Documentação Técnica do Projeto NAO - Desvio de Obstáculos Reativo

Disciplina: Robótica Sensorial e Controle

Data: 04 de novembro de 2025

Alunos: José Mateus Francisco

Yuta Katagiri

Leandro Aparecido

Felipe Teixeira

---

1. Visão Geral da Solução Proposta

Este documento descreve a implementação de um sistema de controle reativo para o robô humanoide **NAO** no ambiente de simulação **CoppeliaSim (V-REP)**. O principal objetivo é permitir que o robô navegue e desvie de obstáculos (representados por pilares cilíndricos) em tempo real, utilizando uma lógica de decisão simples baseada em prioridade (Arquitetura Reativa de Baixo Nível).

A solução não depende de planejamento de rotas global ou mapeamento, focando em uma resposta imediata às informações sensoriais. O controle é implementado através de um *script* em linguagem Lua, acoplado ao objeto raiz do NAO.

2. Especificação dos Sensores Utilizados

Para o desvio de obstáculos, foram adicionados e utilizados **três sensores de proximidade** (assumidos como sensores infravermelhos ou ultrassônicos emularam no simulador) montados na parte frontal do robô.

Nome do Objeto (Script)	**Posição no Robô**	**Finalidade**	**Tipo de Sensor (Simulação)**
/NAO/proximitySensor1	**Direita** (Frontal-Lateral)	Detectar obstáculos à direita.	Proximidade (Cônico/Direcional)
/NAO/proximitySensor2	**Esquerda** (Frontal-Lateral)	Detectar obstáculos à esquerda.	Proximidade (Cônico/Direcional)
/NAO/proximitySensor3	**Central** (Frontal)	Detectar obstáculos diretamente à frente.	Proximidade (Cônico/Direcional)

Funcionamento dos Sensores:

A função sim.readProximitySensor() retorna um resultado booleano (res) que indica se algo foi detectado e uma distância (dist). O controle utiliza o resultado booleano (res > 0) para tomar decisões imediatas.

3. Especificação do Script de Controle (Lua)

O controle é encapsulado em um *script* não *thread* acoplado ao objeto /NAO e utiliza as *callbacks* de sistema do CoppeliaSim.

3.1. Variáveis e Inicialização (sysCall_init)

- **Parâmetros de Movimento:**
    - VELOCIDADE_AVANCO = 0.01: Define o deslocamento linear para a função moverParaFrente().
    - VELOCIDADE_GIRO = 0.05: Define o deslocamento angular (em radianos) para a função girar().
- **Mapeamento:** Os objetos do robô (/NAO) e os sensores são obtidos e mapeados para variáveis locais.

3.2. Funções de Atuadores (Movimento)

- moverParaFrente(): Calcula a nova posição do NAO (X e Y) aplicando VELOCIDADE_AVANCO na direção determinada pela orientação atual do robô.
- girar(direcao): Atualiza a orientação do NAO, girando no sentido horário (direcao = -1) ou anti-horário (direcao = 1) pelo valor de VELOCIDADE_GIRO.

3.3. Lógica de Controle Reativa (sysCall_sensing)

Esta função é o coração do controle, executada a cada ciclo de simulação, implementando uma **lógica de prioridade estrita**:

1. **Leitura dos Sensores:** Os três sensores são lidos simultaneamente.
2. **Decisão Sequencial (Prioridade):**
    - **Prioridade Máxima (Risco Central):** Se o sensor **central** detectar (res3 > 0), o robô **gira para a direita** (girar(-1)).
    - **Prioridade Intermediária (Risco Esquerdo):** Se o sensor **central NÃO** detectar, mas o **esquerdo** detectar (res2 > 0), o robô também **gira para a direita** (girar(-1)).
    - **Prioridade Mínima (Risco Direito):** Se apenas o sensor **direito** detectar (res1 > 0), o robô **gira para a esquerda** (girar(1)).

- **Caminho Livre:** Se nenhum sensor detectar, o robô **avança** (moverParaFrente()).

4. Código Fonte (Lua)

```
-- ==============================
-- Controle de desvio do NAO (simples, sem cair)
-- ==============================

-- Sensores
local proximitySensor1 -- direita
local proximitySensor2 -- esquerda
local proximitySensor3 -- central

-- Parâmetros de movimento
local VELOCIDADE_AVANCO = 0.01   -- deslocamento para frente por
passo
local VELOCIDADE_GIRO   = 0.05   -- giro por passo (rad)
local nao

-- Inicialização
function sysCall_init()
    nao = sim.getObject('/NAO')

    -- Mapeamento dos Sensores
    proximitySensor1 = sim.getObject('/NAO/proximitySensor1') --
direita
    proximitySensor2 = sim.getObject('/NAO/proximitySensor2') --
esquerda
    proximitySensor3 = sim.getObject('/NAO/proximitySensor3') --
central

    sim.addLog(sim.verbosity_scriptinfos, "Controle de desvio
seguro inicializado.")
end

-- Função para mover o robô para frente
function moverParaFrente()
    local pos = sim.getObjectPosition(nao, -1)
    local orient = sim.getObjectOrientation(nao, -1)
    -- Calcula novo X e Y baseado na orientação (yaw) do robô
    pos[1] = pos[1] + VELOCIDADE_AVANCO * math.cos(orient[3])
    pos[2] = pos[2] + VELOCIDADE_AVANCO * math.sin(orient[3])
    sim.setObjectPosition(nao, -1, pos)
end

-- Função para girar o robô
function girar(direcao)
    -- direcao: 1 = esquerda, -1 = direita
    local orient = sim.getObjectOrientation(nao, -1)
    orient[3] = orient[3] + direcao * VELOCIDADE_GIRO
    sim.setObjectOrientation(nao, -1, orient)
end

-- Loop de controle (Executado a cada passo de simulação)
function sysCall_sensing()
    local res1, dist1 = sim.readProximitySensor(proximitySensor1)
    local res2, dist2 = sim.readProximitySensor(proximitySensor2)
    local res3, dist3 = sim.readProximitySensor(proximitySensor3)
```

```
    -- Decisão de desvio (Lógica Reativa com Prioridade)
    if res3 > 0 then
        -- Central detecta: Gira para direita
        sim.addLog(sim.verbosity_scriptinfos, "Obstáculo central ->
desviando para direita")
        girar(-1)

    elseif res2 > 0 then
        -- Esquerdo detecta: Gira para direita
        sim.addLog(sim.verbosity_scriptinfos, "Obstáculo à esquerda
-> desviando para direita")
        girar(-1)

    elseif res1 > 0 then
        -- Direito detecta: Gira para esquerda
        sim.addLog(sim.verbosity_scriptinfos, "Obstáculo à direita
-> desviando para esquerda")
        girar(1)

    else
        -- Caminho livre: Anda para frente
        moverParaFrente()
    end
end

-- Finalização
function sysCall_cleanup()
    sim.addLog(sim.verbosity_scriptinfos, "Sistema de desvio seguro
finalizado.")
end
```