**FUNDAÇÃO EDUCACIONAL INACIANA "PE. SABÓIA DE MEDEIROS” (FEI)**

Felipe Orlando Lanzara

João Vitor Governatore

**PROJETO DE ROBÓTICA**

Relatório sobre o projeto Webots

São Bernardo do Campo

2025

# Introdução

O objetivo deste projeto é desenvolver, em C, um controlador para o robô e-puck no simulador Webots, capaz de patrulhar automaticamente um ambiente composto por várias caixas dispostas em grade, das quais apenas uma se movimenta. Ao identificar essa caixa em movimento, o e-puck interrompe seu deslocamento e passa a girar sobre o próprio eixo, sinalizando a detecção do objeto deslocado.

Abaixo está a imagem do mapa, mostrando as caixas e o robô em sua posição inicial:

Uma imagem contendo Forma

O conteúdo gerado por IA pode estar incorreto.

# Inicializações e Configuração de LEDs

## Inicialização

Configura o controlador do Webots e a semente aleatória para decisões de giro.



## Configuração de LEDs

Armazena em leds[] as tags dos 8 LEDs do e-puck, usados depois para sinalizar a detecção de caixa deslocada.

Texto

O conteúdo gerado por IA pode estar incorreto.

# Movimentação do Robô

## Detecção de Obstáculos

A cada ciclo de simulação, o controlador varre todos os sensores de proximidade (ps0 a ps7) e, sempre que qualquer leitura ultrapassa o limiar definido por OBST\_THRESHOLD, a variável *obstacle* é imediatamente setada para 1 e o laço interrompido. Enquanto *obstacle* permanecer 0, o caminho é considerado livre de colisões e o e-puck prossegue sem alterar sua trajetória.

Texto

O conteúdo gerado por IA pode estar incorreto.

## Vagueio em Linha Reta

Sempre que nenhum obstáculo é detectado, ambos os motores recebem a mesma velocidade máxima (MAX\_VELOCITY), fazendo com que o robô avance em linha reta. Esse comportamento contínuo garante que o e-puck percorra de forma uniforme os corredores formados pelas caixas, mantendo uma exploração sistemática do ambiente até encontrar um obstáculo.

Tela preta com letras brancas

O conteúdo gerado por IA pode estar incorreto.

## Giro Aleatório e Fase de “Escape”

Texto

O conteúdo gerado por IA pode estar incorreto.Ao identificar um obstáculo, o e-puck escolhe aleatoriamente uma direção de giro e mantém esse movimento por um tempo também randômico, garantindo que o padrão de evasão nunca seja previsível. Logo em seguida, executa uma breve fase de “escape” avançando por 0,2 segundos, o que evita que ele fique preso no próprio obstáculo e retoma automaticamente o vagueio em linha reta. Essa combinação de giro estocástico e fuga rápida cria trajetórias variadas, fazendo com que o tempo até encontrar a caixa móvel seja diferente a cada execução.

# Supervisor e Detecção de Caixa Movimentada

## Utilização do Supervisor para Leitura das Posições

Na primeira etapa o controlador recorre ao nó Supervisor do Webots para obter, em tempo de execução, referências a cada caixa definida em cena (DEF “CAIXA00” a “CAIXA17”). Com *wb\_supervisor\_node\_get\_from\_def()* armazenamos essas referências em um vetor, e a cada ciclo de simulação usamos *wb\_supervisor\_node\_get\_position()* para ler as coordenadas atuais x, y, z de cada caixa diretamente do mundo virtual, sem interromper o comportamento reativo de navegação do robô.

Tela de computador com texto preto sobre fundo branco

O conteúdo gerado por IA pode estar incorreto.

## Estrutura de Dados para Armazenar as Posições Originais/Iniciais

Na segunda etapa definimos uma *struct* *PosicaoCaixas* com três campos — x, y e z, e criamos um array *originais[]* capaz de guardar as posições iniciais de cada uma das 18 caixas. Logo após carregar todas as referências, lemos uma vez *pos[0]*, *pos[1]*, *pos[2]* de cada nó e as atribuímos a *originais[i]*, garantindo um registro imutável do estado inicial do ambiente.

Texto

O conteúdo gerado por IA pode estar incorreto.Texto

O conteúdo gerado por IA pode estar incorreto.

## Detecção de Movimento Comparando com o Valor Original

Na terceira etapa, a cada passo de simulação voltamos a chamar *wb\_supervisor\_node\_get\_position()* e calculamos o desvio absoluto (fabs) entre a posição atual e a original em cada eixo. Utilizando um pequeno epsilon (1e-6) para filtrar ruídos numéricos, disparamos o alerta quando qualquer uma dessas diferenças ultrapasse o limite, sinalizando que a caixa foi efetivamente movimentada.

Quando o e-puck encosta na caixa que foi deslocada, seu controlador imediatamente zera as velocidades dos dois motores para parar o deslocamento, acende todos os LEDs do chassi como sinal visual de alerta e passa a girar continuamente sobre o próprio eixo, indicando que identificou a movimentação não autorizada da caixa.

Texto

O conteúdo gerado por IA pode estar incorreto.

# Considerações Finais

## Variabilidade no Tempo de Detecção

Por se tratar de um procedimento de exploração puramente aleatória, o intervalo até que o e-puck encontre a caixa móvel pode oscilar amplamente entre diferentes execuções. Em alguns casos o robô encontra o objeto deslocado em poucos minutos, enquanto em outros percorre grande parte do mapa antes de acioná-lo.

## CAIXA00 não Utilizada

No mapa de teste disponibilizado, a referência CAIXA00 foi mantida apenas para preservar a sequência dos identificadores, mas não está presente como obstáculo físico. Todas as verificações ocorrem sobre as caixas numeradas de 01 a 17.

## Vídeo Demonstrativo e Repositório do GitHub

Para acompanhar o sistema em funcionamento, assista ao vídeo demonstrativo em: <https://www.youtube.com/watch?v=tNnDUJCGiAY>.

Além disso, é possível consultar o código-fonte completo no GitHub: <https://github.com/jvgoverna/CC7711-Inteligencia-Artificial-e-Robotica/tree/main/ProjetoIA>