Instituto Tecnológico de Aeronáutica - ITA
Inteligência Artificial para Robótica Móvel - CT213 (Ano: 2025)
Aluno: Gustavo Ferraresi Gottschild

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

1.1.1 Estrutura de Classes

Na implementação da *DFSM* ("*Deterministic Finite-State Machine*"), empregaram-se tanto classes de estado como classes que representam a *DFSM* em si. As classes de estado foram: `State` (abstrata), bem como suas classes derivadas `MoveForwardState`, `MoveInSpiralState`, `MoveBackState` e `RotateState`; fazendo-se, ainda, uso de polimorfismo para implementação dos métodos das classes derivadas. Os métodos de tais classes derivadas são `__init__(self)`, `check_transition(self, agent, state_machine)` e `execute(self, agent)`. Para a implementação em *OOP* ("*Object-Oriented Programming*") da *DFSM*, usou-se a classe `FiniteStateMachine`.

1.1.2 Funcionamento do Robô

O funcionamento do *roomba* se dá, como mostrado na Fig. 2 do roteiro, da seguinte forma: o robô fica alternando entre se mover para frente (estado `MoveForwardState`) por $t = t_2 = 3\,s$ e se mover conforme a espiral logarítmica descrita na Eq. 1 (estado `MoveInSpiralState`) por $t_1 = 20\,s$, sendo que, caso seja detectada colisão em qualquer um dos estados, o *roomba* executa o seguinte:

1. **Estado** `GoBackState`**:** Andar para trás por um intervalo de tempo $t_3 = 0,5\,s$;
2. **Estado** `RotateState`**:** Girar de um ângulo $\varphi \in [-\pi, \pi)$, com distribuição de probabilidades uniforme em tal intervalo;
3. Ir para o estado `MoveForwardState`.

E, assim, volta-se ao estado inicial da *DFSM*.

1.1.3 Geometria da Trajetória

Quanto à trajetória do *roomba*, ela é tal que descreve, em um sistema (r, θ) de coordenadas polares:

- Do enunciado, tem-se que
$$r = r(t) = r_0 + bt$$

- Como $\omega = \dot{\theta} = v/r$ e como $v \doteq v_0 =$ constante, então

$$\omega = \frac{\mathrm{d}\theta}{\mathrm{d}t} = \frac{v_0}{r_0 + bt} \Rightarrow \theta - \theta_0 = \int_{t_0}^{t} \frac{v_0}{r_0 + b\eta}\, d\eta \Rightarrow \boxed{r = r(\theta) = r_0 \cdot exp\left(\frac{b}{r_0}(\theta - \theta_0)\right)} \quad (1)$$

- Finalmente, conclui-se que o *roomba* descreve uma **espiral logarítmica**.

1.2 *Behavior Tree*

1.2.1 Estrutura de Classes

Obviamente, tal estrutura visa implementar a *behavior tree* mostrada na Fig. 3 do roteiro.

O status de execução retornado pelos nós da árvore foi implementado, via tipos enumerativos, com uma classe `ExecutionStatus`, a qual é derivada da classe `Enum` da biblioteca `enum`.

Os nós da árvore de comportamentos são implementados por meio da classe abstrata `TreeNode`, a qual tem como classes filhas `LeafNode` e `CompositeNode`. Já os nós compostos da *behavior tree* do tipo *sequence* e *selector* são implementados, respectivamente, por meio das classes `SequenceNode` e `SelectorNode`, ambas filhas da classe concreta `CompositeNode`. Ademais, os nós folha são implementados via as classes `MoveForwardNode`, `MoveInSpiralNode`, `MoveBackNode` e `RotateNode`; todas filhas da classe concreta `LeafNode`.

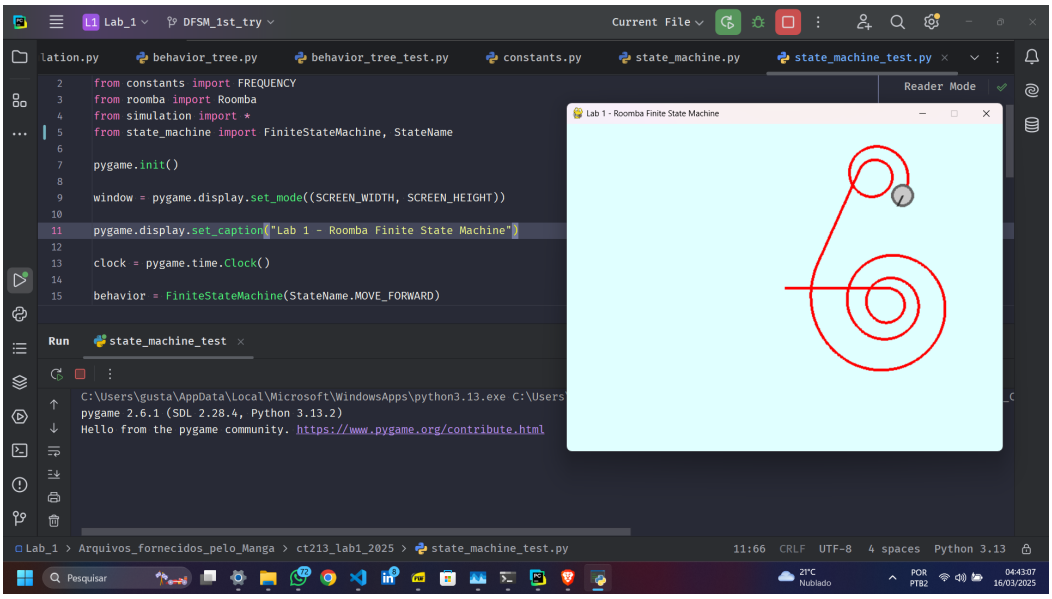
Quanto à *behavior tree* propriamente dita do agente, essa é implementada como uma instância da classe `RoombaBehaviorTree`, a qual é filha da classe concreta `BehaviorTree`.

1.2.2 Funcionamento do Robô e Geometria da Trajetória

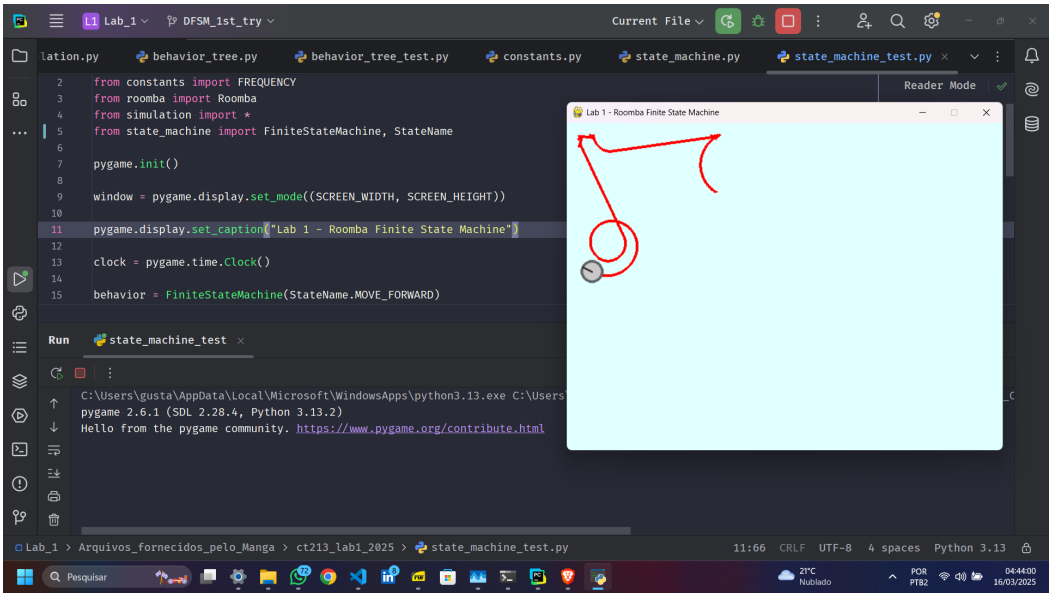
São idênticos aos da implementação do agente via *DFSM*. A única diferença entre as implementações com *DFSM* e com *behavior tree* é a forma como o agente é implementado.

2 Figuras Comprovando o Funcionamento do Código

2.1 Máquina de Estados Finita



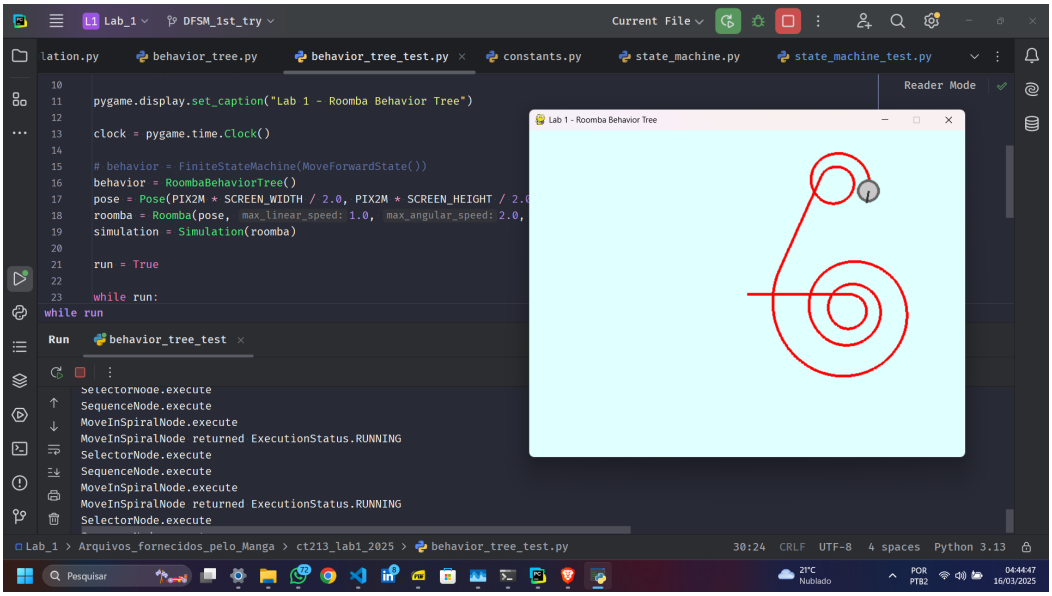
(a) Estados "`move_forward`" e "`move_in_spiral`".



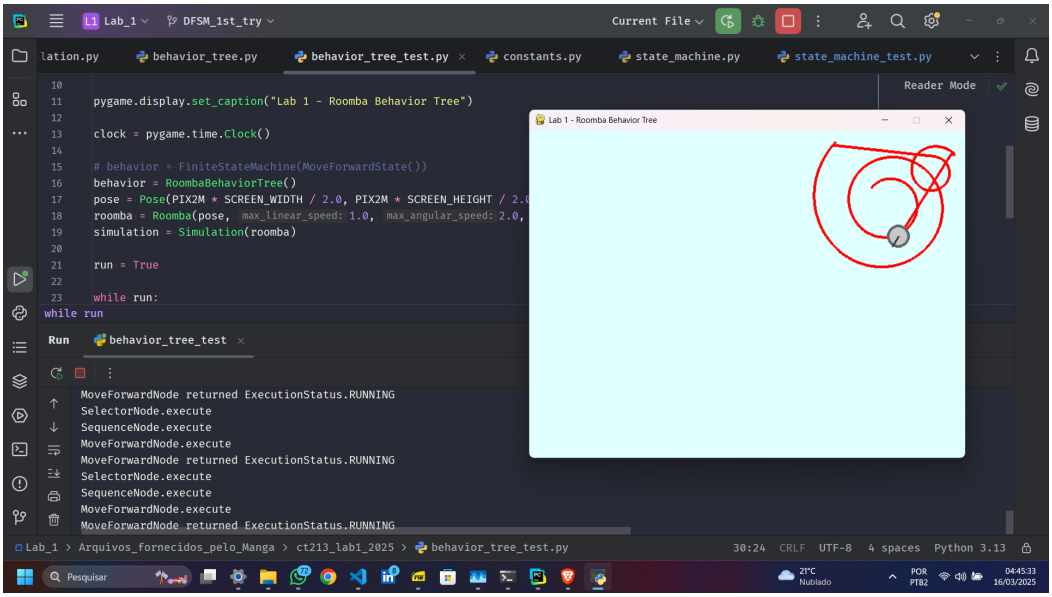
(b) Estados "`go_back`" e "`rotate`".

Figura 1: Simulação, no *engine Pygame*, do agente controlado pela *DFSM*.

2.2 *Behavior Tree*



(a) Estados "`move_forward`" e "`move_in_spiral`".



(b) Estados "`go_back`" e "`rotate`".

Figura 2: Simulação, no *engine Pygame*, do agente controlado pela *behavior tree*.

Referências

[1] M. R. O. A. Máximo, "Laboratório 1 – Máquina de Estados Finita e *Behavior Tree*," pp. 1–6, Março de 2025.