ASTEROIDS GAME

by Tutku Tashkan

I used finite-state machines (FSMs) to create AI solutions that simulate intelligence in my NPCs. There are 2 NPC's in my game and one of them is an UFO which Chase the player and shoot the other one is a special Asteroid that again chase the player and collide with it.

There are 2 States for the UFO's AI "Roaming State" and a "Chase State"

In the game's initial state (the "Roaming" state) the UFO is roaming around. At first we get a roaming position with GetRoamingPosition() function which is a random position inside the game area(14*9) and then we use that position and tell the UFO's to go to that roaming position. If the UFO reached that position we give it a new roaming position. This cycle is continues until the UFO is close to the Player (distance of 5) we check this in FindTarget() function then we go into the "Chase" state.

In the "Chase" state the UFO is chasing the player. If the UFO is still coming closer to Player and the distance between them is less than 3 the UFO will go into the "Shooting" state. If the player fly away from the UFO and the distance between them is greater than 5 than the UFO will stop chasing the player and go back to the "Roaming" state

In the "Shooting" state the UFO is shooting the player. We get the position of the player with and UFO will shoot the bullets on that position every 2 seconds. If it got closer than distance of 2 it will retreat so that the distance between the UFO and the player is not so close and we can see the UFO shooting. If the player fly away and it is not in the shooting distance but still in chase distance UFO will go back to "Chase" state.

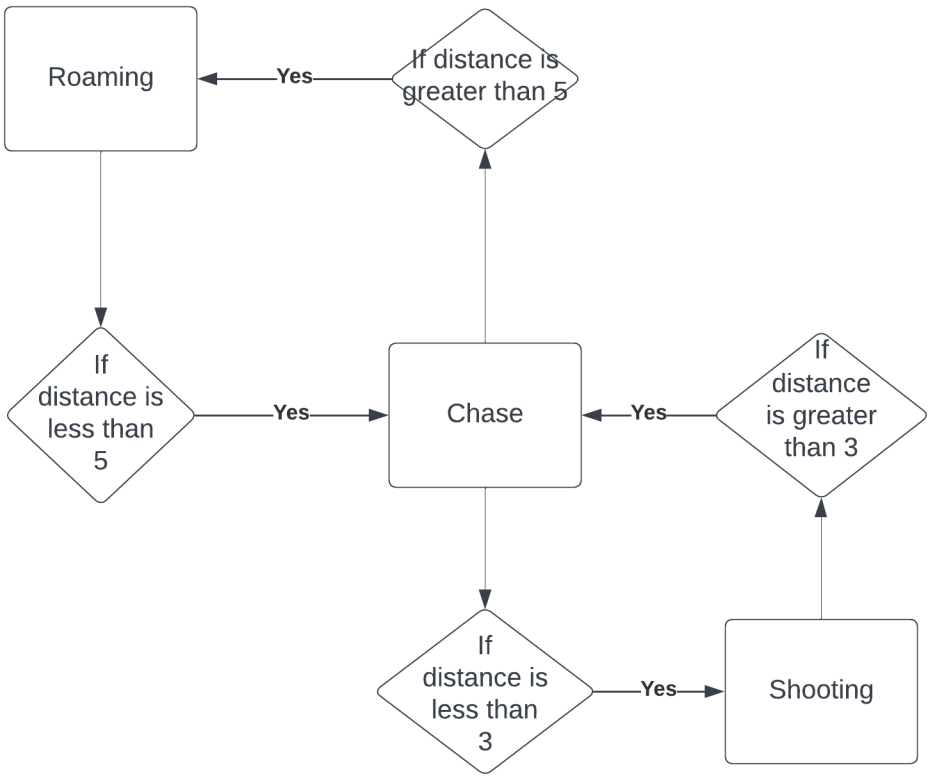
The AI of the asteroid has the same logic but the asteroid does not have the "Shooting" state. When the asteroid is close to the Player it goes in to the "Chase" state and tries to collide with the player. If the player fly away from the asteroid the asteroid will go into the "Roaming" state

```
private enum State {

  Roaming,
  ChaseTarget,
  ShootingTarget,

}
private void Awake() {

  // initial state
  state = State.Roaming;
}

private void Start() {
  startingPosition = transform.position;
  roamPosition = GetRoamingPosition();}
```

```
    private void Update() {
      switch (state) {
      default:
      case State.Roaming:

          transform.position = Vector2.MoveTowards(transform.position,
           roamPosition, speed * Time.deltaTime);

          float reachedPositionDistance = 3f;
          if (Vector2.Distance(transform.position, roamPosition) <
           reachedPositionDistance)
           {
              // Getting new Position to go
              roamPosition = GetRoamingPosition();
          }
          FindTarget();
          break;

       case State.ChaseTarget:

           transform.position = Vector2.MoveTowards(transform.position,
           target.position, speed * Time.deltaTime);

           float attackRange = 3.0f;
           if (Vector3.Distance(transform.position, target.position) <
           attackRange) {
               // attack
               state = State.ShootingTarget;
           }
           float stopChaseDistance = 5f;
           if (Vector3.Distance(transform.position, target.position) >
                  stopChaseDistance) {
               // Too far, stop chasing
               state = State.Roaming;
           }
           break;

       case State.ShootingTarget:

           if(Time.time > nextShotTime){
               Instantiate(ufoBullet,transform.position,Quaternion.identity);
               nextShotTime = Time.time + timeBetweenShots;
           }

           float goBack = 2.0f;
           if (Vector3.Distance(transform.position, target.position) <
           goBack) {
               // Retreat
```

```
        transform.position = Vector2.MoveTowards(transform.position,
        target.position, (-speed*2) * Time.deltaTime);
           }
           if (Vector3.Distance(transform.position, target.position) > 3.0f)
           {
               state = State.ChaseTarget;
           }
           break;
       }
   private Vector3 GetRoamingPosition()
     {
      Vector3 nextDirection = Random.insideUnitCircle.normalized;
      Vector3 nextPoint = nextDirection * Random.Range(13f,8.0f);
      return nextPoint;
     }
     private void FindTarget() {

      float targetRange = 5f;
      if (Vector2.Distance(transform.position, target.position) <
           targetRange) {
          // Player within target range
          state = State.ChaseTarget;
      }
   }
```

FSM of UFO



Levelling of the Game

Player starts at level 1 and pass the level 1 at score 1500. Score needed to pass the levels are increasing exponentially by 50% more. So level 2 is passed at 2250 point and level 3 3375 so at first it's easier to pass levels but then it gets harder. Every level the speed of the asteroids increase 0.5 more. Every 5 levels the amount of asteroids spawned in every 2 seconds is increased by 1, the Health of the UFO increase by 3 which at the start was 10 so you will need to shoot the UFO more to kill it.