



MECATRÔNICA

APRENDENDO CIÊNCIA E TECNOLOGIA

JOVEM



ROBÔ LIXINHO

ROBÔ RPI PICO 1

SENSOR DE
CORES

ROBÔ ARDUÍNO
BLUETOOTH



ROBÔ LIMPA AREIA

ROBÔ CONDUTOR DE PACIENTES

**EMPOWERING INNOVATION
TOGETHER™**

Fornecendo energia para um futuro mais sustentável com sistemas de armazenamento de energia verde.

A Mouser Electronics® e a série Juntos Incentivando a Inovação™ exploram a crescente separação entre a captura de energia verde e o armazenamento na rede enquanto apresentam as tecnologias inovadoras que ajudam a fechar esta lacuna.

Ouçã o podcast, assista aos vídeos, leia os artigos técnicos e muito mais em:

br.mouser.com/empowering-innovation



EXPEDIENTE

Revista Mecatrônica Jovem
Revista do Instituto Newton C. Braga
Ano 2 – Edição nº 11 - 2023

Editorchefe

Luiz Henrique Correa Bernardes

Produção Gráfica

Wilson Caruso Junior - @wcjrdesign

Atendimento ao leitor

leitor@newtonbraga.com.br

Atendimento ao cliente

publicidade@newtonbraga.com.br

Conselho editorial

Marcio Jose Soares
Newton C. Braga
Renato Paiotti

Administração

Newton C. Braga (CEO)
Marcelo Lima Braga
(Gerente Administrativo)

Jornalista Responsável

Marcelo Lima Braga
MTB 0064610SP

Autores

Eng. Clarice Barreto - @claricebarretoeng
Julian C. Braga - @julianbraga
Luiz Henrique - Moocalab - @moocalab
Murilo Pinheiro - @eng_murilobrito
Newton C. Braga - @newtonbraga_
Otavio Lemes - @taviolemes3101
Pedro Bertoleti - LinkedIn (Pedro Bertoleti)
Raul Júnior - @teclencois22
Renato Paiotti - @renatopaiotti
Reginaldo Resistrônico - @reginaldo-resistronic
Tio Rafa - @prof.tiorafa
Vander Lab - @vander_lab
VanderTronic - @vandertronic_

Não é permitida a reprodução das matérias publicadas sem prévia autorização dos editores. Não nos responsabilizamos pelo uso indevido do conteúdo de nossos artigos ou projetos.

Atendimento

publicidade@newtonbraga.com.br



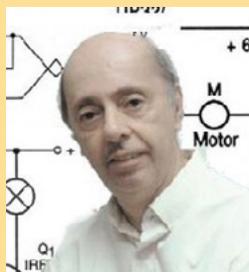
ROBÓTICA

Mais uma edição da Revista do Clube da Mecatrônica Jovem, feita com muito carinho com artigos enviados pelos nossos colaboradores, presentes em todas as regiões do Brasil. Foi explorado o Tema: “Robótica”. Paralelo fizemos um desafio na Live do Clube, onde duas equipes teriam que montar um robô. A Equipe A teria que fazer um robô que levaria os pacientes da recepção ao consultório e a Equipe B teria que fazer um robô que limpasse a areia de uma praia. Durante as lives, foi discutido como montar, realizadas várias alterações no projeto e o resultado você pode ver no artigo nesta edição. Muita diversão na Live com o Raul fazendo demonstração de seus experimentos quânticos. Acompanhe e participe do Clube da Mecatrônica Jovem pelo Discord, o Clube está crescendo e muita gente nova ajudando. No mais, deixe marcado o dia 16 de Setembro que irá acontecer o Segundo Encontro da Mecatrônica Jovem que será no Senai da Vila Mariana em São Paulo. Boa leitura, boas montagens e muita diversão !



LUIZ HENRIQUE CORRÊA BERNARDES

DISCORD <https://discord.gg/sHmBawH6dT>



Sempre gostei de desafios, tanto que por diversas vezes os fiz através de histórias do Professor Ventura, artigos nas nossas revistas do passado e até mesmo numa seção que mantive em que cheguei a dar prêmios. Desta vez não vou dar prêmios, mas certamente os desafios desta edição especial de robótica vão surpreender vocês. Com diversos projetos interessantes para todos os níveis, muitos sugeridos nas nossas lives das quarta feiras você pode adotá-los em sua escola, no seu grupo de makers e onde quiser. Chamamos a sua atenção para o grau de improvisação de cada um, o que demonstra que os projetos descritos não são definitivos, mas estão abertos para uma infinidade de variações. Ponham sua criatividade em ação e nos conte sobre seus resultados. Podemos até publicar seu projeto numa edição futura.

NEWTON C. BRAGA

PARA OS
PROFESSORES

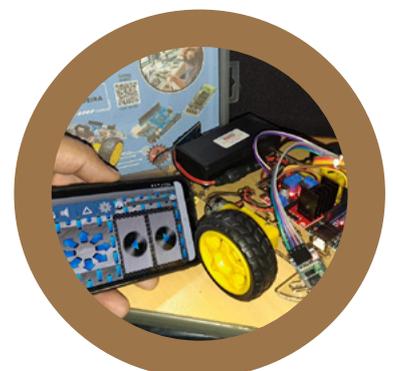


PALAVRAS DO MESTRE

ÍNDICE

MONTAGEM

- 06 ROBÔ CONDUTOR DE PACIENTES
- 10 ROBÔ LIMPA AREIA
- 14 A ESTRUTURA DE PROJETOS DE ROBÓTICA E MECATRÔNICA
- 20 AS LIVES DO DESAFIO DE ROBÓTICA
- 24 ENERGIA QUE MOVE MEU ROBÔ
- 28 PAFÚNCIO
- 36 ROBÔ ARDUINO BLUETOOTH V2 DO TIO RAFA
- 42 ROBÔ CONTROLADO VIA INTERNET COM MÓDULO WIFI LORA 32
- 58 ROBÔ LIXINHO
- 64 ROBÔ RPI PICO 1 - UM ROBÔ BASE PARA SUAS EXPERIÊNCIAS
- 74 SENSOR DE CORES
- 80 TIPOS DE MOVIMENTOS ROBÓTICOS
- 82 MAMUTE CAR

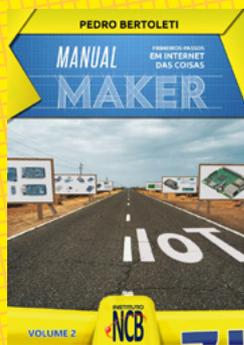


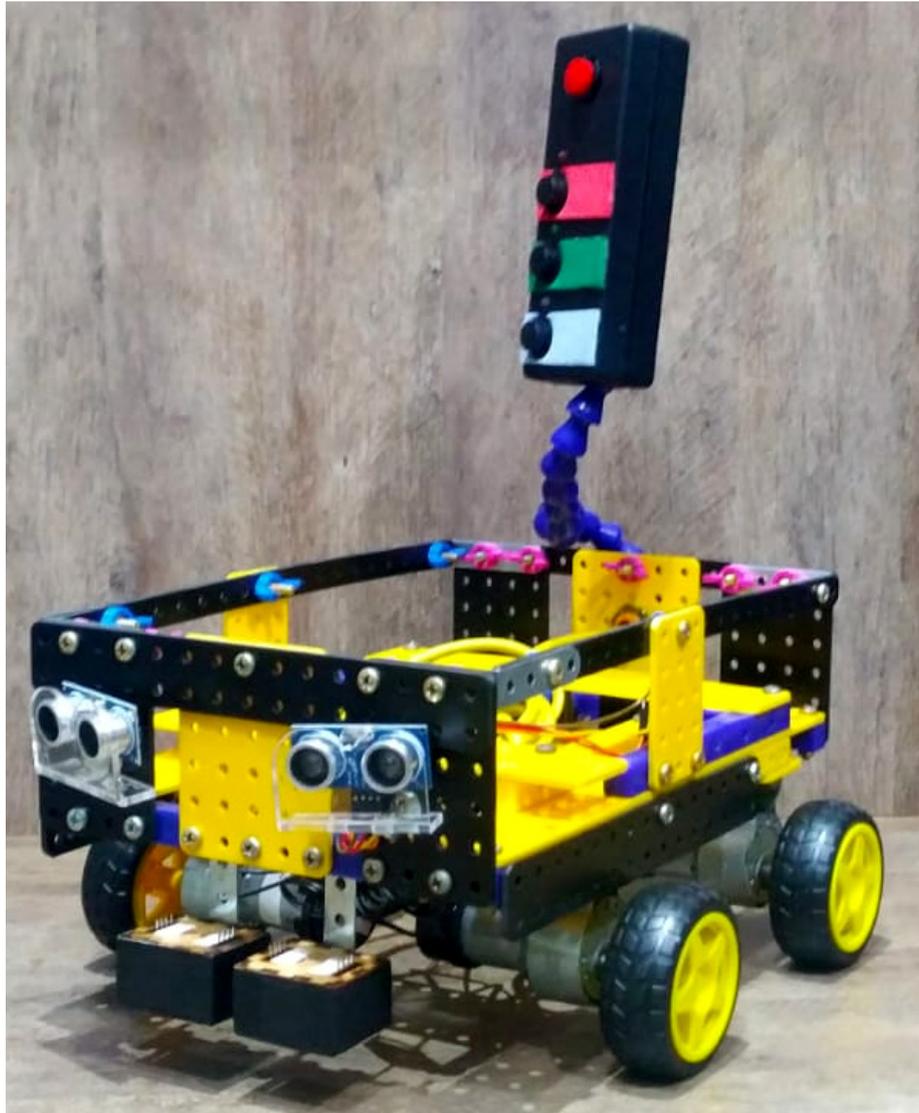
APRENDA ELETRÔNICA



No formato
Impresso e e-Book

newtoncbraga.com.br/livros





ROBÔ CONDUTOR DE PACIENTES

Luiz Henrique

Instagram: @moocalab

VanderLab

Instagram: @vander_lab

Otávio Lemes Rodrigues

Instagram: @taviolemes3101

Reginaldo Resistronic

Instagram: @reginaldoresistronic

Murilo Pinheiro Brito

Instagram: @eng_murilobrito

Este robô foi montado pela equipe A durante as nossas lives sobre o desafio robótico que acontece todas as quartas-feiras às 20 horas. Esta equipe era formada pelo Luiz Henrique, VanderLab, Otávio Lemes Rodrigues, Reginaldo Resistronic e Murilo Pinheiro Brito.

O DESAFIO

Criar um robô que levasse pacientes por dentro de um hospital de forma autônoma. O Robô deveria reconhecer o local e levar o paciente para um ponto específico do hospital.



Figura 1 - O chassi do robô.

SOLUÇÃO

A equipe optou por criar uma cadeira de roda robótica que serviria como um cão guia, através de um painel o paciente ou alguém do hospital apertaria o botão da cor que aquele robô deveria percorrer. O robô faria o percurso através de faixas coloridas no chão (verde, azul ou vermelho) e pararia no local indicado quando chegasse. Sensores de distância evitariam colisões com outros pacientes e funcionários.

DIVIDINDO AS FUNÇÕES

O Vanderlab foi designado o Gerente de Projeto, onde sua função seria de reunir em seu laboratório todas as partes desenvolvidas pelos membros da equipe.

O Murilo ficou responsável pelo sensor de distância e desenvolvimento da placa.

O Otávio ficou responsável pelo sensor de cor e sua lógica.

O Reginaldo da base e motores como o painel de instruções.

O Luiz ... bem... ele deu apoio moral para a equipe.

A BASE

No princípio o Reginaldo Resistrônic montou uma base de acrílico com 4 motores, o que ficou bem maker e muito legal, mas com o andar das lives o VanderLab optou por usar Modelix e assim mais didático, onde os alunos poderiam montar o seu, conforme podemos ver na figura 1.

O robô teria 4 rodas que seriam controladas pelo Arduino Uno através dos módulos de controle de motores. Porém, quando a turma A foi colocando cada vez mais sensores, ficou claro que o Arduino não seria a placa ideal para o controle de todo o sistema, o código ficaria complexo demais para o seu funcionamento. Como o intuito era criar uma plataforma que fosse simples e didática, a equipe optou por usar o ESP32, pois atenderia a demanda, com um código mais simples e ainda poderíamos aprender uma nova plataforma de desenvolvimento.

O SENSOR DE CORES

O reconhecimento das faixas ficou por conta do Otávio, que desenvolveu todo um estudo sobre o TSC3200. Você poderá aprender como ele funciona no artigo publicado nesta edição (Sensor de cor TCS3200).

Com a equipe fazendo reuniões semanais, eles resolveram trocar este sensor por outro com protocolo de comunicação I²C, o sensor TCS34725, porém a física do funcionamento é a mesma, somente a forma como os sensores se comunicam com o microcontrolador é diferente. Na figura 2 podemos ver os dois sensores lado a lado.

OS SENSORES DE DISTÂNCIA

A equipe optou em usar o sensor ultrassônico simples, o mais utilizados nas montagens makers e mais fácil de encontrar nas lojas de componentes. Na figura 3 é possível ver o sensor utilizado.



TSC3200 - TCS34725

Figura 2 - Os sensores de cores.



Figura 3 - Sensor Ultrassônico.

O PAINEL DE CONTROLE

O painel de controle foi também desenvolvido pelo Reginaldo Resistrionic que utilizou botões coloridos para representar qual faixa o robô deveria seguir. Se o paciente apertar o botão azul, o robô deveria seguir a linha azul, e assim com as outras cores. Na figura 4 podemos ver como ficou a versão do controle.



Figura 4 - Controle do Robô.

O CIRCUITO

A equipe pensou em colocar 4 sensores ultrassônicos com sincronismo defasado para não dar interferência entre eles. Nas lives é possível ver o próprio Murilo testando

todos eles ao mesmo tempo. No início ele utilizou dois Arduinos Nano para gerenciar os sensores, como também acoplar os sensores de cor desenvolvidos pelo Otávio. Como o Murilo tem grande conhecimento no desenvolvimento de placas, ele desenvolveu a placa para o robô, e depois de algumas reuniões do grupo no Discord, ele desenvolveu mais duas placas bem mais simples para que possam ser aplicadas nas escolas, uma com dois sensores ultrassônicos e outra com somente uma.

Na figura 5 temos o desenho da placa e na figura 6 o diagrama elétrico.

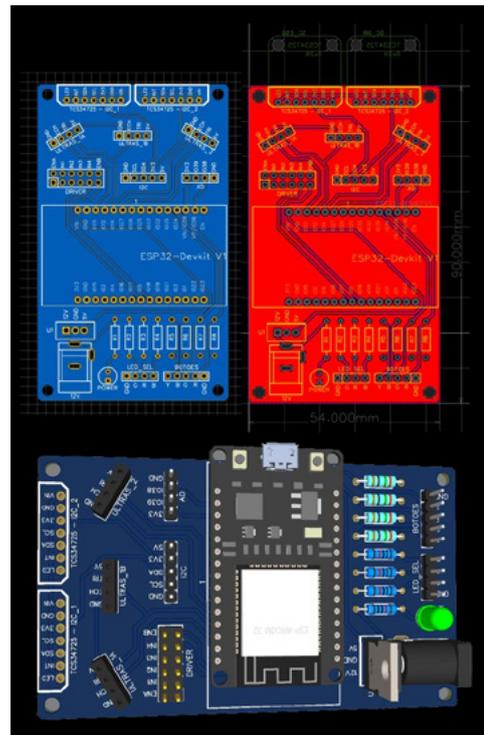


Figura 5 - A placa final.

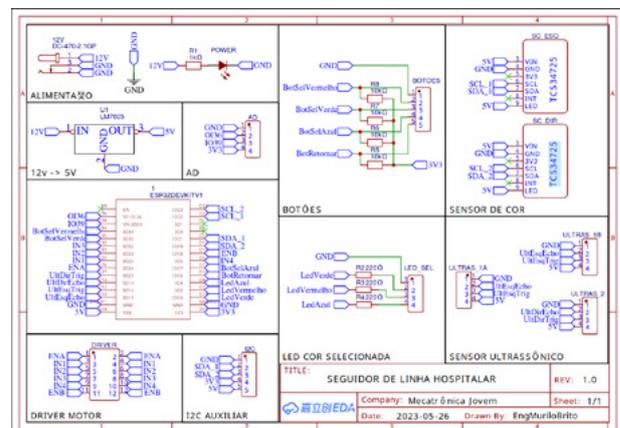


Figura 6 - Diagrama elétrico do robô.

O CÓDIGO

O código aqui foi o utilizado pela equipe durante as lives, vale a pena salientar que dependendo do robô que for montando, sempre alguns ajustes devem se fazer necessários.

A lógica do código consiste na seguinte em acender o LED na cor branca indicando que o robô precisa ser colocado sob a faixa amarela, com as linhas coloridas atrás dele. Uma vez detectado a luz amarela, o LED começará a piscar indicando que algum botão precisa ser selecionado e acionado. Ao pressionar a cor desejada o robô fará a busca pela faixa da cor escolhida, uma vez encontrada a faixa de cor selecionada, o robô seguirá a linha até encontrar o fim dela que é indicado pela faixa amarela, parando totalmente e voltando para a fase em que o LED fica piscando esperando que outra vez o botão da cor desejada seja pressionado.

No link do QR-Code você encontrará o código utilizado no projeto, como também os arquivos gerber para a produção das placas e um arquivo PDF com o esquemático do projeto.



CONCLUSÃO

Este projeto deu várias voltas e é impressionante como a turma conseguiu resolver o problema, várias mudanças ocorreram e muito se aprendeu durante a construção deste robô hospital. Claro que este é apenas um projeto, mas com grandes chances de um dia se tornar viável comercialmente. Lógico que um projeto de tal magnitude não pode ser feito num espaço de 2 meses, mas sim de 2 anos ou mais, mas fica a dica para você aperfeiçoar este projeto, seja para o seu trabalho de escola, TCC ou mesmo para a sua empresa. Um abraço da equipe A.

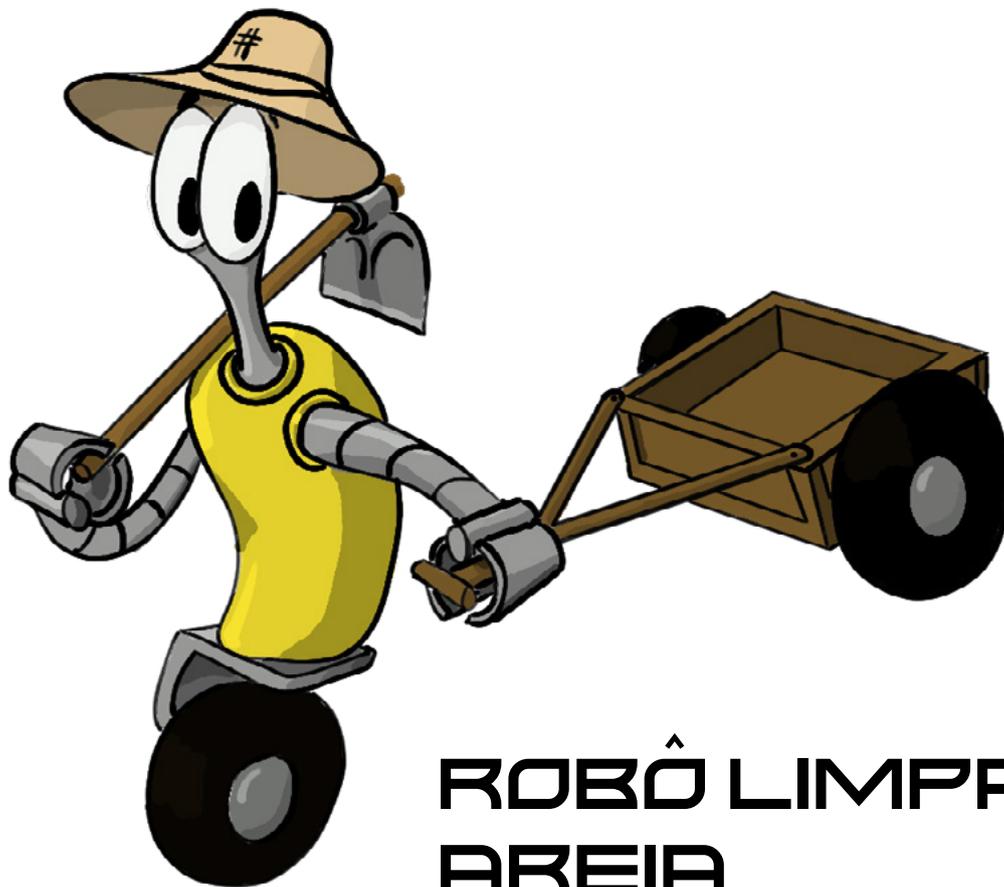
PROJETOS ELETRÔNICOS PARA O ENSINO DE FÍSICA E CIÊNCIAS



Durante muitos anos o autor tem treinado professores de física e ciências para aplicar a eletrônica no ensino de diversas disciplinas, através de experimentos práticos. Neste período ele criou centenas de projetos práticos simples que podem ser implementados com facilidade pelos professores, usando material de baixo custo e sendo fáceis de montar e entender até mesmo os estudantes do nível fundamental.

No formato
Impresso ou
e-Book





ROBÔ LIMPA AREIA

VanderTronic

Instagram: @vandertronic_

Tio Rafa

Instagram: @prof.tiorafa

Raul Júnior

<https://youtube.com/@raulaser>

José Igor Modesto Cruz

Salvador da Equipe B

Este robô foi desenvolvido durante as lives que realizamos todas as quartas-feiras às 20 horas da noite. Depois que a equipe B foi formada por Vandertronic, Tio Rafa e o Raulaser, eles receberam o desafio de criar um robô que limpasse a areia das praias, mas tempos depois a equipe B recebeu a grande ajuda do José IGOR Modesto Cruz. O artigo a seguir mostra como foi realizado este desafio que precisou de muita engenhosidade.

O PROBLEMA

Muitos os chamados “seres humanos” não cuidam das praias da forma que deveriam, sujando a areia com os mais diversos tipos de lixo. Para deixar as praias limpas

e assim ajudar a natureza, a equipe B teria que desenvolver um robô que retirasse o lixo das praias de forma autônoma.

PENSANDO NA SOLUÇÃO

A equipe B pensou em desenvolver um robô que tivesse uma esteira rotativa que varresse a areia e tudo que nela tivesse para dentro de uma caixa, esta caixa teria um fundo onde uma peneira que teria que reter o lixo e soltar a areia.

Porém o robô estaria andando por locais onde outras pessoas estariam descansando e curtindo a praia, logo

o robô teria que tomar cuidado para onde ir. Para este problema a solução é equipar o robô com sensores ultrassônicos.

A ESTRUTURA

A estrutura foi feita com tubos de ferro, uma caixa de madeira com o fundo vazado, porém com uma rede para segurar o lixo, sensores ultrassônicos para proteger o robô de colisões e para medir o nível de lixo dentro da caixa, uma esteira feita com uma rede e tubos de PVC, rodas, motores para as rodas e esteira.

Ao todo foram colocados 3 motores de 12 V, sendo 2 para as rodas e um para a esteira.

A areia e o lixo caem numa caixa onde o fundo é uma tela, com o movimento, a areia escorre pela tela deixando somente o lixo. Para aproveitar o movimento do carro e do motor da esteira, a caixa estaria presa na estrutura por elásticos, conforme podemos ver na figura 1.

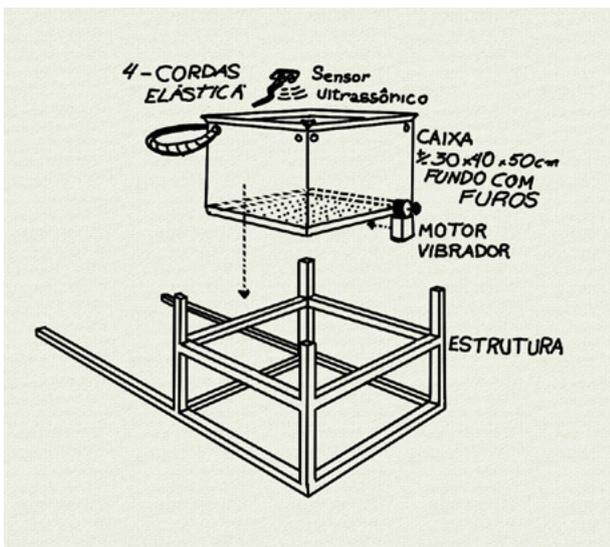


Figura 1 - Desenho da estrutura da caixa de depósito.

O CIRCUITO

O circuito foi projetado utilizando o ESP32 como o microcontrolador principal, o circuito é apresentado na figura 2. Em nossa página no site você encontrará o andamento e a finalização deste projeto proposto nas lives.

A PROGRAMAÇÃO

O Robô Areia possui dois tipos de programas, o primeiro que roda no ESP32 e outro no telefone celular que controla o robô. Na figura 3 temos a tela e a programação do controle, um arquivo na extensão APK que deve ser instalado no aparelho celular. A tela do aplicativo foi desenvolvida no App Inventor, que está livre para ser usado e modificado. Todos os arquivos poderão ser baixados, clicando ou fotografando o QR-code "Arquivos do projeto".

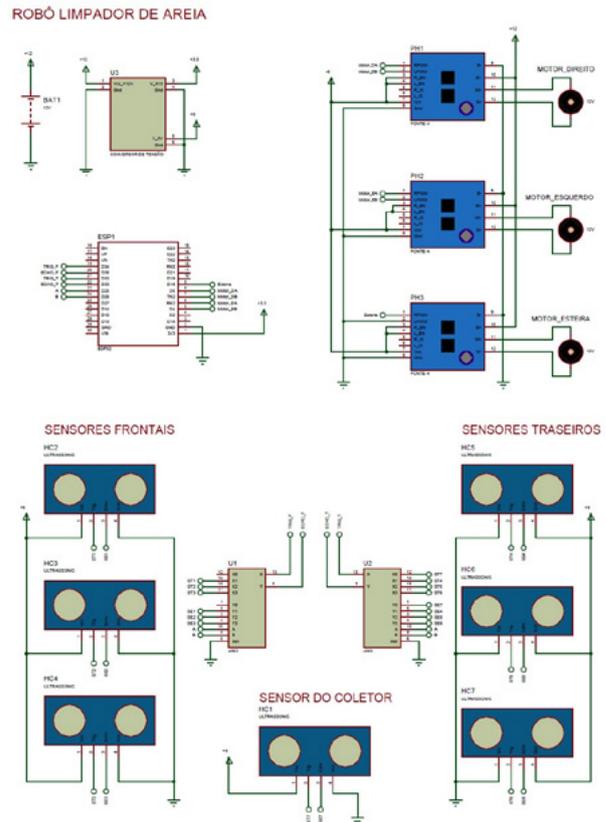


Figura 2 - O circuito do robô areia.

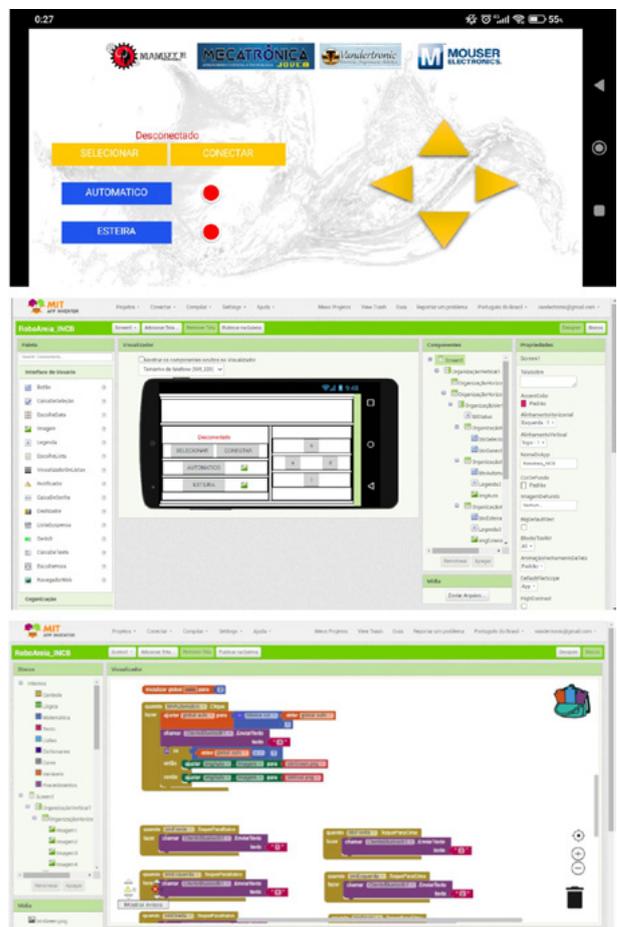


Figura 3 - Tela do aplicativo desenvolvido pela equipe B.

O QUE VOCÊ PODERÁ FAZER?

Este projeto foi executado com a equipe B num prazo bem curto de 2 meses, mas você poderá aprimorar o projeto e apresentar como seu trabalho de ciência, TCC ou mesmo um projeto para a prefeitura da sua cidade, com mais tempo e testes. Vimos que a ideia é viável e aprimorar pode ser a sua chance de mostrar o que você pode fazer. Quando terminar o seu robô limpador de areia, envie uma foto ou filme para podermos compartilhar aqui na nossa revista e em nosso Clube da Mecatrônica Jovem.

CONCLUSÃO

O projeto foi bem divertido de ser feito e a turma aprendeu bastante com ele, a troca de conhecimento entre os membros da equipe foi fantástica onde todos aprenderam e ensinaram alguma coisa interessante.

A equipe B agradece a todos que colaboraram na construção deste grande robô limpador de praias.

ARQUIVOS DO PROJETO

Para encontrar todos os arquivos utilizados neste projeto, basta clicar ou fotografar o QR-Code deste quadro e baixar o arquivo ZIP, dentro dele você encontrará os esquemas elétricos feitos em Protheus, o APK para o celular e a sua programação no Inventor.



PROJETOS ELETRÔNICOS EDUCACIONAIS COM ENERGIA ALTERNATIVA

Newton C. Braga



PROJETOS DIDÁTICOS PARA OS FUTUROS ENGENHEIROS

No formato
Impresso ou
e-Book



WWW.MAMUTEELETRONICA.COM.BR

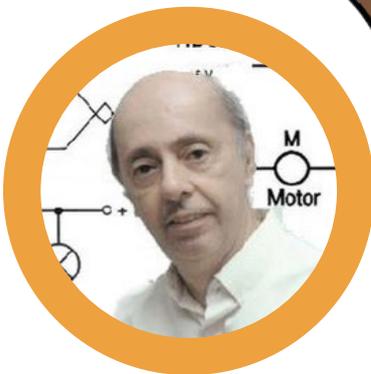


MAMUTE Eletrônica



RUA: VITÓRIA 125
SANTA IFIGÊNIA - SÃO PAULO
CEP: 01210-001
(11) 3222-8816 | (11) 3222-8774





Newton C. Braga

www.newtoncbraga.com.br

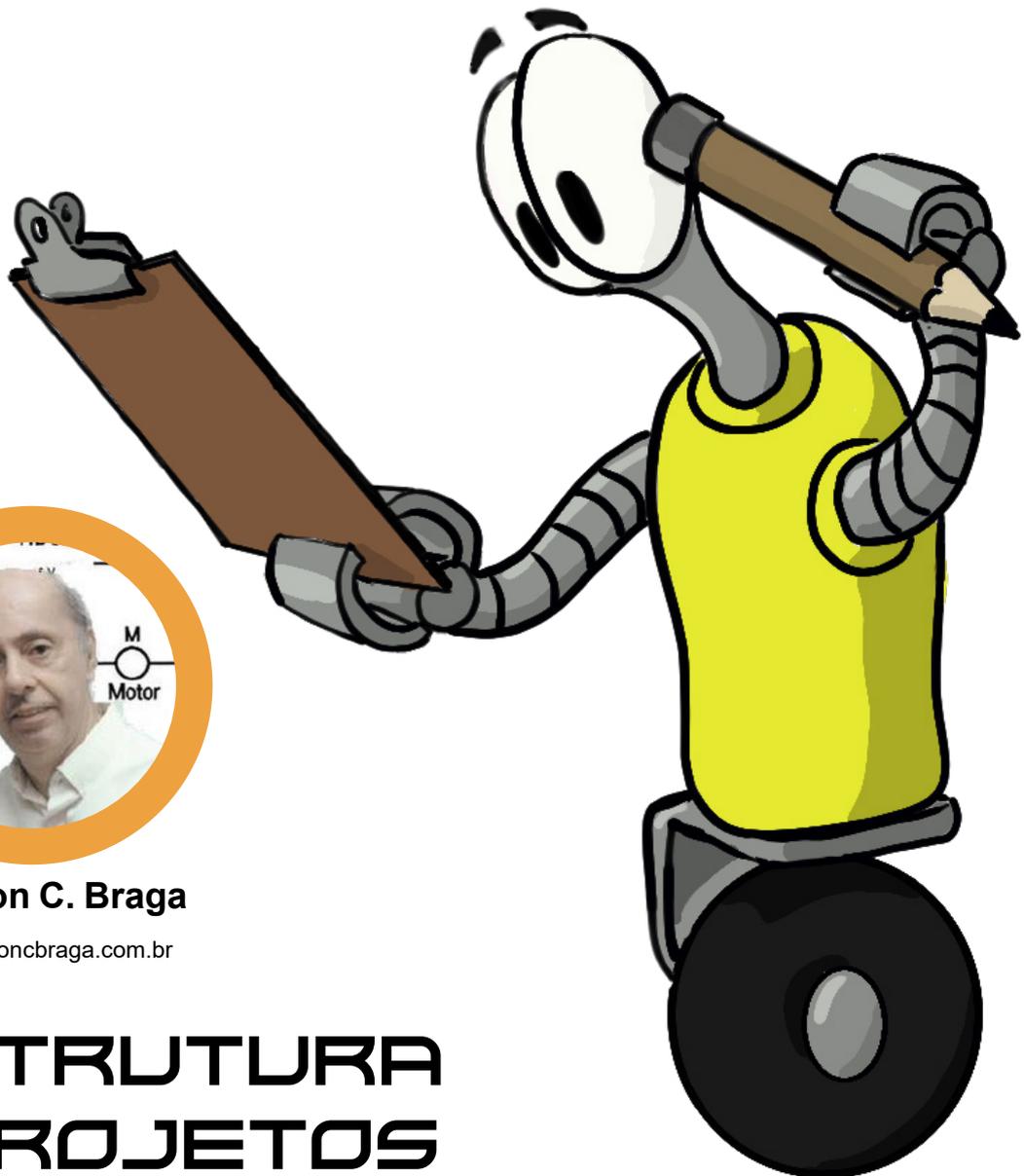
A ESTRUTURA DE PROJETOS DE ROBÓTICA E MECATRÔNICA

Nosso ponto de partida é a ideia de que robôs e dispositivos mecatrônicos são máquinas que usam eletrônica e mecânica em combinação, e que são criados para executar tarefas normalmente realizadas por pessoas. A partir dessa suposição básica, podemos separar as funções necessárias em blocos. A estrutura geral é mostrada na Figura 1.

O número e a seleção de blocos usados em um projeto específico são determinados pelo resultado final previsto pelo projetista. Um braço fixo, por exemplo, ou um elevador automático não precisa de rodas ou pernas. Uma

cabeça com olhos eletrônicos programados para “ver” e detectar coisas não precisa de armas. Os blocos comuns usados em todos os projetos são descritos abaixo.

Este texto foi obtido em uma edição antiga do meu livro Robotics, Mechatronics and Artificial Intelligence publicado nos Estados Unidos em 2002 (esgotado). Os upgrades foram feitos usando novas tecnologias.



CONTROLE

Este é o “cérebro” de qualquer projeto em um sistema robótico ou mecatrônico. Todas as partes elétricas de um robô ou qualquer outro projeto são controladas por circuitos eletrônicos.

Os controles básicos para robôs e projetos mecatrônicos são os seguintes.

a) Controle de posição. Braços com garras ou outros manipuladores de objetos devem ter controles precisos para colocá-los na posição correta. O movimento de uma cabeça com os olhos é controlado por um bloco de controle de um eixo.

b) controle cinemático. Qualquer projeto com partes móveis precisa desse tipo de controle.

A velocidade de qualquer parte móvel deve ser determinada e controlada com precisão pelos circuitos. Um dos controles mais importantes deste grupo rege a velocidade do motor que move um robô

(c) controle dinâmico. Muitas partes de um robô ou projeto mecatrônico incorporam forças que devem ser controladas durante a operação. Quando a pegada de um robô manipula um objeto, é necessário usar algum tipo de controle que determine qual o grau de força necessário para agarrar o objeto com firmeza sem quebrá-lo. Um desafio real para o construtor do projeto é construir um braço robótico que possa pegar um ovo de um cesto e colocá-lo em outro sem quebrá-lo. Tais tarefas exigem controle dinâmico preciso.

d) Controle adaptativo. O controle adaptativo é necessário quando uma função de um robô ou dispositivo mecatrônico deve mudar durante a execução de um processo. Um exemplo é a necessidade de maior força de prensão ao pressionar contra uma mola.

Mais força é necessária quando a mola é comprimida. Outro exemplo de controle adaptativo é a aplicação de mais potência a um motor para manter uma velocidade constante, como quando um robô se move de um terreno plano para um declive ou quando deve mover um objeto pesado.

e) Controle externo. Controles externos são usados quando um operador humano é usado para comandar todas as tarefas de um robô. O humano é o “cérebro” e usa esses sensores (sentidos) como visão no controle das operações do robô.

Para transferir comandos para um robô ou dispositivo mecatrônico, o operador pode usar uma variedade de “interfaces”. As opções básicas são links que empregam rádio, infravermelho (IR), fios e até mesmo comandos de voz. Projetos modernos podem incluir circuitos de reconhecimento de voz para receber pedidos diretamente de

um operador. Um computador também pode ser usado para interligar a unidade de controle e o robô ou dispositivo mecatrônico.

Neste ponto, é importante considerar o grau de inteligência que pode existir em um robô. Controles complexos podem dar ao observador a falsa impressão de que o robô é “inteligente”. No entanto, um bloco de controle que emprega muitas funções não é um bloco inteligente. A inteligência pode ser adicionada se o robô tiver que tomar decisões com base na entrada de seus próprios sensores ou de um operador que emprega um bloco de entrada de dados específico.

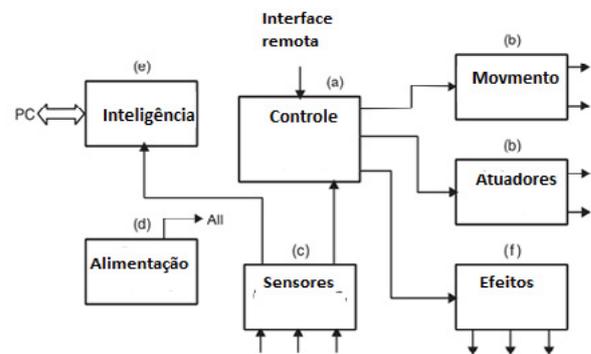


Figura 1 - Organização básica de um projeto em mecatrônica ou robótica.

ATUADORES

Robôs e máquinas mecatrônicas devem ter alguma maneira de manipular objetos ou realizar algum tipo de ação no mundo externo. Como descrito abaixo, muitos tipos de atuadores são encontrados em projetos práticos.

MOVIMENTO

Os robôs podem se mover de um lugar para outro usando pernas, rodas ou trilhas. As pernas podem ser movidas usando motores, solenóides ou ligas com memória de formato * (SMAs).

MANIPULAÇÃO

Robôs e dispositivos mecatrônicos não têm mãos. Eles usam garras para manipular objetos e essas garras são controladas por circuitos eletrônicos. O movimento das garras pode ser acionado com solenóides, motores ou SMAs. A figura 2 mostra alguns tipos de garras.

A manipulação também pode ser efetuada usando equipamentos especialmente projetados para uma tarefa específica, como vemos em muitos robôs industriais. Em muitas aplicações, peças mecanicamente acopladas podem ser adaptadas para se ajustar ao tamanho e à forma de qualquer objeto que precise ser manipulado. Isto é sugerido pela Figura 3.

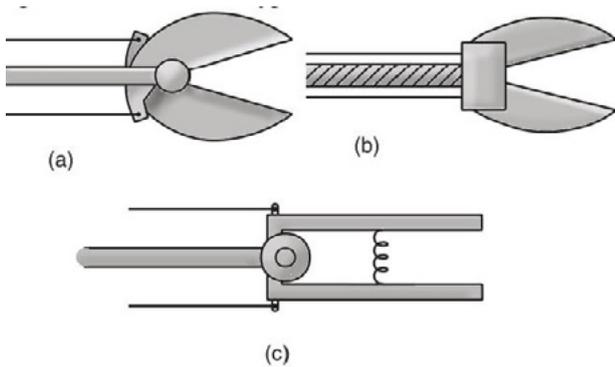


Figura 2 - Vários tipos de garra.

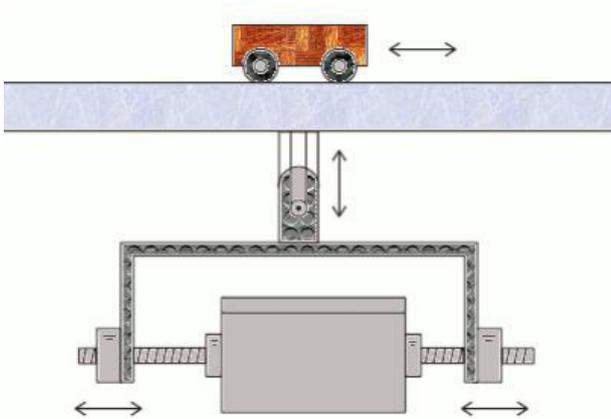


Figura 3 - Manipulador adaptado para o tamanho e forma do objeto

SENSORES

Robôs e dispositivos mecatrônicos detectam o que acontece no mundo real usando sensores. Os sensores são muito importantes, pois podem transmitir informações sobre a posição de um robô ou um braço, o tamanho e a forma de um objeto sendo manipulado, a presença de obstáculos (se for um robô em movimento) e muitas outras informações, incluindo o reconhecimento de objetos por tamanho e forma, como encontrado em alguns projetos “inteligentes”.

Uma câmera de TV pode ser conectada a um circuito inteligente, permitindo assim que um braço automático selecione peças que tenham tamanho e forma específicos dentre muitas peças mistas, como mostrado na figura 4.

Os sensores básicos encontrados em projetos de robótica e mecatrônica são os seguintes:

- Luz: resistores dependentes de luz (LDRs, por exemplo, células CdS ou fotorresistores), fotodiodos, células fotoelétricas e fototransistores
- Pressão: espuma condutora, sensores eletromecânicos, sensores semicondutores

- Temperatura: NTC (**), PTC, diodos ou transistores
- Imagem: dispositivos acoplados a carga (CCDs), fotodiodos ou matrizes fototransistor
- Posição: potenciômetros, sonar, radar, sensores de infravermelho (IR)
- Contato: microswitches, pêndulos
- Proximidade: capacitiva, indutiva ou IR

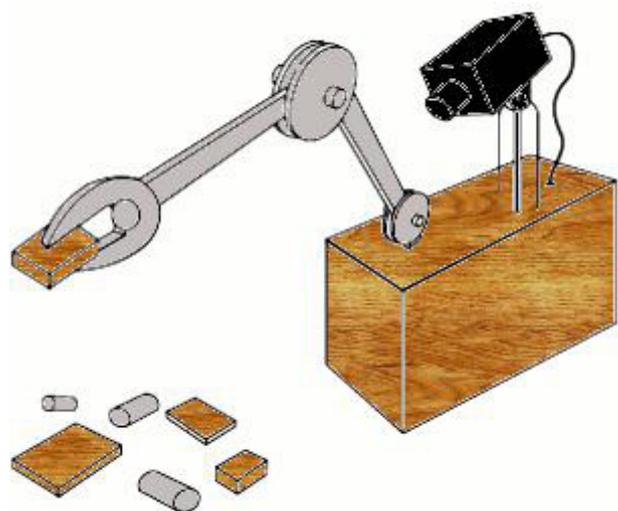


Figura 4 - Câmera de TV usada como sensor de imagem

ALIMENTAÇÃO

Qualquer projeto envolvendo componentes eletrônicos e móveis precisa de algum tipo de fonte de energia elétrica. Se o projeto for um robô que se movimenta, é ideal se você puder colocar a fonte de alimentação dentro dele. Baterias e células podem ser usadas dessa maneira. O tamanho e o tipo de bateria dependem da potência necessária ao robô, de quanto tempo ela deve operar sem recarregar e das tarefas que deve executar.

Se o dispositivo for fixo (por exemplo, um braço robótico) ou relativamente fixo (por exemplo, um elevador), a energia elétrica pode ser fornecida pela linha de energia CA. Uma vez que os circuitos eletrônicos e dispositivos eletromecânicos usados no projeto geralmente usam energia DC, os circuitos que fornecem retificação, filtragem e estabilização de tensão devem ser incluídos se uma fonte de energia CA for usada. Fontes alternativas de energia, incluindo células solares, geradores acoplados a motores de combustão interna, células de combustível e muitos outros, podem ser empregadas em alguns projetos.

INTELIGÊNCIA

Este é um bloco importante que é usado em muitos projetos de mecatrônica e robótica. Inteligência muitas vezes pode ser considerada como um bloco independente.

A inteligência processa informações que são captadas por sensores ou recebidas de outras fontes externas (por exemplo, um computador ou o operador humano) e tomam decisões sobre as tarefas que o sistema deve executar. O bloco de inteligência pode ser tão simples quanto um comparador neural básico que, por exemplo, detecta a luz ambiente e determina a distância do robô a partir da fonte de luz ou de uma parede. Também pode abranger configurações muito mais complexas, envolvendo decisões de alto nível com microcontroladores.

Duas formas de inteligência artificial são adequadas para aplicações em robótica e mecatrônica:

INTELIGÊNCIA DE SOFTWARE

A inteligência de software é fornecida por um computador, microprocessador ou microcontrolador em que qualquer software inteligente é executado. Links de hardware fornecem os dados que o processador precisa para tomar decisões e se comunicar com o bloco de controle.

As decisões são programadas em uma estrutura básica e, em alguns casos, podem ser alteradas de acordo com os dados recebidos. Nesse caso, o programa pode “aprender” com a experiência, que é considerada uma característica básica dos sistemas inteligentes.

Programas que simulam redes neurais são os preferidos por estudantes, pesquisadores e desenvolvedores no campo da inteligência artificial. Outra ferramenta importante para o design de sistemas inteligentes é a lógica fuzzy.

A inteligência de software pode estar localizada dentro do próprio robô ou máquina mecatrônica quando microprocessadores e microcontroladores são usados. O chip Arduino ou PIC* fornece uma maneira simples de adicionar algum grau de inteligência a uma máquina. Pode ser programado para tomar algumas decisões a partir das entradas de sensores de controle externo.

O experimentador pode encontrar muitos programas que simulam redes neurais e lógica difusa. Muitos podem ser usados para adicionar inteligência a computadores, autômatos e outras máquinas.

The image shows the cover of a course titled "CURSO MATEMÁTICA PARA ELETRÔNICA". The cover is primarily blue and white with geometric shapes. It features several mathematical diagrams and formulas: a cylinder with height 'h' and radius 'r' and the formula $V = \pi r^2 h$; the quadratic formula $X = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$; the equation $\frac{x}{a} + \frac{y}{b} = 1$; a right-angled triangle with hypotenuse 'hyp', adjacent side 'adj', and opposite side 'opp', with an angle θ and the formula $\sin(\theta) = \frac{opp}{hyp}$; and a sphere with radius 'r' and the formula $V = \frac{4}{3} \pi r^3$. There is also a small logo of a stylized 'V' in a circle. At the bottom, a blue banner contains the text: "APRENDA A USAR A MELHOR FERRAMENTA PARA DESENVOLVIMENTO E ANÁLISE DE CIRCUITOS ELETRÔNICOS".

PICs e Arduinos são pequenos microcontroladores que executam programas softwares especiais muito fáceis de usar

* As ligas com memória de forma são materiais extraordinários que, por meio de um processo de transformação, retornam a uma forma predeterminada quando aquecidas. Quando um SMA está frio (isto é, abaixo da temperatura de transformação), pode ser deformado facilmente para uma nova forma. Quando o material é aquecido acima de sua temperatura de transformação, ele sofre uma alteração na estrutura cristalina que faz com que ele retorne à sua forma original. Se um SMA encontra resistência a essa transformação, pode gerar forças muito grandes. O material de memória de forma mais comum, chamado Nitinol, é uma liga de níquel e titânio de 50:50.

*NTC = coeficiente de temperatura negativo, PTC = coeficiente de temperatura positivo. Com um dispositivo NTC, a resistência diminui à medida que a temperatura aumenta.

INTELIGÊNCIA POR HARDWARE

Outra maneira de adicionar inteligência a uma máquina é usando circuitos que podem aprender. A ideia básica é imitar a forma como os seres vivos processam as informações que recebem por meio dos sentidos, ou seja, usando o sistema nervoso. A analogia é mostrada na Figura 5.

O ser vivo recebe a informação do mundo externo por sensores acoplados a células nervosas. As células enviam a informação para o cérebro, formada por uma rede de células, onde é processada. O resultado é enviado de volta para os efetores (músculos, por exemplo) por outra rede de células nervosas.

Usando neurônios artificiais, o projetista pode conectar os elementos, formando redes neurais que imitam a operação do sistema de processamento de informações do ser vivo, adicionando algum grau de inteligência a um robô.

É claro que um único neurônio eletrônico não pode produzir o que podemos definir como um comportamento inteligente. É o comportamento de uma ameba. Mesmo usando milhares ou milhões de neurônios, a inteligência que você adicionará ao seu projeto não excederá a de uma minhoca.

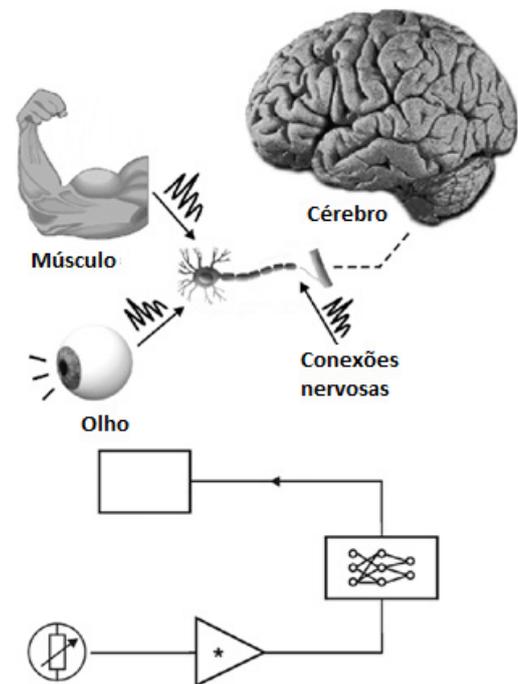


Figura 5 - Adicionando inteligência através de uma rede neural.

Uma rede neural pode aprender e tomar decisões, tornando a máquina um pouco inteligente. As redes neurais na forma de circuitos integrados podem ser compradas de revendedores de componentes, como a Mouser Electronics. Começando com CIs deste tipo, os projetistas podem criar redes complexas de neurônios com recursos como aprendizado, tomada de decisões e muito mais.

EFEITOS

Dependendo da aplicação, o robô ou o dispositivo mecatrônico deve ter algum meio especial de atrair a atenção das pessoas ou até mesmo simular o comportamento humano. Em projetos práticos, podemos considerar a inclusão de blocos que produzem sons, falam palavras ou respondem a perguntas, produzem efeitos de luz ou até mesmo fornecem um mecanismo de autodefesa. Basicamente, os efeitos que podem ser usados nesses projetos envolvendo robótica e mecatrônica são os seguintes:

- Sons: sirenes, síntese de voz, batimentos cardíacos, respiração etc. (O batimento cardíaco e a respiração são usados em algumas bonecas para adicionar um pouco de realismo.)
- Efeitos de luz: pisca-pisca, pisca-pisca, sinalização estroboscópica etc.
- Defesa: alta voltagem

Entregando mais

A mais ampla seleção de semicondutores e componentes eletrônicos em estoque e prontos para envio™



[mouser.com](https://www.mouser.com)





Adobe Stock | #394989787



Raul Júnior

<https://youtube.com/@raulaser>

- Quem vai ganhar time A ou time B?

Essa é a pergunta que assombrou os participantes dessa edição da “Cube Mecatrônica Jovem”.

Mas, outro questionamento que foi levantado:

- De quem vai funcionar time A ou time B?

Se você acompanha o “Clube Mecatrônica Jovem” já deve saber do que estou falando, mas se é a primeira vez que você está em contato com esse material riquíssimo em informações sobre tecnologia... Calma, relaxe, segure na minha mão e vem que vou te levar para se sentar na “janelinha”.

Vem comigo e saiba desde já que o mais importante é: Divertir-se!

AS LIVES DO DESAFIO DE ROBÓTICA

Somos um grupo formado por pessoas apaixonadas por criar coisas de preferência replicáveis que utilizem eletrônica, mecânica, programação e tudo mais que a imaginação permitir, para ficar bonito vou chamar de equipe multidisciplinar.

Na nossa equipe multidisciplinar temos professores, alunos, engenheiros, “makers”, profissionais de diversas áreas da tecnologia.

A partir dessas informações fica mais fácil você entender o porquê de time A e time B.

Fomos divididos assim, em dois grupos. E, alguns mais experientes foram denominados os “Oráculos”.

Os “Oráculos” com a missão orientar as equipes na sua trajetória e escolhas durante as etapas de projeto e execução.

Nosso Chassis

Chassis Comuns



Figura 1 - A estrutura do nosso Robô Areia

Os desafios para as equipes foram divididos nos seguintes temas:

Time A – Desafio do Robô para transporte de pessoas com limitações visuais e de locomoção pelos corredores do Hospital, carinhosamente chamado de Robô Hospital.

Time B – Desafio do Robô para remoção de lixo da AREIA da praia, carinhosamente chamado de Robô AREIA.

Tudo isso com o prazo de dez semanas para terminar as etapas de concepção, montagem, testes e validação do projeto.

Muito tempo não acha?

A construção dos chassis dos robôs foi desafiadora, pois fugimos um pouco do tradicional. Um pouco não... Muito.

Na figura 1 temos exemplos dos chassis tradicionais ou comuns e o nosso Robô Areia, veja como o nosso ficou gigante.

Baseado nesse contexto, nós do “Clube Mecatrônica Jovem” resolvemos dar essa temática para esse módulo do clube.

Digo esse módulo porque nosso clube está dividido em várias frentes e você não pode ficar de fora dele.

Venha participar conosco e gratuitamente.

Outra forma de estar com a gente é na nossa sala no “DISCORD” (“Clube Mecatrônica Jovem”) e criando artigos para essa revista.

Se a dúvida bater aí... Acredite em mim... Vem! Você consegue!

Existem muitas formas para aprender e se desenvolver. Posso citar como exemplo rápido: uma aula teórica bem elaborada, uma aula prática que transmita vivência real, um filme, um vídeo, um livro, uma matéria de revista, uma conversa, uma “live” ... Aproveite para deixar o convite para você acompanhar as “lives” do “Clube Mecatrônica Jovem”, toda quarta-feira às vinte horas no canal “Instituto Newton C Braga” no Youtube e lá você encontrará conteúdo de altíssima qualidade.

Durante as “lives” do Desafio Robótica, fiz alguns experimentos interessantes relacionados com física quântica, bem parecido com o que acontece no filme do “Homem Formiga” da “Marvel”, isso ao vivo e sem edição.

Com meus experimentos no “Mundo Quântico” consegui viver dois momentos únicos.

Do meu laboratório no interior do estado de São Paulo através da reação do primeiro experimento criei um portal quântico que me levou direto para o MoocaLab do professor Luiz Henrique.



Figura 2 – Meu teletransporte para o MoocaLab.

Mesmo assim, percebi que podia continuar avançando nos experimentos e logo comecei a testar a paridade de átomos no “Mundo Quântico”.

Assim, na última “live” dessa série, criei um portal quântico que me levou direto ao escritório do nosso mestre, o professor Newton C. Braga. A energia desprendida dos núcleos dos átomos foi tamanha que interferiu na transmissão da “live”. Veja com seus próprios olhos as fotos que comprovam o experimento na figura 3.

SENSACIONAL!!!!!!!

Não tem como ficar de fora, se programe e vem com a gente toda quarta-feira.

E, para assistir as “lives” que já passaram é fácil, estão todas disponíveis no acervo do canal (“playlists”).

Criador do Projeto Raulaser, que tem como finalidade principal o estímulo do ensino da Robótica Educacional no Brasil.

Como apaixonado por heróis e histórias em quadrinhos, criei juntamente com minha filha, uma família de personagens que nos ajudarão nessa jornada, são eles: Raulaser, LED Day, Juquinha, Gata e o terrível Curto-Circuito.

Acredito que o conhecimento pode transformar realidades. E, acredite em mim, quem optar por uma carreira nas áreas de tecnologia precisará estudar para sempre, pois a evolução é crescente, cada vez mais rápida e é mais que preciso estar atualizado.

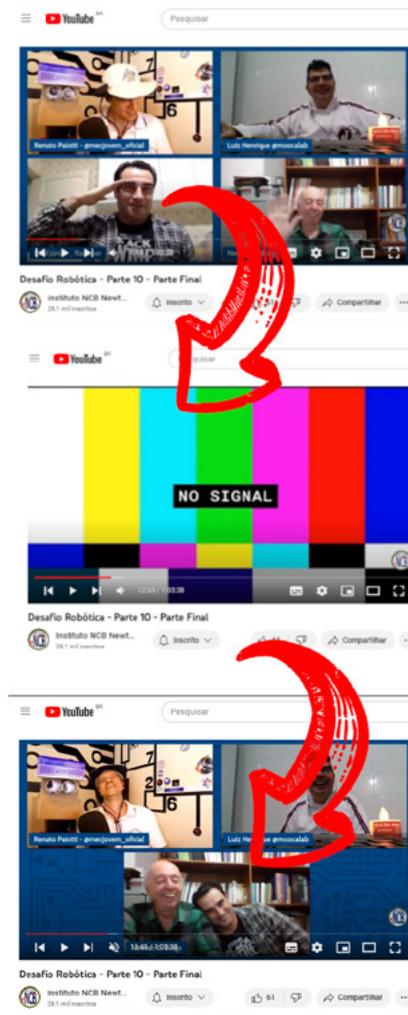


Figura 3 – O dia que me teletransportei para a casa do Prof. Newton C. Braga

Como citei em artigos passados, estamos em processo de recrutamento de heróis para a equipe do Raulaser.

Com a chegada da LED Day e do Juquinha, a equipe ficou ainda mais técnica e multidisciplinar.

Apesar desses sensacionais reforços, a equipe ainda não está completa.

E assim, com muita alegria apresento o novo integrante da equipe Raulaser, seja bem-vindo:

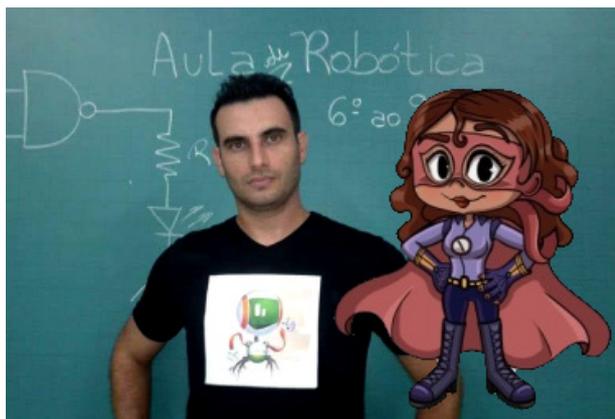


Figura 4 - Eu sou Raul Júnior do Time B...

Espero que tenham curtido e espero você em nossa live ou no meu canal.

No QRCode abaixo você encontrará o meu canal no Youtube, acesse e pense em inscrever-se ... Sua inscrição ajudará demais.

Desejo bons estudos e pode contar comigo!

Fica bem!

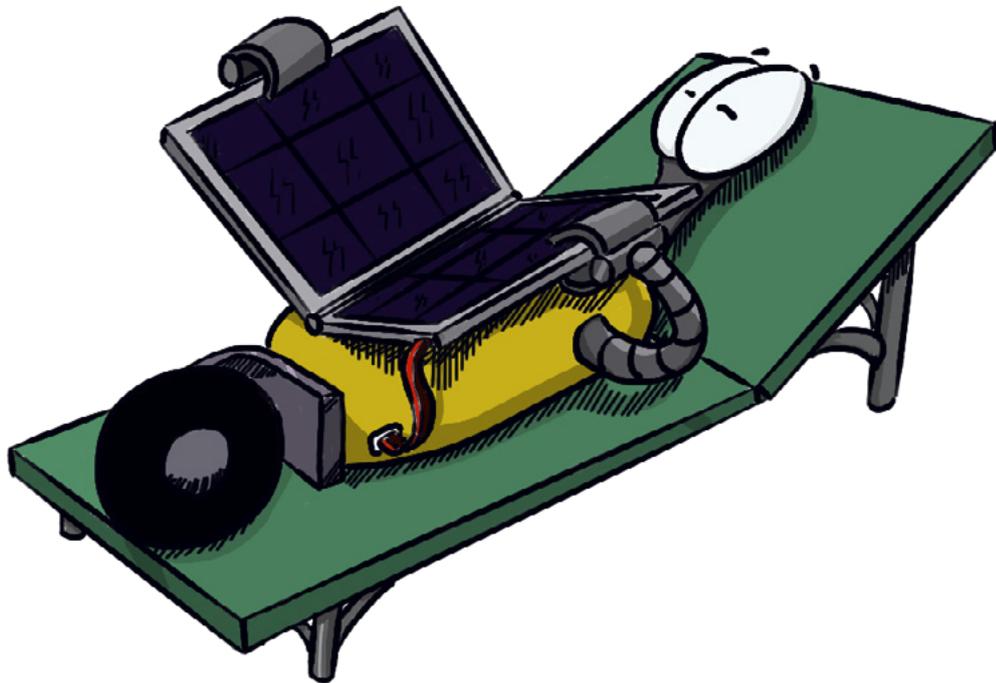


Seu amigo,

Raul Júnior – Raulaser



Figura 5 – O ESTEVÃO, para nós que somos seus amigos podemos chamá-lo de ESTEVINHA.



Raul Júnior

<https://youtube.com/@raulaser>

ENERGIA QUE MOVE MEU ROBÔ

No artigo da edição passada, escrevi o seguinte parágrafo:

“Existem vários simuladores via software para eletrônica e todos são muito úteis, cada um com suas particularidades. Mas, nada substituirá o momento da montagem física onde você irá se deparar com o desafio real de fazer funcionar circuitos. – Raul Júnior.”

Em qualquer montagem de projeto que tenha deslocamento de posição, precisamos pensar em um item extremamente importante para o funcionamento, que é a fonte de energia. Por exemplo, um veículo, aeronave ou em-

barcação, eles precisam de energia para movimentar-se, mas seu movimento não pode ser limitado pela existência de cabos de alimentação. Portanto, é extremamente inevitável pensarmos nas pilhas ou baterias.

“Nossa parou de funcionar... O que será que aconteceu?... Deve ter “acabado” a pilha... #triste.”

- Como pedir mais pilhas para meu pai, se ontem ele tinha me dado as pilhas novas do seu rádio?

As pilhas que usamos tradicionalmente se esgotam de acordo com o consumo do nosso circuito, embora sigam



Figura 1 - Meu carrinho, pilhas e mais pilhas e o radinho do meu pai.

esse processo de esgotamento de acordo com a carga instalada e o tempo que estão em operação, nunca nos conformamos fácil com o fim da sua vida útil.

Geralmente, as pilhas que utilizamos são de 1,5Vcc e podem ser encontradas dos tipos zinco carbono (400 a 1700 mAh) ou as alcalinas (1800 a 2600 mAh). Também existe a versão recarregável que opera em tensão de 1,2Vcc, sendo fabricada com níquel-hidreto metálico (Ni-MH).

As pilhas que se esgotam, infelizmente precisarão ser descartadas, já as recarregáveis é só colocar no carregador e logo estarão disponíveis para uso novamente.

Tudo isso não muda o fato de que quando elas se esgotam ficamos tristes.

Não esqueça que mesmo triste temos que dar destino correto para as pilhas, é nosso dever cuidar do planeta.

Você talvez possa estar se perguntando se usar pilhas é errado ou frustrante para uma montagem.

Em nenhum momento eu escrevi que usar pilhas é errado. Se você utiliza, confesso que eu também e caso esteja pensando em fazer seu primeiro projeto/montagem e pretende usar pilhas, não tem problema nenhum.

Foi que... Em uma manhã fria do mês de maio visitei a loja “Mamute Eletrônica”.

Você precisa viver essa experiência, que lugar incrível. Mas, se você estiver muito longe no momento, não tem crise... Acessa depois que terminar de ler o artigo <https://www.mamuteeletronica.com.br/>, já estou finalizando, só mais dois minutos de leitura.

Na minha visita vi uma plaquinha, bem pequena, passaria despercebida se não fosse meu poder quântico para detectá-la.

Essa placa é muito relevante, a referência dela é TP4056, parece nome dos robôs de “Star Wars”. Seu funcionamento é basicamente carregar baterias de íons de lítio, a bateria geralmente usada é conhecida pelo código 18650 que tem como característica 3,7Vcc, eu usei a de 2600 mAh. O módulo pode trabalhar com outros modelos de bateria também.

Para ficar bonito, vamos chamar de módulo TP4056. Esse módulo pode estar incorporado ao seu robô, bastando apenas colocar o carregador do seu celular para iniciar a carga da bateria.

O meu módulo é USB – V8 e existe a versão usb – tipo C.



Figura 2 - Eu com o Rodrigo e o Rogério da Mamute Eletrônica

E o que achei muito aplicável é o fato de o módulo funcionar como um nobreak para aplicação em circuitos que trabalham alimentados por uma fonte de corrente contínua. Com ausência da alimentação o módulo faz a comutação da bateria no circuito e o mantém em funcionamento.

Essa função não se aplica em um robô em movimento, pois seria necessário o carregador ou fonte externa comutado no módulo TP4056.

Olha só o que dá para fazer com esse módulo. Tornar seu projeto independente por completo usando energia solar para o carregamento da bateria.

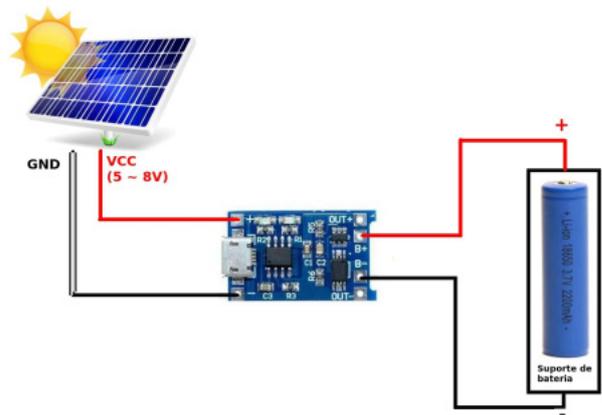


Figura 4 - O módulo acoplado num painel solar.

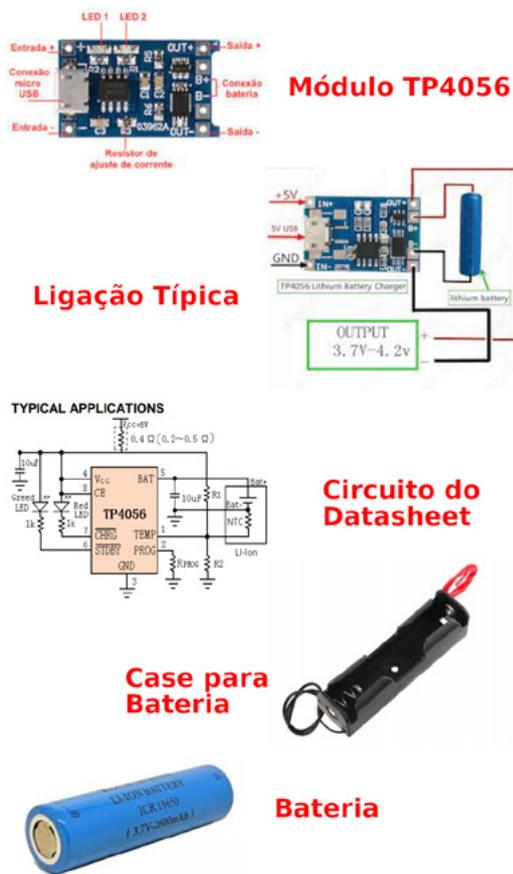


Figura 3 - Etapas de um projeto.

O passo a passo está nesse link, onde o Professor Pedro Bertoleti arrasou no artigo.

<https://www.makerhero.com/blog/aprenda-a-fazer-seu-carregador-solar-de-baterias/>

Sempre destacamos a importância em desenvolver montagens de baixo custo, embasadas tecnicamente e que sejam principalmente divertidas.

Espero que tenha gostado.

Não se esqueça que a "Mamute Eletrônica" tem tudo isso e muito mais.

Vamos no "DISCORD"?

Monta aí e posta lá e me marca...

Compartilhe essa revista com seus amigos e contatos para que ela tenha o maior alcance possível, leve-a aos seus professores e ao diretor ou diretora da sua escola, seja um agente multiplicador do nosso clube.

Se você ainda não tem todas as edições das revistas do "Clube Mecatrônica Jovem", é só acessar o site do "Instituto Newton C Braga" e baixar a coleção completa gratuitamente.

Parabéns!!!!

No QRCode abaixo você encontrará o meu canal no Youtube, acesse e pense em inscrever-se ... Sua inscrição ajudará demais...

Desejo bons estudos e pode contar comigo!

Fica bem!



Seu amigo,

Raul Júnior – Raulaser

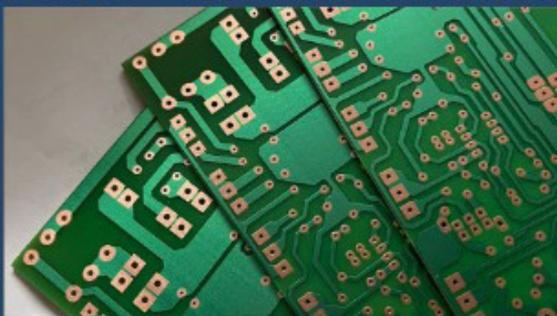
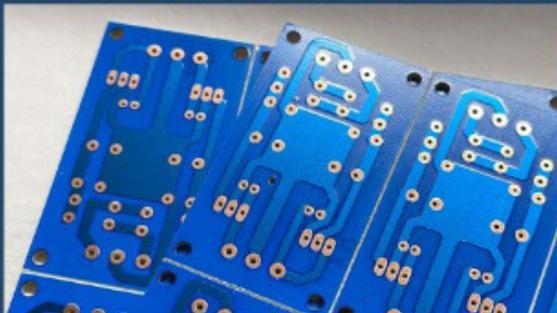
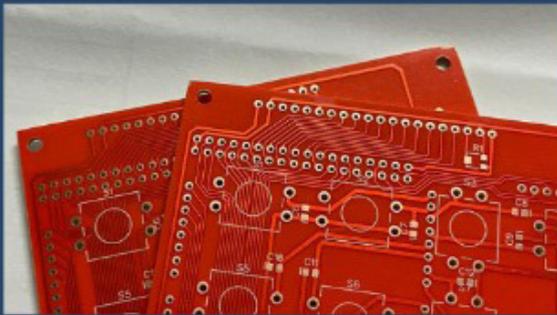


LUCHTEC

A solução ideal para o seu projeto

Confecção de placas de circuito impresso com alta qualidade (PCB/PCI)

A Luchtec traz qualidade nas confecções de circuitos impressos em diversos tamanhos e formatos, preservando o meio ambiente e levando ao cliente produtos com qualidade. Assim se tornando referência nacional em confecção de circuito impresso.



Confecção de Placas de Circuito Impresso

Trabalhamos com materiais de alta qualidade. Aqui na Luchtec prezamos pela qualidade em todos nossos serviços, por isso trabalhamos apenas com materiais de alta qualidade.

- FACE DUPLA
- FACE SIMPLES
- FIBRA
- COMPOSITE
- FENOLITE



Leia o QRCode ao lado e fale conosco diretamente no WhatsApp.

Conheça mais em:
<http://www.luchtec.com.br>



PAFÚNCIO

Olá, Fazedores e Fazedoras de coisas legais, também resolvi criar um robô para recheiar esta edição sobre robótica da Mecatrônica Jovem, e como de costume irei utilizar o papelão como material.

Para quem acompanha as nossas lives, a partir do episódio número 7, apareceu ao meu lado o Pafúncio, um robô que fica na live vendo as trapalhadas da turma e lendo os seus comentários no chat, algumas vezes ele resolve abrir a boca para resmungar algo que não compreendo. Se você gostou dele (ou não) e gostaria de montá-lo, é só seguir as dicas deste artigo e assim montar o seu.



Renato Paiotti

Instagram: @mecjovem_oficial

MATERIAL

Neste robô utilizei papelão, 1 tubo de papelão (daqueles que vem em papel alumínio ou papel toalha), duas bolinhas de isopor, arame, cola, 3 servomotores, uma Raspberry Pi Pico (mas pode ser Arduino) e fios.

A ESTRUTURA

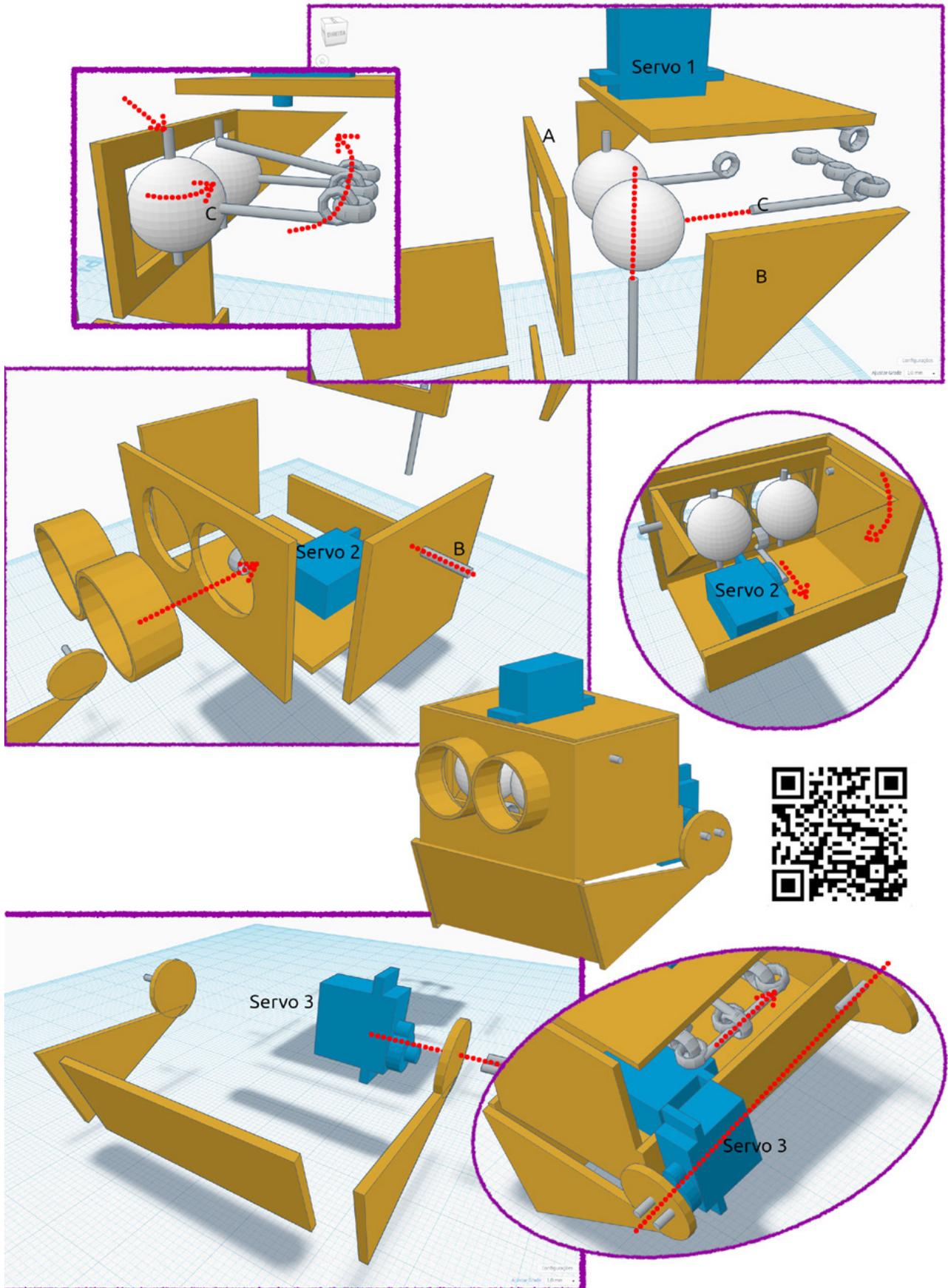


Figura 1

Na figura 1 temos a estrutura do Pafúncio estourada para que você possa ter uma ideia de como montar. Poderia passar as medidas do meu robô, porém não seriam as mesmas que você poderia utilizar, pois o olho pode variar e não encaixar na estrutura que eu fiz. Então aconselho começar a montagem pelos olhos e então desenhar e montar as demais peças, que você poderá customizar. Como disse antes, a montagem é uma sugestão que serve de base para a sua montagem.

Neste momento seria importante você decidir se o seu robô terá ou não movimento nos olhos, ou somente movimentar a boca. É possível trocar todo o sistema do movimento dos olhos com as bolinhas de isopor, por uma animação de olhos piscando e se movendo na tela de um celular, como também um display de LEDs, onde todo o movimento é feito pela programação.

O ponto mais importante aqui são os olhos, pois ele tem um movimento um pouco complexo, que é olhar para todos os lados e copiar o movimento dos olhos humanos. Mas você também poderá utilizar somente um dos movimentos.

Para criar o movimento realista, preendi as duas bolinhas dentro de um quadro (a), mas as bolinhas estão livres para rodar livres neste eixo, o que faz o movimento horizontal dos olhos com o auxílio do servo 1. Este quadro por sua vez faz parte de uma caixa e esta caixa gira preso sobre um eixo (b). Esta caixa que é responsável pelo movimento horizontal dos olhos está presa numa segunda caixa que é a cabeça do Pafúncio, como ela está presa apenas por um eixo, ela poderá girar facilmente, criando o movimento dos olhos na posição vertical com o auxílio do servo 2.

Resumindo: a caixa interna onde o servo 1 está fixo faz o movimento horizontal e a caixa externa onde temos o servo 2 preso na sua base, mexe a caixa interna no movimento vertical dos olhos.

Na caixa externa fiz dois furos redondos para os olhos e para “encobrir” o mecanismo interno, cortei dois pedaços de tubo de papelão para criar uma espécie de óculos e assim dar um ar mais engraçado. No meu robô achei duas lentes de binóculos de brinquedo para fechar os tubos. Lógico que os furos que fiz na caixa externa tem o mesmo diâmetro do tubo.

A boca é mais simples, porém é importante deixar o espaço para ela não enroscar na cabeça. Colei o servo 3 na parte traseira da caixa com uma fita dupla-face de modo que este servo movesse um dos lados da boca, o outro lado fica preso com um outro pedaço de arame,

lembrando que o alinhamento dos dois lados tem que estar perfeito, senão a boca abrirá de forma meio torta.

AJUSTES TÉCNICOS (OK, PODE CHAMAR DE GAMBIARRA)

Como comentei, toda a sua estrutura terá como base as bolinhas de isopor que você conseguir, logo, separe pedaços de papelão para fazer ajustes de largura ou para prender algum tipo de material.

Furar o isopor requer a ajuda de um adulto, pois será necessário pegar um pedaço reto de arame, esquentar e calmamente perfurar as bolinhas de isopor. Cola-quente derrete o isopor e super-colas não grudam em metal (C), então, depois que você furou a bolinha e precisar prender o arame de movimento do olho, aconselho colocar super-cola no buraco e enfiar o arame de movimento em seguida, segurando-o até secar.

Usei um papelão grosso, mas é possível usar papel Paraná, capa de caderno ou livro capa dura ou mesmo MDF, pode ser até impresso em 3D, se você montar um publique nas redes sociais com a #mecatronicajovem que teremos o prazer de mostrar em nossas lives e na próxima edição.

O ajuste do movimento dos olhos é obtido de duas formas: pelo código que abordaremos mais adiante, e com um alicate modelando os arames, onde uma ligeira curvatura faz toda a diferença.

NUNCA MONTEI, MAS QUERO COMEÇAR

Um conselho que dou para quem nunca montou um robô é tentar montar a estrutura, sem se preocupar com a parte elétrica e movimentos do sistema. Feito isso, tentar criar apenas um movimento, onde consiste em pegar esta estrutura, colocar sobre uma base com um pequeno servo, fazendo o movimento da cabeça parecida com o movimento de dizer “não”. Passando esta etapa, tentar fazer o movimento de abrir e fechar a boca, feito isso, criar um segundo movimento, um terceiro e aí caro fazedor ou fazedora, logo montará um robô humanóide.

PARTE ELETRÔNICA

Usei como cérebro para o Pafúncio uma Raspberry Pi Pico e programei seus movimentos em Micropython. Na figura 2 temos um esquema elétrico da montagem onde temos os pinos de controles para os 3 servos, ligados aos pinos 1, 2 e 4, a alimentação no pino 40 e o terra (GND) no pino 38 em nossa matriz de contatos.

Vale lembrar que se você pensa em animar somente a boca, apenas 1 servo já serve, e que um pino de saída aciona o controle deste servo.

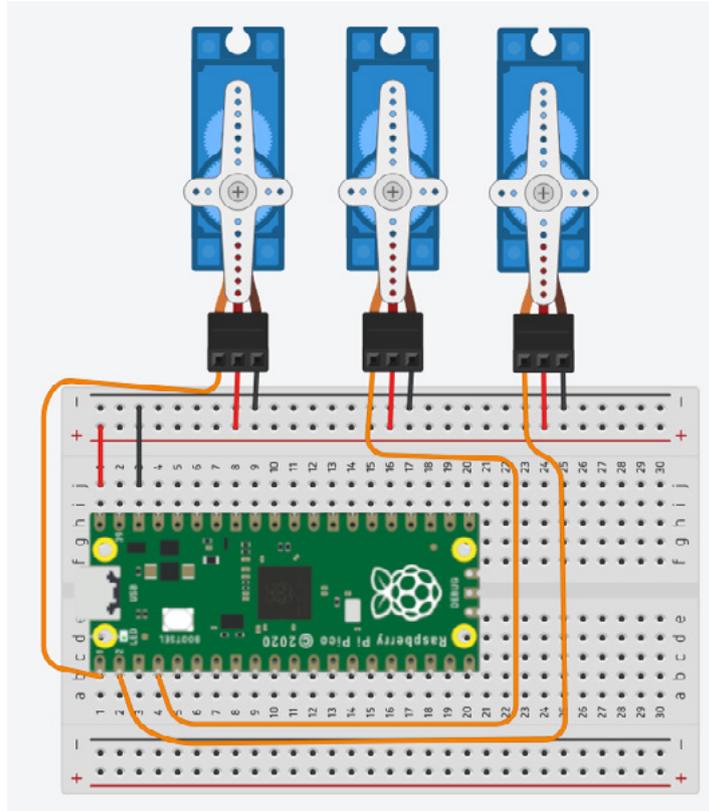


Figura 2

Apoio à Decisão em Manutenção na Gestão de Ativos Físicos

Autor: Rui Assis



O CÓDIGO

Por ser o mais usado para a programação do Raspberry Pi Picco, utilizaremos o Thonny (thonny.org) para programar em Micropython os movimentos do nosso Pafúncio. Não iremos demonstrar aqui como instalar o Thonny, mas no site deles tem muitos exemplos.

No código 1 temos um programa que movimenta todos os 3 servomotores, mas se você está começando com o movimento da boca, apenas o controle de um deles se faz necessário, o servo que está preso à boca e no nosso exemplo é o pino 0, linkado à variável pwm0. No código temos as explicações de cada linha de comando.

CÓDIGO 1

```

-----
from time import sleep # Importando a função tempo
from machine import Pin # Importando as diretrizes dos pinos
from machine import PWM # Importando a função PWM

pwm = PWM(Pin(1)) # Associando à variável pwm o pino 1 da placa
pwm.freq(50) # A frequência que o pino 1 deverá funcionar
pwm.duty_u16(5000) # Posição que o servo da boca deverá começar
pwm0 = PWM(Pin(0)) # Associando à variável pwm o pino 0 da placa
pwm0.freq(50) # A frequência que o pino 0 deverá funcionar
pwm4 = PWM(Pin(4)) # Associando à variável pwm o pino 4 da placa
pwm4.freq(50) # A frequência que o pino 4 deverá funcionar
pwm4.duty_u16(4000) # Posição que o servo 4 deverá começar
def rir (tempo): # Função criada para o Pafúncio rir
    for vezes in range(5400,4800,-50): # Um loop da posição aberta para fechada
        pwm0.duty_u16(vezes) # Move o servo no valor da variável "vezes"
        sleep(tempo) # uma parada definida pela variável "tempo"
    for vezes in range(4800,6000,50): #Agora um loop para abrir
        pwm0.duty_u16(vezes)
        sleep(tempo)
    for vezes in range(6000,4800,-50): # Agora um loop para fechar
        pwm0.duty_u16(vezes)
        sleep(tempo)

def olhar (tempo): # Função para mover os olhos
    for vezes in range(6000,7000,50):
        pwm.duty_u16(vezes)
    sleep(tempo)
    for vezes in range(7000,6000,-50):
        pwm.duty_u16(vezes)
        sleep(tempo)
    for vezes in range(3600,5200,50):

```

```

        pwm4.duty_u16(vezes)
        sleep(tempo)
    for vezes in range(5200,3600,-50):
        pwm4.duty_u16(vezes)
        sleep(tempo)

while True:          # Função principal
    olhar(0.09)      # Chamada da função que mexe os olhos
    sleep(5) # Uma parada
    rir(0.01) # Chamada da função que mexe a boca

```

Note que dentro das funções “olhar” e “rir” é passado um valor dentro dos parênteses, esses valores são passados para dentro das funções através da variável “tempo”. Esta variável ditará o tempo que a função “sleep(tempo)” deverá operar.

Os valores que estão dentro dos loops, são as posições que os servos deverão operar, ou seja, nos meus servos, o máximo da boca fechada era de 4.800 e o boca totalmente aberta o servo deveria estar em 6000. Então é importante você testar os valores em que os seus servos deverão operar. Um conselho importante é testar os servos fora do seu robô e com eles calcular os valores aproximados. Quando eles tiverem colados no robô, é só fazer o ajuste fino dos valores.

No código 2 temos um programa que faz com que o Pafúncio opere de forma autônoma com os olhos, ou seja, de tempos em tempos ele mexe os olhos para uma direção aleatória e só abre a boca quando um botão é acionado. Coloquei um simples botão entre o GND da

placa e o pino 16, e quando ele é acionado, a função rir é chamada e o Pafúncio mexe a boca.

Para que os olhos fiquem se mexendo de forma aleatória, precisamos importar a biblioteca “random”. Com essa função é possível gerar números aleatórios que estejam dentro da faixa de movimento dos servos, e assim fazer com que os olhos se movimentam até aquela posição.

Exemplo: o servo do movimento vertical dos olhos pode se mover entre 6000 e 6800, então peço para a função random gerar um número entre estes dois valores, se ele gerar o número 6.345, o servo irá mover-se até a posição 6.345. Na próxima rodada do programa, ele gerará outro valor dentro desta faixa e movimentará novamente o servo com a nova posição. O mesmo acontece com o servo que faz o movimento horizontal.

Note que dentro de cada função é verificado se o botão foi pressionado, e se ele foi pressionado, a função rir é chamada.

CÓDIGO 2

```

from time import sleep
from machine import Pin
from machine import PWM
import random          # Biblioteca para a geração de números randômicos

pwm = PWM(Pin(1))
pwm.freq(50)
pwm0 = PWM(Pin(0))
pwm0.freq(50)
pwm4 = PWM(Pin(4))
pwm4.freq(50)

```

```

button = Pin(16, Pin.IN, Pin.PULL_UP)    # Atribuindo o pino 16 ao botão

def rir (tempo):
    for x in range(10): # Um loop que gira 10 vezes
        pwm0.duty_u16(2700)
        sleep(tempo)
        pwm0.duty_u16(3200)
        sleep(tempo)

def olhar (tempo):
    pwm.duty_u16(random.randint(6000,6800))    # Gera uma posição randômica entre 6000 e 68000
    pwm4.duty_u16(random.randint(3600,5200))
    sleep(tempo)
    if get_button() == 1:                    # Verifica se o botão foi acionado
        rir(0.2)                            # Se o botão é acionado - chama a função rir

def get_button():                          # Função que confirma o acionamento do botão
    return not button.value()

while True:
    olhar(0.09)
    sleep(3)

```

CONCLUSÃO

É claro que o Pafúncio é único, mas você poderá montar o seu. Este artigo é para demonstrar como é possível com apenas 3 servos, uma placa microcontrolada e papelão é possível criar um robô. Você ou a turma poderão acrescentar voz, luzes e mais servos.

Se você montar um robô com a sua turma da classe, é possível desmembrar as ações do robô, onde cada um fica responsável com algum movimento e assim utilizar cada um à sua placa, e no final unir todas as partes num único robô. É fácil? Sim. Um pouco trabalhoso? Também sim, mas o resultado é muito bacana, e para provar isso aconselho você assistir às nossas lives no canal do INCB e ver o Pafúncio funcionando.

Se você montar o seu Pafúncio, compartilhe nas redes sociais com a #mecatronicajovem que teremos o prazer de mostrar aqui no Clube.



www.ipesi.com.br

VERSÃO IMPRESSA E DIGITAL

REVISTA
IPESI

ELETRÔNICA
& INFORMÁTICA

SUA EMPRESA
PARA O SEU
MERCADO DE
MANEIRA DIRETA

CONSULENTES
ESCOLHIDOS A DEDO.
DIRETORES, GERENTES,
COMPRADORES, TÉCNICOS,
PROJETISTAS ELETRÔNICOS
E PROFISSIONAIS COM
PODER DE DECISÃO





**Prof. Rafael
Oliveira**

Instagram: @prof.tiorafa

ROBÔ ARDUINO BLUETOOTH V2 DO TIO RAFA

Robôs feito com o Arduino estão cada vez mais populares nas instituições de ensino, seja para o aprendizado de automação, mecatrônica, eletrônica ou sistemas embarcados. Este que iremos mostrar hoje por exemplo, foi utilizado no primeiro Campeonato de Robótica realizado no primeiro semestre de 2022 na Escola Técnica Estadual (ETEC), Professora Ermelinda Giannini Teixeira, localizada no Centro Histórico da Cidade de Santana de Parnaíba, para este evento trouxemos para a Etec a modalidade duelo de bexigas onde cada equipe pode acrescentar ao seu robô 2 palitos de churrasco e uma bexiga, os robôs foram controlados por aplicativo instalado em seus smartphones usando a comunicação serial Bluetooth.

Conseguimos, através de pesquisas e testes, encontrar as melhores e mais econômicas peças e então evoluir nosso robô Arduino para a versão 2, onde acrescentamos baterias recarregáveis e usamos o próprio robô para realizar a carga.



Primeiro Campeonato de Robótica Etec Ermelinda -
Santana de Parnaíba



Figura 1 – O Kit.

PRIMEIRO PASSO, A MONTAGEM DO CHASSI

Monte o kit chassi do carrinho de acordo com as instruções que o acompanham. A estrutura básica inclui o chassi, as rodas, eixos, suportes e placas de suporte (figura 2A). Você precisará de uma chave Philips.

Conecte os motores DC que estão nesta caixa de redução, as “caixinhas amarelas” aos suportes do chassi, verifique se as rodas giram suavemente e se os motores estão alinhados corretamente (figura 2B). Caso necessário, ajuste a posição dos motores e a tensão nos parafusos de fixação, para que não fique com folgas, encaixe as rodas amarelas.

Agora parafuse a “roda boba” ao chassi e pronto, seu robô, já tem toda uma estrutura mecânica (figura 2C)

Agora vamos para a parte eletrônica. Primeiro dê um jeito de fixar os fios nos motores, se tiver um ferro de solda e estanho, você pode soldar, caso não possua, descasque um pouco mais e enrole o fio no terminal do motor (figura 2D).

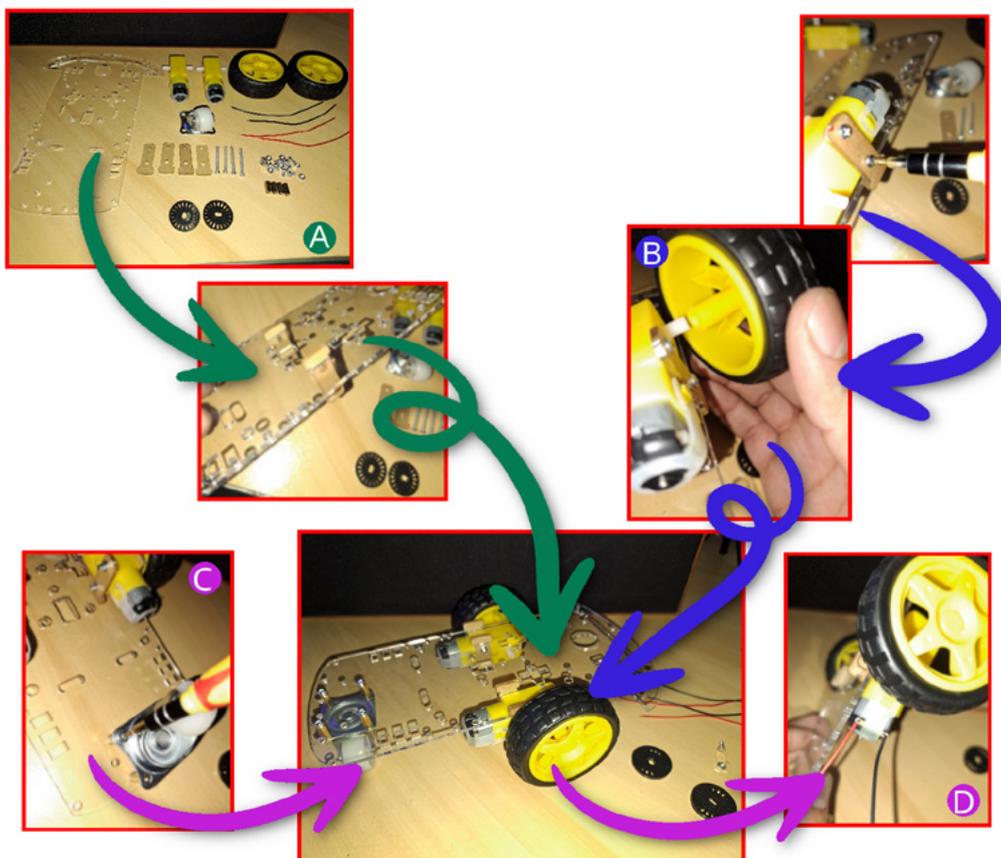


Figura 2 – Montando o chassi e as rodas

SEGUNDO PASSO, CONEXÕES

Conecte esses fios na ponte H L298n. O L298n é um controlador de ponte H, que permite controlar a direção e velocidade dos motores, para isso, conecte os fios dos motores aos terminais correspondentes do L298n, ela possui 2 conectores com 2 parafusos, é aí que vão os 2 fios do motor, um de cada lado (figura 3).

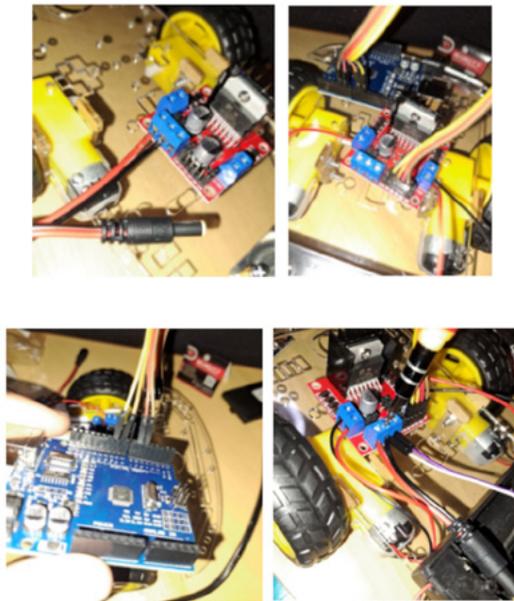


Figura 3 – Montando o Crive do Motor.

Conecte o plug P4 na ponte H L298n, você encontrará um terminal com 3 parafusos, conecte o positivo do plug, o fio vermelho na entrada de voltagem, no parafuso que está mais próximo a lateral da placa, o fio preto, o negativo, conecte no parafuso do meio. Agora vamos conectar ao Arduino. Em seguida, conecte os pinos de controle do L298n aos pinos digitais do Arduino de acordo com o seguinte esquema:

- IN1: pino 5;
- IN2: pino 6;
- IN3: pino 9;
- IN4: pino 10;

Também precisará conectar um jumper do L298n, o 3º parafuso que está no terminal onde conectou o plug P4, ao terminal VIN do Arduino, e outro jumper, junto com o negativo do plug P4 ao GND do Arduino.

TERCEIRO PASSO, MÓDULO BLUETOOTH

Conecte o módulo Bluetooth HC-06 ao Arduino. O módulo Bluetooth permite controlar o robô por meio do smartphone. Conecte os fios do módulo aos seguintes pinos do Arduino:

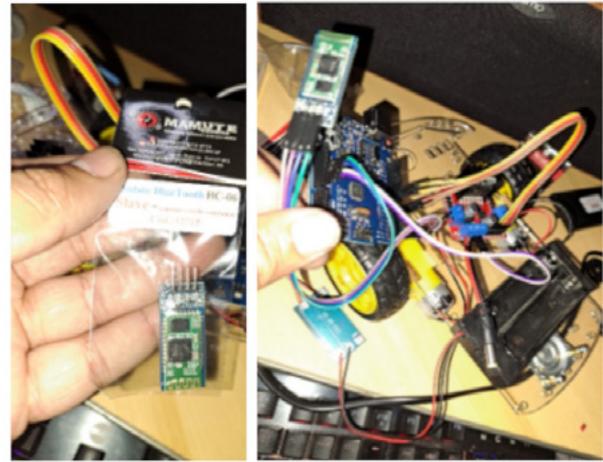


Figura 4 – Instalando o módulo Bluetooth.

- TX do HC-06 para o RX do Arduino (pino 0);
- RX do HC-06 para o TX do Arduino (pino 1);
- VCC do HC-06 para o 5V do Arduino;
- GND do HC-06 para o GND do Arduino.

O módulo Bluetooth possui uma senha que pode ser facilmente alterada através de uma programação adequada, mas que como padrão é 1234, você utilizará esta senha no seu smartphone quando for parear com o robô.

QUARTO PASSO, ALIMENTAÇÃO

Tudo montado e conectado, falta apenas a parte da alimentação, que começaremos com nossa placa de controle de carga 18650 BMS, fixaremos nela o Jack P4 ou P4 fêmea no lado placa indicado por B+ e B-, sendo o positivo, o fio vermelho no B+ e o fio preto, o negativo no B-, do outro lado, indicado pela letra P+ e P-, iremos fixar os fios do suporte de bateria, seguindo a mesma lógica, o fio vermelho é o positivo, que deve ser fixado no P+ e o fio preto é o negativo no P-.

Agora é só fixar o suporte de baterias ao chassi e toda montagem está completa, quando for utilizar seu robô, junte os plug P4 macho e fêmea, quando acabar a carga da bateria, desconecte os plugs P4 e conecte o plug P4

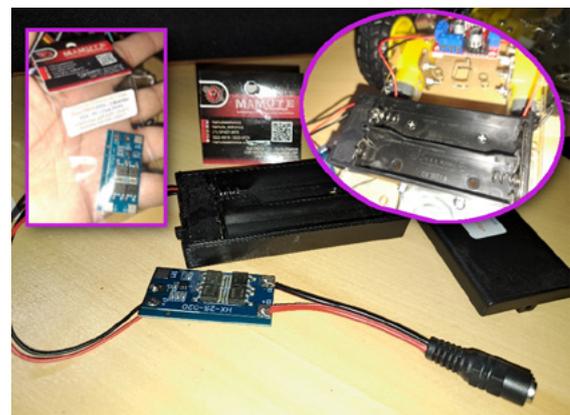


Figura 5 – A parte da alimentação de energia.

macho da fonte de alimentação que acompanha o kit, no plug P4 fêmea, assim irá realizar a carga das baterias.

Você pode organizar os fios e prender as placas no chassi, usando arames encapados que vem em pacotes de pão entre outros, ou ainda elásticos, dupla face, durex, fita isolante etc. Afinal você é maker, certo?

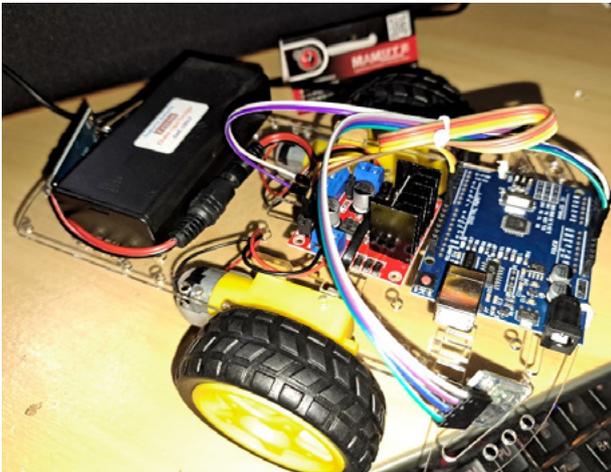


Figura 6 – Organizando toda a fiação.

Essa montagem ficou assim, mas você pode mudar tudo de lugar em cima do chassi e deixar o robô com a sua cara, ou ainda criar uma carenagem e cobri tudo isso.

QUINTO PASSO, PROGRAMAÇÃO

Para que tudo funcione corretamente, você deve carregar o código no seu Arduino, e baixar o aplicativo RC Car no seu Smartphone, mas antes instale o drive CH340 caso seu Sistema operacional seja Windows, você encontra esse drive neste link:

https://drive.google.com/file/d/1uep_2XcInLIY8_j_tbil4kdoYx7p8VdD/view

Para Instalação do ambiente de desenvolvimento integrado (IDE) do Arduino, acesse: <http://www.arduino.cc/> e depois de instalado, na Aba Software, escolha a versão de acordo com seu sistema operacional, por fim, conecte seu Arduino em porta Usb do Computador, na IDE do Arduino, vá em ferramentas e verifique o nome da placa e porta, e então carregue o código no Seu Arduino (figura 7).



Figura 7 – Instalando o IDE do Arduino.

Para enviar o código, o Arduino precisa estar conectado e confirmado a porta em que ele está configurado, como mostrado na figura 8.

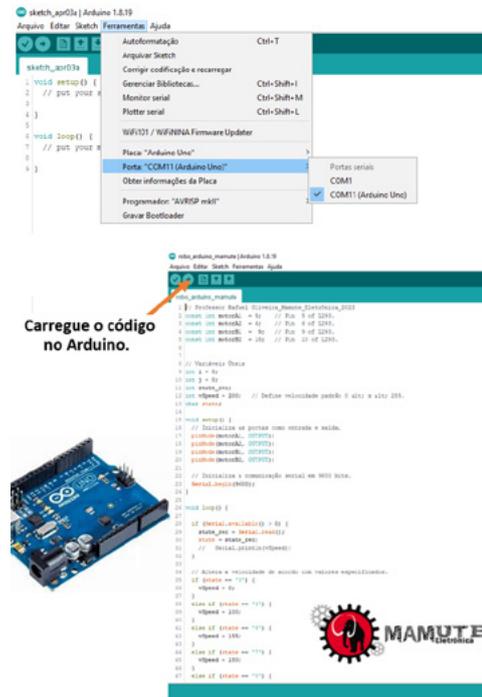


Figura 8 – Carregando o código para o Arduino.

No seu Smartphone, vá até a loja de softwares e baixe o APP RC CAR <https://play.google.com/store/apps/details?id=braulio.calle.bluetoothRCcontroller>

Para configurar o seu controle é só seguir os passos das figura 9A e 9B, Configurar e Conectar.

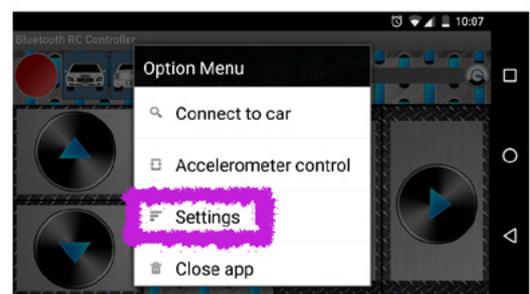
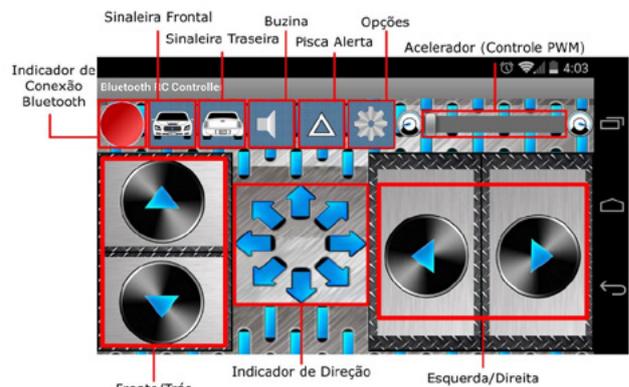


Figura 9A – Configurando o controle.

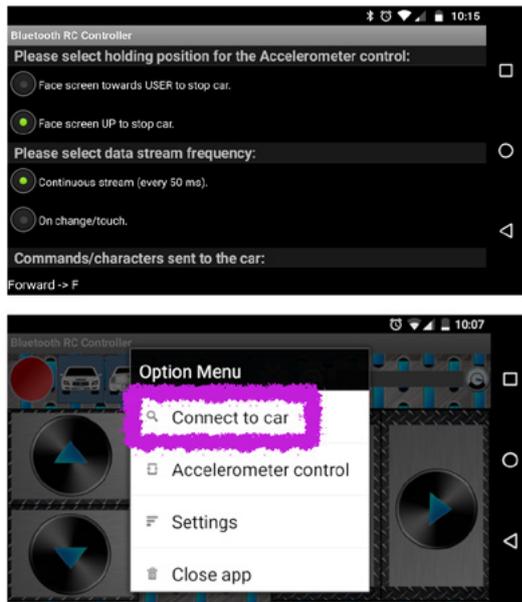


Figura 9B – Configurando o controle.

Lembre-se, acabou a bateria, é só conectar o carregador, como mostrado na figura 10.

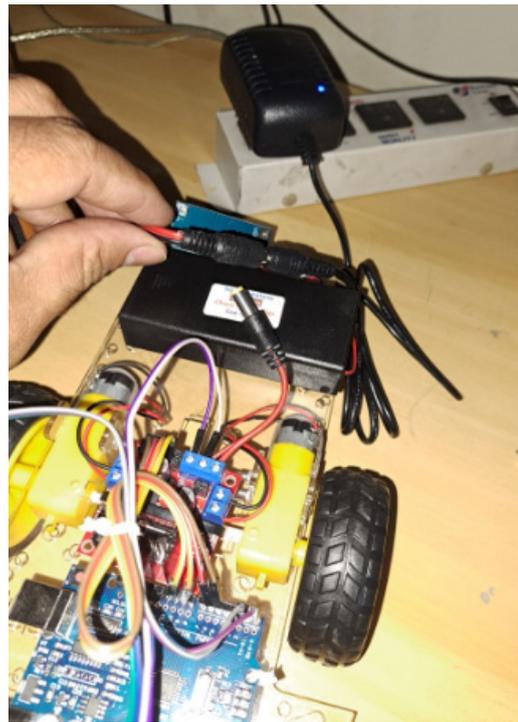


Figura 10 – Recarregando a bateria do nosso robô.



Professor Rafael Oliveira ou Tio Rafa

Mestrando em Informática e Gestão do Conhecimento (PPGI) na Universidade Nove de Julho, (Uninove), Especialista em Tecnologias para Indústria 4.0, Licenciado em Tecnologia da Informação, Pós-graduado em Educação de Jovens e Adultos (EJA). Graduação em Gestão da Tecnologia da Informação. Atualmente, Professor de Ensino Médio e Técnico no Centro Estadual de Educação Tecnológica Paula Souza, a mais de 12 anos, com experiência na área de Educação, como Gestor, Coordenador e Docente, com ênfase em Ensino Aprendizagem, atuando principalmente nos temas mais atuais como Robótica Educacional, Sistemas Embarcados com Arduino/ESP32/ Raspberry e Internet das Coisas (IoT). Pesquisador do Grupo de Pesquisa em Educação, Tecnologias e Cultura Digital (GRUPETeC), Membro da equipe de Robótica do Centro Estadual de Educação Tecnológica Paula Souza, responsável por implantação da robótica educacional, torneios de robótica e capacitações para professores. Desenvolvedor de kits de robótica educacional em parceria com a empresa Mamute Eletrônica.

FAÇA JÁ O
DOWNLOAD
DAS SUAS
REVISTAS
GRATUITAMENTE

INCBELETRÔNICA

Uma revista bimestral com artigos e projetos de eletrônica especialmente selecionados para você aprender e ficar por dentro dos novos projetos.





ROBÔ CONTROLADO VIA INTERNET COM MÓDULO WIFI LORA 32



Pedro Bertoleti

Linkedin: Pedro Bertoleti

Conforme demonstra este nosso artigo sobre o módulo WiFi LoRa 32(V2) (<https://newtoncbraga.com.br/index.php/microcontrolador/143-tecnologia/16326-moduloesp-32-heltech-mec218>), ela consiste numa placa completa para quem quer realizar projetos completos inseridos no contexto de IoT (Internet of Things - Internet das Coisas). Indo desde interação com sensores, controle de periféricos diversos até comunicação wi-fi e LoRa, o módulo WiFi LoRa 32(V2) atende perfeitamente às necessidades da maioria dos projetos, sendo uma solução compacta e poderosa em termos de recursos de comunicação e poder computacional.

Dando continuidade aos usos deste módulo, vamos utilizá-lo para criar um robô! Tal robô é do tipo rover, ou

seja, um robô do tipo explorador, inspirado principalmente nos robôs de exploração espacial que utilizam rodas para locomoção. Tal robô será controlado pela Internet, permitindo que o controle seja feito de qualquer lugar do planeta que possua conectividade à Internet. Como forma de explorar o maior número de possibilidades deste projeto, serão mostradas duas formas de fazer o acionamento e controle dos motores do rover: acionamento direto com relés e com um módulo amplificador/driver e ponte-H. Os motores deste robô serão pequenos motores DC comumente encontrados no mercado maker.

Ao fim deste artigo, o leitor será capaz de montar seu próprio robô, podendo escolher o acionamento e controle do motor DC mais adequado ao seu uso e controlá-lo de qualquer lugar do planeta via Internet.

Material necessário

Para reproduzir o projeto deste artigo, você precisará de:

- Um módulo WiFi LoRa 32(V2)
- Uma fonte / carregador de celular (conector micro-USB) do tipo powerbank - tensão de 5V, com pelo menos 2A de corrente de saída
- Um chassi de robô com duas rodas (<https://www.filipeflop.com/produto/kit-chassi-2wd-robo-para-arduino/>) . Chassis deste tipo já são vendidos com motores DC (de 3V até 6V de tensão de operação, já com caixa de redução acoplada)
- Um cabo micro-USB (para programação do WiFi LoRa 32(V2))
- Dois transistores BC548 (ou equivalente NPN, como o BC337, por exemplo)
- Dois resistores 1K/0,25W
- Dois resistores 2K2/0,25W
- Módulo relé / Shield relé de 2 canais
- Jumpers macho-macho
- Um driver de motor DC (com Ponte H) usando o circuito integrado L298N. Tal driver é encontrado no mercado maker com o nome módulo L298N

Primeira forma de controle e acionamento de motores DC: via relés

Iniciaremos com a forma mais simples e intuitiva de controle e acionamento de motores DC: acionamento via relés ou controle on/off. Esta forma é a mais simples de controle e acionamento de motores DC existente, pois funciona da mesma forma de que se ligássemos uma “chave” para acionar o motor. Ou seja, trata-se de um controle do tipo on/off, puro e simples, utilizando como “chave” um relé.

Neste tipo de acionamento e controle de motores DC, no quesito de dimensionamento dos relés, devemos nos atentar não somente à tensão suportada na carga, mas também quanto à corrente de operação máxima suportada. Esta não deve ser menor que a corrente máxima (de pico) consumida pelo motor. Para o caso dos motores usados nesse projeto, a corrente de pico (na partida, por pouquíssimos milissegundos) atinge algo em torno de 1,2A, portanto qualquer relé capaz de suportar 1,2A ou mais de corrente elétrica e com tensão da carga igual ou superior a 6V pode ser utilizado.

Como esta forma de acionamento e controle de motor se resume basicamente em alimentar (on) e tirar a alimentação (off) do motor, a movimentação do rover é definida e restrita conforme a seguir:

Ir para frente: acionamento dos dois motores de forma simultânea (figura 1.a)

Girar para direita: acionamento do motor da esquerda somente (figura 1.b), de forma que o rover gire sobre seu próprio eixo.

Girar para esquerda: acionamento do motor da direita somente (figura 1.c), de forma que o rover gire sobre seu próprio eixo.

Parar: desacionamento de ambos os motores.

Ré: a forma de acionamento e controle simples on/off não contempla controle de sentido de rotação dos motores. Logo, não é possível mover o robô diretamente para trás. Logo, como estratégia para ir para trás, devemos mover o robô para a direita ou esquerda até que este dire 180°.

Observe a figura 1, onde é possível vermos a movimentação possível de ser feita com este tipo de controle e acionamento dos motores DC.

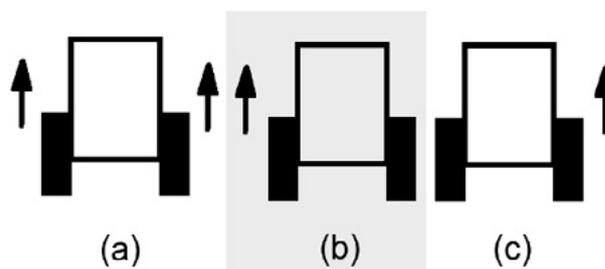


Figura 1 - Movimentação do rover

Circuito esquemático

O circuito esquemático do rover com controle e acionamento de motores com relés é mostrado na figura 2.

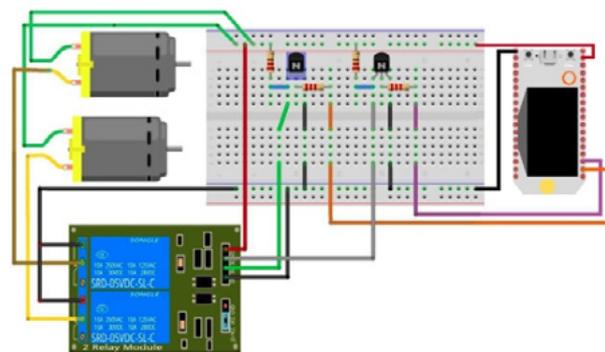


Figura 2 - circuito esquemático do projeto (robô do tipo rover)

Porém, as seguintes observações são pertinentes:

Para a alimentação dos motores é necessário utilizar um carregador de baterias de celular portátil (conhecido no mercado como powerbank).

O motivo disso é que os motores, quando acionados, consomem muita energia, levando ao rápido consumo de baterias. Portanto, a utilização de carregadores portáteis de celular minimizará custos do projeto (e torná-lo recarregável). Tais carregadores portáteis possuem capacidade de fornecimento de corrente mais do que suficientes para o referido projeto.

Aqui, o objetivo do controle e acionamento dos motores com relés foi a simplificação do circuito.

Você notará que foram utilizados transistores para acionar o Shield de Relés a partir do WiFi LoRa 32(V2). Isso foi necessário, pois o Shield de Relés necessita de uma tensão de acionamento maior que a tensão de saída de um GPIO do módulo WiFi LoRa 32(V2) (o módulo WiFi LoRa 32(V2) utiliza 3,3V, enquanto o Shield de Relés necessita de 5V). Como não é uma boa ideia acioná-lo com tensão menor que a especificada, os transistores serão responsáveis por “converter” os níveis de tensão para aqueles dentro do especificado.

Controle da movimentação via Internet (com MQTT)

Agora que já definimos o circuito esquemático e movimentação do rover, falta somente uma definição antes de partirmos para o código-fonte do rover (versão do controle e acionamento de motores via relés): definir como controlar o robô através da Internet (via MQTT). Se você não está familiarizado com MQTT no módulo WiFi LoRa 32(V2), recomendo que leia o artigo de Monitoramento de Temperatura Ambiente e Umidade Relativa do Ar via Internet (<https://newtoncbraga.com.br/index.php/microcontrolador/143-tecnologia/14409-monitoramento-de-temperatura-ambiente-e-umidade-relativa-do-ar-via-internet>).

Para o controle via Internet do rover, primeiramente precisamos definir 3 caracteres / letras que, enviadas ao rover via MQTT, irão fazê-lo ir para frente, direita ou esquerda. Além disso, é necessário também um comando para fazê-lo parar (desligar os dois motores). Sendo assim, vamos definir estes caracteres / letras conforme o seguinte:

- Caractere que, quando recebido, move o rover para frente: F (figura 3.a)
- Caractere que, quando recebido, move o rover para direita: D (figura 3.b)
- Caractere que, quando recebido, move o rover para esquerda: E (figura 3.c)
- Caractere que, quando recebido, faz o rover parar: P

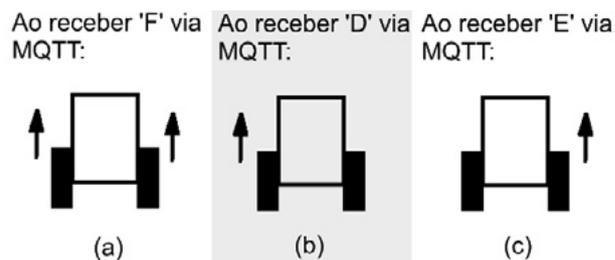


Figura 3 - movimentação do rover mediante comando recebido via MQTT

Observe a figura 3 para mais detalhes.

Além disso, via MQTT, é possível também monitorar o robô (monitorar sua movimentação). Para isso, o robô envia 3 caracteres em sequência via MQTT, conforme abaixo:

Estado do motor direito: caractere '1' para acionado ou caractere '0' para desacionado

Separador: caractere '-'

Estado do motor esquerdo: caractere '1' para acionado ou caractere '0' para desacionado

Código-fonte do rover

É chegada a hora de escrever o código-fonte do projeto.

IMPORTANTE 1: leia atentamente os comentários do mesmo para completa compreensão de seu funcionamento.

IMPORTANTE 2: instale a biblioteca relativa ao MQTT informada no artigo Monitoramento de Temperatura Ambiente e Umidade Relativa do Ar via Internet (<https://newtoncbraga.com.br/index.php/microcontrolador/143-tecnologia/14409-monitoramento-de-temperatura-ambiente-e-umidade-relativa-do-ar-via-internet>)

Código-fonte do Rover

/ Programa: código-fonte do robô tipo rover (controle e acionamento de motores com relés Descrição: código-fonte do robô tipo rover, controlado por Internet via MQTT. Sua movimentação é feita conforme a recepção de caracteres via MQTT: F: rover se move para frente D: rover se move para a direita E: rover se move para a esquerda P: o rover para (os motores são desacionados) Autor: Pedro Bertoleti*/*

```
/* includes */

#include <WiFi.h>
#include <PubSubClient.h> /* defines gerais */
#define BAUDRATE_SERIAL_DEBUG 115200 /* defines de id mqtt e tópicos para publicação e subscribe */
#define TOPICO_SUB "MQTTRoverEnvia" /* tópico MQTT de escuta */
#define TOPICO_PUB "MQTTRoverRecebe" /* tópico MQTT de envio de informações para Broker */
#define ID_MQTT "RoboRoverINCB" /* id mqtt (para identificação de sessão) */

/* IMPORTANTE: este deve ser único no broker (ou seja, se um client MQTT tentar entrar com o mesmo id de outro já conectado ao broker, o broker irá fechar a conexão de um deles). */

/* defines - motores */
#define MOTOR_DIRETO 12 /* GPIO 12 */
#define MOTOR_ESQUERDO 13 /* GPIO 13 */
/* defines - movimentacao */
#define CHARACTER_MOVE_FRENTE "F"
#define CHARACTER_MOVE_DIREITA "D"
#define CHARACTER_MOVE_ESQUERDA "E"
#define CHARACTER_PARA_ROVER "P"

/* Variáveis e objetos globais: */
/* WIFI */
const char* SSID_WIFI = "";
/* Coloque aqui o SSID_WIFI / nome da rede WI-FI que deseja se conectar */
const char* PASSWORD_WIFI = ""; /* Coloque aqui a senha da rede WI-FI que deseja se conectar */
/* MQTT */
const char* BROKER_MQTT = "iot.eclipse.org"; /* URL do broker MQTT que se deseja utilizar */
int BROKER_PORT = 1883; /* Porta do Broker MQTT */
/* Variáveis e objetos globais */
WiFiClient espClient; /* cria o objeto espClient */
PubSubClient MQTT(espClient); /* objeto PubSubClient */
char estado_motor_direito = '0'; /* estado atual do motor da direita */
char estado_motor_esquerdo = '0'; /* estado atual do motor da esquerda */
/* Prototypes */
```

```

void init_serial_debug(void);
void init_wifi(void);
void init_MQTT(void);
void connect_wifi(void);
void init_outputs(void);
void mqtt_callback(char* topic, byte* payload, unsigned int length);
void verifica_conexoes_wifi_e_MQTT(void);
void envia_estado_output_MQTT(void);
/* * Implementações das funções */
void setup()
{
    //Faz as inicializações necessárias
    init_outputs();
    init_serial_debug();
    init_wifi();
    init_MQTT();
}

/* Função: inicializa comunicação serial com baudrate 115200 (para fins de monitorar no terminal serial o que está
acontecendo. Parâmetros: nenhum Retorno: nenhum*/
void init_serial_debug(void)
{
    Serial.begin(BAUDRATE_SERIAL_DEBUG);
}/* Função: inicializa e conecta-se na rede WI-FI desejada Parâmetros: nenhum Retorno: nenhum*/
void init_wifi(void)
{
    delay(10);
    Serial.println("-----Conexao WI-FI -----");
    Serial.print("Conectando-se na rede: ");
    Serial.println(SSID_WIFI);
    Serial.println("Aguarde");

    connect_wifi();
}

/* Função: inicializa parâmetros de conexão MQTT(endereço do broker, porta e seta função de callback) Parâ-
metros: nenhum Retorno: nenhum*/
void init_MQTT(void)
{
    MQTT.setServer(BROKER_MQTT, BROKER_PORT); //informa qual broker e porta deve ser conectado
    MQTT.setCallback(mqtt_callback); //atribui função de callback (função chamada quando qualquer informação
de um dos tópicos subscritos chega)

```

```

}

/* Função: função de callback esta função é chamada toda vez que uma informação de um dos tópicos subscritos
chega) Parâmetros: nenhum Retorno: nenhum*/
void mqtt_callback(char* topic, byte* payload, unsigned int length)
{
    String msg;
    int i;
    /* obtem a string do payload recebido */
    for(i = 0; i < length; i++)
    {
        char c = (char)payload[i];
        msg += c;
    }

    /* toma ação dependendo da string recebida: */
    if (msg.equals(CARACTER_MOVE_FRENTE))
    {
        //para ir para frente, os dois motores são ligados
        digitalWrite(MOTOR_DIRETO, HIGH);
        digitalWrite(MOTOR_ESQUERDO, HIGH);

        estado_motor_direito = '1';
        estado_motor_esquerdo = '1';
    }
    if (msg.equals(CARACTER_MOVE_DIREITA))
    {
        /* para ir para a direita, somente o motor da esquerda é ligado */
        digitalWrite(MOTOR_DIRETO, LOW);
        digitalWrite(MOTOR_ESQUERDO, HIGH);

        estado_motor_direito = '0';
        estado_motor_esquerdo = '1';
    }
    if (msg.equals(CARACTER_MOVE_ESQUERDA))
    {
        /* para ir para a esquerda, somente o motor da direita é ligado */
        digitalWrite(MOTOR_DIRETO, HIGH);
        digitalWrite(MOTOR_ESQUERDO, LOW);
    }
}

```

```

    estado_motor_direito = '1';
    estado_motor_esquerdo = '0';
}
if (msg.equals(CARACTER_PARA_ROVER))
{
    /* para parar, os dois motores são desligados */
    digitalWrite(MOTOR_DIRETO, LOW);
    digitalWrite(MOTOR_ESQUERDO, LOW);

    estado_motor_direito = '0';
    estado_motor_esquerdo = '0';
}
}

/* Função: reconecta-se ao broker MQTT (caso ainda não esteja conectado ou em caso de a conexão cair) em
caso de sucesso na conexão ou reconexão, o subscribe dos tópicos é refeito. Parâmetros: nenhum Retorno: nenhum*/
void reconnectMQTT()
{
    while (!MQTT.connected())
    {
        Serial.print("** Tentando se conectar ao Broker MQTT: ");
        Serial.println(BROKER_MQTT);
        if (MQTT.connect(ID_MQTT))
        {
            Serial.println("Conectado com sucesso ao broker MQTT!");
            MQTT.subscribe(TOPICO_SUB);
        }
        else
        {
            Serial.println("Falha ao reconectar no broker.");
            Serial.println("Haverá nova tentativa de conexão em 2s");
            delay(2000);
        }
    }
}

/* Função: reconecta-se ao WiFi Parâmetros: nenhum Retorno: nenhum*/
void connect_wifi(void)
{
    /* se já está conectado a rede WI-FI, nada é feito. Caso contrário, são efetuadas tentativas de conexão */
    if (WiFi.status() == WL_CONNECTED)

```

```

return;

WiFi.begin(SSID_WIFI, PASSWORD_WIFI); // Conecta na rede WI-FI

while (WiFi.status() != WL_CONNECTED)
{
    delay(100);
    Serial.print(".");
}

Serial.println();
Serial.print("Conectado com sucesso na rede ");
Serial.print(SSID_WIFI);
Serial.println("IP obtido: ");
Serial.println(WiFi.localIP());

/* Função: verifica o estado das conexões WiFi e ao broker MQTT. Em caso de desconexão (qualquer uma das
duas), a conexão é refeita. Parâmetros: nenhum Retorno: nenhum*/
void verifica_conexoes_wifi_e_MQTT(void)
{
    if (!MQTT.connected())
        reconnectMQTT(); //se não há conexão com o Broker, a conexão é refeita

    connect_wifi(); //se não há conexão com o WiFi, a conexão é refeita
}

/* Função: envia ao Broker o estado atual do output Parâmetros: nenhum Retorno: nenhum*/
void envia_estado_output_MQTT(void)
{
    char estados_motores[4];
    char separador = '-';
    estados_motores[0] = estado_motor_direito;
    estados_motores[1] = separador;
    estados_motores[2] = estado_motor_esquerdo;
    estados_motores[3] = '\0';
    MQTT.publish(TOPICO_PUB, estados_motores);
    Serial.print("- Estados dos motores enviados via MQTT (");
    Serial.print(estados_motores);
    Serial.println(")");
    delay(1000);
}

```

```

}

/* Função: inicializa os outputs em nível lógico baixo (desliga os dois motores) Parâmetros: nenhum Retorno: nenhum*/
void init_outputs(void)
{
  pinMode(MOTOR_DIRETO, OUTPUT);
  pinMode(MOTOR_ESQUERDO, OUTPUT);

  digitalWrite(MOTOR_DIRETO, LOW);
  digitalWrite(MOTOR_ESQUERDO, LOW);
}

/* * Programa principal */
void loop()
{
  /* Ações a serem feitas no loop principal: - garante funcionamento das conexões WiFi e ao broker MQTT - envia o status de todos os outputs para o Broker - envia keep-alive da comunicação com broker MQTT */

  verifica_conexoes_wifi_e_MQTT();
  envia_estado_output_MQTT();
  MQTT.loop();
}

```

Resultado - movimentação e monitoramento via MQTT

Conforme mostrado no artigo Monitoramento de Temperatura Ambiente e Umidade Relativa do Ar via Internet (<https://newtoncbraga.com.br/index.php/microcontrolador/143-tecnologia/14409-monitoramento-de-temperatura-ambiente-e-umidade-relativa-do-ar-via-internet>), é possível utilizarmos o cliente MQTT MQTTLens (<https://chrome.google.com/webstore/detail/mqttlens/hem-jaaeigabkbcookmlgmdigohjobjm?hl=pt-BR>) para enviar e receber informações via MQTT do módulo WiFi LoRa 32(V2). Veja o resultado da recepção dos estados dos motores na figura 4.

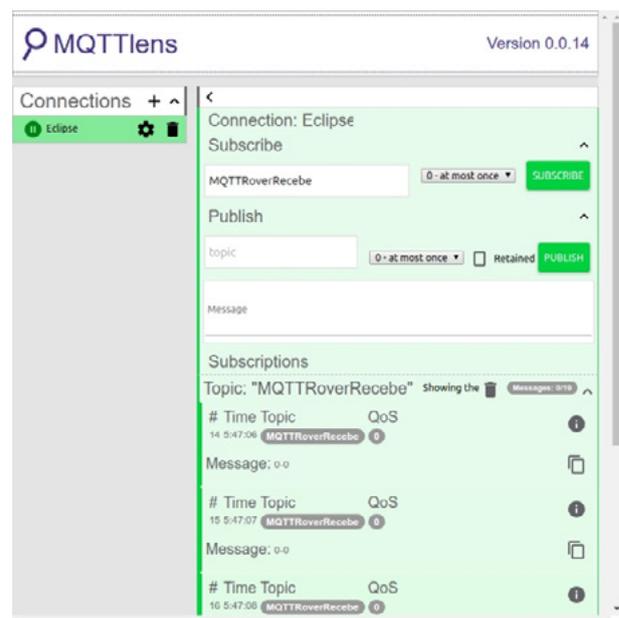


Figura 4 - estado dos motores (recebidos via MQTT no cliente MQTTLens)

Segunda forma de controle e acionamento de motores DC: via L298N

Aqui, acionaremos e controlamos o motor utilizando um módulo que contém o circuito integrado L298N, um driver de motor e ponte H. Este circuito integrado permite o controle de dois motores (cujo consumo de corrente elétrica de cada motor seja, no máximo, 2A) de forma a, além de acionar e “desacionar” os mesmos, controlar os sentidos de rotação de cada motor (horário e anti-horário). Facilmente encontrado no mercado maker, ele é vendido na forma de um módulo, já com tudo necessário para operar. Tal módulo pode ser visto na figura 5, e é comumente encontrado como módulo L298N.

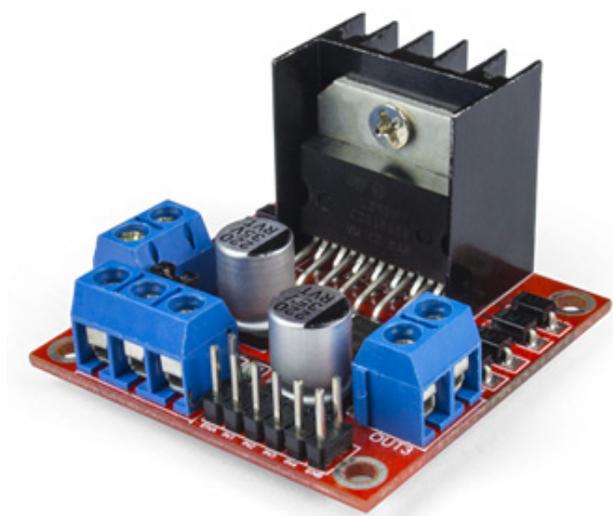


Figura 5 - módulo de acionamento de motores (com ponte H) feito com o circuito integrado L298N

As especificações do módulo L298N podem ser vistas a seguir:

- Tensão de operação: 4V até 35V
- Permite controle simultâneo de 2 motores DC ou 1 motor de passo
- Corrente de operação máxima por motor/canal: 2A
- Tensão da parte lógica/controladora: 3,3V
- Limites de temperatura de operação: -20°C a +135°C
- Potência máxima: 25W (com uso de dissipador, já contido no módulo)

Funcionamento do módulo de driver de motor e ponte H

O módulo de driver de motor e ponte H tem as entradas e saídas definidas conforme mostrado na figura 6.

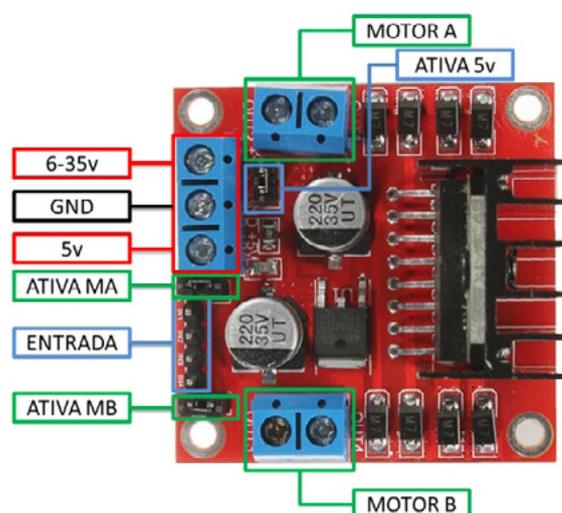


Figura 6 - entradas e saídas do módulo de driver de motor e ponte H contendo o L298N (fonte da imagem: <https://www.filipeflop.com/blog/motor-dc-arduino-ponte-h-l298n/>)

As entradas e saídas, bem como suas funcionalidades, estão detalhadas abaixo:

MOTOR A e MOTOR B: borne para ligação de dois motores DC (ou um motor de passo, se utilizado ambos os canais). Em cada canal (MOTOR A ou MOTOR B) é possível acionar e controlar o sentido de rotação de um motor DC de consumo de corrente de até 2A (isso é possível graças ao dissipador do L298N que acompanha o módulo)

ATIVA MA e ATIVA MB: jumpers para os motores A e B (ATIVA MA e ATIVA MB, respectivamente) que indicam se estes terão sua velocidade controlada por PWM (jumper ausente) ou se não haverá controle de velocidade dos motores (jumper presente).

ENTRADA (conjunto de entradas IN1, IN2, IN3 e IN4): conjunto de quatro entradas com finalidade de controlar tanto o acionamento quanto o sentido de rotação de cada motor individualmente (a lógica de acionamento das entradas será detalhada mais adiante neste mesmo artigo).

ATIVA 5V: este módulo permite, conforme especificação, controlar motores DC de tensão de operação de 4V até 35V. Além disso, o módulo conta com um regulador de tensão de 5V. No caso de controle de motores DC de tensão de operação entre 6-35V (ou seja, jumper presente em ATIVA 5V), o regulador de 5V disponibiliza 5V no pino correspondente (chamado 5V). Isso tem uma utilidade muito interessante para o caso de, a partir da alimentação do motor, alimentar outros componentes eletrônicos externos com 5V (poupando a necessidade de se utilizar um regulador de tensão de 5V externo adicional). Com

jumper ausente em ATIVA 5V, o módulo espera que a alimentação para os motores venha de uma fonte externa, ligada ao pino 5V.

Controle da movimentação via Internet (com MQTT)

Neste ponto, o controle do robô via MQTT se difere um pouco do visto na versão com relés previamente neste artigo. A diferenciação se dá pelo fato de que, ao invés de se ter somente motor desligado e motor ligado como estados possíveis, com o módulo L298N existem os seguintes estados possíveis de acionamento e controle dos motores DC: motor parado, motor com giro em sentido horário e motor com giro em sentido anti-horário.

Além disso, como há a possibilidade de mudança no sentido de rotação dos motores, o robô ganha um novo movimento, o qual não era possível em sua versão de controle e acionamento de motores DC com relés: ir para trás (ré).

Desta forma, os caracteres utilizados para controlar o rover são:

- Caractere que, quando recebido, move o rover para frente: F (figura 7.a)
- Caractere que, quando recebido, move o rover para direita: D (figura 7.b)
- Caractere que, quando recebido, move o rover para esquerda: E (figura 7.c)
- Caractere que, quando recebido, move o rover para esquerda: R (figura 7.d)
- Caractere que, quando recebido, faz o rover parar: P

Observe as figuras 7A e 7B para mais detalhes.

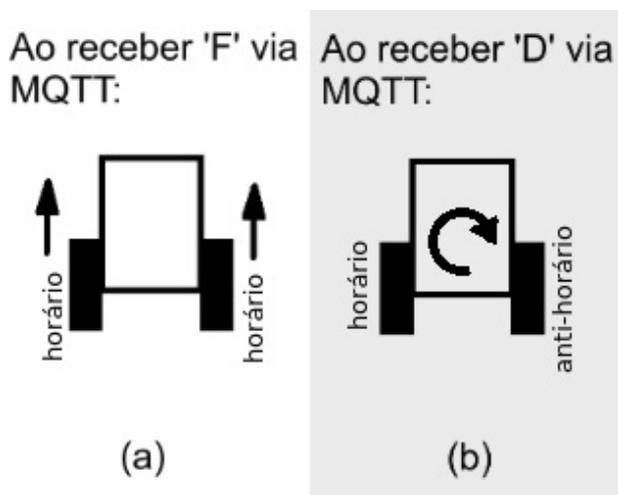
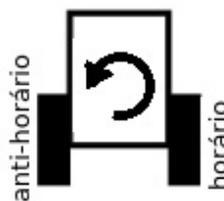


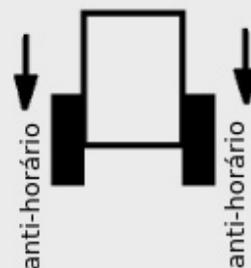
Figura 7A - movimentação do rover mediante comando recebido via MQTT (com controle de motores utilizando módulo L298N)

Ao receber 'E' via MQTT:



(c)

Ao receber 'R' via MQTT:



(d)

Figura 7B - movimentação do rover mediante comando recebido via MQTT (com controle de motores utilizando módulo L298N)

Além disso, também há pequenas diferenças quanto às informações enviadas pelo rover à via MQTT. Agora, cada motor pode ter seu estado representado por três caracteres distintos:

- Motor parado: caractere 'P'
- Motor com giro em sentido horário: caractere 'H'
- Motor com giro em sentido anti-horário: caractere 'A'

Portanto, a mensagem enviada pelo rover via MQTT tem formato definido conforme abaixo:

- Estado do motor direito (caractere 'P', 'H', ou 'A')
- Separador: caractere '-'
- Estado do motor esquerdo (caractere 'P', 'H', ou 'A')

Acionamento de motores e controle de sentido de rotação com entradas IN1, IN2, IN3 e IN4

Para entender por completo o módulo, resta apenas compreender como acionar os motores e controlar o sentido de rotação de cada um deles. Para isso, iremos fazer uso das entradas IN1, IN2, IN3 e IN4. De forma geral, as entradas IN1 e IN2 controlam o acionamento e sentido de rotação do motor A, enquanto IN3 e IN4 controlam o acionamento e sentido de rotação do motor B.

Observe a tabela 1. Nela, são informadas as condições de IN1, IN2, IN3 e IN4 para cada situação de acionamento e controle de rotação dos motores. Nela '1' significa nível lógico alto (3,3V), '0' significa nível lógico baixo (0V) e 'X' é uma don't care condition.

IN1	IN2	IN3	IN4	Comportamento esperado
0	0	X	X	Motor A em “ponto morto” (sem giro e sem freio)
1	1	X	X	Motor A com freio (sem giro)
1	0	X	X	Motor A acionado e com giro em sentido horário
0	1	X	X	Motor A acionado e com giro em sentido anti-horário
X	X	0	0	Motor B em “ponto morto” (sem giro e sem freio)
X	X	1	1	Motor B com freio (sem giro)
X	X	1	0	Motor B acionado e com giro em sentido horário
X	X	0	1	Motor B acionado e com giro em sentido anti-horário

Circuito esquemático

O circuito esquemático do rover utilizando o módulo de driver de motor e ponte H com L298N pode ser visto na figura 8.

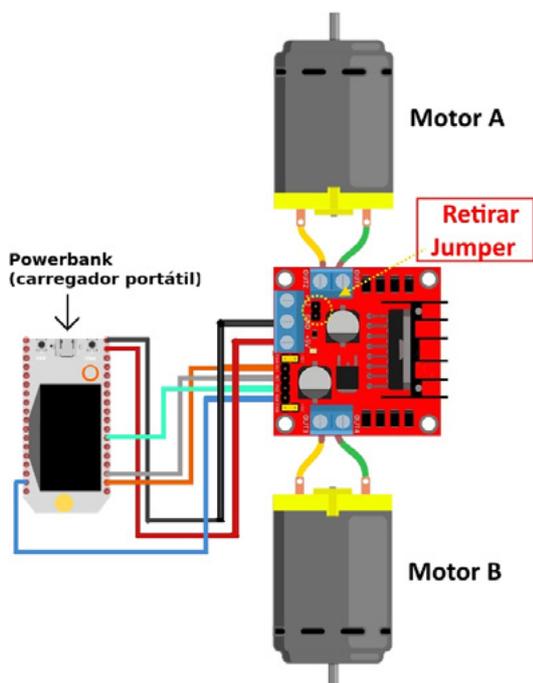


Figura 8 - circuito esquemático (com módulo L298N)

A seguinte observação é pertinente: para a alimentação dos motores é necessário utilizar um carregador de baterias de celular portátil (conhecido no mercado como powerbank). O motivo disso é que os motores, quando acionados, consomem muita energia, levando ao rápido consumo de baterias. Portanto, a utilização de carregadores portáteis de celular minimizará custos do projeto (e torná-lo recarregável). Tais carregadores portáteis possuem capacidade de fornecimento de corrente mais do que suficientes para o referido projeto.

Código-fonte

É chegada a hora de escrever o código-fonte do projeto utilizando o módulo L298N para controlar os motores DC do rover.

IMPORTANTE 1: leia atentamente os comentários do mesmo para completa compreensão de seu funcionamento.

IMPORTANTE 2: este projeto também faz uso da biblioteca relativa ao MQTT informada no artigo Monitoramento de Temperatura Ambiente e Umidade Relativa do Ar via Internet (<https://newtoncbraga.com.br/index.php/microcontrolador/143-tecnologia/14409-monitoramento-de-temperatura-ambiente-e-umidade-relativa-do-ar-via-internet>).

Há muito tempo, numa galáxia muito, muito distante acontecia a saga de Star Wars. Mas na MOOCA centro do universo foi desenvolvido o Robô Lixinho em homenagem a um dos robôs mais queridos do universo.



**Luiz Henrique
Corrêa Bernardes**

Instagram: @moocalab

ROBÔ LIXINHO

O Robô Lixinho, é um carismático Robô que tem semelhança do R2D2, ele pode ser usado como Lixinho de pia de cozinha, Lixinho de banheiro, de recepção de empresa ou em qualquer lugar inclusive no seu quarto! E toda vez que você jogar algo nele, fazendo com que a tampa se movimente, ele vai falar algo na linguagem do R2D2. Legal! Então vamos montá-lo.

Vamos precisar:

- Lixeira pequena branca
- Arduino Nano
- Buzzer
- Suporte de 4 pilhas AA
- 04 Pilhas AA
- Reed Switch
- Imã
- Fios para montagem



A figura 1 ilustra o circuito eletrônico, note que temos o Reed Switch que irá funcionar como uma chave, quando o ímã estiver próximo ao Reed Switch (tampa na posição fechada) ele fecha os contatos, caso contrário (tampa na posição aberta) ele abre os contatos. O Reed Switch é ligado direto no GND e no pino 10 do Arduino. O buzzer irá gerar o som, ele é ligado direto na alimentação (pino positivo "+") e no pino 11 do Arduino.

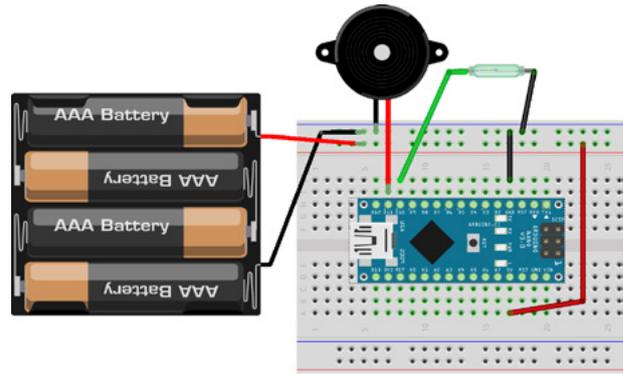


Figura 1 - O circuito

Agora vamos ao programa do Arduino:

Eu adaptei o programa do Marcelo Larios para fazer os sons do R2D2.

CÓDIGO COMPLETO

```
/******
```

Project R2D2 Sound Generator - To all fans of StarWars and Arduino!

Written by Marcelo Larios

BSD license, all text above must be included in any redistribution Participated in the Instructable Arduino Contest 2019 <https://www.instructables.com/R2D2-Sound-Generator/>

Adaptado por Luiz Henrique Corrêa Bernardes para o Artigo Robô Lixinho da Mecatrônica Jovem da Edição Robótica.

```
*****/
```

```
#define speakerPin 11 // Ligar Buzzer nesse pino
```

```
#define ledPin 13 // Led do Arduino
```

```
#define reed_rele 10 // Conectar Reed Rele
```

```
void setup() {
    pinMode(speakerPin, OUTPUT);
    pinMode(ledPin, OUTPUT);
    pinMode(reed_rele, INPUT_PULLUP);
    randomSeed(analogRead(0));
}
```

```
void phrase1() {
    int k = random(1000,2000);
    digitalWrite(ledPin, HIGH);
    for (int i = 0; i <= random(100,2000); i++){
        tone(speakerPin, k+(-i*2));
        delay(random(.9,2));
    }
    digitalWrite(ledPin, LOW);
}
```

```

    for (int i = 0; i <= random(100,1000); i++){
    tone(speakerPin, k + (i * 10));
    delay(random(.9,2));
    }
}

void phrase2() {
int k = random(1000,2000);
digitalWrite(ledPin, HIGH);
for (int i = 0; i <= random(100,2000); i++){
tone(speakerPin, k+(i*2));
delay(random(.9,2));
}
digitalWrite(ledPin, LOW);
for (int i = 0; i <= random(100,1000); i++){
tone(speakerPin, k + (-i * 10));
delay(random(.9,2));
}
}

void loop() {
if (digitalRead(reed_rele)) { // Se tampa aberta executa sons
int K = 2000;
switch (random(1,7)) {
case 1:phrase1(); break;
case 2:phrase2(); break;
case 3:phrase1(); phrase2(); break;
case 4:phrase1(); phrase2(); phrase1();break;
case 5:phrase1(); phrase2(); phrase1(); phrase2(); phrase1();break;
case 6:phrase2(); phrase1(); phrase2(); break;
}
for (int i = 0; i <= random(3, 9); i++){
digitalWrite(ledPin, HIGH);
tone(speakerPin, K + random(-1700, 2000));
delay(random(70, 170));
digitalWrite(ledPin, LOW);
noTone(speakerPin);
delay(random(0, 30));
}
noTone(speakerPin);
delay(random(2000, 4000));
}
}
}

```

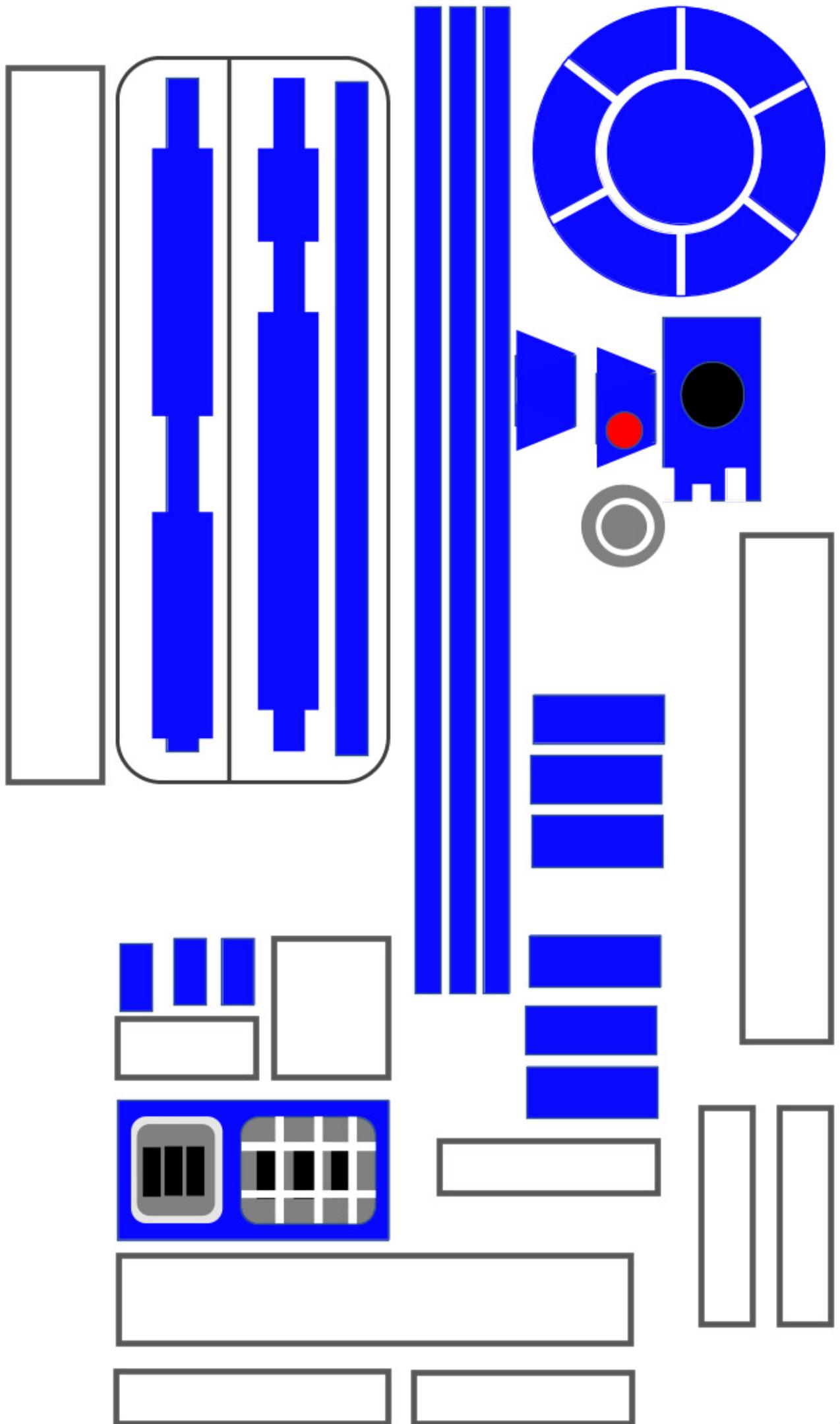


Figura 3



Figura 2 - O Reed-Switch

Faça o teste, se necessário ajuste o ímã para ficar próximo do reed-Switch (figura 2).

Agora na parte de customização do Lixinho, eu fiz o desenho (figura 3) que está em tamanho real e imprimi

em uma folha de etiqueta A4 da Pimaco, aí foi só adesivar o Robô Lixinho!!

Pronto , ficou bem legal ! Que tal montar o seu ? O que você faria diferente ? Você pode adaptar para colocar na porta do seu quarto, na sua gaveta de objetos preciosos. Poste seu projeto no Discord da Mecatrônica Jovem .

Se você traduzir o que ele está falando, em uma das frases ele canta um trecho do hino do Juventus da Mooca

Juventus , Juventus

Eu estou aqui

Boas montagens e nos encontramos aqui no centro do universo, ou nas lives de quarta feira do Clube da Mecatrônica Jovem.

Veja esse vocalizador para Apple ou Android.

<https://humancyborgrelations.com/r2d2/>

Hino do Juventus (Música e Letra) Compositor: Waldemar Leopoldo

Esse moleque travesso
Que tem nome e tradição
Merece nosso respeito
É a força jovem da nação

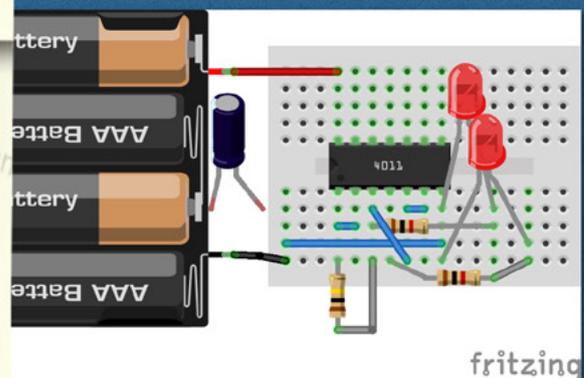
Que belo time
Que belo esquadrão
Juventus amigo
Do meu coração

Juventus, Juventus
Eu estou aqui
Vamos torcer juntos Juventus
E daqui nunca mais sair



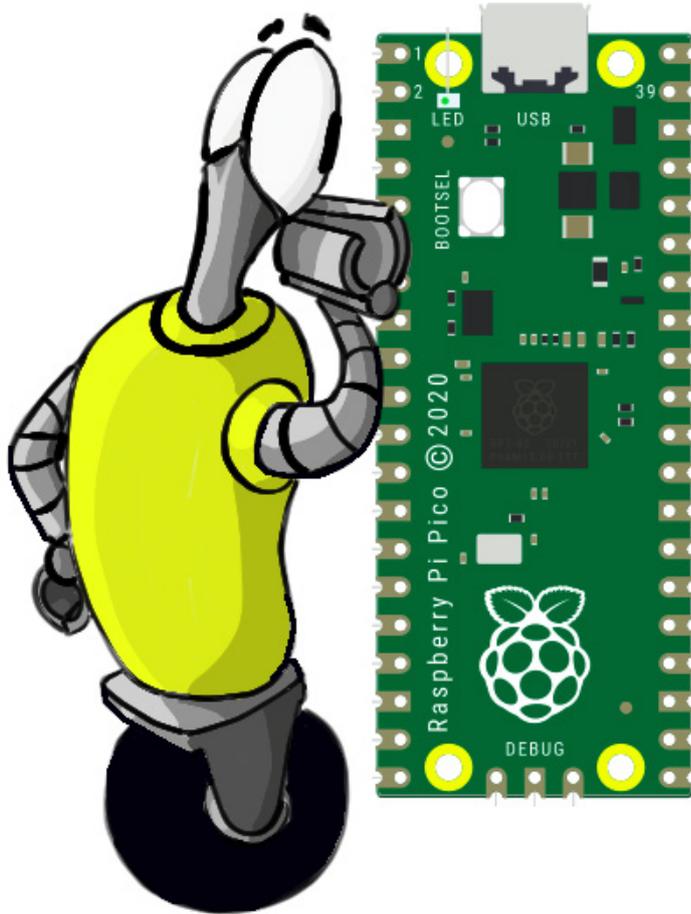
Hino do Juventus da Mooca

PROJETOS EDUCACIONAIS DE ROBÓTICA E MECATRÔNICA



e-Book
e Impresso

<-- mais detalhes



Eng. Márcio José Soares

- Página Web - <http://www.arnerobotics.com.br>
- Instagram - <https://www.instagram.com/arnesake/>
- YouTube - <https://www.youtube.com/c/arnesake>
- Thingiverse - <https://www.thingiverse.com/arnesake/designs>

ROBÔ RPI PICO 1 - UM ROBÔ BASE PARA SUAS EXPERIÊNCIAS

INTRODUÇÃO

Que tal montar um pequeno robô que poderá servir de base para inúmeros projetos para o estudante/professor na escola/sala de aula?! O maker também poderá utilizá-lo como base para projetos mais sofisticados. Essa é a proposta deste artigo.

CARACTERÍSTICAS

A principal característica desse pequeno robô é sua simplicidade além da facilidade na montagem, já que o mesmo utiliza apenas alguns poucos módulos “prontos” de baixo custo. O uso destes módulos permite, principalmente para aqueles que estão em sala de aula, chegar com maior velocidade no resultado final. Dessa forma os professor/aluno não precisam perder tempo “criando” os

vários circuitos necessários. Porém para os que desejam e também para o estudante dos cursos técnicos em eletrônica e/ou engenharia é recomendado o estudo do circuito eletrônico dos vários módulos envolvidos para uma melhor compreensão do seu funcionamento.

Uma outra característica é que ele foi desenvolvido com base no novo “embarcado” Raspberry Pi Pico que tem em sua placa o microcontrolador RP2040 dual core ARM Cortex M0+ com 2MB de memória flash, 256kB de SRAM e clock de 133MHz. Além dessas, ele pode ser programado em MicroPython que tem muitos recursos interessantes e com certeza despertará o interesse do leitor em aprender mais esta linguagem de programação que desponta hoje como uma das mais promissoras no mundo dos microcontroladores.

CIRCUITO

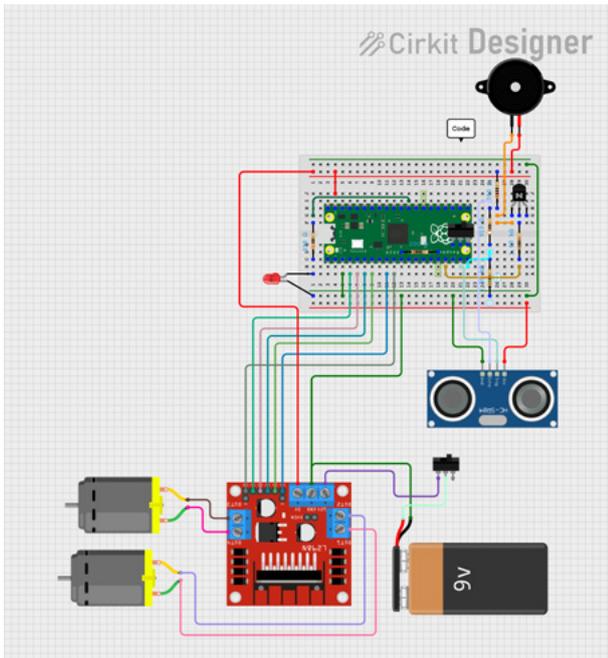


Figura 1 - circuito do robô

A figura 1 mostra o circuito eletrônico do robô. O cérebro do mesmo é um Raspberry Pi Pico programado em MicroPython. É ele quem controla os motores, sonar, buzzer e LED presentes no circuito. O módulo de controle para os motores é um L298N e ele também é responsável por fornecer ao microcontrolador e ao resto do circuito a tensão de 5VDC.

Obs.: A tensão de operação do microcontrolador RP2040 presente no Raspberry Pi Pico é de 3.3V! Porém o módulo possui um pino de entrada VSYS que permite sua conexão a alimentação externa em 5VDC máximos!

O módulo L298N pode ser alimentado com uma pequena bateria de 9V ou ainda com duas pilhas recarregáveis LiPo de 3.7V (com carga completa atingem 4.2V cada) ou ainda através de um conjunto de 8 pilhas (12VDC). No protótipo o autor usou um conjunto de 2 pilhas de LiPo com 3000mA.

O sensor de obstáculo escolhido foi um módulo sonar HC-SR04 (figura 2). Com ele o robô consegue “ouvir” através dos “ecos” emitidos e calcular a distância de um

obstáculo realizando o desvio antes da colisão com o mesmo.

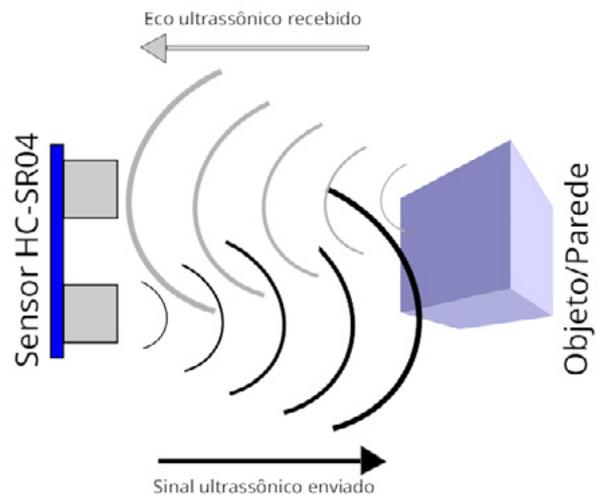


Figura 2 - sinal do sensor HC-SR04

Para entender como o sensor trabalha internamente veja a figura 3. Tem-se então que:

1. aplica-se um pulso com 10us no pino de Trigger (disparo) para iniciar a transmissão do sinal ultrassônico;
2. são emitidos então 8 pulsos de 40kHz (sinal ultrassônico);
3. o pino Echo será mantido em nível lógico alto (high) durante o tempo de ida e volta do sinal de saída.

Considerando que o sinal que nos interessa é apenas o retorno, tem-se então que a distância poderá ser calculada através da fórmula:

$$\text{Distância} = (\text{“Velocidade do som”} \times \text{“Tempo – pino Echo em High”}) / 2 \text{ [m/s]}$$

A “Velocidade do som” no ar é de aproximadamente 340 m/s e bastará determinar o tempo em que o pino “Echo” se mantém em “High” para que todos as variáveis da fórmula estejam presentes. A distância será obtida realizando o cálculo. A partir de algumas conversões é possível chegar a distância em centímetros ou polegadas, conforme o desejado por cada um.

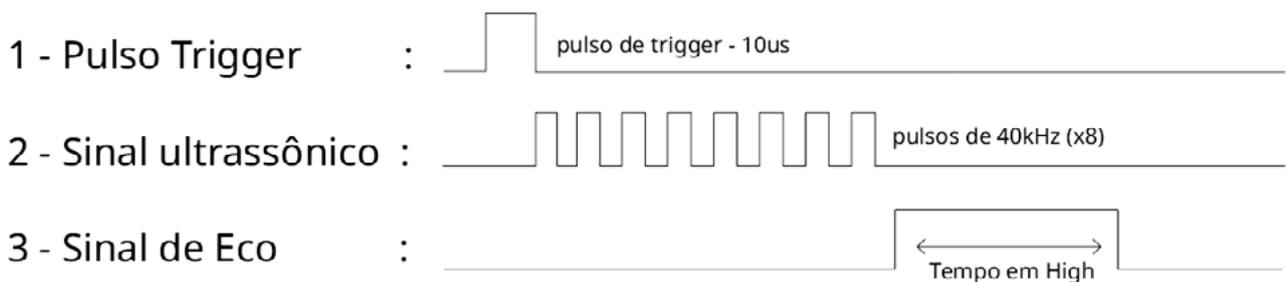


Figura 3 - funcionamento básico do sensor HC-SR04

O buzzer utilizado no circuito não possui oscilador interno. O transistor Q1 é o responsável pelo chaveamento do buzzer em 5VDC. O par de resistores R2 e R3 formam um divisor resistivo para a saída Echo do sonar que trabalha em 5VDC proporcionando uma queda de tensão para 3,3VDC, adequada aos pinos de I/O do Raspberry Pi Pico.

O LED1 serve para indicar o funcionamento do circuito (Led Live) e R4 é seu resistor limitador de corrente.

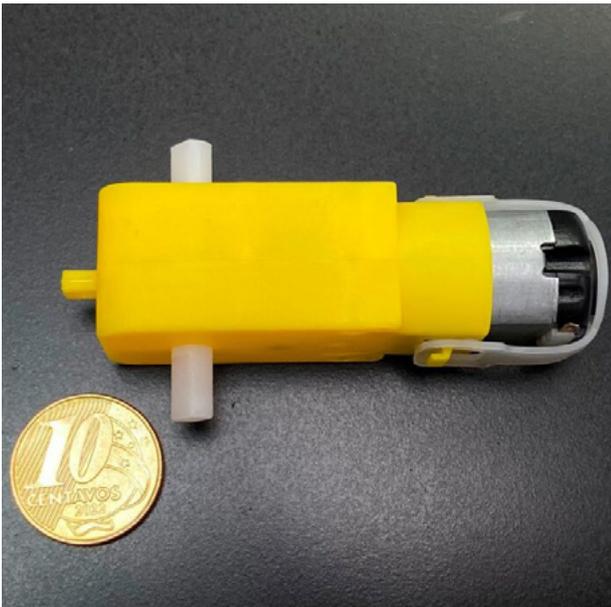


Figura 4 - motor DC com caixa de redução utilizado no projeto

Os motores DC devem possuir acoplados aos mesmos caixas de redução para aumentar o torque, conforme a figura 4.

Montagem eletrônica

A montagem pode ser feita utilizando uma pequena matriz de contatos (proto-board) de 440 pontos (opção do autor). Porém é possível também montar o circuito em uma placa padrão ou ainda preparar um lay-out de circuito impresso considerando todos os componentes presentes nos módulos. A escolha é livre e cada um deve optar por aquela que lhe for mais conveniente.

O leitor se optar pela montagem conforme descrita nesse artigo, pode usar cabinhos tipo jumper comuns no mercado de componentes eletrônicos e também na internet. Apenas tome cuidado e habitue-se a medir a resistência de tais cabinhos. Em alguns casos ela é alta demais e isso pode prejudicar algumas conexões. Verifique sempre!

O Raspberry Pi Pico, juntamente com o transistor, LED e resistores devem ser montados na matriz de contatos. O buzzer, dependendo do modelo, pode ou não ser montado também no proto-board. Certifique-se de ligar os

pontos certos do sonar e também de ligar corretamente a saída Echo a entrada do divisor resistivo. Cuidado ao ligar os 5VDC vindos da placa L298N. Certifique-se que ele tem 5VDC.

Uma chave liga/desliga pode ser utilizada interrompendo o positivo do suporte de bateria. Assim fica fácil desligar o robô quando necessário.

Montagem mecânica

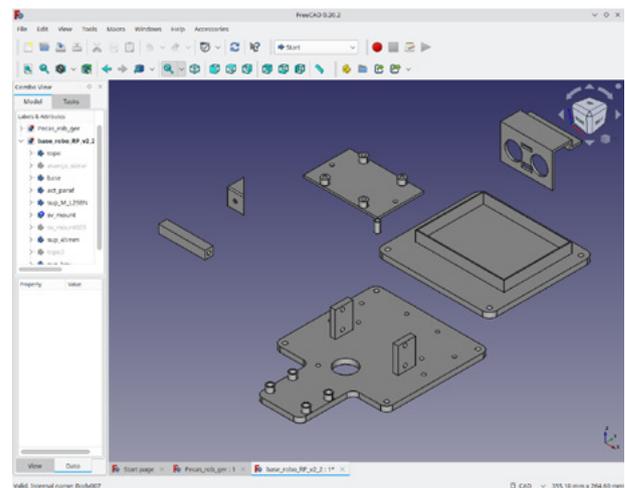


Figura 5 - desenho das peças do chassi

O chassi do protótipo foi feito com o auxílio de uma impressora 3D (figura 5). Para aqueles que dispõem de uma, os arquivos STL estão disponíveis para download no perfil do autor no Thingiverse. Porém os professores/alunos poderão preparar o chassi a partir de outros materiais como plástico, madeira, papelão ou ainda montá-lo utilizando kits de robótica educacional disponíveis em sua casa/escola.

Obs.: No perfil do Thingiverse o leitor encontrará também a lista de parafusos necessárias para esta montagem.

O PROGRAMA

No quadro 1 é possível ver o código fonte em MicroPython desenvolvido pelo autor (disponível também para download em <http://www.arnrobotics.com.br>). O editor utilizado foi o Thonny 3.3.13 Linux (também disponível para Windows) e o mesmo pode ser obtido através do link <https://thonny.org/>. No canal do YouTube do autor o leitor poderá ver um vídeo com a operação do robô e também uma breve explicação sobre o funcionamento do programa. Veja no link <https://www.youtube.com/c/arnesake>.

Quadro 1

```
#-----  
# Robô com RaspberryPi Pico  
# Arquivo: robo_raspPico.py  
# Versão : 1.0  
# Plataforma: Raspberry Pi Pico - Thonny  
# Desenvolvido por: Eng. Márcio José Soares  
# Pinos utilizados:  
# GPIO02 - motor In4 - pino 4  
# GPIO03 - motor In3 - pino 5  
# GPIO04 - motor In2 - pino 6  
# GPIO05 - motor In1 - pino 7  
# GPIO06 - motor ENB - pino 9 (PWM)  
# GPIO07 - motor ENA - pino 10 (PWM)  
# GPIO13 - Buzer - pino 17  
# GPIO22 - LED Live2 - pino 29  
# GPIO25 - LED Live1 - LED Built-in  
#-----  
  
#-----  
# Bibliotecas incluídas  
#-----  
from machine import Pin  
import time  
import utime  
#-----  
  
# Define pinos  
#-----  
pinIni1Esq = 2 #  
pinIni2Esq = 3  
pinIni3Dir = 4  
pinIni4Dir = 5  
pinTrig = 14  
pinEcho = 15  
pinBuzzer = 13  
pinLED_Live1 = 25  
pinLED_Live2 = 22  
#-----
```

```

#                               Configura pinos
#-----

In1Esq                          = Pin(pinIni1Esq, Pin.OUT) # cria pinos de controle da ponte H
In2Esq                          = Pin(pinIni2Esq, Pin.OUT)
In3Dir                          = Pin(pinIni3Dir, Pin.OUT)
In4Dir                          = Pin(pinIni4Dir, Pin.OUT)
EnEsq                          = machine.PWM(machine.Pin(6)) # pinos com PWM
EnDir                          = machine.PWM(machine.Pin(7))
EchoS                          = Pin(pinEcho, Pin.IN) # pinos para uso no sonar
TrigerS                        = Pin(pinTrig, Pin.OUT)
LedLive1                       = Pin(pinLED_Live1, Pin.OUT)
LedLive2                       = Pin(pinLED_Live2, Pin.OUT)
Buzzer                         = machine.PWM(machine.Pin(pinBuzzer)) # pino do buzzer
EnEsq.freq(1000);              # frequência nos pinos de PWM
EnDir.freq(1000);
Buzzer.freq(1000);
#-----

#                               Vars globais
#-----

duty                            = 23000 # Duty cicle                inicial
duty_05                         = 21000
#-----

#                               Função sonar
#                               Adaptada do exemplo obtido em Brincando com ideias
#                               Entradas - nenhuma
#                               Saídas - distância medida
#-----

def                               sonar():
new                               = False
counter                          = 0
result                           = 0
dur                              = 0

TrigerS.value(0)                 # Pino em Low
time.sleep_us(2)                 # aguarda
TrigerS.value(1)                 # levanta pino
time.sleep_us(10)                # aguarda
TrigerS.value(0)                 # pino em low
time.sleep_us(2)                 # aguarda

```

```

while          not EchoS.value(): # se pino em low
pass
counter        += 1 # incrementa contador
if            counter == 5000: # se chegou no limite
new           = True # prepara retorno
break        # sai do laço
if           new: # retorna falso
return       False
inicio        = time.ticks_us()/1000000 # pega tempo
while        EchoS.value(): # lê sinal de retorno
pass         # sai quando em low
ret          = time.ticks_us()/1000000 # lê novo tempo
if          ret == inicio: # se tempo igual, retorna
result       = "N/A"
else:
dur          = ret - inicio # do contrário, calcula distância
soundSpeed   = 34300 # velocidade do som
result       = dur * soundSpeed / 2 # só interessa tempo de ida
result       = round(result, 1) # uma casa decimal, arredonda se necessário
return       result # retorna a distância medida

#-----
#           Função stopMotors
#           Entradas - nenhuma
#           Saídas - nenhuma
#-----

def          stopMotors():
print("pare          o robô") # mensagem na tela
In1Esq.low()          # acerta pinos
In2Esq.low()
In3Dir.low()
In4Dir.low()
EnEsq.duty_u16(0)          # duty em zero!
EnDir.duty_u16(0)

#-----
#           Função para frente
#           Entradas - valor do duty cycle desejado
#           Saídas - nenhuma
#-----

```

```

def                                goForward(myduty):
print("p/ frente") # mensagem na tela
In1Esq.high()  # acerta pinos
In2Esq.low()
In3Dir.low()
In4Dir.high()
EnEsq.duty_u16(myduty)                # liga PWM
EnDir.duty_u16(myduty)

#-----
#                                Função para trás
#                                Entradas - valor do duty cycle desejado
#                                Saídas - nenhuma
#-----
def                                goBack(myduty):
print("p/                tras") # mensagem na tela
In1Esq.low()                # acerta pinos
In2Esq.high()
In3Dir.high()
In4Dir.low()
EnEsq.duty_u16(myduty)                # liga PWM
EnDir.duty_u16(myduty)

#-----
#                                Função para direita
#                                Entradas - valor do duty cycle desejado
#                                Saídas - nenhuma
#-----
def                                goRight(myduty):
print("p/ direita") # mensagem na tela
In1Esq.high()                # acerta pinos
In2Esq.low()
In3Dir.high()
In4Dir.low()
EnEsq.duty_u16(myduty)                # liga PWM
EnDir.duty_u16(myduty)

#-----
#                                Função para esquerda
#                                Entradas - valor do duty cycle desejado
#                                Saídas - nenhuma

```

```

#-----
def          goLeft(myduty):
print("p/ esquerda") # mensagem na tela
In1Esq.low()          # acerta pinos
In2Esq.high()
In3Dir.low()
In4Dir.high()
EnEsq.duty_u16(myduty)          # liga PWM
EnDir.duty_u16(myduty)

#-----
#   Função "main" (não existe em Python, mas sou programador "old school" de C!!!)
#           Entradas - nenhuma
#           Saídas - nenhuma
#-----

mycount          = 0
myLED            = False
LedLive1.low()
LedLive2.low()
stopMotors()     # inicia com motores   parados
Buzzer.duty_u16(0) # buzzer desligado
utime.sleep(2)   # aguarda 2 segundos para iniciar
print("Inicio do andar do robô!!!") # mensagem inicial

while True: # laço eterno!
dist          = sonar() # mede a distância
print("Dist: "+str(dist)) # print
time.sleep(0.25)   # aguarda 0.25s
mycount = mycount + 1 # soma 1
if mycount == 2: # se ja passou 0.5s
mycount = 0
if myLED: # verifica estado do LED e troca
myLED = False # o estado
LedLive1.low()
LedLive2.high()
else:
myLED          = True
LedLive1.high()

```

```

LedLive2.low()
#continue

if dist >= 10.0: # se a distância maior ou igual a 10cm
    goForward(duty)      # robô a frente
else:                 # do contrário
    Buzzer.duty_u16(1000) # liga buzzer
    stopMotors()        # para os motores
    goBack(duty_05)     # anda para trás
    time.sleep(1) # por 1 segundo
    stopMotors() # para
    goLeft(duty_05)     # vira para esquerda
    time.sleep(1)      # por 1 segundo
    stopMotors()      # para
    Buzzer.duty_u16(0) # desliga buzzer e começa tudo novamente
    
```

TESTE E USO

Após a montagem do circuito confira tudo. Não tenha pressa nessa etapa. Lembre-se que o Raspberry Pi Pico não aceita tensões maiores que 3,3VDC em seus pinos! Uma boa conferida pode evitar problemas!

Ligue o circuito, conecte a RBPi Pico ao seu PC através de um cabo USB e carregue o programa usando o botão Run script do editor Thonny . Esse é um bom momento para verificar se os motores estão girando no sentido “correto” de maneira que o robô se mova a frente. Se algo não parecer normal verifique as conexões entre o RBPi Pico e o módulo L298N. Você também poderá inverter as conexões dos motores a placa L298N e/ou os próprios motores em relação à placa.

Coloque um obstáculo a 10 cm do sensor e verifique se o desvio está sendo feito (ambos os motores giram para trás e em seguida os motores giram de maneira a fazer um giro à esquerda.

Tudo verificado é hora de gravar o programa de forma definitiva. Pare a execução do script. Eventualmente a execução será parada no ponto onde o programa estava, mas o microcontrolador não é resetado pelo editor mantendo os motores como estavam. Remova o cabo USB e o recoloque para que o reset seja efetivo. Agora selecione a opção no menu do editor “File → Save as...” (Arquivo → Salvar como...) e pressione o botão Raspberry Pi Pico (figura 6)

Agora basta salvar o programa com o nome main.py (veja a figura 7). Após salvar basta remover o cabo USB, desligar e ligar novamente o robô e ele estará pronto para a ação.

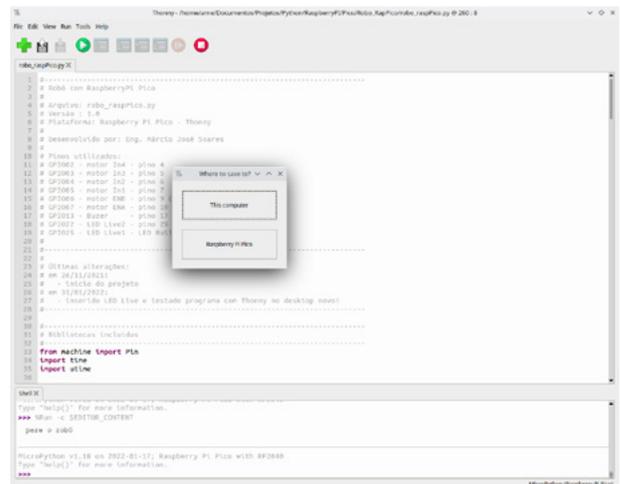


Figura 6 - botão de escolha para o nome do programa no RBPi Pico

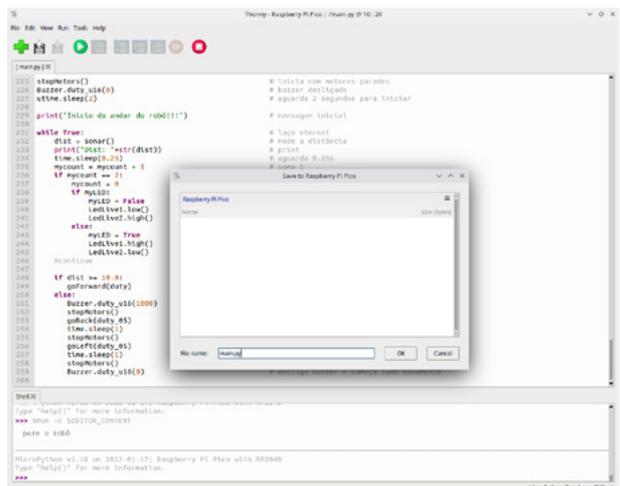


Figura 7 - salvando o programa como main.py

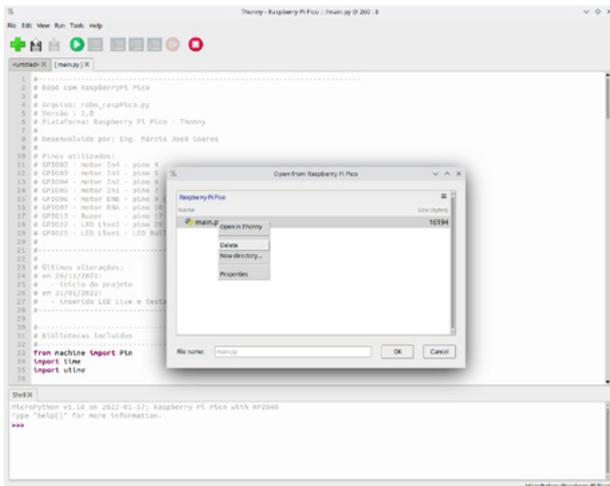


Figura 8 - apagando o programa main.py do RBPi Pico

Caso queira apagar o programa do seu RBPi Pico basta abrir novamente o editor Thonny com o microcontrolador conectado ao PC, selecionar o botão “Open → Raspberry Pi Pico” (Abrir → Raspberry Pi Pico) e clicar com o botão direito do mouse sobre o programa main.py que aparece na lista de arquivos dentro do RBPi Pico e selecionar “Delete” (figura 8). Pronto! Seu arquivo foi apagado! Reinicie o Thonny se quiser gravar um novo main.py!

CONCLUSÃO

Como dito no início desse artigo o robô não traz nenhuma novidade na sua operação, mas graças ao poder do microcontrolador presente no mesmo e a linguagem MicroPython com seus muitos recursos o leitor poderá estender as capacidades desse pequeno robô muito além do demonstrado aqui. Muitos são os recursos que podem ser implementados! Compartilhe suas ideias, modificações e outras relacionadas a este robô e também outras montagens com nossa turma no Discord:

(<https://discord.com/invite/sHmBawH6dT>).

Boa montagem e até a próxima!

PARA OS PROFESSORES

- Ciências/física poderá estimular os alunos a pesquisar mais sobre o som, seu deslocamento nos meios (ar e água), o eco, etc;

- Ciências/física/Matemática poderá estimular os alunos na pesquisa sobre robôs e IA aplicada aos mesmos;

- Língua inglesa poderá auxiliar os alunos na tradução dos datasheets necessários a montagem;

- Matemática/física poderá estimular a pesquisa sobre as conversões métricas.

LISTA DE MATERIAIS:

Módulos

1 – Módulo Sensor Ultrassônico HC-SR04

1 – Raspberry Pi Pico

1 – Módulo para controle de motores L298N

Componentes eletrônicos

Q1 – transistor BC337 NPN ou equivalente

R1 – resistor de 1k (marrom, preto, vermelho)

R2, R3 – resistores de 330R (laranja, laranja, marrom)

R4 – resistor de 4k7 (amarelo, violeta, vermelho)

LED1 – LED redondo 5mm difuso vermelho

Diversos

1 – Matriz de contatos com 440 pontos

2 – Micro Motores DC 3V-6V com Redutor 208 Rpm 1:48 (veja texto)

1 – cabinhos tipo jumper para interligar os itens

1 – suporte para bateria (veja texto)

1 – chassi para o robô (veja texto) e parafusos para montagem

REFERÊNCIAS BIBLIOGRÁFICAS:

- BRAGA, Newton C. - Os ultrassons (ART587) – Disponível em: <https://www.newtonbraga.com.br/index.php/52-artigos-tecnicos/artigos-diversos/4261-art587.html?highlight=WyJzb25hciJd> – Acesso em 12 de Maio de 2023

- BRAGA, Newton C. - Medida de Distancias Através de Sons (INS283) - Disponível em: <https://www.newtonbraga.com.br/index.php/usando-os-instrumentos/10868-medida-de-distancias-atraves-de-sons-ins283.html?highlight=WyJzb25hciJd> – Acesso em 12 de Maio de 2023

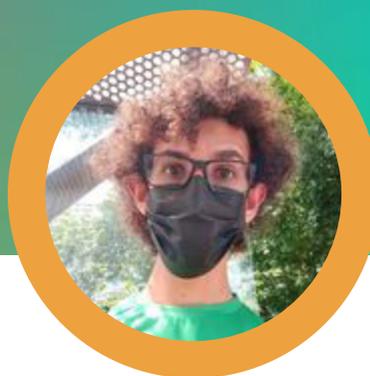
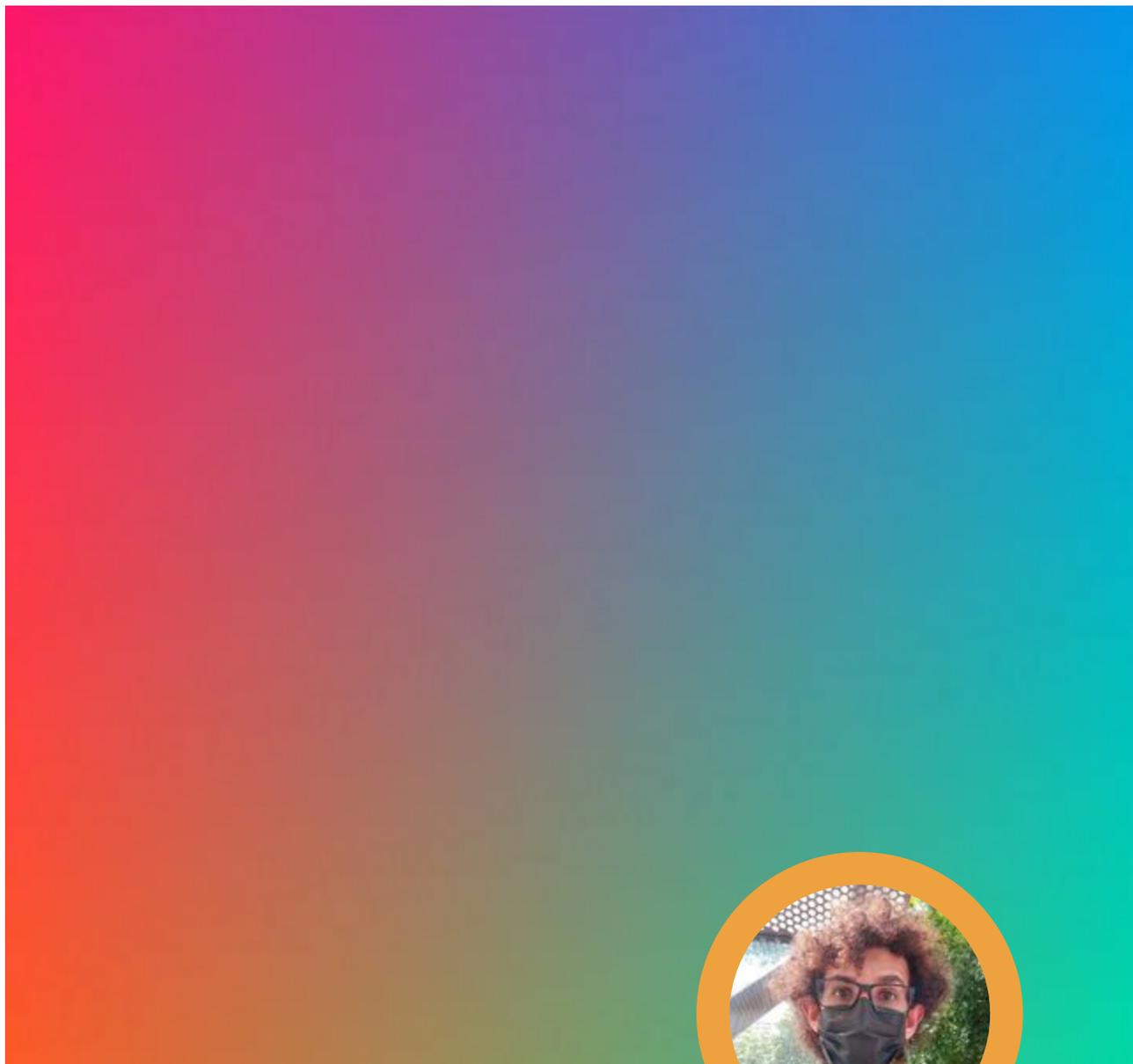
- GUIMARÃES, Flávio – Canal Brincando com ideias - Programe o Sensor de Distância HC-SR04 no Pico! YouTube – 24/11/2021 – Acesso em 26/11/2021

- BRAGA, Newton C. - Sensores ultrassônicos (MEC081) – Disponível em: <https://www.newtonbraga.com.br/index.php/robotica-e-mecatronica/3484-mec081.html?highlight=WyJzb25hciJd> - Acesso em 12 de Maio de 2023

- RASPBERRY – Raspberry Pi Pico – Datasheet – Disponível em: <https://datasheets.raspberrypi.com/pico/pico-datasheet.pdf>

- SPARKFUN – Ultrasonic Distance Sensor HC-SR04 – Datasheet – Disponível em: <https://cdn.sparkfun.com/datasheets/Sensors/Proximity/HCSR04.pdf>

538 473 252



**Otávio Lemes
Rodrigues**

Instagram: @taviolemes3101

SENSOR DE CORES

Por ser muito preciso e capaz de identificar diferentes níveis de cores, o sensor de cores TCS3200 é muito utilizado na indústria como também no mundo maker. Seu funcionamento é simples, se eu colocar um papel colorido na frente deste sensor, com o auxílio de um microcontrolador (como o Arduino) podemos fazer um led RGB acender de acordo com a cor ou fazer um separador de tampinhas coloridas, ou até mesmo identificar se a palheta de cores de uma embalagem está correta. Mas como ele faz isso? É que o veremos neste artigo.

DETECÇÃO DE CORES

Sabe quando levantamos de noite com luz apagada e parece que tudo é preto, até mesmo aquele vaso colorido que você sabe que está em cima da mesa, e momentos depois quando acendemos a luz, conseguimos ver nitidamente as cores da mesa, da parede e inclusive as do vaso brega da nossa avó?

O nosso sol e outras fontes de luz, emitem uma série de ondas eletromagnéticas, em diferentes frequências –

chamamos isso de espectro de emissão -, que se espalham pelo ambiente e ao entrarem em contato com um determinado material, uma parcela dessas frequências-ondas são absorvidas e a outras refletidas, dependendo do material o espectro de absorção muda. Assim, quando uma parcela de ondas eletromagnéticas, que não são absorvidas pelo material, é refletida e chega até os nossos olhos, essas ondas passam pela córnea, depois pela íris, a pupila, o cristalino e por fim chegam a retina.

Na retina existem células fotorreceptoras, que são sensíveis a determinados níveis de luz, sendo divididas em bastonetes e cones, em geral os bastonetes são usados em ambientes com baixa luminosidade e os cones em ambientes com alta, que são sensíveis a diferentes frequências de cor.

O TCS3200 funciona de uma forma bem similar, de uma forma geral, no centro da placa temos um sensor de luz, que consiste em uma matriz de 64 fotodiodos (como o LDR), divididos em 4 grupos de 16. Podemos pensar que cada um desses fotodiodos é como um cone do olho humano, cada um deles contém um filtro de luz, permitindo que apenas uma certa parcela de frequência de onda seja absorvida (o azul, vermelho e verde) e dependendo da cor que colocamos na frente do sensor, ela afeta uma quantidade "x" de fotodiodos dentro da matriz. Na figura 1 temos o esquema da matriz de cores os sensores.

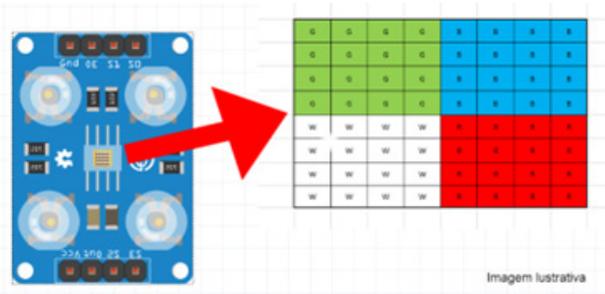


Figura 1 - Os esquemas de sensores são divididos em 4 matrizes de 16.

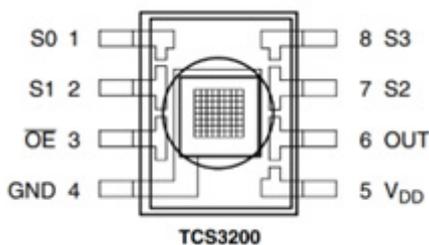


Figura 2 - Pinagem do TCS3200

Seria complicado criar um sensor de cor que tivesse um fotodiodo que detectasse cada frequência de onda

do espectro visível - imagina a complexidade da leitura de dados - por esta razão o sensor utiliza apenas três filtros, que correspondem ao padrão RGB (Red, Green and Blue, traduzindo, Vermelho, Verde e Azul), no qual dependendo da quantidade de "cones" acionados, ou seja, ao misturar estas três cores, podemos identificar qualquer cor. Uma informação importante, embora o sensor utilize o padrão RGB, a saída de dados não obedece a escala linear de 0 a 255 para cada cor, caso queira usar este padrão é necessário tratar estes dados.

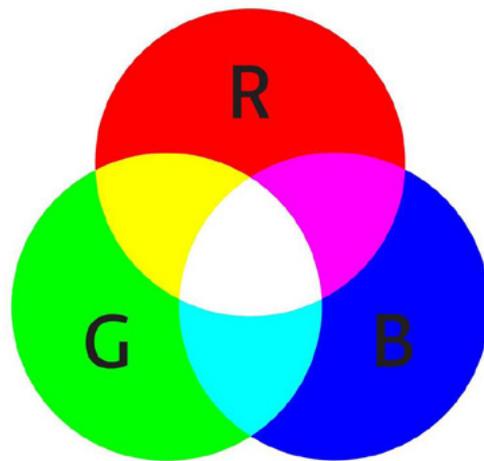


Figura 3 - Diagrama da escala de cores RGB.

ESTRUTURA

A placa TCS3200 é bem pequena e leve, contendo 4 leds brancos de iluminação e 8 pinos, como visto na figura 4.

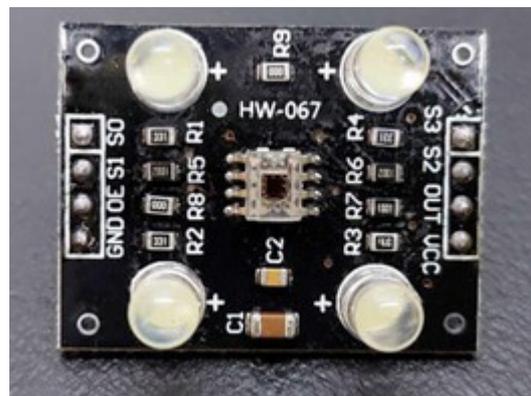


Figura 4 - Shield com 4 LEDs e o sensor TCS3200 ao centro.

- GND e VCC; são usados para alimentar o módulo (que funciona tanto em 3.3 V quanto em 5 V);
- S2 e S3; são usados para selecionar a cor que o sensor irá ler, seguindo a tabela:

S2	S3	Cor
Desligado	Desligado	Vermelho
Desligado	Ligado	Azul
Ligado	Desligado	Sem filtro
Ligado	Ligado	Verde

- S0 e S1; determina o nível de sensibilidade do sensor, seguindo a lógica da tabela:

S0	S1	Nível de Sensibilidade
Desligado	Desligado	Sensor desligado
Desligado	Ligado	2%
Ligado	Desligado	20%
Ligado	Ligado	100%

- OE; este pino inverte a lógica das saídas digitais do sensor (Se elas ligam em HIGH, com ele ativado, elas passam a ligar em LOW).

- OUT: é o pino de saída de dados, por meio de pulsos, na forma de onda quadrada.

Uma coisa interessante de se notar, é que este sensor só tem um cabo de transmissão de dados, no qual trafega informações de três conjuntos de fotoresistores. Esta capacidade de transmitir várias mensagens simultaneamente por um meio de comunicação, é chamada de multiplexação.

Antes de usar o sensor, é preciso tomar alguns cuidados, já que ele sofre muito com a interferência de fontes emissoras (que variam muito quanto ao espectro de emissão), a mudança do índice de reflexão de cada material (o verde escuro de um pedaço de fita adesiva não reflete a luz da mesma forma que o verde escuro de um pedaço de papel sulfite) e os próprios sensores podem ter variações nas leituras. Por isso é muito importante você calibrar o seu sensor de acordo com a realidade da sua aplicação.

Junto com o sensor de cor, é possível encontrar 4 leds brancos acoplados à placa, que auxiliam no balanceamento de cor, mas isso não é o suficiente, já que fontes externas, como a luz do seu quarto, podem incidir sobre o material e interferir nas leituras. Uma solução bem eficaz é fazer uma caixinha do tamanho do sensor, - nós usamos uma caixinha de mdf, cortada no laser, mas também é possível fazer uma usando papelão - que impeça que a luz externa chegue até o sensor, além de deixar o sen-

sor a uma distância constante do material a ser analisado (figura 5).

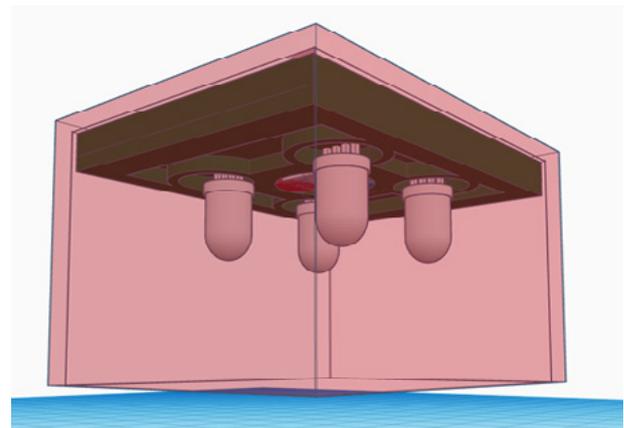


Figura 5 - Sugestão de aplicação no uso do sensor de cores.

Também é muito importante saber quais materiais o sensor irá analisar e tê-los em mãos no momento de calibrar. Uma dica é usar materiais com cores bem vivas (saturadas) e evitar cores mais escuras (menos saturadas), isso facilita muito na hora de calibrar o sensor.

Lista de Materiais

- Arduino Uno;
- TCS3200;
- Led RGB (pode ser substituído por 3 leds coloridos e os seus respectivos resistores);

- Resistor de ~330 ohms;
- Caixa para o sensor de cor;
- Jumpers;

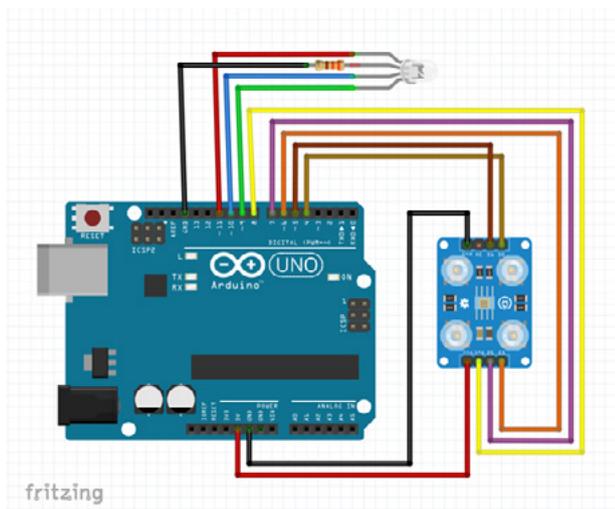


Figura 6 - Esquema elétrico do projeto.

Antes de mais nada, coloque o sensor dentro da caixinha e siga o diagrama de conexões da figura 6, as conexões entre o sensor e o módulo são bem simples, só tome cuidado para não inverter o GND com o VCC (um curto sempre é prejudicial a um equipamento eletrônico), e certifique-se de que as conexões dos pinos do LED RGB estão corretas antes mesmo de alimentar o arduino - não queremos que o sensor esteja identificando a cor vermelha e o LED fique azul. Um protoboard pode te ajudar a posicionar o LED RGB e os resistores, mas infelizmente o TCS3200 é muito grande (seriam necessárias 2 protoboards), então use jumpers do tipo macho-fêmea neste caso.

Após fazer todas as conexões, abra a Arduino IDE (em uma tradução direta, ambiente de Desenvolvimento do Arduino), leia os comentários do código acima e tente entender o que cada bloco do código faz (isso fica como um desafio).

Lembre-se que a linguagem de programação C++, é só a forma como os programadores escrevem as instruções para a máquina - quando o código for compilado para o arduino, vai ser convertido em uma linguagem de máquina, que só entende zeros e uns (o famoso código binário) -, o mais importante é compreender o algoritmo (a sequência de passos para solucionar qualquer problema) e a lógica por trás de cada uma das etapas, sobretudo como o sensor funciona. Com o Arduino conectado a sua máquina, carregue o código, e abra o monitor serial.



Figura 7 - QR-Code de acesso ao código.

No Monitor Serial, você encontrará os valores que estão sendo medidos, e a sua respectiva cor, e também será mostrado o valor da cor predominante.

A princípio, pode ser que o sensor fique meio “desorientado” e que caso você coloque algum objeto colorido na frente, ele mostre uma cor incorreta. Isso é normal, afinal você ainda não “ensinou” para o sensor a frequência de cor que ele deve considerar vermelho, azul ou verde.

Repare que se tu colocares algo branco, que reflete todas as frequências de cor, na frente do sensor, irá obter as maiores leituras e caso substitua por um objeto preto, no qual absorve quase todas as frequências de cor, irá obter as menores leituras (figura 7).

```
//Variáveis para calibrar o sensor
int vermelho = 95;
int verde = 100;
int azul = 110;
```

Figura 7 - Valores de calibração.

Para calibrar o sensor de cor, pegue o material da cor desejada e coloque o sensor em cima dele, quando os valores estabilizarem anote o valor da cor verde, que aparece no monitor serial. Faça o mesmo para as outras três cores primárias e ao final altere os valores das variáveis para cada cor no espaço reservado.

Pode ser que apareçam valores iguais ou muito parecidos, se isso acontecer, vá ajustando esses três campos, até obter o resultado desejado, caso não consiga tente alterar o material - Tive problemas com o verde escuro e com o azul, pois as leituras deles ficaram muito parecidas. Ao final, a cor do LED RGB deve mudar conforme a cor lida pelo sensor.

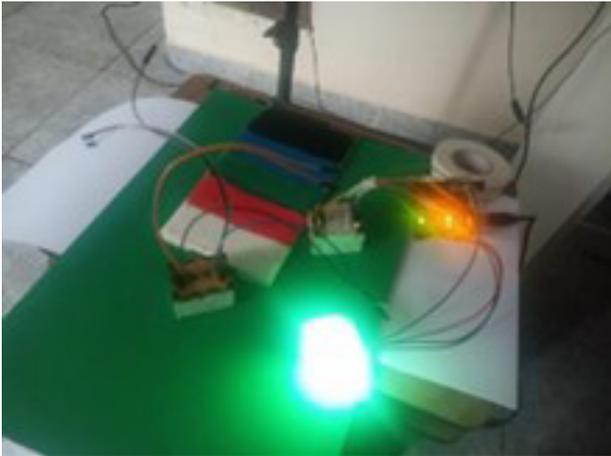


Figura 8 - Testando meu sensor

Agora segue um desafio, e se quisermos adicionar uma nova cor para ser reconhecida? Em um pedaço do código, nós realizamos uma comparação entre a intensidade das três cores primárias, para determinar qual estava em predominância.

```
//Verifica se a cor vermelha foi detectada em predominância
if ( 500 < vermelho < azul && vermelho < verde && vermelho)
{
  Serial.println("Vermelho");
  digitalWrite(pinoledverm, HIGH); //Acende o led vermelho
  digitalWrite(pinoledverd, LOW);
  digitalWrite(pinoledazul, LOW);
}
```

Figura 9 - Verificando a cor predominante.

Este código diz: “Se o valor de vermelho for maior que 500, menor que azul, ao mesmo tempo for menor que verde, e igual ao valor lido por vermelho. Quero que exiba ‘Vermelho’ no monitor serial e mude o LED RGB para vermelho”.

Para adicionar um amarelo, por exemplo, precisamos analisar as leituras feitas, eu consigo saber se ele tem mais vermelho, verde ou azul e usando os operadores lógicos (e/ao mesmo tempo, ou, diferente) e operacionais relacionais (>, <, =, != etc..) você consegue ensinar o sensor a reconhecer as cores.

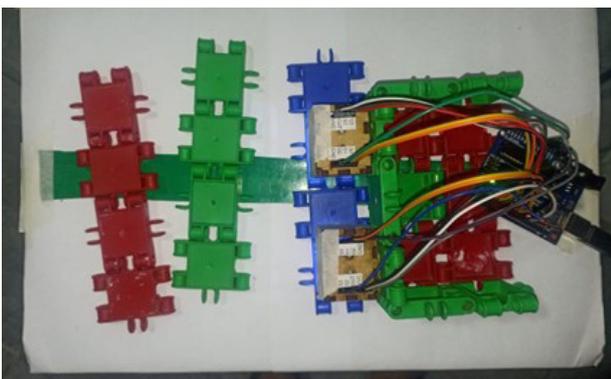


Figura 10 - Fazendo o sensor reconhecer a linha verde.

CONCLUSÃO

Trabalhar com algo novo, que você nunca sequer tinha ouvido falar, pode parecer uma tarefa um pouco assustadora - de fato é -, ainda mais quando se está trabalhando em grupo, todavia, também te obriga a sair da sua zona de conforto e enfrentar o desafio, te levando a desbravar novos horizontes. A minha experiência com o TCS3200, foi mais ou menos parecida.

Conectar um sensor no Arduino é a parte simples, o complicado é entender como o sensor funciona, uma vez que é preciso entender o estudo no qual o sensor se baseia (no caso do TCS3200 a óptica) e a lógica por trás do seu funcionamento. Só assim você consegue entender como o sensor interpreta os fenômenos do nosso mundo e converte em sinais elétricos/leituras, que são usados nas mais diversas aplicações.

Como nas linhas de produção industrial, em que o sensor de cor é muito útil para analisar os padrões de cores das embalagens ou de produtos, pois o sensor é bem mais sensível às cores do que o olho humano.

Um exemplo de aplicação é na agricultura este sensor poderia ajudar a monitorar as folhas de uma planta e descobrir se faltam nutrientes (algumas plantas ficam com folhas amareladas quando estão precisando de adubo, mas isso exige um maior aprofundamento).

Esses foram apenas algumas ideias, mas pense em quantas informações úteis as cores podem nos fornecer e se associarmos a mais sensores? Como o ultrassom que usamos no carrinho seguidor de linha? Deixe a sua imaginação fluir!

Por fim gostaria de fazer um convite para você professor, aluno, maker, entusiasta ou curioso de plantão. Venha fazer parte do clube da Mecatrônica Jovem, nos sigam nas redes sociais, entrem na comunidade do Discord, assistam as lives, toda quarta-feira, levem as suas dúvidas, compartilhem os seus projetos conosco. Como o senhor Julian C Braga citou uma vez nas lives “Ninguém é tão grande que não possa aprender, nem tão pequeno que não possa ensinar”. Nos vemos na próxima!

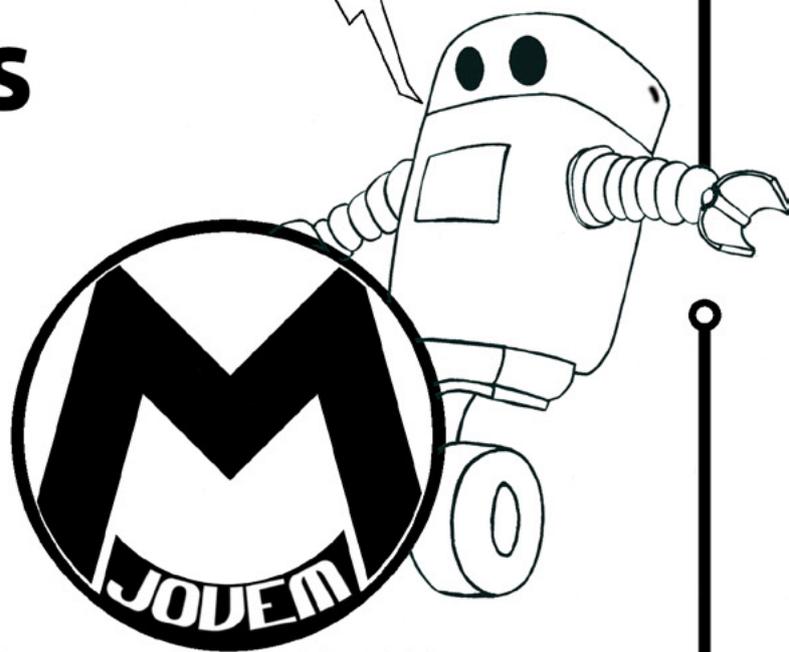
MECATRÔNICA

APRENDENDO CIÊNCIA E TECNOLOGIA

JOVEM

**Entre para
este Clube e
venha criar
projetos
tecnológicos
com uma
galera
nota 10**

Montagens,
Dicas,
Desafios,
Histórias,
Eventos e
Bate-Papo



REVISTAS
DOWNLOAD GRATIS

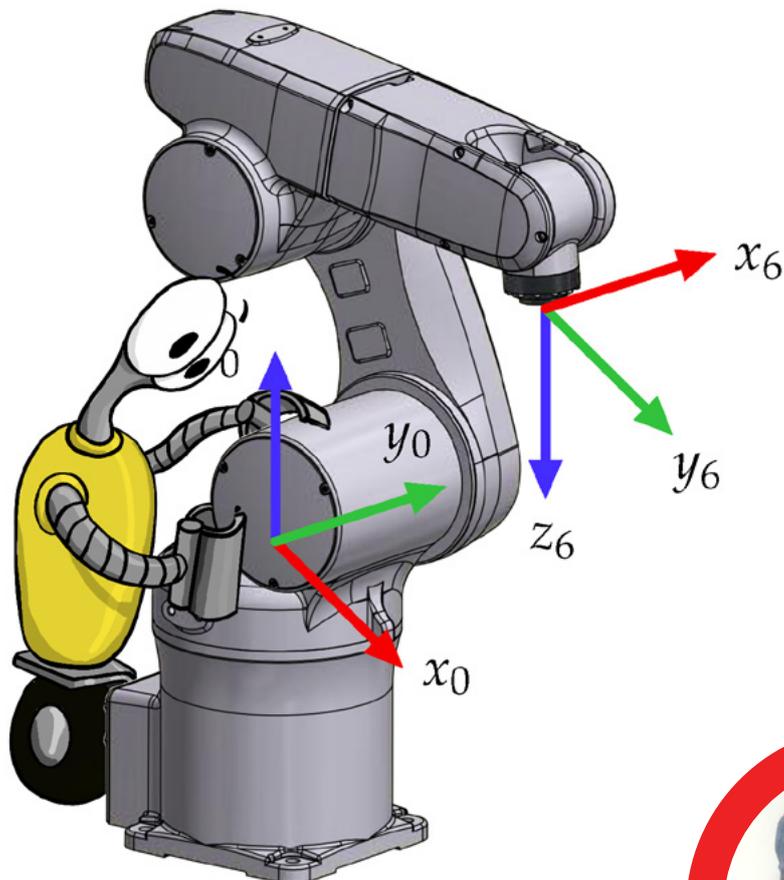


DISCORD



TWITCH





TIPOS DE MOVIMENTOS ROBÓTICOS



Renato Paiotti

Instagram: @mecjovem_oficial

Quando precisamos criar um robô, temos antes de tudo saber se ele transitará por algum lugar, pois dependendo do terreno que ele irá percorrer, as rodas podem não ser uma boa opção de locomoção. Para resolver este problema precisamos pesquisar e encontrar a melhor solução para movimentar do ponto A para o ponto B as estruturas metálicas, controladores, motores, garras, câmeras, antenas e muitos fios, por terrenos que até mesmo nós humanos não conseguimos trilhar. Por isso temos neste artigo algumas formas que engenheiros e cientistas criaram para locomoverem seus robôs.

ROBÔS BÍPEDES

Vamos começar esta lista com o mais próximo que nós conhecemos, nós mesmos, os bípedes. Desenvolver

um robô que tenha os movimentos idênticos aos dos humanos, é um objetivo almejado por muitos cientistas e projetistas, mas sabemos, porém, que o corpo humano é uma máquina muito bem elaborada, e copiar estes movimentos requer muita destreza, materiais leves, uma lógica de programação que atenda as tomadas de decisão e principalmente de equilíbrio. Um exemplo disto é o Asimo (figura 1), onde seus graus de liberdade de movimentos são perfeitos, mas é importante notar que a bateria que ele carrega é muito grande, pois precisa alimentar todos os sensores e motores para manter o equilíbrio de todo o conjunto.



Figura 1 - Robô bípede da Honda
Fonte: <https://pt.wikipedia.org/wiki/ASIMO>

ROBÔS RASTEJANTES

Dependendo do local onde o robô deverá caminhar, os movimentos que o Asimo executa hoje (bípede) são ineficientes. Um exemplo disto é a necessidade de termos um robô que ande pelos dutos de ar ou por dentro de sistemas líquidos, pernas não seria uma boa opção, para este tipo de trabalho uma boa alternativa seria o sistema que copia os movimentos de uma cobra ou outro animal rastejante, como podemos ver na figura 2.



Figura 2 - Usando o modelo da centopeia - Fonte: <https://www.tamiya.com/english/products/70230/top.jpg>

Outro sistema que copia o método de andar de uma cobra é o robô do projeto Anna Konda, que foi criado com o intuito de auxiliar bombeiros no combate ao fogo, transportando a mangueira de água em locais muito quentes onde colocariam em risco a vida dos bombeiros, outro ponto importante é o acesso deste equipamento em locais em que o homem não chega. Na figura 3 temos a foto deste robô conceito.



Figura 3 - Imitando o rastejar da cobra -
Fonte: [http://www.sintef.no/upload/IKT/9023/Prosjekter/Slangeroboter/Anna%20Konda%20med%20hud%20forfra%20\(HiRes\)](http://www.sintef.no/upload/IKT/9023/Prosjekter/Slangeroboter/Anna%20Konda%20med%20hud%20forfra%20(HiRes))

Atuando no mesmo conceito de pressão, porém do lado externo de um cano, temos outro robô conceito chamado HyDRAS, este robô foi criado com o intuito de salvar a



Figura 4 - Robô Hydra - Usando a pressão dos módulos para fixar ao cano - The Robotics and Mechanisms Laboratory (RoMeLa) of the College of Engineering at Virginia Tech. fonte <http://www.romela.org/hydras-hyper-redundant-discrete-robotic-articulated-serpentine>

vida de operários de obras. A sua forma de se locomover se baseia na pressão que os seus anéis exercem para se prender a um cano, como faria uma jiboia a sufocar a sua presa ou subir numa árvore. Para se locomover, este robô libera anel após anel, com um anel solto ele se projeta para a frente enquanto outros o prendem ao cano, e os que se soltaram, prendem-se mais adiante para dar suporte aos outros anéis que serão soltos e impulsionados para o mesmo caminho. Na figura 4 temos uma foto deste protótipo.

ROBÔS COM ESTEIRAS

O movimento por esteiras já é muito usado por tanques de guerra, onde a ideia é ter uma sequência de peças de metal por debaixo da roda, num processo contínuo, sendo muito útil em terrenos acidentados.

Um exemplo de robô que emprega esteira é o HD2-S Doberman, que tem o intuito de poupar a vida de soldados no campo de batalha, estes robôs soldados são equipados com esteiras iguais aos tanques de guerra, pois o uso de esteiras possibilita levar uma quantidade grande de peso sobre diversos tipos de terrenos acidentados, na figura 5 temos um exemplo deste tipo de robô.

Os robôs que utilizam esteira têm um funcionamento simples, pois se você precisa que o robô faça uma curva para a direita, reduzimos a velocidade do motor que movimentam a esteira do lado direito. O mesmo processo funciona para virar para o outro lado.



Figura 5 - DEMO HD2-S Doberman - Fonte: <https://www.superdroidrobots.com/robots/prebuilt-robots/prebuilt-tactical-robots/product=3314>

Na mesma área militar, mas com a finalidade de desarmar bombas, temos o EOD robot PIAP GRYF, que também funciona com esteiras, mas quatro esteiras independentes e que duas são móveis para subir desníveis, atuando como pseudo-patas, podendo ser movidas para

cima ou para baixo atuando como uma força extra em locais difíceis de chegar, na figura 6 temos uma foto deste sistema de esteiras.



Figura 6 - EOD robot PIAP GRYF - Conjunto de duas esteiras e rodas para desarmar bombas. - Fonte: https://www.antiterrorism.eu/piap_summarizes_2021/

ROBÔS COM QUATRO OU MAIS PATAS

Nos movimentos que utilizam pernas temos os tipos que são bípedes (2 pernas) como o Asimo, de 4 pernas (quadrúpede) como o BigDog, os de 6 e 8 que imitam o andar de insetos como formigas e aranhas, e os robôs que parecem uma centopeia de tantas pernas que possuem.

Nos robôs bípedes o ponto mais importante que um projetista deve ter em mente é o equilíbrio, a utilização de sensores e de giroscópios são necessários para manter o senso de equilíbrio, mas não é só isso que ajuda um robô bípede a ficar de pé, existe toda uma programação de locomoção de massa e ponto de gravidade que necessitam ser colocados no sistema do robô. Os robôs quadrúpedes também precisam ter em mente o equilíbrio, porém não tão complexa em comparação de um robô bípede, sua locomoção se dá uma perna dianteira no chão como apoio enquanto a outra segue para a frente, o mesmo processo se dá nas pernas traseiras, mas de lados trocados, ou seja, enquanto a perna dianteira esquerda se encontra no chão a traseira direita é que fica no chão como base, formando um X, conforme a figura 7.

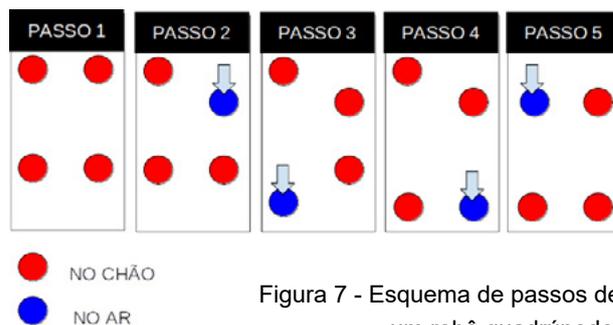


Figura 7 - Esquema de passos de um robô quadrúpede.

Um robô quadrúpede que fez um certo sucesso na internet e serve como referência para o nosso assunto é o Spot, alguns vídeos no Youtube mostram como o processo de equilíbrio e locomoção deste robô são fantásticos. Na figura 8 temos a foto deste robô que segundo a Boston Dynamics foi criado para ajudar nos mais diversos tipos de serviço de vigilância e monitoramento.



Figura 8 - O Spot da Boston Dynamics - Fonte: <https://www.bostondynamics.com/press-release-spot-commercial-launch>

Para os robôs de 6 pernas o ponto de equilíbrio não é um ponto crítico do projeto, porém uma atenção especial ao sincronismo das pernas deve ser tomada. O funcionamento pode ser visto na figura 9, onde as pernas 1 e 3 do lado esquerdo e a perna 2 do lado direito estão se movimentando para a frente, a perna 2 da esquerda deve servir de base juntamente com as pernas 1 e 3 da direita, depois a perna 1 e 3 do lado esquerdo e a perna 2 do lado direito alcançam o destino e se apoiam ao chão, a perna 2 do lado esquerdo e 1 e 3 do lado direito se levantam e avançam para frente, assim sucessivamente. Fazer uma curva é algo um pouco complexo, pois dependendo do lado que for feito a curva, as pernas deste mesmo lado deverão ter uma trajetória menor que a do outro lado. Ou se houver a necessidade de fazer uma curva mais fechada as pernas que estão levantadas deverão fazer uma trajetória oposta a outra, para que rode em um eixo. Para o engenheiro que deseja montar um robô com este sistema de locomoção, é importante estudar os movimentos dos insetos com seis patas.

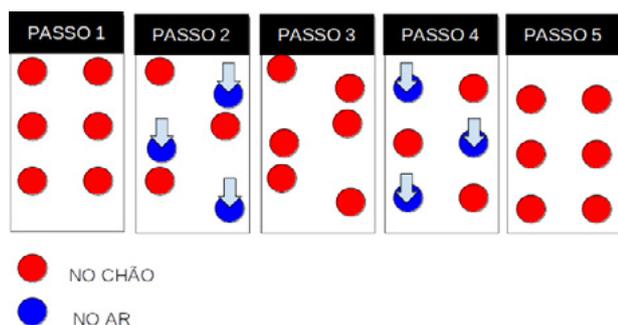


Figura 9 - Esquema da trajetória do robô de seis pernas.

Para aumentar o poder de locomoção de um robô de seis pernas a Boston Dynamics colocou no robô RHex, pernas em formato de meia lua, possibilitando o robô a superar obstáculos de qualquer tamanho que esteja à sua altura, caso tivesse pernas retas, seria preciso calcular a altura que deveria ser ultrapassada e levantar a perna o suficiente para tal. Outro item interessante neste robô é que as pernas do robô estão localizadas no centro da altura, ou seja, se ele porventura vier a ficar de ponta cabeça a sua locomoção não será prejudicada (figura 10).



Figura 10 - Rhex Mug - Híbrido de pernas que giram como rodas. - Fonte: <https://www.defensemedianetwork.com/stories/project-minotaur-project-sand-flea-and-project-rhex/>

ROBÔS ESCALADORES

Nem sempre andar para frente é o caminho, e em alguns casos os robôs precisam subir, e nem sempre possuem canos ou cordas para isso. Para este tipo de desafio, engenheiros e cientistas estão desenvolvendo robôs que escalam. Os sistemas utilizados podem ser através

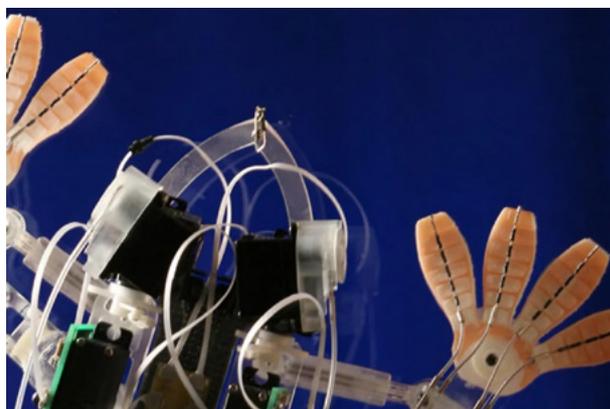


Figura 11 - Robô que imita o movimento das lagartixas. <https://spectrum.ieee.org/gecko-adhesives-moving-from-robot-feet-to-your-walls>

de sucção, como os utilizados pelos polvos, por pequenos ganchos, como os velcros e por contato eletromagnético. Lógico que o método dependerá do tipo de material utilizado na parede. Na figura 11 temos o Stickybot que se baseia na “força de Van der Wall”, o mesmo aplicado pelas lagartixas.

ROBÔS COM SISTEMAS HÍBRIDOS

Quando você precisa criar um robô que terá de andar em todos os tipos de terrenos, quais são os sistemas que você poderia utilizar? Se possível todos, e já foram criados alguns deles que possuem mais de um sistema de locomoção. Um exemplo é o veículo robótico que a Nasa criou para andar sobre Marte, onde ele é composto de pernas e rodas para enfrentar as pedras e o terreno desconhecido daquele planeta (figura 12).



Figura 12 - Pathfinder da missão da Nasa a Marte.
Fonte: http://mars.jpl.nasa.gov/MPF/roverctrlnav/images/fur_corner.jpg

Outro exemplo interessante é um robô criado pela Petrobras para enfrentar terrenos irregulares e ainda andar por rios e pântanos, note que estas rodas são ocas e possuem aletas que servem de remos para impulsão na água. Vale a pena notar que o sistema de direção se dá pelo movimento independente de cada roda (figura 13).



Figura 13 - Robô com sistema híbrido de rodas da Petrobras. Fonte: Petrobras.

Claro que existem casos em que certamente ficaríamos com uma questão em mente: Será que 4 rodas não seria suficiente? Como projetar robôs é uma arte, fica a critério de quem vê e a necessidade de quem o criou. Este é um caso do Robô Barata, onde temos 8 pernas com uma roda espetada nas pontas, quando o terreno é liso, usamos rodas, quando não, usamos as patas (figura 14).



Figura 14 - Robô Barata onde pernas e rodas atuam conforme a necessidade.

Fonte: <https://furo.org/en/works/halluc2x/halluc2x.html>

ROBÔS SALTADORES

Copiando os movimentos dos animais, alguns cientistas e engenheiros imitaram a forma de alguns animais que utilizam os pulos como locomoção. Assim como cangurus, sapos e grilos. Com isso eles tentam pular os obstáculos ao invés de passar por eles. Na figura 15 temos um robô da Festo imitando um canguru.



Figura 15 - Canguru da Festo - Fonte Festo.

CONCLUSÃO

Escolher qual é o melhor sistema de direção para um robô não envolve somente o tipo de terreno que será trilhado, é importante ver qual o sistema que gasta menos energia, qual suporta a carga desejada, qual possui mais precisão e fazer as contas para ter um sistema ideal para aquele determinado robô.

Se comparado com um robô de pernas, um robô com rodas é muito mais rápido, é mais simples de construir e programar, mas nem sempre as rodas são a solução, e é aí que entra a criatividade e conhecimento do engenheiro mecatrônico.

Antes de colocar servos e parafusos, o engenheiro precisa ver os locais e atuação do robô e aí escolher qual é a melhor maneira de locomover.

Além dos sistemas apresentados aqui, faltou falar de robôs que se locomovem na água e no ar, mas isto ficará para outro artigo, pois existem outros sistemas de locomoção para cada um deles.

Espero que este artigo tenha trazido um resumo dos meios de locomoção utilizados em diversos tipos de robôs para que você possa tomar a decisão correta na hora de projetar o seu.

REFERÊNCIAS BIBLIOGRÁFICAS

<http://world.honda.com>

<http://www.engineersedge.com>

<http://www.sintef.no/>

<http://www.sciencedaily.com>

<http://www.armedforces-int.com>

<http://www.bostondynamics.com>

<http://mars.jpl.nasa.gov>

<http://www.labor.unicamp.br>

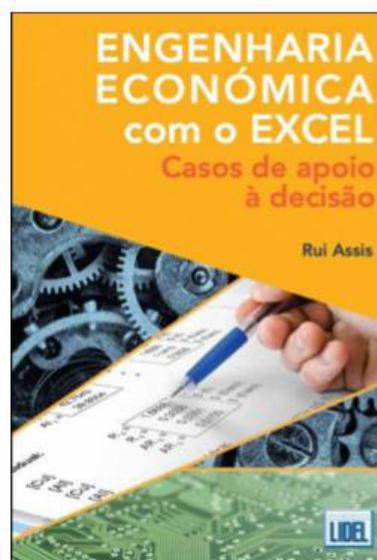
<http://rayandlaura.com>

<https://www.festo.com>

Engenharia Económica com o Excel

Casos de Apoio à Decisão

Autor: Rui Assis





Associação Agentes da Paz - AGEPAZ Paranavaí (foto: Andréia S. Avanci)



**Vander da Silva
Gonçalves**

YouTube: @Vander Lab

MAMUTE CAR

O DESAFIO

No mês abril foi realizado o Primeiro Desafio de Traje-to Mecatrônica Jovem/Vander LAB/ Mamute Eletrônica, figura 1, realizada no CECAP de Paranavaí no Estado do Paraná, que teve transmissão ao vivo através do canal do Instituto Newton C. Braga.

Que teve como principal objetivo, integrar Escolas e Entidades do Município de Paranaíba em uma competição divertida, onde com seus carros, as crianças e adolescentes tiveram que passar por obstáculos e assim cumprir objetivos pré-determinados. A Pista de Trajeto, como os Carros de competições, foram desenvolvidos pelas crianças e adolescentes da AGEPAZ, figura 2, assim colocando em prova seus conhecimentos adquiridos em sala.



Figura 2 - A montagem do Mamute CAR

O PROJETO MAMUTE CAR

As ideias acontecem sempre no primeiro semestre do ano aqui na AGEPAZ, onde damos início às atividades de robótica com as Crianças e Adolescentes.

As primeiras aulas são de Física Eletricidade e montagens com os kits de Robótica. E foi daí que surgiu a ideia de montar um carro de competição, evoluindo assim, para o que temos hoje (figura 3).



Figura 3 - Carrinho de competição desenvolvido pela equipe de Robótica da AGEPAZ (foto: Andréia S. Avanci)

Na Figura 4, podemos ver a pista de competição desenvolvida pela Equipe de Robótica do CECAP, onde aconteceu o Primeiro desafio de Trajeto de Robótica.



Figura 4 - Pista de competição desenvolvida pela equipe de robótica do CECAP.

A MONTAGEM

Para melhor compressão, a montagem se dá através do infográfico (figura 5). Toda a montagem poderá ser vista também no Canal Vander LAB no Youtube. Utilizamos o Kit de Robótica da Modelix para o desenvolvimento do carro.



Figura 5 - Vistoria dos carros com as equipes.

DISCO DE HOCKEY "BOLETA"

Os discos de hockey, que na verdade recebeu o nome de "Boletas" pelas crianças, foram confeccionados na impressora 3D (figura 6).



Figura 6 - As Boletas

A COMPETIÇÃO

O objetivo do desafio era levar o Mamute CAR até um lugar demarcado, capturar a boleta e em seguida levar ao segundo lugar demarco, deixando a boleta neste lugar, e voltar ao lugar de onde os Carrinhos saíram, tudo com um tempo máximo de percurso de 3 minutos, quem fizesse o percurso em menos tempo se classifica.

A competição envolvia 4 Carrinhos (figura 7) simultâneos na pista, onde tivemos corridas memoráveis e muito divertidas, tudo isso nosso leitor poderá acompanhar na íntegra no link e QR-code ao lado.



Figura 7 - Carrinhos na linha de partida .



CIRCUITO ELÉTRICO

Utilizamos o Kit de Robótica da Modelix para o desenvolvimento do carro, e a parte elétrica foram fornecidos pela Mamute Eletrônico, como o Arduino UNO, Módulo HC-05, Driver L298 Mini entre outros.

O aplicativo utilizado para controlar o Mamute CAR foi o Dabble (figura 8). O passo a passo dessa montagem, podemos conferir no canal do Vander LAB.

https://www.youtube.com/@vander_lab

Lista

- 1 - Celular para o aplicativo
- 1 - Arduino UNO
- 1 - HC-05 Bluetooth
- 1 - Drive L298 mini
- 1 - Suporte de 4 pilhas AA
- 1 - Suporte de Bateria 9V
- 8 - Jumper Macho/femea

Peças da Modelix *infográfico na figura 9.

CONHEÇA NOSSAS OUTRAS MONTAGENS?



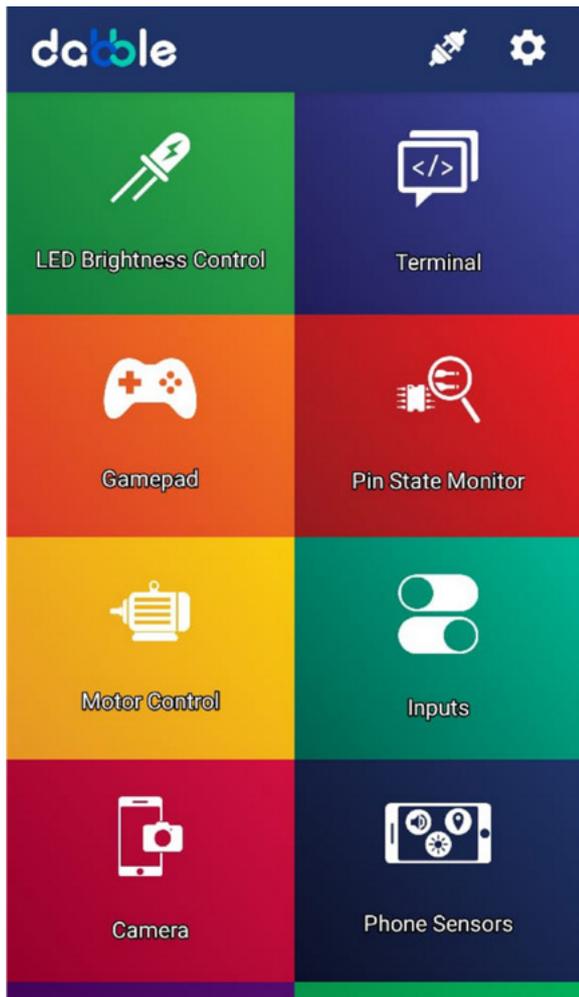


Figura 8 - Aplicativo Dabble

FONTE DE ALIMENTAÇÃO

Cada Mamute Car compõe de um conjunto de pilhas, totalizando 6V (4 Pilhas) para o Drive e uma bateria de 9V para a alimentação do Arduino e Módulo HC-05, um detalhe que devemos levar em consideração são as pilhas e baterias, todas devem ser alcalinas e de boa qualidade.

Equipe de Robótica

Ana Clara Torres Macedo de Brito.

Samuel Lopes da Silva.

Yasmin Eloisa Albarello da Cruz.

Rafael Dias de Souza.

Victoria Chiquinquirá Leon Chacin.

Isabelly Ayka Noguti.

Matheus Ferreira da Cruz.

Douglas Ribeiro Fernandes Dultra.

Rebeca Kamili de Mello Furukawa.

Luan Vermieiro Shuster.

Daniel Serafim Porto da Silva.

Mônica Martins Maciel.

Mariana Vitória da Costa Briscola.

DEPOIMENTO

A Oficina de Robótica na Assistência Social, mais especificamente no Serviço de Convivência e Fortalecimento de vínculos, que tem, como eixos orientadores: “Eu comigo”, “Eu com os outros” e “Eu com a cidade”, contribui imensamente não somente para o desenvolvimento de habilidades, como também, estimula a organização, criatividade, autoconfiança, cooperação, resolução de conflitos, respeito, sociabilidade, sentimento de pertença e tantas outras vantagens.

A Agepaz-Associação Agentes da Paz foi assertiva na escolha desta oficina e também do instrutor que é sem dúvida, um apaixonado pelo que faz, tendo em vista a participação prazerosa das nossas crianças e adolescentes e os resultados alcançados.

A Diretoria, Equipe técnica, crianças e adolescentes da AGEPAZ, agradece ao Vander e patrocinadores esta oportunidade, de ampliarmos a participação social, não só no limite do nosso território, mas sermos vistos pelo mundo.

Silvana Luci Romero

Coordenadora da Agepaz



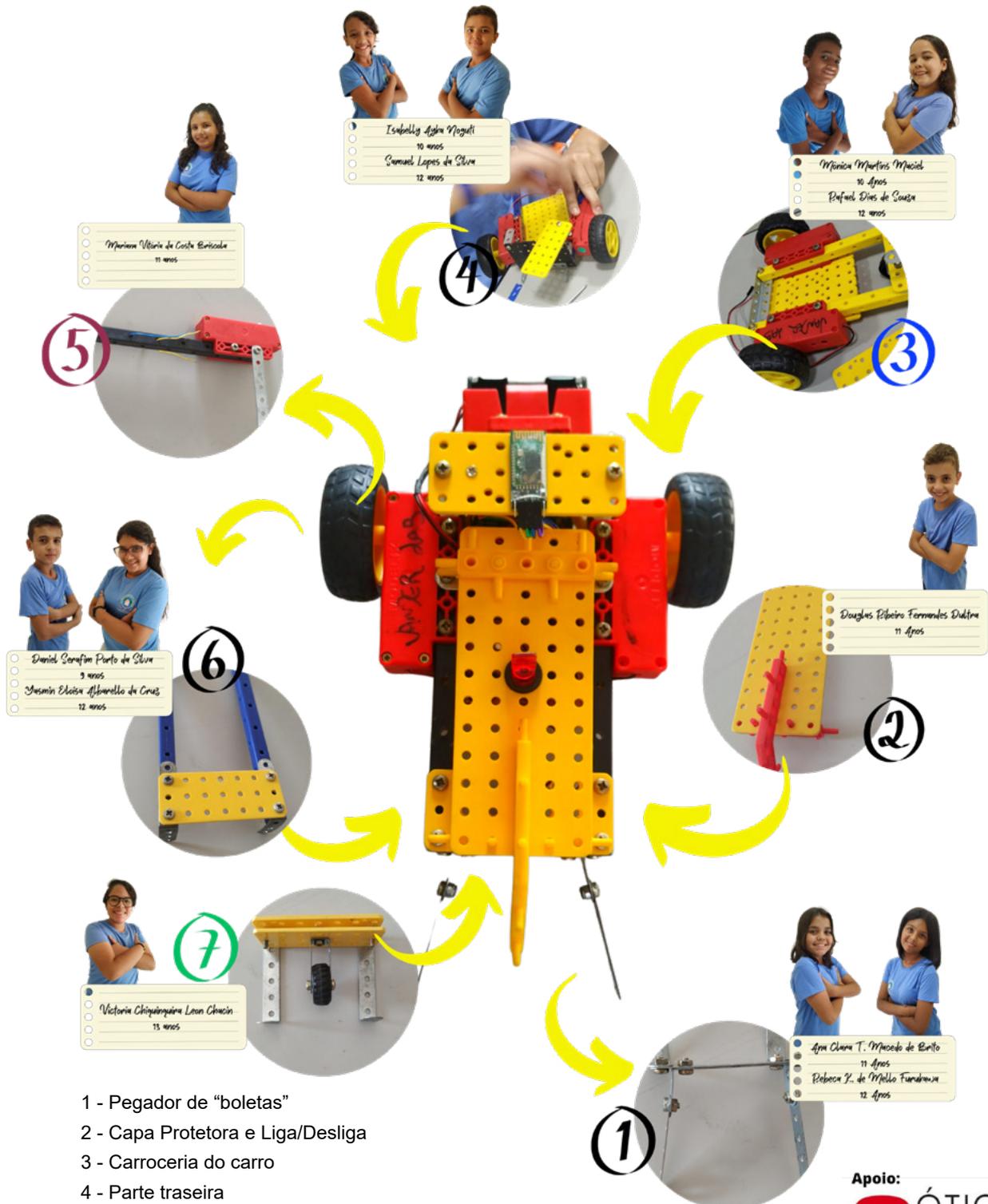
FINALIZANDO

Foi um ótimo desafio, e espero que nosso leitor tenha gostado, deixo vocês com o infográfico (figura 9) e a gente se vê na próxima edição, aquele abraço!

Tivemos como participantes do Evento: Colégio Bonaparte Objetivo de Paranavaí Colégio Estadual de Paranavaí Centro de Atendimento à Criança e ao Adolescente

de Paranavaí – CECAP de Paranavaí Agentes da Paz – AGEPAZ de Paranavaí - Apoio: Mamute Eletrônica – São Paulo SP - INCB – Instituto

Newton C. Braga – São Paulo SP Revista Mecatrônica Jovem – São Paulo SP Vander LAB Channel – Youtube CECAP – Paranavaí Pr. AGEPAZ – Paranavaí Pr. Colégio Bonaparte Objetivo de Paranavaí – Pr. Colégio Estadual de Paranavaí Pr.



- 1 - Pegador de "boletas"
- 2 - Capa Protetora e Liga/Desliga
- 3 - Carroceria do carro
- 4 - Parte traseira
- 5 - Motor
- 6 - Estrutura de sustentação
- 7 - Suporte da Roda Boba

REVISTA

Elevador Brasil

REVISTA PROFISSIONAL - Fabricantes - Conservadoras - Consultores - Construtoras



**A melhor vitrine para o seu
produto na América Latina**

www.elevador.com.br