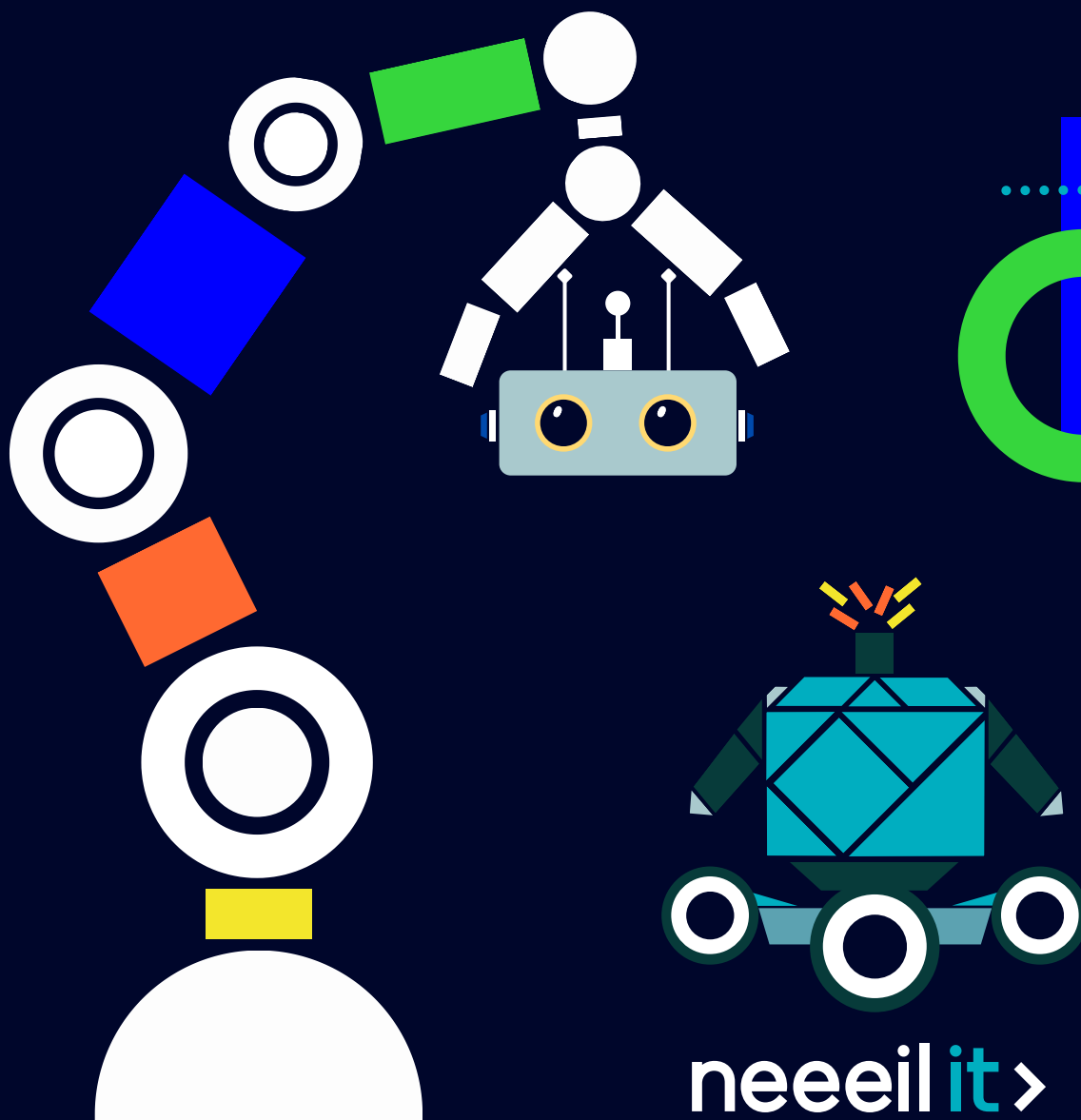


NEEEIL IT '24

Guião da Competição



neeeil it >



Índice

Desafio	4
Conceitos Teóricos	5
Soldadura:	5
Pratear fios:	6
Soldar fio a pino:	6
Retirar solda:	7
Encoders	7
Controlador PID	8
Como usar o PID	9
Algoritmos de resolução de labirintos:	10
Estruturas de Dados	10
Grafo:	10
Stack:	11
Priority queue:	12
Algoritmos:	13
Segue Parede:	13
DFS (Depth First Search):	13
Dijkstra / BFS:	14
A* / Greedy:	14
DFS vs BFS vs Dijkstra vs A*:	15
Material	16
Avaliação	21
Desafio Principal	21
Bónus	22
Pontuação Final	22
Momentos da competição	23
Feedback	23
Testes	23
Submissão	23
Avaliação	23
Procedimentos da Avaliação	24
Regras	24
Penalizações	25
Contactos	26

Desafio

Enfrentando uma crise financeira devido à inflação galopante e aos preços exorbitantes da habitação, Neeelito, um jovem robô da classe operária, viu-se desesperado por uma solução para garantir uma melhor qualidade de vida para si e para sua esposa, Neeelita Al Waifu. Numa decisão desesperada, optou por cometer fraude fiscal, mas logo foi descoberto pela Autoridade Tributária e Aduaneira. Num ato de pânico, o Neeelito utilizou a sua viatura autónoma para escapar aos agentes do Estado.

Na agitação da fuga, a viatura acabou por se perder nas ruas confusas do bairro do Cerco. Agora, o Neeelito encontra-se numa corrida contra o tempo para completar este labirinto urbano antes que lhe roubem o catalisador, a sua última esperança de recuperação financeira.

Então vocês têm como objetivo montar um pequeno carrinho para conseguir resolver um labirinto cujo a sua configuração é desconhecida. O carro ideal é autónomo, não bate nas paredes do labirinto, e segue o caminho mais rápido. Para alcançar a melhor performance terão de:

1. **Soldar uma PCB:** Vocês receberão uma PCB e os respectivos componentes para soldar. Uma bancada estará disponível para essa tarefa.
2. **Montar o Carrinho:** Receberão as peças impressas, os parafusos e as ferramentas necessárias para montar o carrinho.
3. **Programar Carrinho:** Será disponibilizado um código base no GitHub para iniciareis a jornada. A partir daí, vocês desenvolverão o vosso próprio algoritmo para controlar o carrinho.
4. **Mapear o Labirinto:** Criar um mapa do labirinto em formato de gráfico com as intersecções e curvas claras e distâncias bem definidas. Este mapa também

deve servir de auxílio na resolução do labirinto, na medida em que torna possível a descoberta do caminho mais rápido.

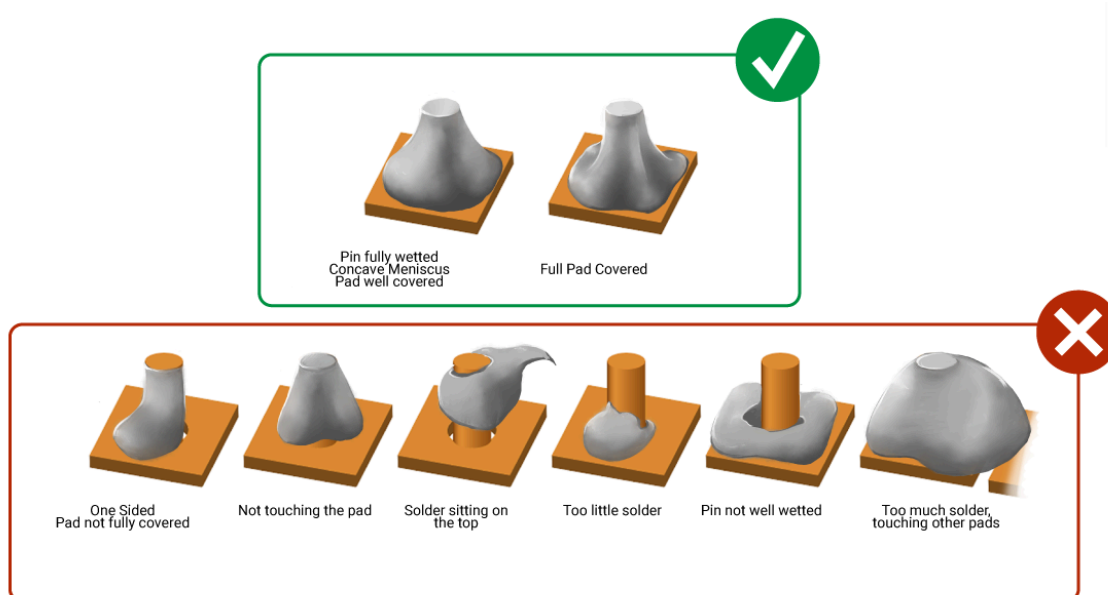
Este género de competições, referidas como “micro-mouse” são muito apreciadas nos Estados Unidos da América e no Japão e a cada década existem inovações quer em termos de hardware como de algoritmia. O seguinte [vídeo do Veritasium](#) ilustra esse desenvolvimento de uma forma sucinta e divertida.



Conceitos Teóricos

Soldadura:

1. Quem não souber ou precisar de ajuda pergunte à staff
2. Deixem o ferro pré-aquecer.
3. Aqueçam o pad de cobre da pcb.
4. Agora podem juntar a solda ao pad.
5. Não abusem da solda mas garantam cobertura da pad.
6. Em baixo têm exemplificadas boas e más soldaduras.



Pratear fios:

A usar os conectores de parafuso devem pratear a ponta dos cabos

1. Enrolar a ponta.
2. Junta solda ao ferro de soldar.
3. Encosta o ferro ao cabo.
4. Juntando mais solda, vai cobrindo a ponta.
5. Deve ficar uma camada fina.



Soldar fio a pino:

No caso de soldar um cabo a um pino na PCB.

1. Pratear o pino.
2. Pratear o cabo.
3. Junte o cabo ao pino e use o ferro para completar a união.

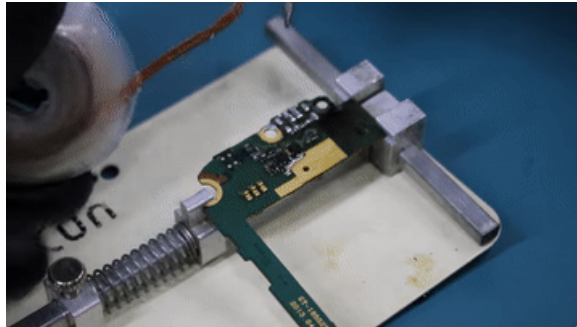


Retirar solda:

No caso de ser preciso retirar solda usa-se a malha de cobre

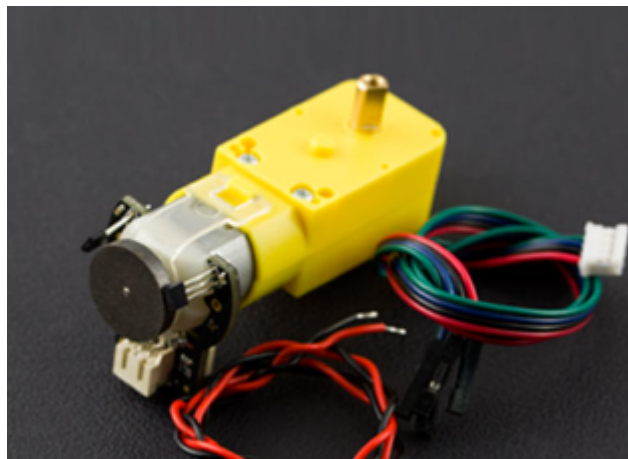
1. Pousa a malha sobre a solda.
2. Usa o ferro de soldar para pressionar e aquecer a malha.
3. Vai se raspando a solda à medida que ela pega na malha.

Se precisares de usar a malha pede a um membro do staff!



Encoders

Os encoders são os componentes que podes ver acoplados ao motor, e servem para ter feedback da posição do eixo do motor. Podes usá-los para saber a posição ou velocidade do motor, o que se vai provar bastante útil. Conforme o eixo do motor roda, o encoder emite pulsos elétricos que podem ser lidos por um microcontrolador, de onde podes concluir a posição. Têm um exemplo de código no [Git](#).



Vídeo explicativo: <https://www.youtube.com/watch?v=oLBYHbLO8W0>

Controlador PID

Quando se pretende ajustar a velocidade de um motor para atingir um valor desejado, é necessário recorrer a técnicas de controlo. Uma abordagem comum e eficaz é a utilização de um controlador PID. O processo inicia-se com o cálculo do erro de velocidade, que consiste na diferença entre a velocidade desejada e a velocidade medida pelo encoder. Em seguida, converte-se o erro numa variável conhecida como PWM, que controla a potência fornecida ao motor, ajustando assim a sua velocidade para o valor desejado.

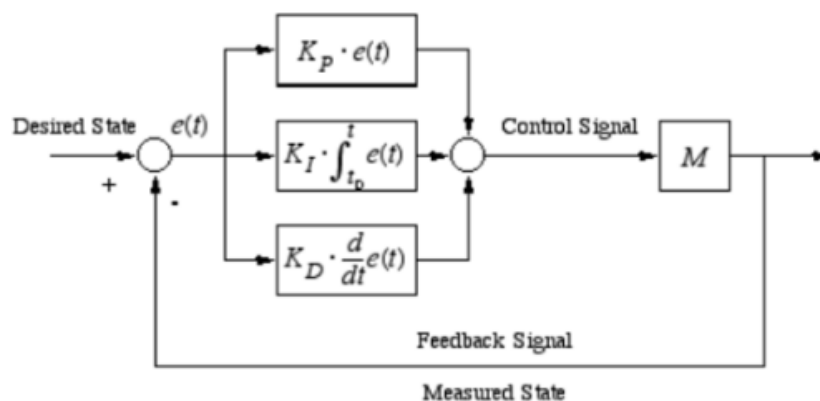
Neste controlador, essa transformação é feita recorrendo a 3 componentes: (Assumindo $e(t)$ como o erro)

- Uma proporcional ao erro: $K_p \cdot e(t)$
- Uma proporcional ao integral do erro: $K_i \cdot \int e(t)$
- Uma proporcional à derivada do erro: $K_d \cdot \frac{d}{dt} e(t)$

Eis um método para definir os valores dos ganhos do controlador (K_p , K_i , K_d):

- Começar com um K_p baixo e os restantes a zero;
- Aumentar a resposta proporcional ao erro gradualmente, até que o motor responda de forma adequada;
- O K_i é a constante que vai “acumulando o erro”, pelo que te permite lentamente ajustar para anular o erro permanente
- O K_d responde à variação do erro. Esta parcela é negativa, permitindo estabilizar o teu controlador. (atenuar grandes variações);

PID CONTROL:

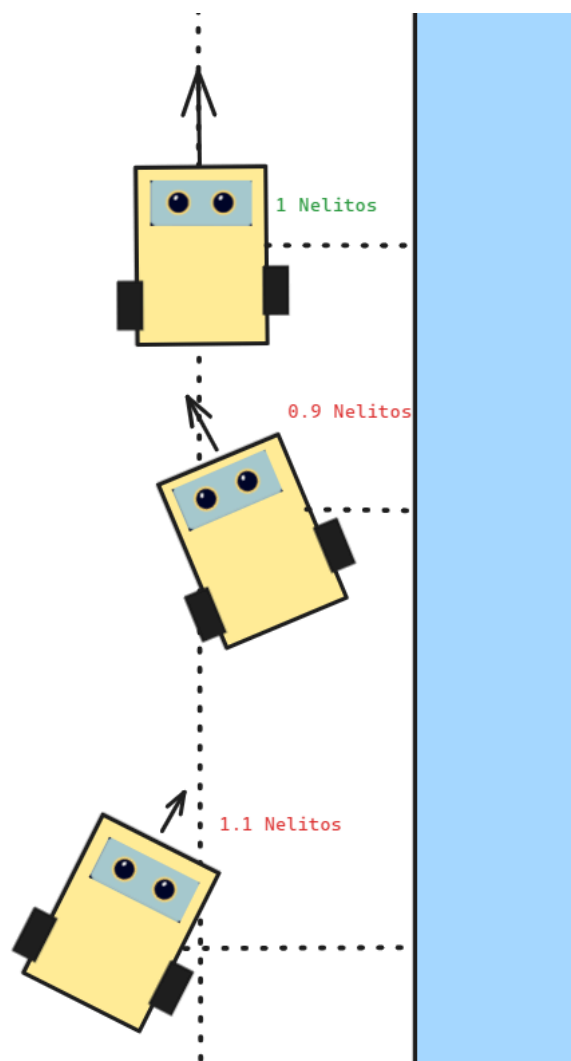


Vídeo explicativo: <https://www.youtube.com/watch?v=dTGITLnYAY0>

Como usar o PID

Para além de ser possível controlar a velocidade das rodas, o PID também consegue controlar a posição. Por exemplo, caso o objetivo seja que o carro do Neeeilito rode exatamente 90° , pode-se calcular quantas rotações do encoder correspondem a uma rotação da roda completa e utilizar essa informação para girar o carro para uma posição específica.

Outra aplicação é utilizar um sensor de distância como entrada para manter a distância entre o robô e uma parede constante, permitindo que o robô acompanhe a parede num movimento retilíneo. Isso pode ser realizado ajustando a velocidade das rodas ou a direção do robô com base nas leituras do sensor de distância. Neste caso, o erro corresponde à diferença entre a distância à parede desejada e a distância inferida através do sonar.



Vídeo explicativo: <https://www.youtube.com/watch?v=4Y7zG48uHRo>

Algoritmos de resolução de labirintos:

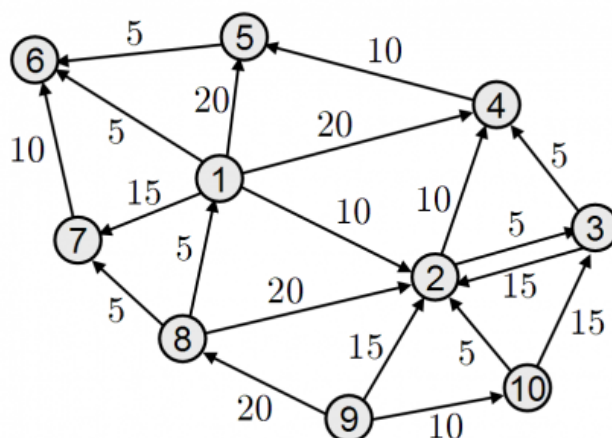
Nesta secção do guião, iremos aprofundar o conhecimento sobre estruturas de dados, algoritmos de mapeamento e pesquisa.

Estruturas de Dados

Escolher uma estrutura de dados é fundamental para implementar com eficácia os algoritmos de resolução de labirintos.

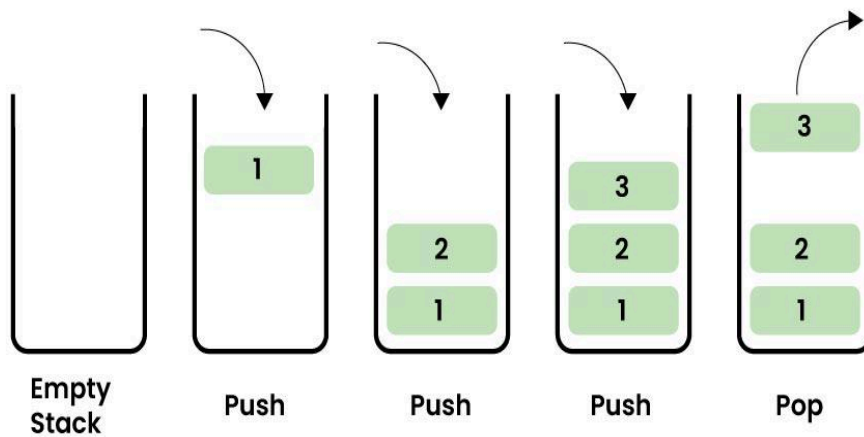
Grafo:

Um grafo é uma estrutura de dados essencial na resolução de labirintos, pois permite representar a estrutura do labirinto de uma forma clara e eficiente. Nele, cada posição é representada por um nó, enquanto que os caminhos possíveis de uma posição até outra são arestas. Analogamente, podemos pensar num mapa como um grafo. Assim como no mapa, onde as localidades são conectadas por estradas, no labirinto os nós e são ligados por arestas, onde o peso atribuído às arestas é a distância entre as posições.



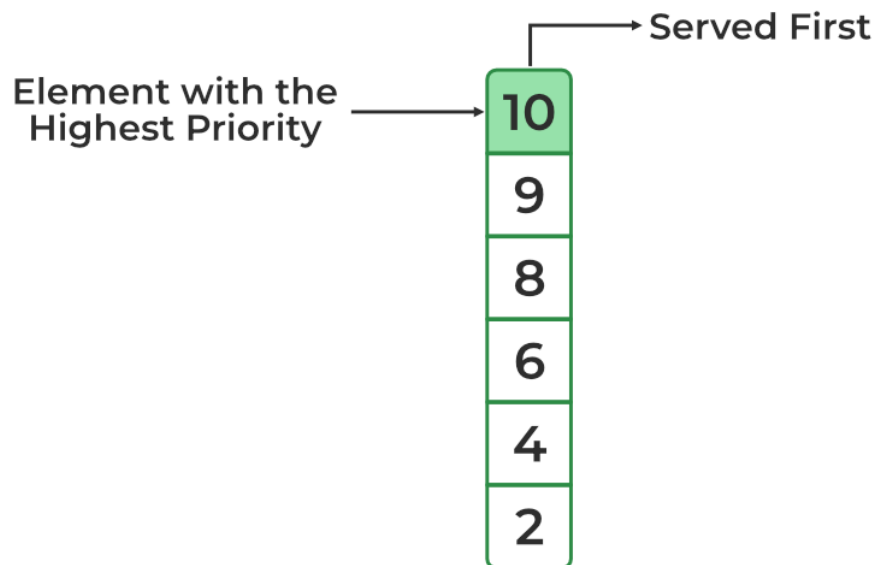
Stack:

Uma stack (LIFO) pode ser vista como um jarro de bolachas. Quando enchemos o jarro de bolachas criamos camadas em que a mais recente se encontra na camada mais superficial. Para chegar às bolachas do fundo do jarro temos que tirar as bolachas da camada mais recente. Isto significa que a última bolacha a entrar é a primeira bolacha a sair (Last In, First Out). O funcionamento de uma stack é semelhante. Logo, uma Stack é uma boa maneira de guardar os nós que o carrinho visitou e que ainda não foram completamente explorados.



Priority queue:

Uma priority queue é uma estrutura de dados que organiza elementos com base em uma prioridade associada a cada elemento. No fundo, é uma fila de elementos organizados de uma certa forma, como por exemplo, uma lista de nomes organizados por ordem alfabética, ou no contexto de um labirinto, uma lista de caminhos não explorados por uma ordem de o caminho ser promissor ou não.



Algoritmos:

Existem inúmeros algoritmos de pesquisa, embora uns sejam computacionalmente mais eficientes do que outros, o que não implica que tenham melhor performance. Existem vários algoritmos que saltam entre nós não explorados de modo que o número de procuras seja menor, mas se o robô fosse a seguir essa linha de pensamento, o seu controlo seria mais complexo.

Alguns algoritmos comuns são:

Segue Parede:

Este método envolve manter-se adjacente a uma parede enquanto explora o labirinto. É um método que funciona para todos os labirintos com entrada e saída na parede exterior. Não é necessário desenvolver uma estrutura de dados para procurar nós não explorados. Embora este método seja simples, a sua eficácia depende da estrutura do labirinto e da posição final.

Vídeo explicativo: https://youtu.be/CB90u22WO8c?si=yjq4c2H88qk_RJK1

DFS (Depth First Search):

DFS explora o labirinto seguindo um ramo até ao fim antes de retornar. É útil para mapear o labirinto e identificar áreas inexploradas. Embora o raciocínio seja relativamente parecido ao segue parede, este guarda nós. Assim podes saber quais nós ainda não foram explorados, o que facilita o facto de podermos explorar caminhos que não são adjacentes a uma parede. Isto pode ser vital caso seja necessário alcançar uma localização em ilha (não conectada a nenhuma parede)

Vídeo explicativo: <https://youtu.be/JFkDh5BSens?si=AtABWAvL9Zu0XN9m>

Dijkstra / BFS:

Dijkstra e BFS são métodos relativamente semelhantes, mas com propósitos diferentes.

BFS tem como principal objetivo explorar todos os nós de um grafo a partir de um nó inicial, visitando todos os nós adjacentes antes de explorar os nós mais profundos. O seu método de procura pode ser visto quase como uma circunferência com centro no ponto inicial a aumentar o raio. Dentro de um espaço confinado age como um fluido que se expande para todos os caminhos ao mesmo tempo.

Vídeo explicativo: https://youtu.be/x-VTfcmrLEQ?si=FEHxULvR7Ho2qHZ_

Dijkstra tem como objetivo encontrar o caminho mais curto entre um nó inicial e todos os outros nós no grafo ponderado, ou seja, onde a distância importa. O método de procura é semelhante ao do BFS, porém diferencia-se ao ter em consideração o custo dos caminhos. Em vez de se expandir em todas as direções, expande-se na direção cujo o custo é menor.

Vídeo explicativo: <https://youtu.be/xU2Z5C4uWp8?si=bPtXhxlYL-3Tn8bl>

A* / Greedy:






No caso do greedy, conhece-se o ponto inicial e o ponto final. Considere-se a linha reta (e abstrata) que une estes dois pontos. O mais provável é que o caminho mais curto siga a direção dessa reta. Assim, o algoritmo Greedy, age tal como um fluído, semelhante a BFS, só que num plano inclinado para o ponto final. Explora primeiro os caminhos que se aproximam do ponto final. A “inclinação”, calculada por uma heurística (distância cartesiana entre dois pontos ou distância Manhattan), pode ou não retornar o caminho mais curto.

O A Star (ou A*), considera tanto o caminho mais curto segundo Dijkstra e o mais simples de percorrer segundo Greedy. Deste modo, o A* surge de forma a combinar os processos. É uma expansão do algoritmo de Dijkstra ao qual lhe é adicionado um parâmetro heurístico (peso_da_ligação + heuristic _do_ ponto) que força a escolha de um caminho que aponta na direção da saída, evitando trajetórias opostas.

Vídeo explicativo: <https://youtu.be/19h1g22hby8?si=2U1A1eLZnRZSDhxs>

DFS vs BFS vs Dijkstra vs A*: <https://youtu.be/GC-nBgi9r0U?si=pvV3HhwKQfK5FWt4>

Material





Materiais	Unidades	Datasheet
Motor DC c/ decoder	2	
Pilha-li-ion	2	
Suporte de Pilhas	1	
Sensor Ultrassom	3	
Sensor Infravermelho	4	
Caster Wheel	1	
Rodas (diametro 65 mm)	2	
Raspberry pi pico	1	
Cabo micro-USB	1	
PCB	1	
Parafusos/anilhas/porcas M3x30	4	
Parafusos/anilhas/porcas M3x12	10	
Parafusos/anilhas/porcas M3x10	3	
Base inferior	1	
Base superior	1	
Pilar traseiro	2	
Pilar central	2	
Suporte sonar	3	
Chave estrela M3	1	


extensão de 4/+	1	
Alicate de pontas	1	
Braçadeiras	3	
Jumpers F/F	15	
Velcro	1	

Existirá uma estação com ferros de soldar em cada sala de competição, onde as equipas poderão soldar os seus componentes.

Material para a PCB

Na placa cada componente tem uma referência - texto a branco junto do local do componente - que ajuda na sua identificação. Na tabela seguinte está listado o material por placa

Referência	Componente/Valor	Datasheet	Quantidade total
R1, R2	10k Ω		x2
R3	560 Ω		x1
R4	330 Ω		x1
C11	22uF		x1
C12	33uF		x1
Q1	BC557		x1
D1, D2, D3, D4, D5, D6, D7, D8	MUR120		x8
D9	STPS1150		x1
U1	BA50DD0WT		x1

U2	1N4735		x1
U3	1N4733		x1
U4	L293DNE; Socket DIP-16		x1 IC x1 socket
MOTOR1, MOTOR2, POWER	Terminal de parafuso de 2 pinos		x3
Headers (GPx, 5V, GND)	6 pinos (macho) 14 pinos (macho) 20 pinos (fêmea)		x2 x1 x2

Antes de começarem a soldar, garantam que o vosso kit tem todos os componentes necessários. O driver de motores L293DNE e o regulador de tensão BA50DD0WT só serão fornecidos depois de já terem o resto dos componentes soldados. Peçam a um membro do staff quando precisarem. Perguntem a um membro do staff caso tenham alguma dúvida sobre um componente antes de soldar!

ATENÇÃO - AVISOS IMPORTANTES QUANTO À MONTAGEM DA PCB:

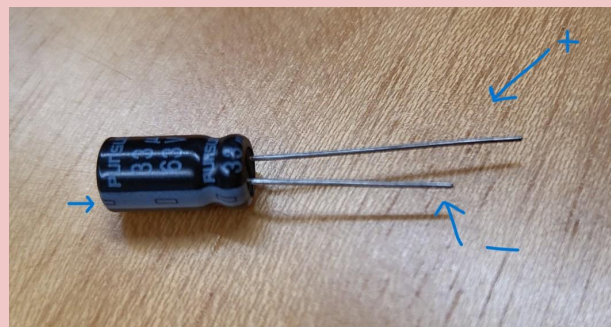
Não devem soldar a PICO W diretamente aos terminais da PCB. No kit estão incluídos dois barramentos de 20 pinos header fêmea para serem soldados na PCB para poderem depois encaixar e desencaixar a PICO W.

Não liguem o cabo USB à PICO W enquanto as pilhas estão conectadas - enquanto o interruptor está ligado. PODEM DANIFICAR O VOSSO COMPUTADOR. O interruptor está desligado se estiver a apontar para dentro da placa.

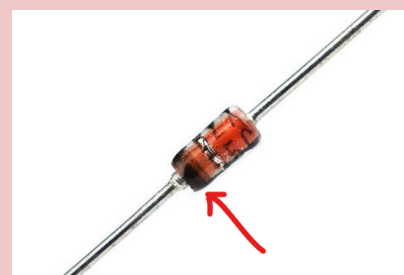
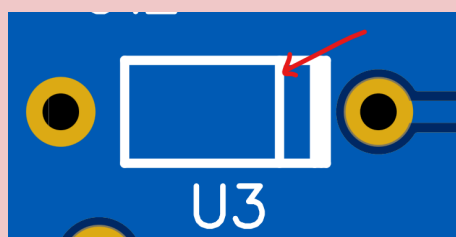


Cuidado com a polaridade dos componentes.

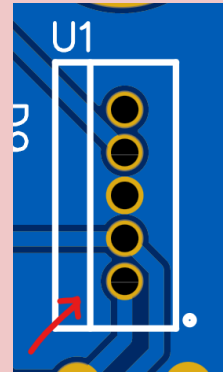
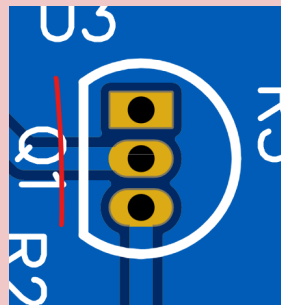
- Verificar no **condensador** de que lado está desenhado o sinal menos “-”.



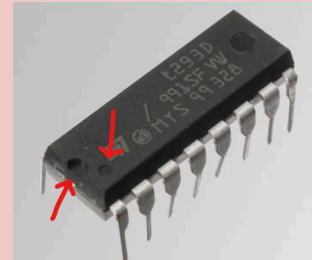
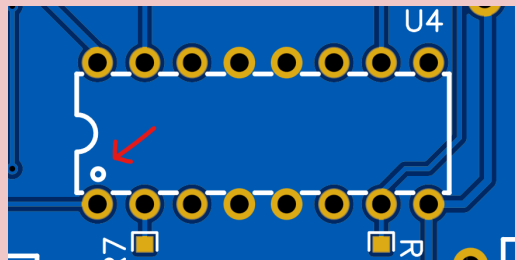
- Verificar nos **díodos** em qual dos terminais está desenhado o risco que indica o cátodo - basta alinhar pelo desenho na placa.



- Verificar a orientação do **transístor** e do **regulador de tensão**. Devem alinhar o transístor pelo desenho - atenção à parte reta - e a heatsink do regulador de tensão - a parte mais alta - pelo risco do desenho da placa.



- Não soldar diretamente o L293DNE à placa. Soldar a socket incluída no kit primeiro. Ao montar o L293DNE verificar que a orientação é a correta - olhar para a bolinha no canto do IC ou semicírculo.



O esquemático da PCB e uma foto do seu modelo 3D estão no link na tabela do material.

Usando esta placa, alguns dos pinos necessários para controlar a driver de motores L293DNE já estão conectados diretamente aos GPIO's da PICO W. Utilizem a lista seguinte para saber quais usar no vosso código.

Pino	GPIO da PICO W
L293DNE - IN1 (Motor1)	GP27
L293DNE - IN2 (Motor1)	GP22
L293DNE - IN3 (Motor2)	GP26
L293DNE - IN4 (Motor2)	GP28
L293DNE - Enable 1	GP15
L293DNE - Enable 2	GP14

Os restantes pinos (GP0 - GP13) são compatíveis com 5V e podem ser usados livremente.

Avaliação

Desafio Principal

O desafio principal consiste nos seguintes parâmetros:

- Navegação - 40 pts
- Melhor tempo de resolução do labirinto (tempo necessário para ir da entrada à saída do labirinto) - 20 pts
- Apresentação - 20 pts

No parâmetro **Navegação** será avaliado o seguinte:

- Qualidade do movimento em linha reta - 5 pts
- Qualidade do movimento nas curvas - 10 pts
- Percentagem do labirinto explorado - 20 pts
- Zona onde o robot finaliza a sua prova - 5 pts

No parâmetro **melhor tempo**, será considerado o melhor tempo de resolução do labirinto, ou seja, o tempo da melhor tentativa no momento de apresentação. E para o efeito de atribuição de pontos será tomado como base o melhor tempo de todas as equipas e deduzir as pontuações das restantes equipas a partir dessa base. Sendo que a primeira equipa terá a pontuação completa (20 pontos) e a última equipa terá zero pontos.

No parâmetro **Apresentação**, será avaliada a capacidade de cada equipa de explicar o método de resolução do desafio. É crucial que a apresentação seja realizada de forma clara e fluente, transmitindo de maneira eficaz os passos seguidos para resolver o labirinto e mapeá-lo, assim como o raciocínio por trás da escolha do caminho mais curto.



Durante os 8 minutos de avaliação, cada equipa deve gerir o seu tempo como considerar melhor, podendo realizar diversas tentativas de resolver ou mapear o labirinto. Recordamos que o código não pode ser alterado após o fim do desafio, logo, caso queiram comutar entre os algoritmos de navegação devem fazê-lo usando inputs, como por exemplo comandos enviados pelo computador.

Bónus

Alguns critérios que irão ser tomados como bónus serão:

- Mapeamento (Fazer uma representação gráfica do labirinto mapeado pelo carrinho) - 15 pts
- Estética - 5 pts

Pontuação Final

A pontuação final (PF) de cada equipa será calculada com base na pontuação obtida em cada parâmetro, levando em consideração os pesos atribuídos a cada subcategoria.

$$PF = NAVEGAÇÃO + MELHOR TEMPO + APRESENTAÇÕES BÓNUS + PENALIZAÇÕES *$$

A pontuação determinará o desempenho geral da equipa no desafio, e as equipas com as maiores pontuações serão selecionadas para avançar para a final (mais informações na próxima secção)..

*Consultar secção das penalizações mais à frente



Momentos da competição

Feedback

Estão alocados momentos de feedback ao longo da competição para poderem partilhar com os técnicos o vosso avanço e poderem receber uma análise mais detalhada sobre as resoluções dos problemas que estiveram a trabalhar. É fortemente aconselhável o uso destes momentos como ferramentas facilitadoras de problemas encontrados. Estas sessões terão lugar às 22h00 e às 8h00 e terão uma duração de 7 minutos por equipa.

Testes

Cada sala terá um exemplo de labirinto de dimensões menores que o final. Poderão testar o protótipo mediante uma marcação prévia com um membro do staff. Apenas é permitida uma marcação em simultâneo por equipa, isto é, cada equipa apenas pode reservar um slot de cada vez. Cada teste tem uma duração máxima de 20 minutos.

Submissão

É necessário submeter o código no chat da vossa equipa no discord, para que este seja avaliado na deliberação. O código tem que ser submetido até ao final da competição (17h00 de Domingo) e não pode ser editado após esta hora. Caso não cumpram este requisito deixa de ser admissível para avaliação, sendo desclassificados. O protótipo deve ser deixado em cima da banca da respetiva equipa até ao momento da apresentação.

Avaliação

Durante esta fase da competição, cada equipa irá colocar os seus carrinhos a resolver o labirinto, executando tarefas de mapeamento e resolução em tempo real. Enquanto os carrinhos estão a resolver o labirinto, os membros da equipa apresentarão o seu trabalho perante o júri.

Procedimentos da Avaliação

Cada equipa dispõe de 8 minutos para resolver o desafio. Durante esse período, espera-se que consigam:

- Resolver o labirinto
- Mapear o labirinto
- Resolver o labirinto seguindo o caminho mais rápido
- Mostrar um gráfico do labirinto

Enquanto o labirinto está a ser resolvido, devem também realizar a apresentação.

A avaliação ocorre num formato de torneio, permitindo que a avaliação seja realizada em paralelo, com duas equipas a serem avaliadas ao mesmo tempo em labirintos iguais.

Todas as equipas serão divididas em dois grupos. Em cada grupo, o Top 3 será selecionado pelo respectivo júri para competir na final. As equipas selecionadas para a final não precisam de realizar o teste novamente, uma vez que as provas serão gravadas para uma análise mais detalhada.

Regras

Quebrar estas regras pode resultar na desqualificação da equipa, perda de pontos, ou perda da caução, de acordo com a decisão da organização ou do júri.

1. Não podem sair da FEUP durante a competição sem autorização da Organização do Evento e devidamente justificado. Se precisares de sair da FEUP fala com um membro do STAFF;
2. Apenas podes trabalhar com a tua equipa. Não podem ajudar membros das outras equipas e as equipas não podem trabalhar em conjunto;
3. Apenas podem utilizar ferramentas e material fornecido pela organização. O uso de outras ferramentas ou materiais é proibido;

4. A equipa não pode fazer nenhuma modificação ao carro ou ao código após o fim das 24 horas de competição;
5. As ferramentas fornecidas não podem ser danificadas e terão que ser deixadas na área de trabalho da equipa no final da competição;
6. O código submetido pela equipa tem que ser da sua autoria;



Penalizações

Cada equipa terá a possibilidade de requisitar material extra ao que inicialmente é disponibilizado (ponto 7), porém dependendo do material pedido haverá penalizações de pontos, nomeadamente:

- Componente - Ponte H (-3 pontos);
- Componente - Regulador de tensão (-3 pontos);
- Sonar (-3 pontos);
- PCB (-15 pontos);
- Peças do carro (-1 a -5 pontos) - para saber o que está disponível e a penalização correspondente falar com o STAFF do evento;
- Pilhas (-10 pontos);
- Rodas (-3 pontos);
- Caster wheel (-3 pontos);
- Suporte para pilhas (-3 pontos)
- Motor DC (-10 pontos);
- Parafusos/anilhas/porcas (0 pontos);
- Infravermelho (0 pontos);
- Braçadeiras (0 pontos);
- Velcro (0 pontos);
- Raspberry Pico W (-15 pontos);

O material que não é referido nesta lista não terá reposição. A lista acima deve ser usada somente para reposição de material danificado, ou seja, o material utilizado no protótipo final não deve exceder o inicialmente fornecido.

Contactos

Preferencial, o contacto é feito por discord via canal da equipa pingando membros do staff. Em caso de problemas relacionados com o material ou dificuldade com o desafio, pingar o role **@STAFF_TEC** no canal da equipa.

Motivos **muito** urgentes contactar:

- Rafael Macedo: 939817320
- António Lopes: 912345966

Outros contactos

- Número interno de emergência: 912233377
- Batalhão de Sapadores Bombeiros: 22 507 3700
- Bombeiros Voluntários Portuenses: 22615 1800
- PSP (Bom Pastor) Police - PSP: 22 557 4900
- GNR:: 22 339 9600

Boa Sorte!