Assignment3-DecisionMaking

ZiShuai Zhang

1. Decision Trees

1.1 Behavior Introduction:

First, determine whether the current state is navigating towards a target. If so, check if the time spent navigating is greater than X. If it is less than X, continue navigating. If it is greater than X, check the distance to the target. If the distance is greater than X, reset the navigation target. If the distance is less than X, stop navigating. If not currently in a navigation state, then check if there is an obstacle ahead and determine if the speed is greater than X. If an obstacle is imminent and the speed is greater than X, move backward. If an obstacle is imminent and the speed is less than X, reset the navigation target. If there is no imminent obstacle and the speed is greater than X, wander. If there is no imminent obstacle and the speed is less than X, accelerate.
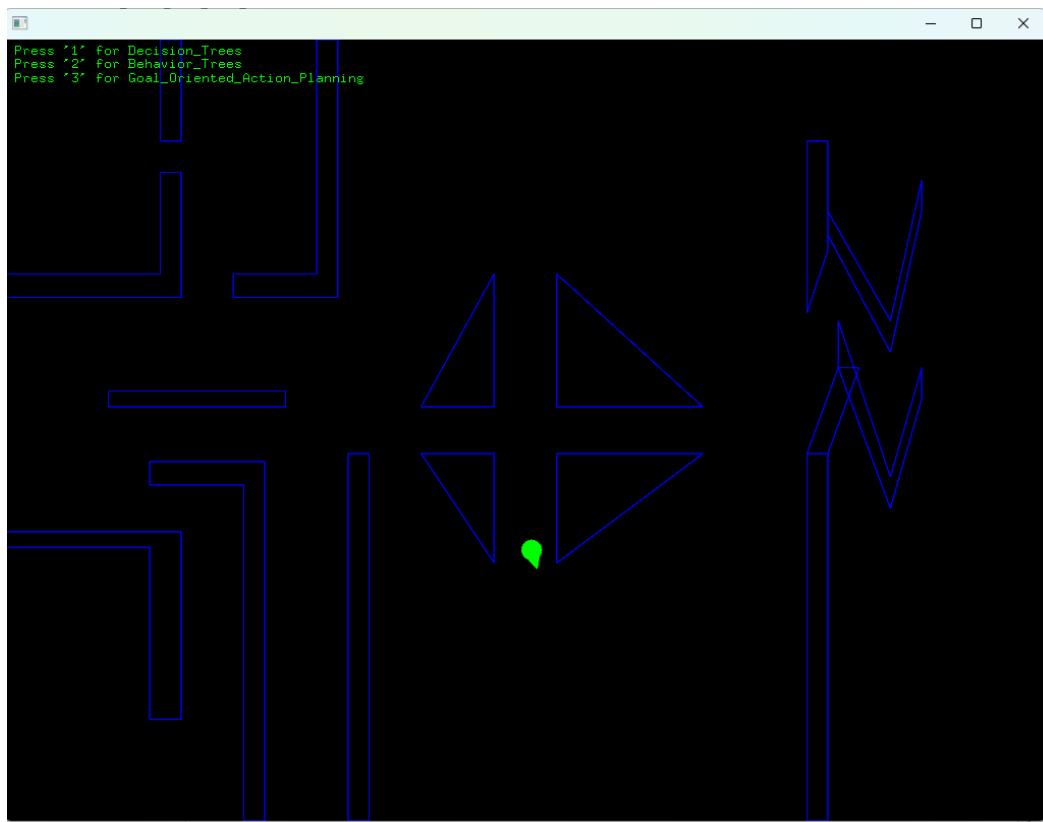

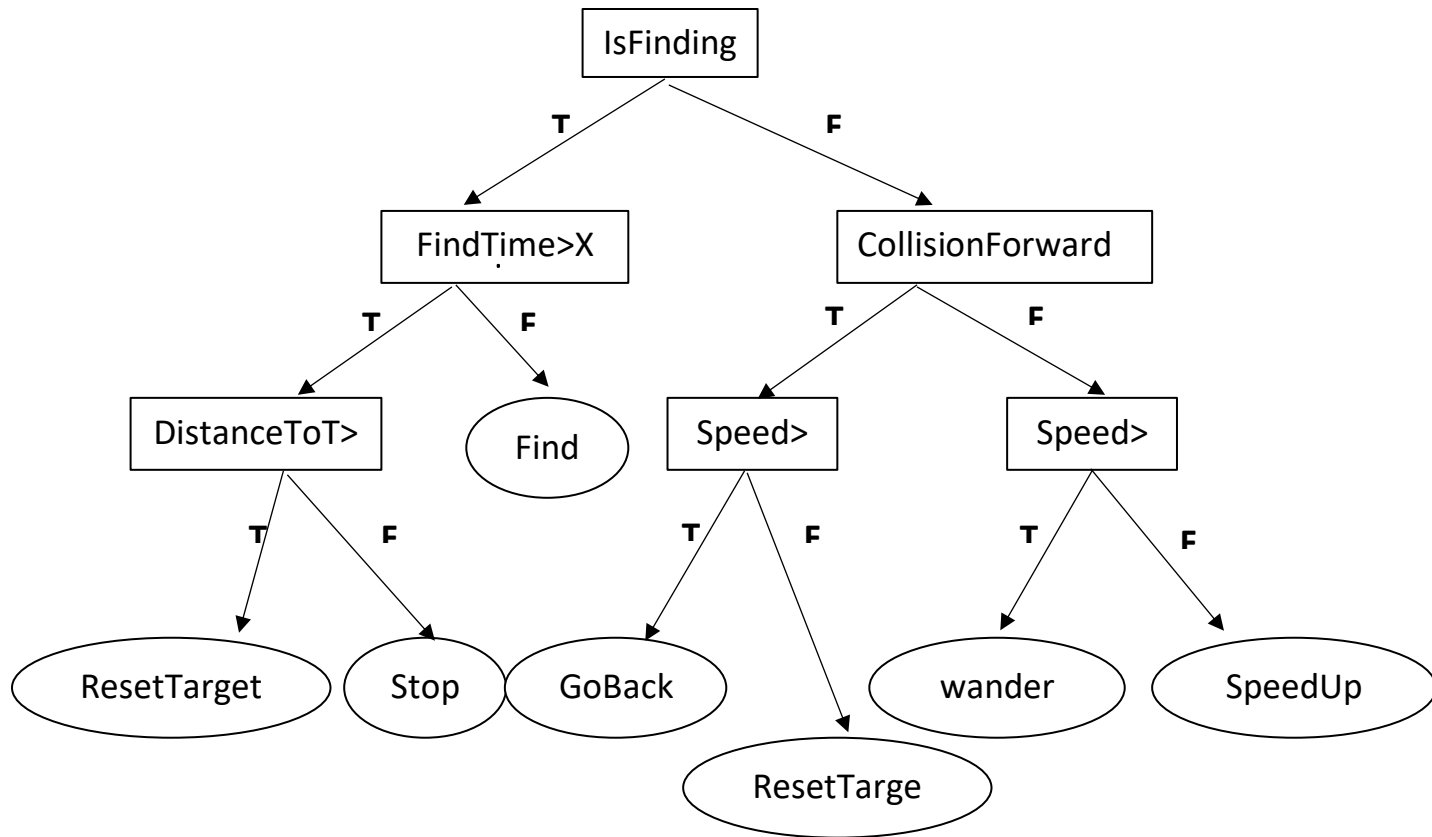**Figure 1:** Screenshot of Decision Tree demo.

Figure 2: Decision Tree Structure.

1.2 Required Variable Information:

Position of the AI-controlled Monster: Coordinates (float x, float y).

Velocity of the AI-controlled Monster: Velocity components (float x, float y).

Position of the collider, defined by vertex coordinates (Position v1, Position v2, Position v3, ...).

Pathfinding State: (bool) Indicates whether pathfinding is active.

Pathfinding Time: (float) Represents the time spent in the current pathfinding state.
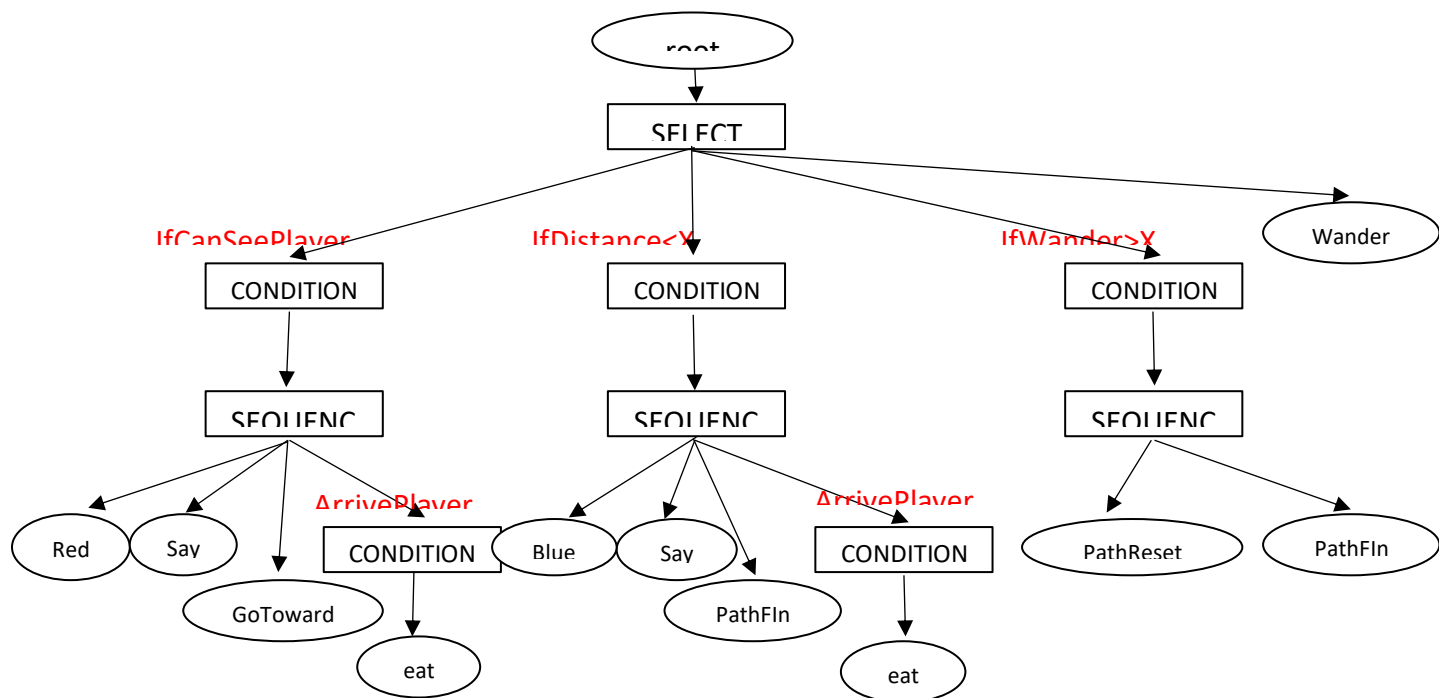
2. Behavior Trees

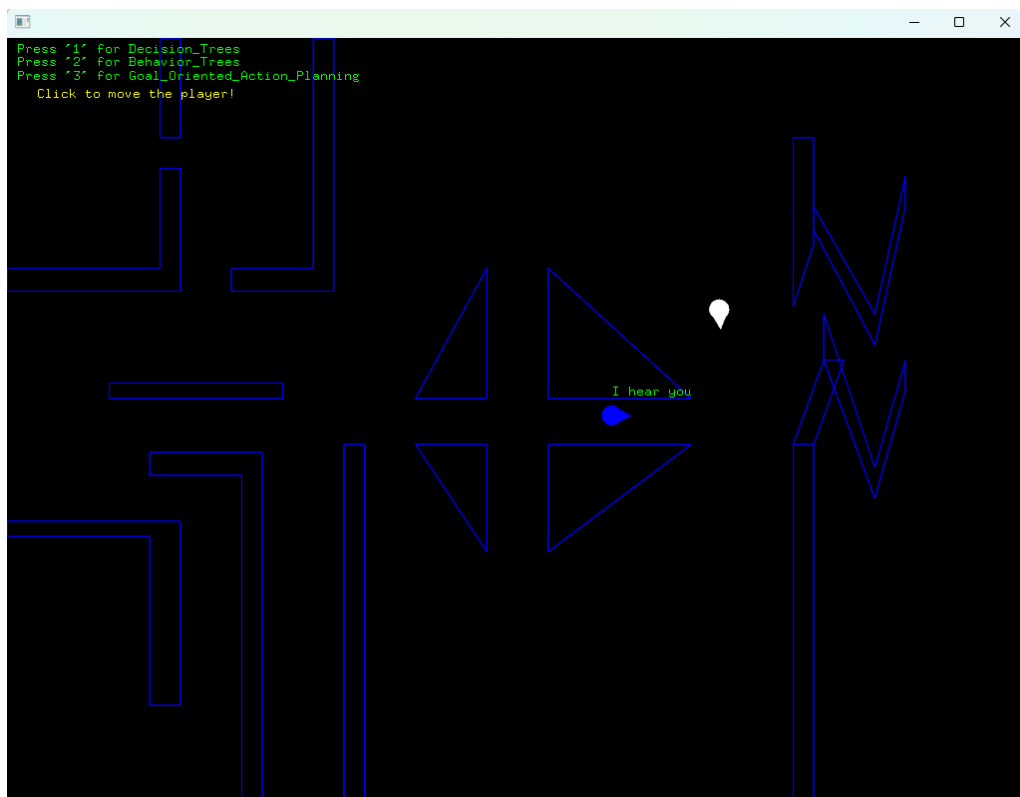**Figure 3:** Behavior Tree Structure.

Figure 4: Screenshot of Behavior Tree demo.

2.2 Behavior Introduction:

First, determine whether the Monster can directly see the Player without any obstacles. If the Monster can see the Player, it turns Red, says "I See You", and then moves towards the Player. If the Monster then arrives at the Player's location, it eats the Player. If the Monster cannot directly see the Player, assess whether the distance between them is sufficiently small. If it is, the Monster turns Blue, says "I Hear You", and then moves towards the Player using pathfinding. If it arrives, it eats the Player. If the distance is also large, check if the Monster's Wander time is sufficiently long. If it is, randomly reset a target and begin pathfinding, like the effect of Wandering. If the Wander time is not long, continue to Wander.

2.2 Required Variable Information:

Position of the AI-controlled Monster: Coordinates (float x, float y).

Velocity of the AI-controlled Monster: Velocity components (float x, float y).

Position of the Player: Coordinates (float x, float y).

Position of the collider, defined by vertex coordinates (Position v1, Position v2, Position v3, ...).

Wander Time: (float) Represents the time spent wandering.

3. Goal Oriented Action Planning
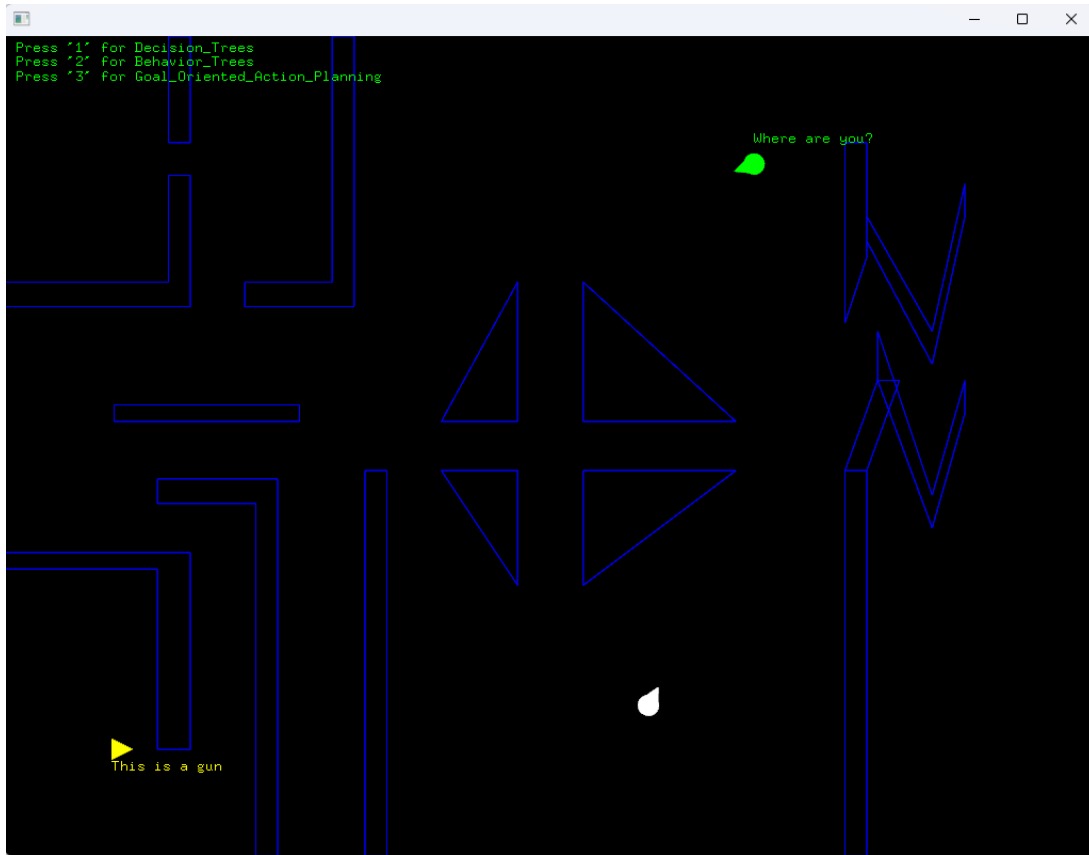
3.1 Player Behavior Introduction

Figure 5: Screenshot of GOAP demo.

If the goal's status "success" is set to true, the following behaviors are initiated: If the player is not within a certain distance of the monster, randomly wander. If the distance to the monster reaches a certain range, navigate towards the gun. After obtaining the gun, if the player is far from the monster, continue to wander; if close to the monster, navigate towards the monster. During this time, if the player can see the monster (i.e., there are no obstacles obstructing the view), shoot to kill the monster. Upon successfully killing the monster, the player flashes colors, and the game status "success" is set to true.

Action Name	Preconditions	Effects
goToGetGun	hasGun: false, findMonster: true	hasGun: true
smallRangeRandom	hasGun: false, seeMonster: false, hearMonster: false	seeMonster: true, hearMonster: true
findMonster	findMonster: false, seeMonster: true	findMonster: true
goToMonster	hasGun: true, findMonster: true	seeMonsterWithGun: true
shootMonster	seeMonsterWithGun: true, hasGun: true	MonsterDie: true
dance	MonsterDie: true	success: true

Table 1: GOAP Actions Overview Table.

Status Name	Data Structure
hasGun	**bool**
findMonster	**bool**
seeMonster	**bool**
hearMonster	**bool**
findMonster	**bool**
seeMonsterWithGun	**bool**
MonsterDie	**bool**
success	**bool**
MonsterHP	**float**
DistanceWithMonster	**float**
DistanceWIthPlayer	**float**
Position	**float2**
MonsterPosition	**float2**
GunPosition	**float2**

Table 2: State of the Environment.

3.2 GOAP Sensor

Called every frame to update the statuses of "hearMonster" and "seeMonster" in real time.

If the distance between the player and the monster is less than X, then set "hearMonster" to true; otherwise, set it to false.

If there is no obstacle blocking the view between the player and the monster, then set "seeMonster" to true; otherwise, set it to false.