1 onde foi encontrado o erro. Caso contrário, é escrito *OK*.

Para dar instruções ao usuário sobre como implementar as funções, outro elemento `<textarea>` foi implementado na página. Este `<textarea>` contém todas as informações e detalhes necessários para utilizar corretamente os comandos criados. Para cada função, um exemplo de implementação é apresentado.

Em seguida, foi implementado na página web a opção de salvar os *scripts* elaborados na memória interna do computador, assim como carregar os *scripts* previamente salvos. Na função de carregar, utilizou-se o elemento do tipo `<input type = "file">`. Este elemento é representado por um botão que possui a função de abrir uma nova janela do sistema operacional e permitir o usuário a ter acesso aos arquivos do computador para selecionar o arquivo a ser carregado. Na função que salva os roteiros, primeiramente utilizou-se o elemento `<input type = "button">` para criar um botão *Save*. Em seguida, adicionou-se um elemento `<input type = "text">` que possibilita ao usuário digitar em um campo de texto o nome do programa a ser salvo. Ao digitar o nome desejado e clicar no botão *Save*, cria-se na memória interna do computador um arquivo de texto ".txt" com o nome do roteiro e texto escrito no `<textarea>`.

Para tornar a página visualmente atrativa, utilizou-se uma imagem de fundo e figuras referentes ao robô Cozmo. As imagens foram baixadas do site da Anki e modificadas pelo autor deste projeto. A Figura 15 ilustra a versão final da página da web desenvolvida.

Figura 15 – Página web desenvolvida.



Fonte: Autor

4.4 Desenvolvimento do Software

O software do projeto consiste no desenvolvimento e implementação do servidor web no computador conectado ao *smartphone*. Para isso, utilizou-se um web *framework* (algoritmo de suporte para o desenvolvimento de sites dinâmicos, aplicativos da Web e serviços da web) chamado Flask. Este *framework* tem a flexibilidade da linguagem de programação Python e fornece um modelo simples para o desenvolvimento web (MAIA, 2015). Com a biblioteca Jinja2 do Flask, é possível implementar a função `render_template` para renderizar a página HTML criada anteriormente. A Figura 16 ilustra o trecho de código utilizado para renderizar a página web index.html. Vale ressaltar que, para implementar a função `render_template`, foi necessário criar uma pasta com o nome *templates*, no mesmo diretório do arquivo ".py", e inserir o arquivo ".html".

Figura 16 – Renderização de um arquivo html pelo Flask.

```
1 @app.route('/')
2 def index():
3     form = InitForm()
4
5     return render_template("index.html", form=form)
```

Fonte: Autor

No algoritmo do servidor web, implementou-se funções que realizam a transferência de dados entre o cliente (usuário) e o servidor utilizando o protocolo HTTP (*Hyper Text*

Protocol). O Flask possui uma biblioteca chamada `request` que realiza as requisições `GET` e `POST` do HTTP. Deste modo, utilizou-se o método `POST` para extrair o texto do *script* escrito no `<textarea>` e interpretar os comandos no algoritmo. A Figura 17 ilustra o trecho de código que implementa o método `POST`.

Figura 17 – Implementação do método `POST` para requisição HTTP utilizando Flask.

```
1 @app.route('/', methods=['POST'])
2 def submit():
3
4     text_textarea = request.form['my_textarea']
5     print(text_textarea)
6     return render_template("index.html")
```

Fonte: Autor

A função do interpretador é traduzir o *script* linha a linha na linguagem Python e executar comandos no Cozmo. Um código foi desenvolvido para efetuar comparações de textos dos *scripts* com textos pré-definidos no algoritmo. Assim, criou-se diversos comandos e jogos educacionais para serem utilizadas pelos usuários. Tais comandos são discutidos nas próximas seções. É importante ressaltar que devido ao Cozmo possuir compatibilidade apenas com os idiomas citados anteriormente, todas as funções foram desenvolvidas somente em inglês.

4.4.1 Funções Matemáticas

Quatro funções matemáticas foram implementadas para auxiliar durante o aprendizado dos alunos: soma, subtração, divisão e multiplicação. Na função de soma, o usuário deve especificar o nome da função `sum` com os dois números a serem somados entre parênteses, ou seja: sum (*n1* + *n2*), onde *n1* e *n2* representam os números a serem somados. Ao executar este código, o algoritmo interpreta os dados recebidos pela solicitação HTTP e executa comandos no Cozmo. Assim, programou-se um código para ativar os alto-falantes do robô para dizer *How much is n1 + n2?*. Ao mesmo tempo, é apresentado no *display* LCD do robô a operação matemática a ser resolvida, ou seja, *n1* + *n2*. A pessoa que está interagindo com o Cozmo deve encontrar a solução e fornecer a resposta via comunicação de voz. Em seguida, o algoritmo de reconhecimento de fala é executado e o sistema compara a resposta dada com a resposta correta. Caso a pessoa acerte, Cozmo reproduz uma animação feliz e parabeniza a pessoa. Caso contrário, Cozmo reproduz uma animação triste e diz a resposta correta.

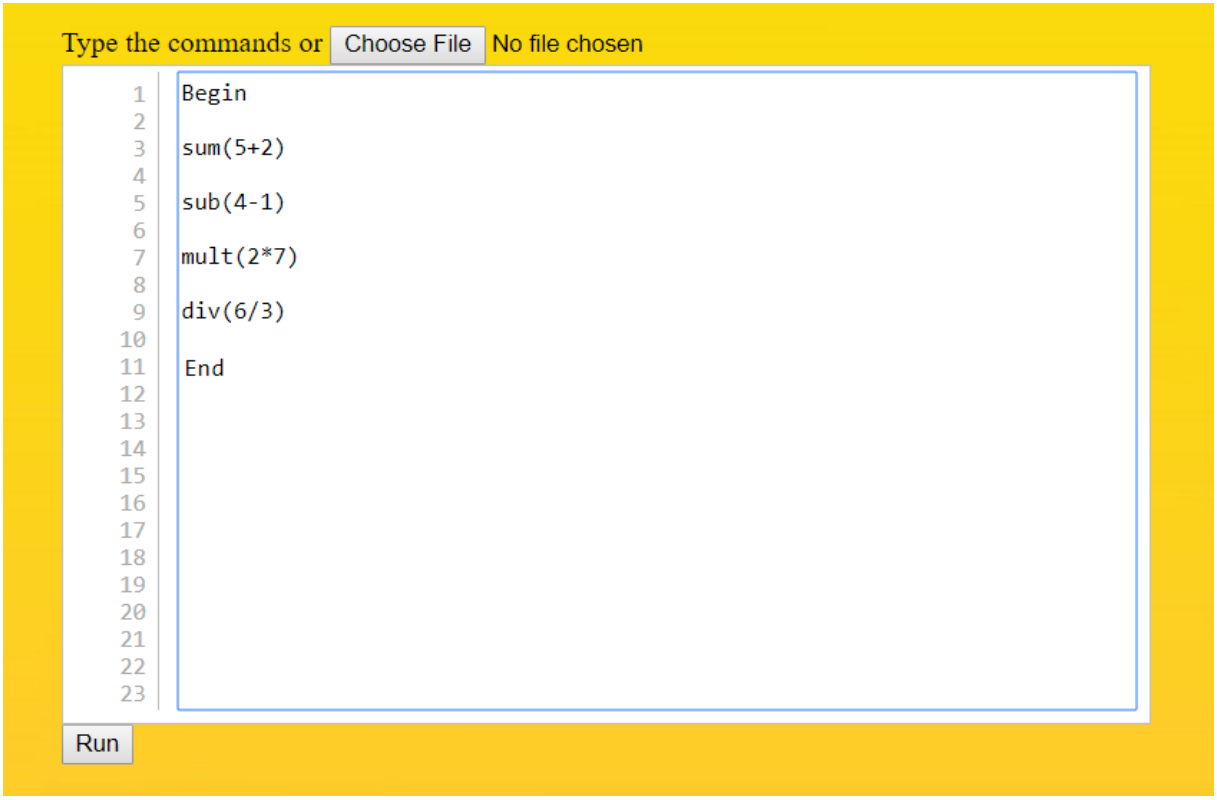
As funções de subtração, divisão e multiplicação funcionam de forma semelhante à função soma. A implementação de tais operações é mostrada a seguir:

- Subtração: sub(*n1* - *n2*)

- Multiplicação: mult(*n1*n2*)
- Divisão: div(*n1*/*n2*)

A Figura 18 ilustra um exemplo de implementação de cada operação matemática na página web.

Figura 18 – Códigos das operações matemáticas desenvolvidas.



Fonte: Autor

4.4.2 Algoritmo de Desenho

Durante os primeiros anos escolares, os alunos aprendem alguns conceitos básicos sobre formas geométricas. Em (FELIX; AZEVEDO, 2015), é citado que o objetivo do ensino de geometria está relacionado ao senso de localização, reconhecimento de figuras, manipulação de formas geométricas, representação espacial e estabelecimento de propriedades. Devido a isto, desenvolveu-se um algoritmo com o objetivo de controlar o os movimentos de Cozmo para desenhar formas geométricas.

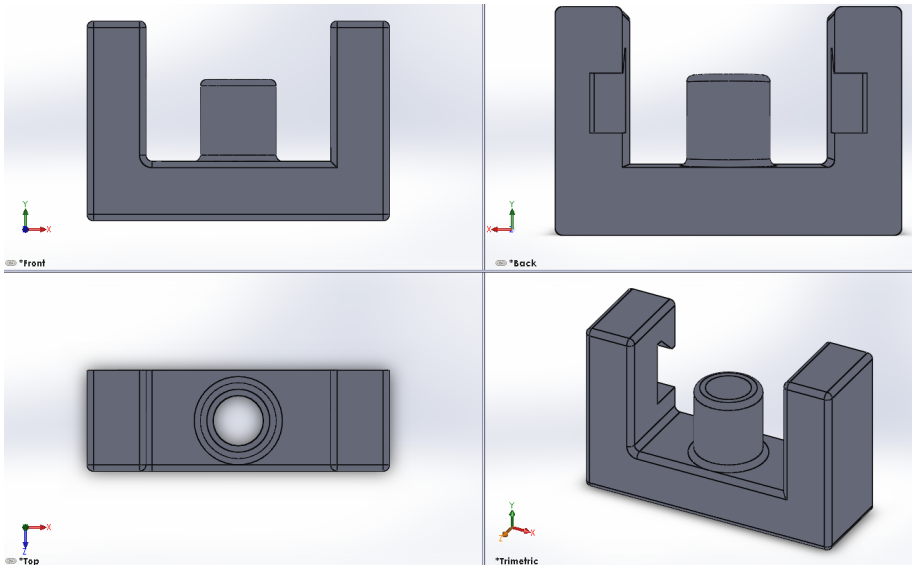
Para isto, primeiramente foi necessário projetar uma peça de acoplamento de canetas de desenho para o robô. O Cozmo possui um braço frontal com uma estrutura de encaixe utilizado para levantar os cubos que acompanham o kit de acessórios do robô. A Figura 19 ilustra a estrutura deste braço em diferentes ângulos.

Figura 19 – Estrutura do braço de Cozmo vista por diferentes ângulos.

Fonte: Autor

O software de modelagem SolidWorks foi utilizado para projetar uma peça de acoplamento com um furo que permite fixar uma caneta de desenho. A Figura 20 ilustra as imagens da peça projetada em diferentes ângulos.

Figura 20 – Design da peça de acoplamento para fixar uma caneta de desenho no braço do Cozmo.

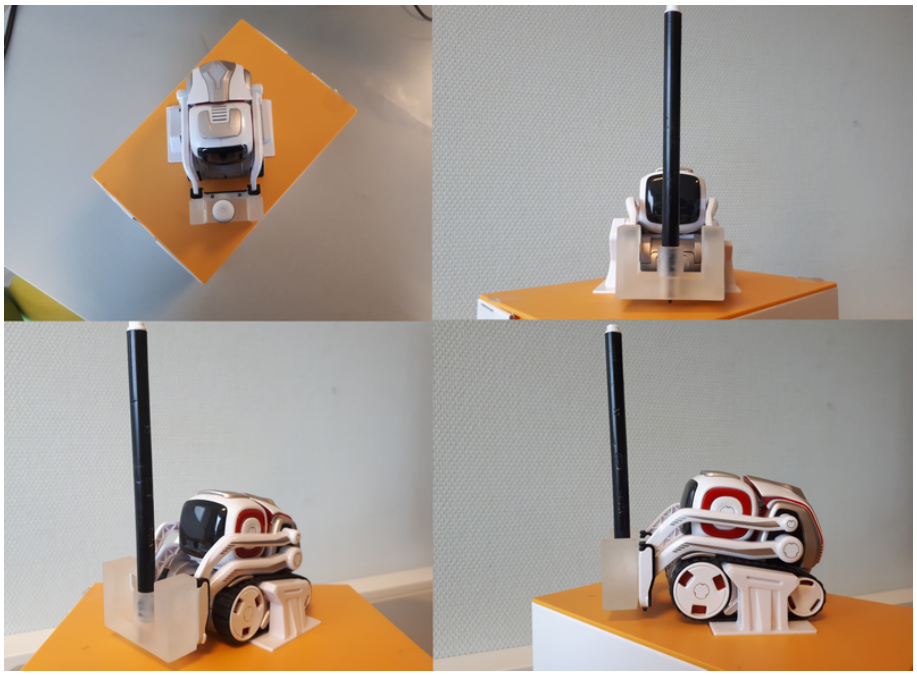


Fonte: Autor

Para produzir a peça da Figura 20, utilizou-se a impressora 3D do modelo Projet 3510 SD com precisão de 0,032 mm e material de impressão VisiJet M3 Crystal (MJP). Ao

fixar a peça impressa no braço de Cozmo, é possível desenvolver algoritmos de movimento para desenhar formas geométricas na superfície de locomoção do robô. A Figura (21 ilustra o Cozmo com a peça de desenho e a caneta fixada.

Figura 21 – Cozmo com a caneta fixada na peça de acoplamento .



Fonte: Autor

Desenvolveu-se um algoritmo com o intuito de controlar os movimentos do robô para desenhar 4 figuras geométricas: quadrado, triângulo, círculo e retângulo. Foi necessário aplicar as funções de movimento disponíveis na biblioteca do Cozmo para controlar os motores e a posição angular do braço.

O código para implementar as funções de desenho na página da web é `draw`, indicando a forma geométrica (*square*, *triangle*, *circle* ou *rectangle*). Para a função de desenho de um triângulo, por exemplo, deve-se digitar: `draw triangle`. Ao clicar no botão *Run* da página, o Cozmo realiza os movimentos programados para desenhar a figura especificada.

Com os mesmos métodos da função de desenho, elaborou-se um jogo de perguntas sobre as características das figuras geométricas. O código para implementar essa função é `game`, indicando a forma geométrica desejada. Para cada forma, são feitas as 3 perguntas mostradas a seguir:

- Quadrado:
  - *What is the name of this geometric shape?*

  - *How many lines does a square shape have?*
  - *How many equal lines does a square shape have?*

- Círculo:
  - *What is the name of this geometric shape?*
  - *Is a circle shape round?*
  - *Does a circle have straight lines?*

- Triângulo:
  - *What is the name of this geometric shape?*
  - *How many lines does a triangle shape have?*
  - *Is a triangle shape round?*

- Retângulo:
  - *What is the name of this geometric shape?*
  - *How many lines does a rectangle shape have?*
  - *Is a rectangle shape the same as a square shape?*

Quando Cozmo finaliza o desenho, todas as 3 perguntas são feitas em sequência, e as resposta devem ser dadas via comunicação em voz. Caso a pessoa acerte, Cozmo reproduz uma animação feliz e parabeniza a pessoa. Caso contrário, Cozmo reproduz uma animação triste, diz a resposta correta, e pede para a pessoa assistir a um vídeo na tela do computador relacionado a tal forma geométrica. Programou-se um algoritmo para abrir uma página do Youtube e exibir um vídeo sobre a forma geométrica utilizada no comando. Os links dos vídeos são mostrados abaixo:

- Quadrado:
  - https://www.youtube.com/watch?v=WHypLi16j4o

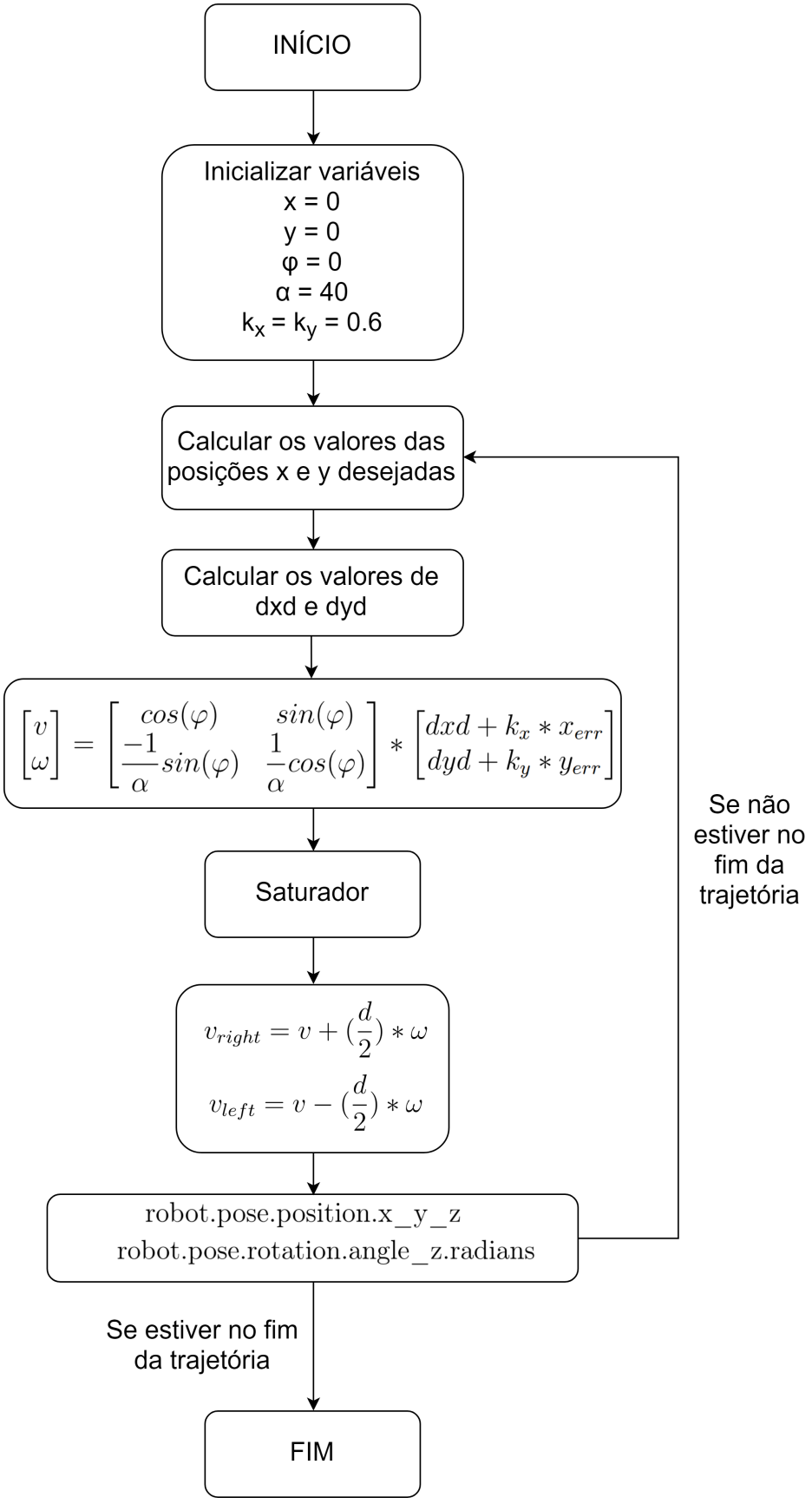
- Círculo:
  - https://www.youtube.com/watch?v=BmtU1SObpKI

- Triângulo:
  - https://www.youtube.com/watch?v=AEpHfWFcfuw

- Retângulo:
  - https://www.youtube.com/watch?v=cW5muVaoK4I

Com o objetivo de garantir uma maior flexibilidade para os usuários desenharem outros tipos de figuras, desenvolveu-se um módulo que permite utilizar o cursor do mouse para elaborar desenhos livres em uma janela da página web e controlar o Cozmo para reproduzir o mesmo desenho. Para isto, foi implementado o controlador de trajetórias descrito na seção 3.3. Os ganhos proporcionais foram determinados empiricamente, obtendo os valores $k_x = 0.6$ e $k_y = 0.6$. A distância entre o centro do robô e a posição da caneta (ponto de controle) foi medida com um micrômetro e obtido um valor igual a $\varphi = 40$ mm. Assim, foi implementado o algoritmo de controle representado pelo diagrama de blocos da Figura 22.

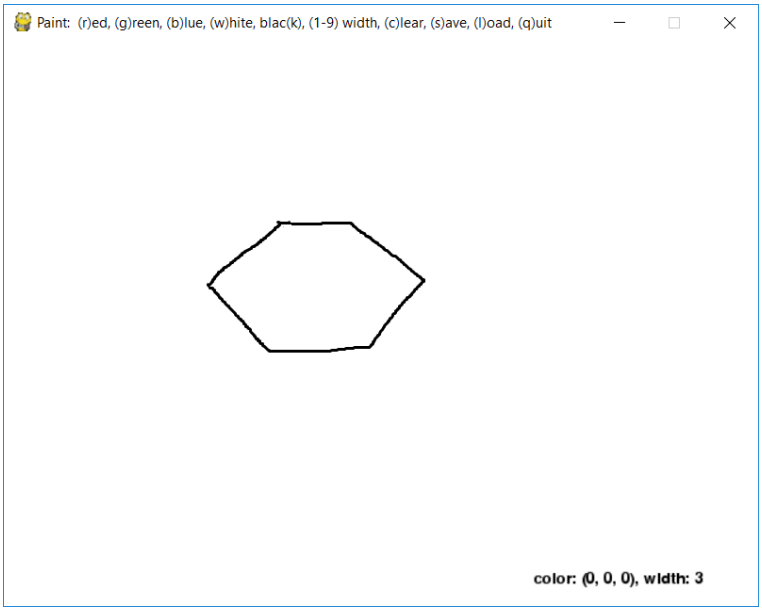
Figura 22 – Diagrama de blocos do controlador de trajetória implementado no robô Cozmo.



Fonte: Autor

Para o módulo de desenho, utilizou-se uma biblioteca *open-source* do Python chamada Pygame. A Figura 23 mostra um exemplo de um desenho hexagonal feito utilizando esta biblioteca.

Figura 23 – Desenho hexagonal feito com a bilioteca Pygame.



Fonte: Autor

Enquanto o usuário desenha, o algoritmo armazena a posição de todos os pixels em uma lista. Quando o desenho é finalizado, a lista de pixels é convertida em coordenadas cartesianas do eixo x e do eixo y para serem utilizadas como os valores de xd e yd (trajetória desejada) no controlador de trajetória. O algoritmo do controlador de trajetória é apresentado no Apêndice A.

4.4.3 Função de Soletração

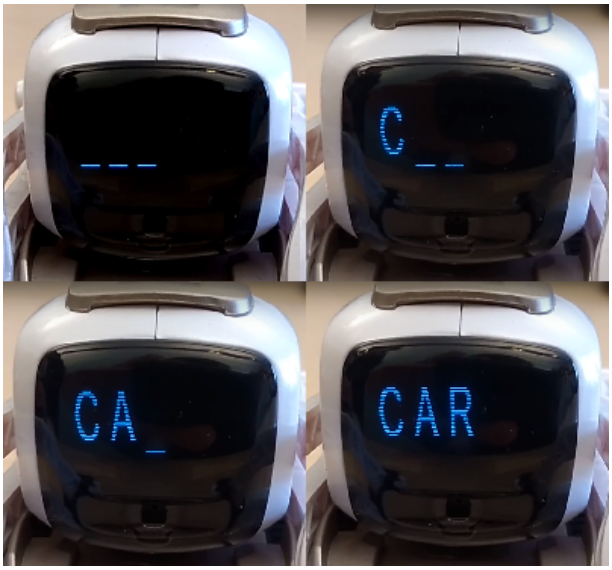
Durante os primeiros anos do ensino fundamental, os alunos aprendem a identificar e pronunciar a letras do alfabeto nativo. Este conhecimento é a base para os alunos aprenderem a ler e escrever palavras e textos. Assim, a dificuldade de alfabetização pode afetar a vida escolar do aluno, pois com a progressão dos anos escolares, o aprendizado de outras disciplinas depende cada vez mais da leitura e da escrita.

Devido a isto, criou-se uma função `spell word ()` para auxiliar no aprendizado do alfabeto latino. Nesta função, o usuário deve indicar entre os parênteses a palavra desejada. Na tela do Cozmo, é mostrado os caracteres "_" indicando as letras que faltam ser soletradas. Quando uma letra é soletrada, os caracteres "_" são substituídos pelas letras correspondentes. Estabeleceu-se uma regra em que deve-se dizer a palavra *Letter* juntamente com a letra. Por exemplo: caso deseja-se soletrar a letra "A", deve-se dizer

Letter A. Estabeleceu-se tal regra para evitar erros no algoritmo de reconhecimento de fala, pois o Google Cloud Speech API possui uma inteligência artificial que interpreta as frases como um todo e verifica a relação entre as palavras. Com isto, o algoritmo pode distinguir a letras de palavras que possuem o mesmo fonema, como por exemplo a letra "c" e a palavra *see* em inglês.

Caso a soletração da palavra esteja correta, Cozmo reproduz uma animação feliz e parabeniza a pessoa. Caso contrário, Cozmo reproduz uma animação triste e apresenta na tela LCD a palavra completa que deveria ter sido soletrada. A Figura 24 apresenta as imagens da tela LCD de Cozmo para um exemplo de implementação da função de soletrar, com a palavra desejada sendo *car*.

Figura 24 – Tela LCD de Cozmo durante a soletração da palavra *car*.



Fonte: Autor

4.4.4 Função para Leitura de Textos

Conforme citado em (FRIGHETTO; SANTOS et al., 2013), as crianças no processo de alfabetização exigem conhecimentos conceituais por parte dos professores para que ocorra a aprendizagem da leitura. Além dos problemas decorrentes da falta de estímulo à leitura, e da adequação dessa criança ao seu nível de aprendizagem, e advindos do ambiente familiar, que dificultam o processo de aprendizagem, é possível afirmar que alguns alunos apresentam transtornos que tornam o aprendizado difícil, e independente do processo de educação (FRIGHETTO; SANTOS et al., 2013). Dessa forma, para incentivar e auxiliar os alunos nas leituras, criou-se a função `read text`.

Na função `read text`, ativa-se a câmera de Cozmo para tirar uma foto a ser utilizada no algoritmo de visão computacional (discutido na seção 4.2) para efetuar o reconhecimento de textos. Em seguida, Cozmo ativa os alto-falantes para reproduzir as

palavras obtidas no resultado do reconhecimento. Deste modo, deve-se posicionar o texto desejado a ser lido na frente do robô e executar o comando `read text`.

4.4.5 Função de Direção

Com o objetivo de auxiliar o ensino de direções, um novo jogo foi desenvolvido. Este jogo consiste em dar instruções a Cozmo para encontrar um objeto especificado pelo usuário. O comando para utilizar esta função é `find object ()`, com o nome do objeto desejado inserido entre os parênteses. Em seguida, o microfone do computador é ativado para iniciar o sistema de reconhecimento de voz. É necessário falar as direções *forward* (para frente), *backward* (para trás), *left* (para a esquerda) ou *right* (para a direita) com o intuito de controlar os movimentos de Cozmo em direção ao objeto. Quando Cozmo está na frente do objeto, deve-se dizer *take a picture* (tirar uma foto) para capturar uma foto com a câmera do robô e enviar para o sistema de visão computacional. Caso o objeto encontrado corresponder a algum objeto reconhecido pelo sistema de reconhecimento, Cozmo reproduz a animação feliz. Caso contrário, é reproduzido a animação triste.

4.4.6 Módulo de Funções de Controle

Como um dos principais objetivos deste projeto é permitir que os professores criem seus próprios *scripts*, desenvolveu-se um módulo de funções que garante uma maior flexibilidade aos usuários para controlarem diversos tipos de ações, com uma linguagem relativamente simples. Este módulo consiste em funções que realizam o controle dos motores das rodas, motor do braço, alto-falante, animações e também a reprodução de vídeos na tela do computador. As funções criadas são apresentadas a seguir:

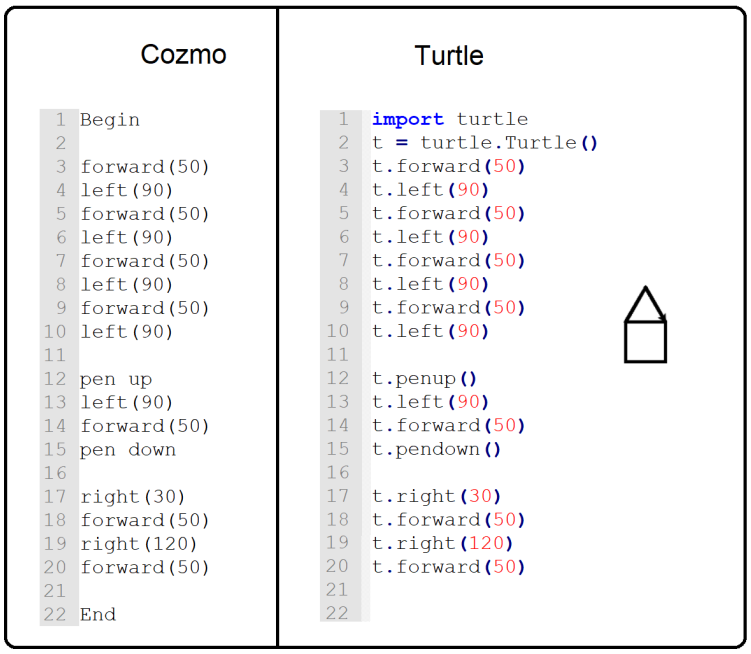
- `forward(Distância em mm)`
- `backward(Distância em mm)`
- `left(Ângulo em graus)`
- `right(Ângulo em graus)`
- `pen up`
- `pen down`
- `head up`
- `head down`
- `play animation (Tipo de animação)`
- `play video (Link do vídeo no Youtube)`

- `speak(Palavras a serem faladas)`
- `ask (Pergunta a ser realizada)`
- `wait for answer`
- `if answer = Resposta correta`
- `end if`
- `if not`
- `end if not`

As funções que controlam os movimentos do robô foram baseadas em uma biblioteca educacional do Python denominada Turtle Graphics. Esta biblioteca foi projetada para controlar uma caneta virtual na tela do computador e elaborar desenhos desejados.

Os comandos `pen up` e `pen down` são utilizados para mover o braço de Cozmo para cima e para baixo, respectivamente. Isso permite controlar os desenhos feitos por Cozmo quando a caneta está fixada em seu braço. Os comandos `head up` e `head down` movem a cabeça de Cozmo para cima e para baixo, respectivamente. A Figura 25 mostra um exemplo de um algoritmo Turtle utilizado para desenhar uma casa, com as respectivas funções de controle de movimentos do Cozmo.

Figura 25 – Figura de uma casa desenhada por um algoritmo Turtle e as funções equivalentes desenvolvidas para o sistema proposto.



Fonte: Autor

A função `play animation (Animação desejada)` foi criada para tornar o sistema de interação de Cozmo mais divertido. A biblioteca do Cozmo fornece diversas animações que podem ser reproduzidas pelo robô. As animações implementadas foram: *Happy* (feliz), *sad* (triste), *angry* (bravo), *surprise* (surpreso), *bored* (entediado), *sleep* (dormir) e *wake* (acordar). Os links dos vídeos de cada animação são apresentados na seção 5.

O comando `play video (Link do vídeo no Youtube)` foi criado para reproduzir um vídeo do Youtube no web *browser* do computador, da mesma forma que é feito no jogo de figuras geométricas mencionado anteriormente.

Para controlar as falas do robô, criou-se a função `speak (Palavras a serem faladas)`. Este comando ativa os alto-falantes de Cozmo para dizer as palavras contidas entre os parênteses. No entanto, apenas os idiomas inglês, francês, alemão e japonês são suportados por Cozmo.

A principal função que permite a criação de *scripts* para aplicar as outras funções discutidas nesta seção é a `ask (Pergunta a ser realizada)`. Esta função é utilizada para o robô realizar qualquer tipo de pergunta (incluindo questões matemáticas), de qualquer tópico que está sendo abordado em sala de aula. Para o comando `ask` funcionar corretamente, deve-se utilizar também os comandos `wait for answer` e `if answer = resposta correta`.

O comando `wait for answer` deve ser utilizado para ativar o sistema de fala de reconhecimento para identificar a resposta do aluno dada via comunicação de voz. O comando `if answer = resposta correta` funciona de maneira semelhante a outras linguagens de programação (como por exemplo, C, C ++ e Python). Após o sinal de "=", o usuário deve informar a resposta da pergunta realizada na função `ask`. Se a resposta for um número, deve-se digitar os números em dígitos. Assim, caso a resposta dada pelo aluno for igual à resposta digitada pelo professor, o algoritmo executa a estrutura condicional `if answer`. Caso contrário, o algoritmo executará a estrutura condicional `if not` (caso seja implementada). Os comandos `end if` e `end if not` são utilizados para indicar quando as estruturas condicionais de `if answer = resposta correta` e `if not` terminam, respectivamente.

Usando estas estruturas condicionais, é possível programar o Cozmo com todas as funções mencionadas nesta seção, possibilitando a criação de diferentes tipos de *scripts* para controlar a fala, as animações, os movimentos do robô e a exibição de vídeos no computador.

Para evitar possíveis erros nos *scipts*, a função `ask` é executada apenas se houver os comandos `wait for answer`, `if answer = resposta correta` e `end if`. Caso contrário, o verificador de códigos mostra *ERROR* na linha de função `ask`. A função `if not` é opcional e não precisa ser utilizada. Para mostrar a implementação correta da função `ask`,

um exemplo de *script* foi criado e é ilustrado na Figura 26.

Figura 26 – Exemplo de *script* com a implementação correta da função `ask`.

```
Begin

ask (How many letters does the Latin alphabet have?)

wait for answer

if answer = 26
play animation (happy)
speak(Congratulations! You are right!)
end if

if not
play animation(sad)
speak(Sorry, that is not the right answer. The right answer is 26)
speak(Look at the computer screen to watch a video about the alphabet)
play video(https://www.youtube.com.br/watch?v=Sw2KZki-eaA)
end if not


End
```

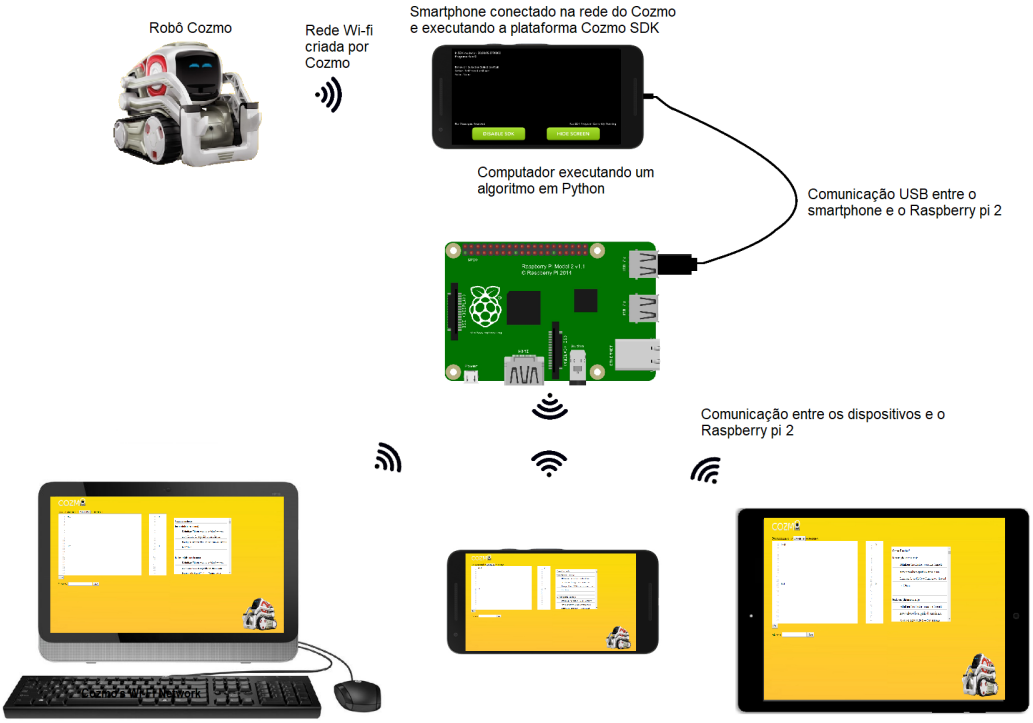
Fonte: Autor

4.5 Sistema Proposto com o Raspberry PI

Após implementar todas as funções descritas na seção 4.4, foi proposto o desenvolvimento de um sistema mais simplificado. Neste novo sistema, utilizou-se um Raspberry PI 2 para atuar como o servidor web e executar o algoritmo Python desenvolvido. Com isto, a utilização de um computador não é mais necessária, diminuindo o custo total do projeto.

Primeiramente, configurou-se o Raspberry como um servidor web com Apache2. O Apache2 é um software de servidor web *open-source* utilizado para servir os arquivos que formam uma página web para usuário. Utilizando este software, é possível acessar o servidor por meio de qualquer computador, *smartphone* ou *tablet* que esteja conectado na mesma rede que o Raspberry. A Figura 27 ilustra a representação do novo sistema proposto.

Figura 27 – Sistema proposto para substituição do computador por um Raspberry PI 2.



Fonte: Autor

O sistema apresentado na Figura 27 possui o mesmo princípio de funcionamento do sistema descrito anteriormente. Porém, devido a substituição do computador que atuava como servidor web, alguns recursos e funções não podem ser utilizados no novo sistema. O sistema desenvolvido anteriormente utilizava o microfone do próprio computador (atuando como servidor) para realizar o reconhecimento de voz. As funções que utilizam esta ferramente não foram implementadas no Raspberry, pois este não possui microfone embutido. Uma solução para integrar esta ferramenta seria alterar o algoritmo para utilizar o microfone dos dispositivos (conectados como clientes) que comunicam-se com o Raspberry.

Uma outra limitação do novo sistema é a impossibilidade de executar o algoritmo que utiliza o controlador de trajetória para reproduzir um desenho feito por um usuário. Como descrito em 4.4.2, a biblioteca utilizada para desenhar é a Pygame da linguagem Python. No algoritmo desenvolvido, é necessário uma interface gráfica no próprio servidor para possibilitar a criação dos desenhos livres. Uma solução para implementar esta ferramenta no novo sistema seria utilizar um canvas de desenho na página web, e enviar para o servidor as posições dos pixels x e y dos desenhos.

É importante salientar que o sistema que utiliza o Raspberry foi desenvolvido como uma prova de conceito para verificar a possibilidade de substituir o computador por um dispositivo de menor custo.

Por fim, o código fonte e arquivos do projeto foram disponibilizados abertamente no endereço: https://github.com/viclpk/CozmoEducationalTool.

5 Resultados e Discussões

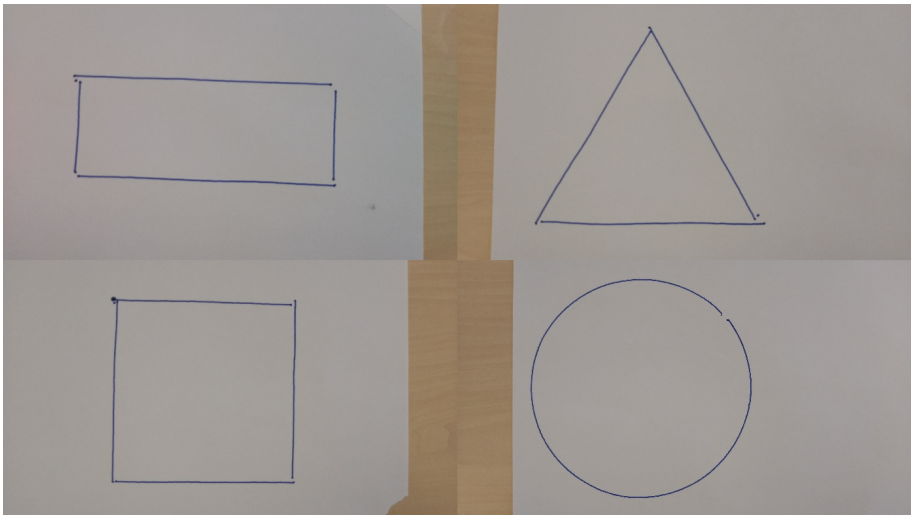
Esta seção descreve, ilustra e discute os resultados obtidos durante os testes realizados.

Como discutido anteriormente, 4 formas geométricas foram escolhidas para fazer o Cozmo desenhar. Para validar essas funções, vídeos do robô desenhando cada figura foram filmados e enviados para o YouTube. Os vídeo estão disponibilizados em:

- Quadrado: https://www.youtube.com/watch?v=NJYDW_IDxvI
- Círculo: https://www.youtube.com/watch?v=h8RV66g1-KQ
- Triângulo: https://www.youtube.com/watch?v=RvKaNv7ySgg
- Retângulo: https://www.youtube.com/watch?v=KM4UMc-H6Pk

A Figura 28 ilustra as formas geométricas desenhadas por Cozmo.

Figura 28 – As 4 formas geométricas desenhadas por Cozmo.



Fonte: Autor

Devido ao Cozmo possuir um controlador de motores sofisticado, as figuras geométricas desenhadas apresentam resultados satisfatórios, como pode ser visto na Figura 28. Vale a pena mencionar que a precisão do desenho é importante para que os alunos identifiquem e memorizem as formas corretamente.

Para ilustrar um exemplo dos jogos das figuras geométricas, filmou-se a implementação da função `game circle`. Como explicado anteriormente, essa função é usada

para o Cozmo desenhar a forma geométrica e fazer três perguntas ao aluno. O vídeo está disponibilizado em: https://youtu.be/LLKLJMn_zwA. Neste vídeo, é possível visualizar o comportamento de Cozmo quando respostas corretas são dadas.

Além disso, criou-se um vídeo do controlador de trajetória com o Cozmo desenhando uma figura desejada. O vídeo está disponível em: https://youtu.be/Nbn1C86MA4c. A Figura 29 mostra os resultados de um outro teste realizado com o controlador.

Figura 29 – Comparação entre uma figura desenhada no computador e a figura desenhada por Cozmo com o controlador de trajetória sendo aplicado.

Fonte: Autor

Como mencionado anteriormente, um módulo de funções foi criado para reproduzir animações no Cozmo e tornar o sistema de interação mais realista. Para ilustrar cada animação, criou-se os vídeos que estão disponíveis nos seguintes links:

- Feliz: https://youtu.be/j05BwBXSkHc
- Triste: https://youtu.be/P_Lv3w4UEBI
- Bravo: https://youtu.be/yDj_NZO33OE
- Surpreso: https://youtu.be/AmOKxQSBa-4
- Entediado: https://youtu.be/ni42l42l8z4
- Dormir: https://youtu.be/vHs6YzvUjJg
- Acordar: https://youtu.be/3wXolHtJ0Us

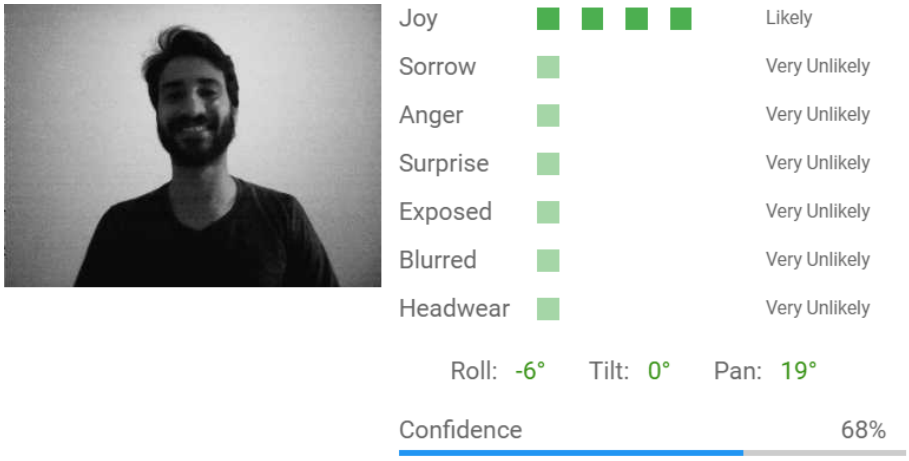
Para demonstrar a funcionalidade dos jogos matemáticos, foram realizados 2 testes: jogos de soma e subtração. No jogo da soma, foi dada a resposta correta, enquanto no jogo de subtração, a resposta errada. Nos vídeos filmados, é possível visualizar as reações de Cozmo em ambas as situações. Os links dos vídeos são apresentados a seguir:

- Soma: https://youtu.be/7zt15YOWyOQ
- Subtração: https://youtu.be/XOnCyyLitnI

Um vídeo de teste da função de soletrar também foi filmado. A palavra especificada foi *car*. O link do vídeo está disponível em: https://youtu.be/EGiHt7ipJvY.

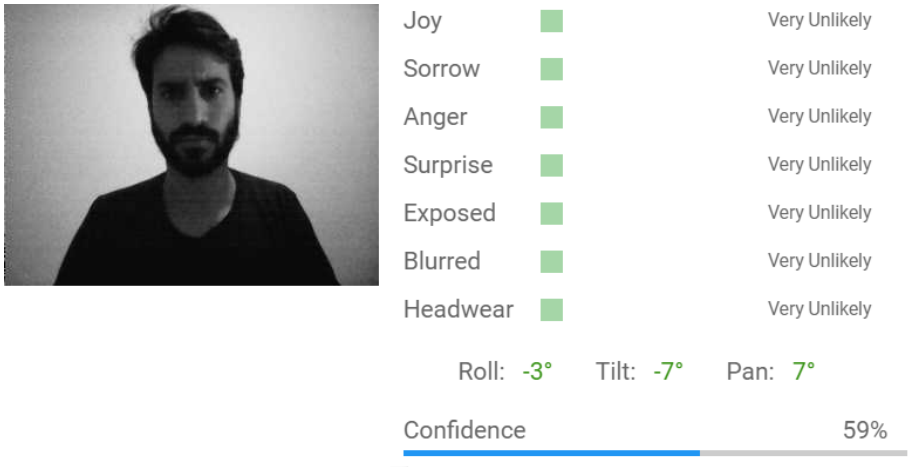
Foram realizados testes com o sistema de visão computacional. Primeiramente, foi realizado testes com sistema de reconhecimento de emoções para identificar alegria, raiva, tristeza e surpresa. As Figuras 30, 31, 32 e 33 apresentam os resultados obtidos.

Figura 30 – Teste de reconhecimento de emoção utilizando o Google Cloud Vision para reconhecer a alegria de uma pessoa.



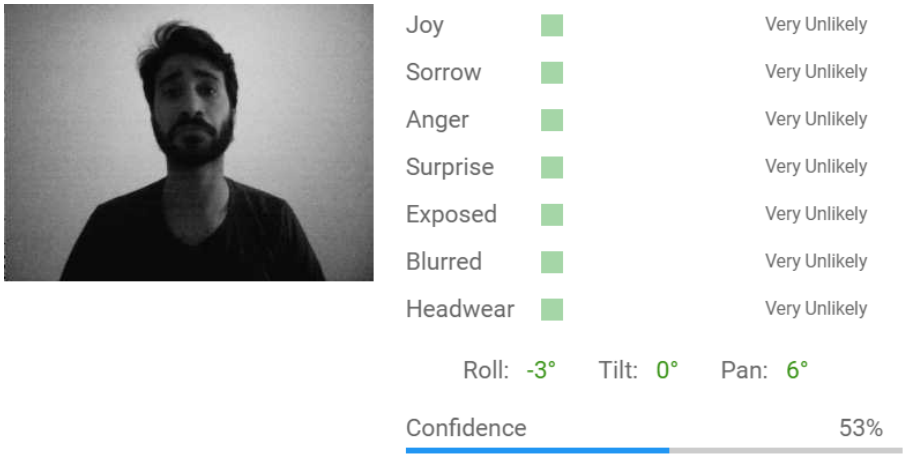
Fonte: Autor

Figura 31 – Teste de reconhecimento de emoção utilizando o Google Cloud Vision para reconhecer a raiva de uma pessoa.



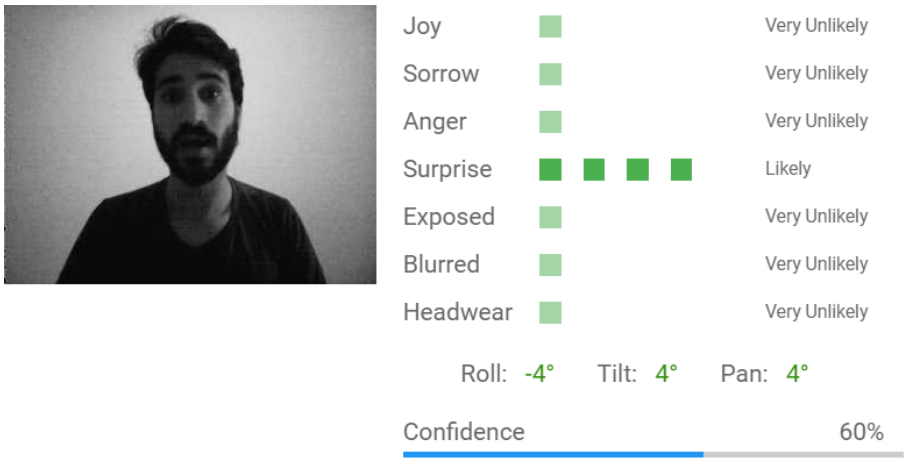
Fonte: Autor

Figura 32 – Teste de reconhecimento de emoção utilizando o Google Cloud Vision para reconhecer a tristeza de uma pessoa.



Fonte: Autor

Figura 33 – Teste de reconhecimento de emoção utilizando o Google Cloud Vision para reconhecer a surpresa de uma pessoa.

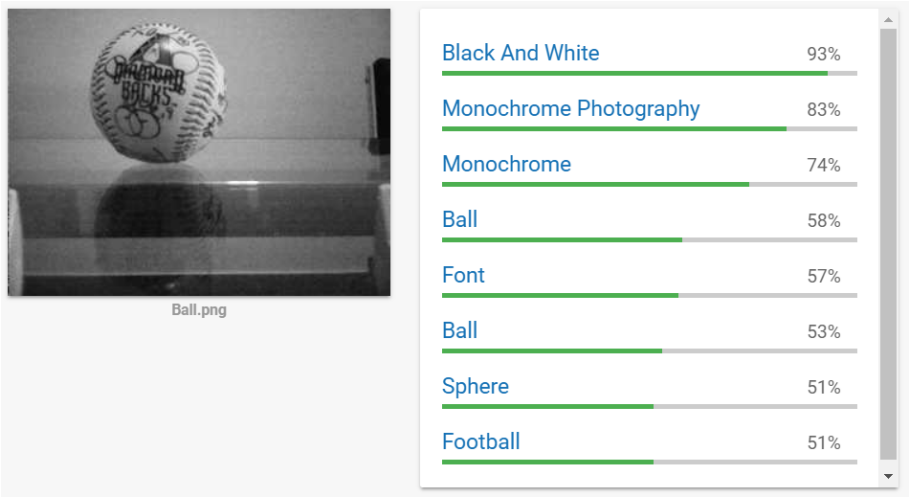


Fonte: Autor

Como pode ser visto nas Figuras 30 e 33, os resultados da identificação de alegria e foram positivos. Porém, devido a baixa resolução da câmera de Cozmo, não foi possível reconhecer as emoções de tristeza e raiva, como ilustrado nas Figuras 31 e 32. Devido a impossibilidade de reconhecer tais emoções, este sistema de reconhecimento não foi implementado no projeto.

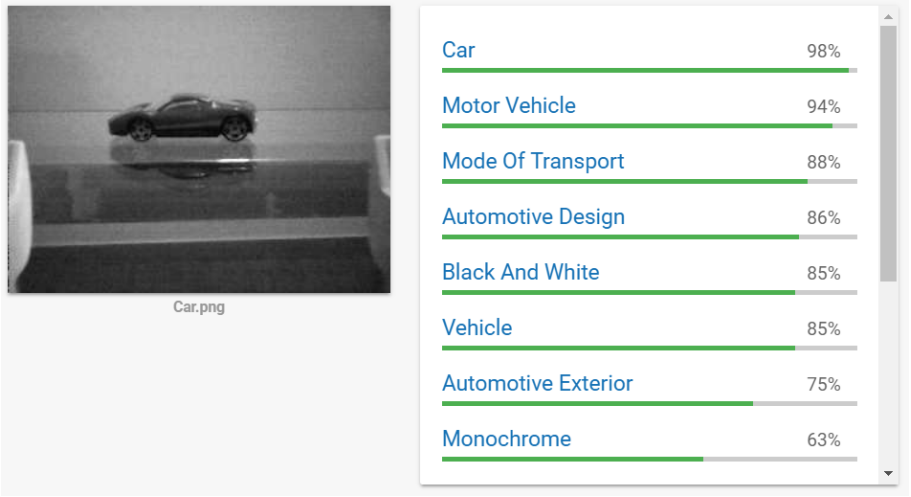
Quatro objetos foram utilizados no método de reconhecimento de objetos: uma bola de beisebol, um carro em miniatura, um maracujá e um cubo de mágico. As Figuras 34, 35, 36 e 37 apresentam os resultados.

Figura 34 – Teste de reconhecimento de uma bola de beisebol utilizando o Google Cloud Vision.



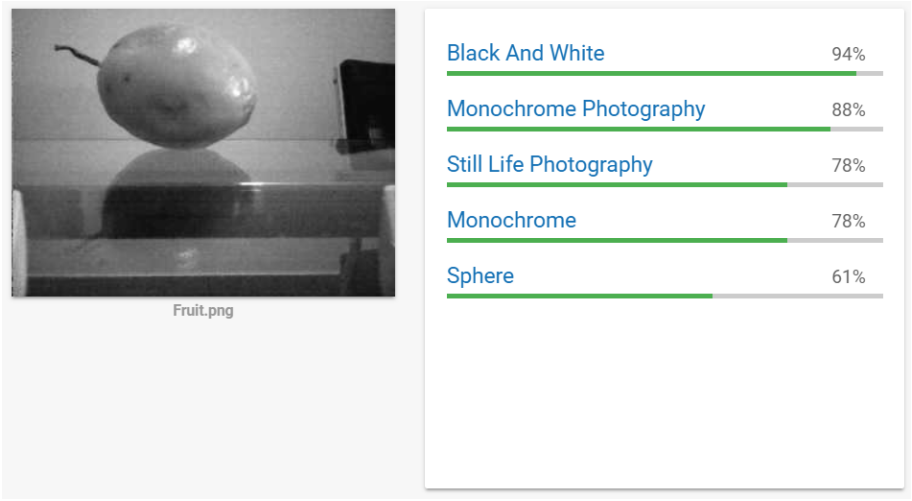
Fonte: Autor

Figura 35 – Teste de reconhecimento de uma carro de miniatura utilizando o Google Cloud Vision.



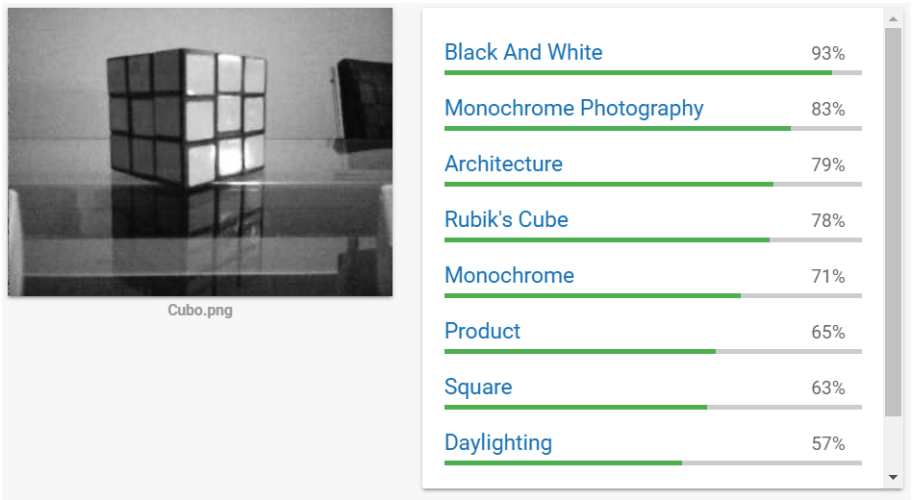
Fonte: Autor

Figura 36 – Teste de reconhecimento de um maracujá utilizando o Google Cloud Vision.



Fonte: Autor

Figura 37 – Teste de reconhecimento de cubo mágico utilizando o Google Cloud Vision.



Fonte: Autor

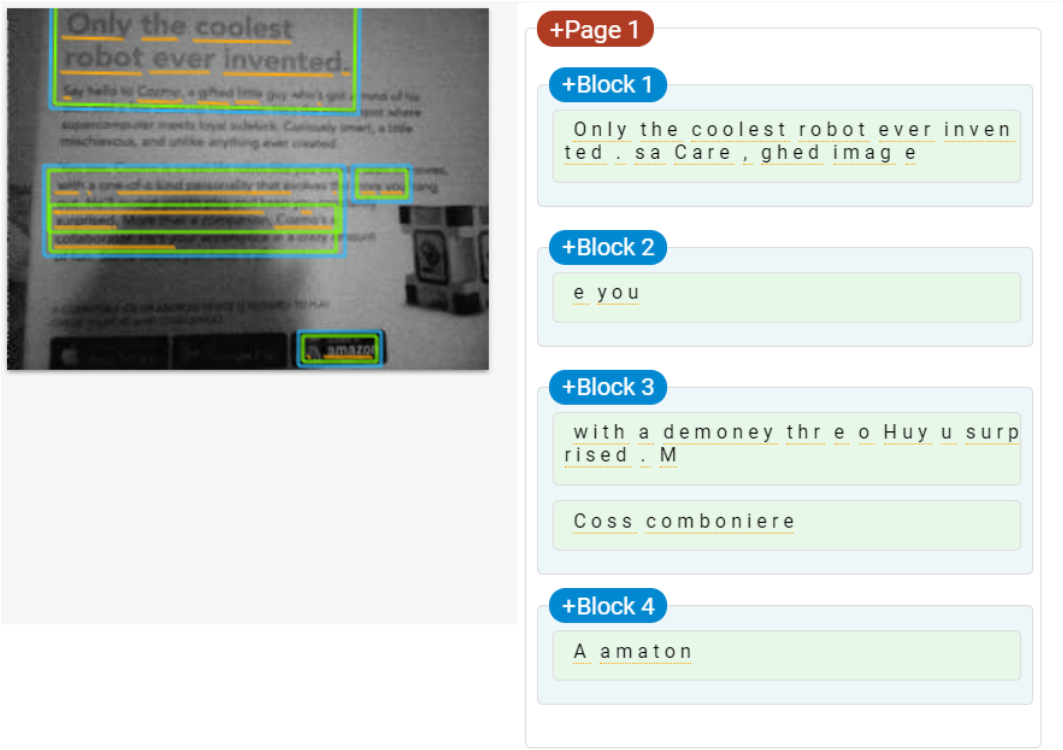
Como pode ser visto nas Figuras 34, 35 e 37, a bola, o cubo e o carro foram reconhecidos com sucesso. O maracujá foi identificado como uma esfera, o que pode-se considerar como um resultado coerente devido ao ser formato aredondado. Os resultados indicam que o sistema de visão computacional implementado foi capaz de identificar todos objetos testados.

A última ferramenta de visão computacional testada foi a de reconhecimento de textos em uma foto tirada pela câmera do Cozmo. Para isto, utilizou-se uma imagem de letras grandes e pequenas. As Figuras 38 e 39 apresentam uma foto tirada por uma câmera de um *smartphone* e os resultados do teste, respectivamente.

Figura 38 – Foto tirada por uma câmera de *smartphone*.

Fonte: Autor

Figura 39 – Resultado do reconhecimento de texto com uma foto tirada pela câmera de Cozmo.



Fonte: Autor

Devido à baixa resolução da câmera de Cozmo, somente as palavras maiores foram identificadas com sucesso, como pode ser visto na Figura 39.

Durante os testes de desempenho, alguns problemas foram encontrados na conexão com o servidor web. Executando as mesmas funções repetidamente, verificou-se que determinadas vezes o sistema apresentou falhas de conexão e não foi possível executar os comandos no Cozmo. Com a ocorrência deste problema, foi necessário reiniciar o algoritmo Python para o sistema voltar a funcionar normalmente.

Para validar o sistema desenvolvido, foram realizados diversos testes com dois professores voluntários do ensino fundamental, uma sala de aula do ensino fundamental e com uma professora de robótica educacional. Todos os testes foram realizados na cidade de Assen, localizada na Holanda. Elaborou-se um questionário para ser respondido pelos professores no final dos testes. A primeira escola a participar foi a CBS Compass.

Na escola CBS Compass, realizou-se os testes apenas com uma professora. Uma breve explicação dos comandos e da página web foi dada e em seguida, pediu-se para a professora executar algumas funções. Após os testes, perguntas foram feitas para avaliar o sistema desenvolvido. A professora não era familiarizada com nenhuma linguagem de programação, mas quando foi perguntado sobre a dificuldade da linguagem criada, a resposta obtida “fácil de se entender”. A professora afirmou que aplicaria o projeto desenvolvido como ferramenta de auxilio de ensino em suas aulas. No entanto, foi citado que pode ser um problema o Cozmo falar apenas os idiomas inglês, alemão, francês e japonês, pois esta restrição limita a aplicação do sistema proposto. A Tabela 2 do Anexo B apresenta as respostas do questionário respondido pela professora da escola CBS Compass.

O segundo teste foi realizado com uma professora de robótica educacional, que utiliza o robô Ozobot para ensinar conceitos básicos de programação a alunos entre 8 e 12 anos de idade. Devido a professora possuir conhecimentos em linguagens de programação, a explicação do projeto foi mais técnica e curta. Além disso, a professora afirmou que todos os alunos ficam muito entusiasmados trabalhando com os robôs Ozobots e isto facilita todo o processo de aprendizagem. A Tabela 3 do Anexo B apresenta as respostas do questionário respondido pela professora.

O último teste foi feito na escola Emmaschool, com uma professora e sua turma de alunos com idades entre 10 e 12 anos. A primeira etapa dos testes foi realizada pelo autor deste projeto, com os alunos da sala. Uma apresentação de Cozmo foi feita para a turma. A primeira função testada foi a de desenhar as figuras geométricas círculo e quadrado. Durante a execução destas funções, observou-se que todos os alunos estavam focados apenas no robô. Após o Cozmo finalizar o desenho, os alunos demonstraram uma reação de surpresa e aplaudiram. A segunda função testada foi a de soma. Solicitou-se um voluntário para participar do teste e a maioria dos alunos demonstraram interesse. Uma aluna foi escolhida pela professora e então, implementou-se a função`sum` para o robô

para pedir a solução de “5 + 1”. A aluna respondeu corretamente e Cozmo reproduziu a animação feliz. Observou-se que a aluna demonstrou uma reação de satisfação por ter acertado, e de felicidade por estar interagindo com o robô. A última função testada foi a de soletrar palavras. Outro estudante voluntário foi escolhido para participar do teste. A palavra a ser soletrada era o nome da aluna escolhida: Anne. A estudante soletrou as letras “A”, “n” e “n” corretamente mas não acertou a última letra “e”. Com isso, Cozmo reproduziu a animação triste para demonstrar que a letra foi soletrada incorretamente.

Devido ao pouco tempo disponível para realizar os testes com a turma, não foi possível aplicar todas as funções desenvolvidas. No entanto, o sistema proposto apresentou resultados satisfatórios durante os testes realizados com os alunos. Foi observado que todos os alunos ficaram muito entusiasmados e demonstraram interesse em interagir com o Cozmo.

Após o teste com a turma, as funções foram mostradas à professora e explicadas como implementá-las no Cozmo. Em seguida, a professora respondeu ao questionário e suas respostas foram registradas na Tabela 4.

Por fim, foram realizados alguns testes com o sistema utilizando o Raspberry PI, conforme foi descrito na 4.5. Os testes foram realizados apenas em ambiente de laboratório com finalidade de validar o sistema desenvolvido. Devido a limitação das funções, executou-se apenas os *scripts* que controlam os movimentos de Cozmo. Todas as funções de movimentos foram executadas com sucesso, comprovando assim que o Raspberry pode ser utilizado como um servidor web para dar comandos ao Cozmo.

6 Conclusão

Neste projeto, desenvolveu-se um sistema de servidor web, reconhecimento de voz, visão computacional e controlador de trajetória para executar funções e jogos educacionais no robô Cozmo.

Devido a baixa resolução da câmera do robô, os testes realizados para o sistema de reconhecimento de emoções utilizando o Google Cloud Vision apresentaram resultados insatisfatórios. Portanto, o reconhecimento de emoções não foi implementado neste projeto. No entanto, o sistema de reconhecimento de objetos apresentou resultados satisfatórios, de forma que a identificação dos objetos testados foi realizada com sucesso. Além disso, o sistema de detecção de texto identificou as palavras com as maiores letras na foto da Figura 3. O sistema de reconhecimento de voz também apresentou resultados positivos. Quando pronunciado as palavras de forma correta, o sistema apresentou os resultados em tempo real.

Os desenhos das figuras geométricas apresentaram resultados precisos, como é ilustrado na Figura 28. Além disso, com o controlador de trajetória, foi possível programar Cozmo para desenhar qualquer figura desejada. Os resultados mostraram uma grande semelhança entre as figuras desenhadas no computador e por Cozmo, como é ilustrado no vídeo disponibilizado na seção 5.

Durante o desenvolvimento do projeto, alguns erros de conexão ocorreram com o servidor Web, sendo necessário reiniciar o sistema. Tais problemas podem causar algumas dificuldades para os professores implementarem o sistema nas salas de aula. No entanto, na maioria das vezes não ocorreu nenhum problema de conexão.

Os testes realizados com os professores apresentaram resultados positivos. As respostas do questionários demonstraram de forma geral que o sistema proposto é aplicável para o uso em salas de aula e a linguagem de programação implementada é simples e intuitiva. Além disso, os três professores afirmaram que aplicariam este projeto em suas aulas.

Os testes realizados com os alunos também apresentaram resultados positivos. Todos os alunos demonstraram interesse em brincar com Cozmo para aprender com as funções educacionais. Além disso, com uma breve explicação, os alunos conseguiram facilmente entender o que tinham que fazer para interagir com Cozmo.

A limitação dos idiomas falados por Cozmo pode ser um problema para implementar o projeto nas escolas. O robô suporta apenas os idiomas inglês, japonês, alemão e francês. Deste modo, seu uso é limitado à escolas que falam essas línguas.

Por fim, verificou-se a possibilidade de utilizar um Raspberry PI 2 para atuar como servidor e executar os comandos no robô Cozmo. Os testes realizados com este sistema apresentaram resultados satisfatórios e foi possível executar todas as funções que controlam os movimentos de Cozmo.

Referências

AHLGREN, D. J. Meeting educational objectives and outcomes through robotics education. In: IEEE. *Proceedings of the 5th Biannual World Automation Congress*. [S.l.], 2002. v. 14, p. 395–404. Citado na página 21.

ALVES, S. F. et al. Educational environment for robotic applications in engineering. In: SPRINGER. *International Conference on Research and Education in Robotics*. [S.l.], 2011. p. 17–28. Citado na página 21.

ALVES, S. F. et al. Proposal of educational environments with mobile robots. In: IEEE. *2011 IEEE 5th International Conference on Robotics, Automation and Mechatronics (RAM)*. [S.l.], 2011. p. 264–269. Citado na página 21.

ANKI. *Cozmo*. 2017. Disponível em: <https://developer.anki.com/>. Citado na página 28.

ANKI. *Cozmo Code Lab*. 2017. Disponível em: <https://developer.anki.com/blog/news/cozmo-code-lab/index.html>. Citado na página 28.

ANKI. *Cozmo SDK Windows Installation*. 2017. Disponível em: <http://cozmosdk.anki.com/docs/install-windows.html>. Citado na página 29.

AROCA, R. V. Plataforma robótica de baixíssimo custo para robótica educacional. Universidade Federal do Rio Grande do Norte, 2012. Citado 2 vezes nas páginas 23 e 24.

BORGES, G. A.; LIMA, A. M.; DEEP, G. S. Controladores cinemáticos de trajetória para robôs móveis com traçao diferencial. *Simpósio Brasileiro de Automação Inteligente–SBAI, Bauru-SP, BRA*, 2003. Citado na página 31.

CONRAD, J. M. Stiquito for robotics and embedded systems education. *Computer*, IEEE, v. 38, n. 6, p. 77–81, 2005. Citado na página 21.

DOCS, M. W. *O que é um servidor web?* 2019. Disponível em: <https://developer.mozilla.org/pt-BR/docs/Learn/Common_questions/o_que_e_um_web_server>. Citado 2 vezes nas páginas 29 e 30.

DOCS, M. web. *Elementos HTML*. 2019. Disponível em: <https://developer.mozilla.org/pt-BR/docs/Web/HTML/Element>. Citado na página 31.

EIS, D. O básico: O que é html. *URL https://tableless. com. br/o-que-html-basico*, 2011. Citado na página 30.

FELIX, E.; AZEVEDO, A. Geometria: como trabalhar os conceitos geométricos nas séries iniciais do ensino fundamental. *Revista Científica de Ciências aplica FAIP. Marília–SP*, p. 1–14, 2015. Citado na página 42.

FRIGHETTO, A. M.; SANTOS, J. C. d. et al. Dificuldade de aprendizagem na leitura. *Nativa-Revista de Ciências Sociais do Norte de Mato Grosso*, v. 1, n. 2, 2013. Citado na página 49.

HAN, J. Emerging technologies: Robot assisted language learning. *Language Learning & Technology*, v. 16, n. 3, p. 1–9, 2012. Citado na página 23.

JEONG, G.-M. et al. A study on the education assistant system using smartphones and service robots for children. *International Journal of Advanced Robotic Systems*, SAGE Publications Sage UK: London, England, v. 11, n. 4, p. 71, 2014. Citado na página 23.

KUSUMOTA, V. et al. Development of a smartphone-based educational robot with cloud architecture and evaluation of its performance. In: IEEE. *2018 Latin American Robotic Symposium, 2018 Brazilian Symposium on Robotics (SBR) and 2018 Workshop on Robotics in Education (WRE)*. [S.l.], 2018. p. 541–546. Citado 3 vezes nas páginas 13, 25 e 26.

MAIA, I. *Building Web Applications with Flask*. [S.l.]: Packt Publishing Ltd, 2015. Citado na página 40.

MANSEUR, R. Development of an undergraduate robotics course. In: IEEE. *Proceedings Frontiers in Education 1997 27th Annual Conference. Teaching and Learning in an Era of Change*. [S.l.], 1997. v. 2, p. 610–612. Citado na página 21.

MARTINS, F. N. et al. An adaptive dynamic controller for autonomous mobile robot trajectory tracking. *Control Engineering Practice*, Elsevier, v. 16, n. 11, p. 1354–1363, 2008. Citado na página 31.

MARTINS, F. N.; GOMES, I. S.; SANTOS, C. R. F. Junior soccer simulation: providing all primary and secondary students access to educational robotics. In: IEEE. *2015 12th Latin American Robotics Symposium and 2015 3rd Brazilian Symposium on Robotics (LARS-SBR)*. [S.l.], 2015. p. 61–66. Citado na página 23.

PI, R. *Raspberry Pi 2 Model B*. 2019. Disponível em: <https://www.raspberrypi.org/products/raspberry-pi-2-model-b/>. Citado na página 33.

PI, R. *What is a Raspberry Pi?* 2019. Disponível em: <https://www.raspberrypi.org/help/what-%20is-a-raspberry-pi/>. Citado na página 33.

PINTEREST. *Anki Cozmo Robot*. 2019. Disponível em: <https://www.pinterest.com/pin/526921225148696006/>. Citado na página 27.

REIS, C. A. d. S.; SARMENTO, H. R.; ZARAMELLA, V. *Ferramenta de auxílio ao desenvolvimento do pensamento computacional: uma plataforma robótica controlada por smartphone*. Dissertação (B.S. thesis) — Universidade Tecnológica Federal do Paraná, 2014. Citado na página 24.

T3N. *Programa Cozmo via Code Lab*. 2019. Disponível em: <https://t3n.de/news/anki-cozmo-code-lab-833493/anki-cozmo-codelab_ipad-challenge/>. Citado na página 28.

TORRES, V.; AROCA, R.; BURLAMAQUI, A. Ambiente de programação baseado na web para robótica educacional de baixo custo/web based programming environment for low-cost educational robotics. *HOLOS*, Instituto Federal de Educacao Ciencia e Tecnologia do Rio Grande do Norte, v. 30, n. 5, p. 252, 2014. Citado na página 25.

TUR, J. M.; PFEIFFER, C. F. Mobile robot design in education. *IEEE Robotics & Automation Magazine*, IEEE, v. 13, n. 1, p. 69–75, 2006. Citado na página 21.

Apêndices

APÊNDICE A – Código Python

Listagem A.1 – Algoritmo do controlador de trajetória

```
#Desired x and y positions
var_x = list_pos_x
var_y = list_pos_y

a = 40
pose = robot.pose
phi_init = pose.rotation.angle_z.radians
x_init,y_init,z_init = pose.position.x_y_z
x_init = x_init + a*math.cos(phi_init)
y_init = y_init + a*math.sin(phi_init)

x = 0
y = 0
phi = 0

dxd = 0
dyd = 0

#Initial robot speeds:
dx = 0
dy = 0

#Initial robot pose ant
x_ant = 0
y_ant = 0
phi_ant = 0



#Controller gains:
kx = 0.6
ky = 0.6
```

```
#Vectors to store simulation data:
x_vector = list()
y_vector = list()
phi_vector = list()
u_vector = list()
w_vector = list()
u_ref_vector = list()
w_ref_vector = list()
dx_vector = list()
dy_vector = list()

xd_vector = list()
yd_vector = list()

t_vector = list()

tempo_perc = 0


time_0 = time.time()
init_var = True

var_dist = False
cal_a_b = True

var_x_final = []
var_y_final = []
ind_x_y = 0

t_delay = 0.03


#Create points between the desired x and y
while (ind_x_y<=len(var_x)-1 and close_window == False):
if (ind_x_y<len(var_x)-1):
       if (math.sqrt(pow(var_x[ind_x_y + 1] - var_x[ind_x_y] ,2) + pow(
          var_y[ind_x_y + 1] - var_y[ind_x_y],2)) > 30):
```

```
                var_x_final = var_x_final + np.linspace(var_x[ind_x_y-1],
                   var_x[ind_x_y],50).tolist()
                var_y_final = var_y_final + np.linspace(var_y[ind_x_y-1],
                   var_y[ind_x_y],50).tolist()
                pass


        else:
                var_x_final = var_x_final + np.linspace(var_x[ind_x_y],
                   var_x[ind_x_y+1],6).tolist()
                var_y_final = var_y_final + np.linspace(var_y[ind_x_y],
                   var_y[ind_x_y+1],6).tolist()

else:

        var_x_final = var_x_final + np.linspace(var_x[ind_x_y-1],var_x[
           ind_x_y],50).tolist()
        var_y_final = var_y_final + np.linspace(var_y[ind_x_y-1],var_y[
           ind_x_y],50).tolist()

ind_x_y = ind_x_y + 1

if (close_window == False):
robot.set_lift_height(0).wait_for_completed()
ind_x_y = 0

#Init the trajectory controller
while (ind_x_y<=len(var_x_final)-2 and close_window == False):

time.sleep(t_delay)

#Get the desired position and speed for time t:
if (var_dist == False):

        xd,yd = var_x_final[ind_x_y],var_y_final[ind_x_y]]

dxd = (var_x_final[ind_x_y+1] - var_x_final[ind_x_y])/t_delay
dyd = (var_y_final[ind_x_y+1] - var_y_final[ind_x_y])/t_delay
```

```
#Call the controller to generate reference commands u_ref and w_ref:
v_ref = traj_tracking_controller(dxd, dyd, xd, yd, x, y, phi, a, kx, ky)


#Apply this signal in the robot
u_ref = v_ref[0][0]
w_ref = v_ref[1][0]

#Saturator
u_max = 220 # max linear speed [mm/s]
w_max = 9.82 # max angular speed [rad/s]

# If speed commands from the controller are higher than the maximum
   speeds
# achivable by the robot, saturate:
u = np.sign(u_ref)*min(abs(u_ref), u_max)
w = np.sign(w_ref)*min(abs(w_ref), w_max)

v_right = u + (44.8/2)*w
v_left = u - (44.8/2)*w


robot.drive_wheels(v_left,v_right)

pose = robot.pose
phi_now = pose.rotation.angle_z.radians
x_now,y_now,z_now = pose.position.x_y_z
x_now = x_now + a*math.cos(phi_now)
y_now = y_now + a*math.sin(phi_now)


x = x_now - x_init
y = y_now - y_init

phi = phi_now - phi_init

x_final = x - x_ant
y_final = y - y_ant
```

```
if (init_var == True):
        time_ant = 0
        init_var = False

time_now = micros()/1000000

tot_time = time_now - time_ant
time_ant = time_now

dx = x_final/tot_time
dy = y_final/tot_time

v_right_real = robot.right_wheel_speed.speed_mmps
v_left_real = robot.left_wheel_speed.speed_mmps

u = (v_right_real + v_left_real)/2
u = float(u)
w = (phi-phi_ant)/tot_time



x_ant = x
y_ant = y
phi_ant = phi

#Keep phi within [-pi,pi]:
if (phi > math.pi):
        phi = phi - 2*math.pi
else:
        if (phi < -math.pi):
                phi = phi + 2*math.pi

if (math.sqrt(pow(var_x_final[ind_x_y + 1] - var_x_final[ind_x_y] ,2) +
   pow(var_y_final[ind_x_y + 1] - var_y_final[ind_x_y],2)) > 20):
        if (math.sqrt(pow(var_x_final[ind_x_y + 1] - x ,2) + pow(
           var_y_final[ind_x_y + 1] - y,2)) > 20):

                if (cal_a_b == True):

                        robot.set_lift_height(0.2).wait_for_completed()
```

```
                    cal_a_b = False
                    dist_x_y = math.sqrt(pow(var_x_final[ind_x_y + 1] -
                       var_x_final[ind_x_y] ,2) + pow(var_y_final[
                       ind_x_y + 1] - var_y_final[ind_x_y],2))
                    dist_x_y = int(dist_x_y)

                    var_x_no_draw = np.linspace(var_x_final[ind_x_y],
                       var_x_final[ind_x_y + 1],dist_x_y).tolist()
                    var_y_no_draw = np.linspace(var_y_final[ind_x_y],
                       var_y_final[ind_x_y + 1],dist_x_y).tolist()
                    ind_no_draw = 0
                    time.sleep(0.5)


            if (ind_no_draw<len(var_x_no_draw) - 2):
                    xd = var_x_no_draw[ind_no_draw]
                    yd = var_y_no_draw[ind_no_draw]

                    dxd = 0
                    dyd = 0
                    var_dist = True

            ind_no_draw = ind_no_draw + 1

    else:
            robot.set_lift_height(0).wait_for_completed()
            var_dist = False
            cal_a_b = True


#Store simulation data:
if (var_dist == False):
    x_vector.append(x)
    y_vector.append(y)
    phi_vector.append(phi)
    u_vector.append(u)
    w_vector.append(w)
    u_ref_vector.append(u_ref)
    w_ref_vector.append(w_ref)
```

```
        dx_vector.append(dx)
        dy_vector.append(dy)

        xd_vector.append(xd)
        yd_vector.append(yd)

        tempo_perc = time.time() - time_0
        t_vector.append(tempo_perc)


        ind_x_y = ind_x_y + 1

#Plot results:

if (close_window == False):
plt.figure(1)
plt.plot(t_vector,u_vector)
plt.title('Linear speed')
plt.ylabel('u [m/s]')
plt.xlabel('time [s]')
plt.grid(True)

plt.figure(2)
plt.plot(t_vector,w_vector)
plt.title('Angular speed')
plt.ylabel('u [rad/s]')
plt.xlabel('time [s]')
plt.grid(True)

plt.figure(3)
plt.plot(x_vector, y_vector)
plt.title('Robot path')
plt.xlabel('x [mm]')
plt.ylabel('y [mm]')
plt.grid(True)

plt.figure(4)
plt.plot(xd_vector, yd_vector)
plt.title('Robot path')
```

```
plt.xlabel('x [mm]')
plt.ylabel('y [mm]')
plt.grid(True)

plt.show()

robot.set_lift_height(0.2).wait_for_completed()
robot.abort_all_actions()
robot.drive_wheels(0,0)
```

APÊNDICE B – Questionário respondido pelos professores

Tabela 2 – Questionário respondido pela professora da escola CBS Compass

Pergunta	**Resposta do professor**
Você já aprendeu programação antes? Qual linguagem de programação?	Eu participei de um *Workshop* antes, mas não conheço nenhuma linguagem de programação.
Você utiliza robôs nas aulas para ensinar? Como é o aprendizado ao usar o robô?	Eu usei bee-bot uma vez. Os alunos amavam porque é algo diferente dos livros e do aprendizado que estão acostumados.
Quão difícil é a linguagem criada neste projeto?	Eu acho que é muito bom e não é muito difícil. Mas talvez para as pessoas mais velhas, um manual pode ser necessário.
As funções de programação são aplicáveis nas aulas de ensino?	Sim. Para minha classe seria possível utilizar apenas as funções matemáticas de soma e subtração. Os alunos não sabem dividir e multiplicar ainda. Mas todas as outras funções podem ser usadas.
Qual é a idade ideal dos alunos para aplicar este projeto?	Eu acho que poderia ser usado em todas as classes. Só precisa estar no nível de aprendizado dos alunos. Para os alunos mais novos, você pode facilitar e, para os alunos mais velhos, tornar um pouco mais difícil.
Você classificaria essa linguagem de programação como flexível?	Sim. Porque você pode escrever suas próprias perguntas.
Para você, qual é a função mais interessante?	Eu gostei de tudo. Eu acho que é possível usar tudo. Mas acho que a parte mais interessante é que você pode realmente fazer suas próprias perguntas para serem feitas pelo robô.
Qual a sua opinião sobre o sistema de comunicação de voz usado para interagir com o sistema?	Eu acho que isso torna a conversa mais natural. É melhor deixar que os alunos falem com o robô do que digitar as respostas.
Quão difícil é entender os itens da página da Web?	Foi fácil de entender.
Existe outro tópico interessante que poderia ser aplicado no projeto?	Talvez criar funções para trabalhar com cores.
Você consideraria usar o projeto em sala de aula?	Sim.

Fonte: Autor

Tabela 3 – Questionário respondido pela professora de robótica educativa.

Pergunta	**Resposta do professor**
Você já aprendeu programação antes? Qual linguagem de programação?	Sim. Pascal, COBOL, C++ e PHP.
Você utiliza robôs nas aulas para ensinar? Como é o aprendizado ao usar o robô?	Sim. Eu gosto, é divertido. Apenas algumas vezes eu percebo que o robô não entende as instruções dadas. Os estudantes ficam bastante entusiasmados.
Quão difícil é a linguagem criada neste projeto?	Até onde eu vi, é simples e bom.
As funções de programação são aplicáveis nas aulas de ensino?	Sim, com certeza. É possível utilizar todas as funções.
Qual é a idade ideal dos alunos para aplicar este projeto?	Estudantes entre 8 e 12 anos de idade.
Você classificaria essa linguagem de programação como flexível?	Sim.
Para você, qual é a função mais interessante?	Eu gostei do sistema de reconhecimento de voz utilizado para interagir com os alunos.
Qual a sua opinião sobre o sistema de comunicação de voz usado para interagir com o sistema?	Eu gostei porque os estudantes podem interagir com o robô. Mas deve-se prestar atenção no caso de houver barulhos na sala que podem interferir com o reconhecimento de voz.
Quão difícil é entender os itens da página da Web?	A princípio, eu não consegui entender a caixa de texto que verificadora de código. Mas após sua explicação, eu pude entender.
Existe outro tópico interessante que poderia ser aplicado no projeto?	Seria interessante se o Cozmo pudesse interagir com o Ozobot, o robô que utilizo com meus estudantes.
Você consideraria usar o projeto em sala de aula?	Sim.

Fonte: Autor

Tabela 4 – Questionário respondido pela professora de escola Emmaschool

Pergunta	**Resposta do professor**
Você já aprendeu programação antes? Qual linguagem de programação?	Não. É a minha primeira vez programando.
Você utiliza robôs nas aulas para ensinar? Como é o aprendizado ao usar o robô?	Esta foi a minha primeira vez utilizando um robô em sala de aula.
Quão difícil é a linguagem criada neste projeto?	É bastante simples e fácil de entender.
As funções de programação são aplicáveis nas aulas de ensino?	Sim. Para a minha classe, principalmente as funções matemáticas.
Qual é a idade ideal dos alunos para aplicar este projeto?	Primeiramente, achei que era mais adequado para as crianças mais novas, porque elas precisam aprender letras e formas. Mas usando esse sistema em minha sala de aula, com alunos de 9 anos, vejo muito potencial para pequenos grupos de crianças que têm dificuldade em assuntos específicos e podem praticar com o robô.
Você classificaria essa linguagem de programação como flexível?	Sim.
Para você, qual é a função mais interessante?	Para minha classe, eu acho que as funções matemáticas
Qual a sua opinião sobre o sistema de comunicação de voz usado para interagir com o sistema?	Até onde eu vi, o sistema funciona e as crianças entenderam facilmente o que elas tinham que fazer.
Quão difícil é entender os itens da página da Web?	Eu acho que é fácil de entender.
Existe outro tópico interessante que poderia ser aplicado no projeto?	Seria interessante se as crianças pudessem fazer perguntas para o robô responder.
Você consideraria usar o projeto em sala de aula?	Sim, especialmente para as crianças que necessitam praticar alguns tópicos um pouco mais.

Fonte: Autor