Instituto Tecnológico de Aeronáutica - ITA
Inteligência Artificial para Robótica Móvel - CT-213
Aluno: Danilo de Farias Matos

Relatório do Laboratório 1 - Máquina de Estados Finita e *Behavior Tree*

1. Breve Explicação em Alto Nível da Implementação

1.1. Máquina de Estados Finita

0) *Comecei o exercício colocando as iniciações dos 4 estados (Forward, Spiral, GoBack e Rotate) apenas setando o timer para zero. Exceto a Classe Rotate que tive que implementar uma escolha aleatória de direção para rotacionar e o tempo de 3 segundos para ser a permanência máxima no estado.*

a) *Depois implementei o check_transition dos movimentos forward e spiral para detectar a colisão com um if statement e caso o booleano retornado pela função get_bumper_state() seja True ele entra no estado GoBack.*

b) *Logo na sequência coloquei um elif para mudar de estado caso chegasse o tempo máximo de execução do estado atual (para o Forward e Spiral) e apenas o If para o Goback e Rotate.*

c) *Depois disso foi a vez de configurar a parte do Execute*

- *Para a Classe Forward apenas coloquei a função set_velocity para copiar a constante FORWARD_SPEED e ficar incrementado o tempo com base na constante SAMPLE TIME*
- *Para a Classe Spiral foi necessário fazer a velocidade linear crescer à medida em que o raio fosse crescendo conforme fórmula proposta no exercício.*
- *A Classe GoBack seguiu o mesmo padrão com a diferença de chamar o Rotate após passar o tempo máximo.*
- *A Classe Rotate também seguiu o mesmo padrão com a diferença de chamar a Forward caso o tempo máximo fosse excedido.*

1.2. Behavior Tree

0) *Comecei configurando o comportamento da ClasseRoombaBehaviorTree com SelectorNode para dois SequenceNode (um que controlaria os nodos para limpeza e outro que controlaria os casos de colisão).*

a) *Na inicialização e no enter de todos os nodos (com exceção do BackNode e RotateNode) apenas zerei o timer. Para o RotateNode eu ainda implementei um tempo máximo de recuo e uma escolha aleatória de direção para rotacionar na inicialização e para o BackNode eu coloquei uma condição na entrada para verificar se a função get_bumper_state() retornaria Falso, caso seja falso o nodo retorna Failure.*

b) *Após isso foi a vez do execute.*

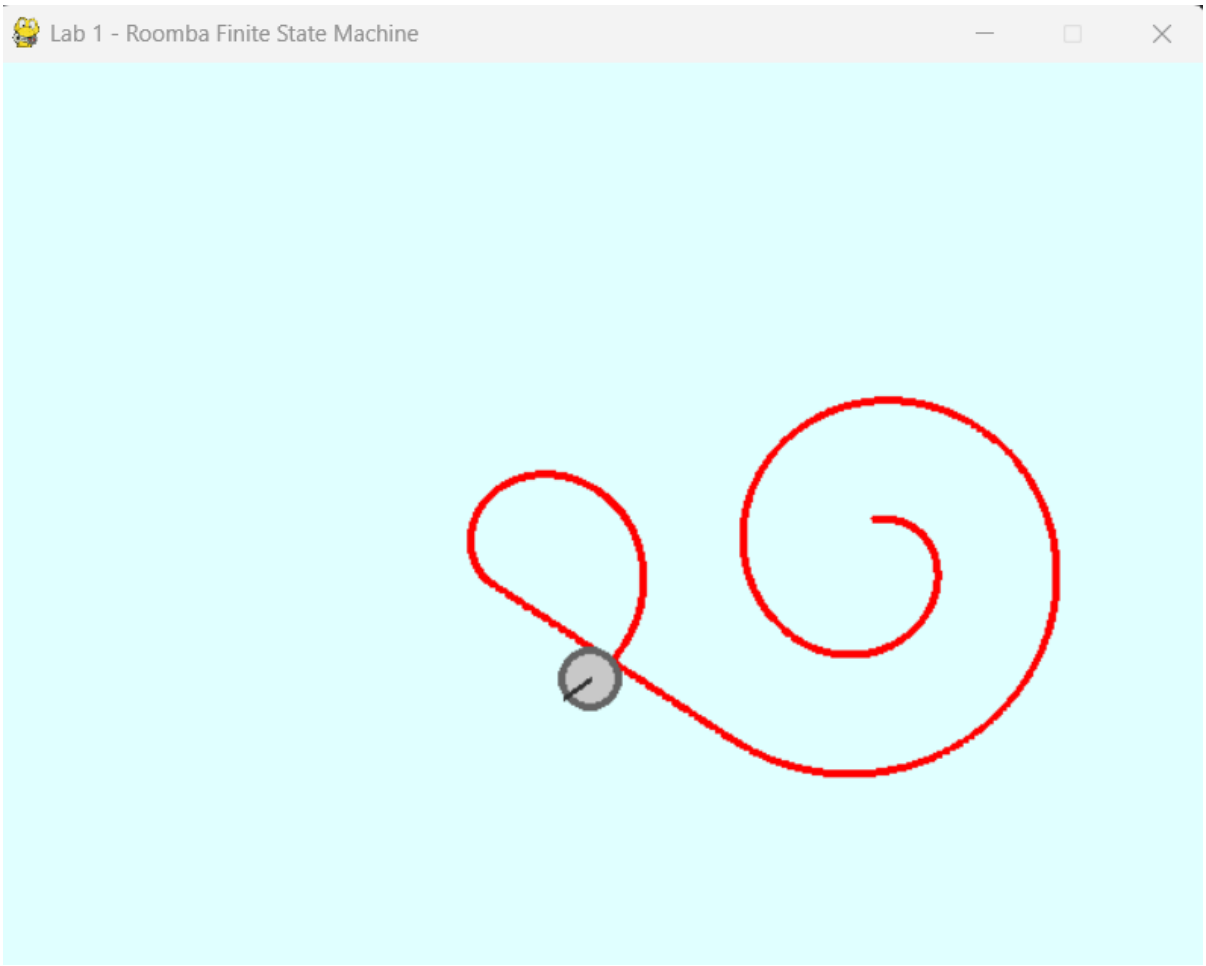
- *Para os nodos responsáveis pelo movimento em linha reta e em espiral eu usei a estrutura condicional if, elif, else para, respectivamente, checar se houve colisão e, caso verdadeiro, retornar Failure, verificar se o tempo máximo de execução transcorreu e retorar Success e, se nenhum desses ocorresse, retornar Running.*
- *Além das condições citadas acima a condição else serviu para implementar o*

comportamento de seguir reto para o MoveForwardNode e em espiral para oMoveInSpiralNode.

- *O execute do GoBackNode inicia com a estrutura if, else, sendo que a primeira condição serve para verificar se o tempo máximo foi excedido e, caso positivo retornar Success e o eles retorna Running.*

2. Figuras Comprovando Funcionamento do Código

2.1. Máquina de Estados Finita



2.2. Behavior Tree