AN AIR DEFENCE SCENARIO SIMULATION WITH SWARM ROBOTS

Burak Eren Dere
Department of Computer Engineering,
Middle East Technical University,
06531 Ankara, Turkey
Email: eren.dere@metu.edu.tr

Abstract

An air defence scenario is being implemented by using an open source 2d physics engine. There are swarm robots patrolling an area and trying to catch incoming rockets by estimating their trajectories. Swarm robots will try to mimic the behavior of flocks. Each robot will follow a modified boids algorithm.

Keywords

Swarm intelligence, Boids Algorithm, aggregate motion, path planning

I. Introduction

In this project I focused on an air defence scenario I came up with in which swarm robots try to intercept incoming missiles. According to the scenario, there will be one attacker side and one defender side. Defender will have a base and a robot swarm, while the attacker will have missiles to fire upon the defender. Robots will move by mimicking the behavior of birds and fish and when they sense that a threat approaches, they will try to intercept it by moving together towards the threat.

II. Background and Related Work

In order to carry on my project, I needed to get used to a physics engine because I wanted to implement my algorithms in a realistic simulation. I am programming the project with c++ language.I have learned Box2D[1] physics engine and SFML[2] for displaying the physics simulated by Box2D. The paper I have found very useful for my project was Craig Reynolds(1987)[3] which explains the aggregate motion of flock of birds or school of fish. Craig Reynolds developed the Boids algorithm which simulates flocking behavior of birds. In this algorithm, velocities of swarm members depend on various rules. Simplest rules are separation for avoiding local flockmates, alignment for steering towards the average heading of local flockmates and cohesion for moving towards the average position of local flockmates[4].

III. Progress

After I finished learning how to use Box2D and SFML with c++, I started implementing the basics of the simulator. I worked on collision listening and filtering and how to get data from censors that I've implemented. I developed an agent based simulation using object oriented approach where each agent's act() method is being called. Motions are implemented using torques, and forces by using differential equation logic to make movements more realistic. So far, I have finished the design of swarm robots and their act() methods by modifying Boids algorithm according to my case.

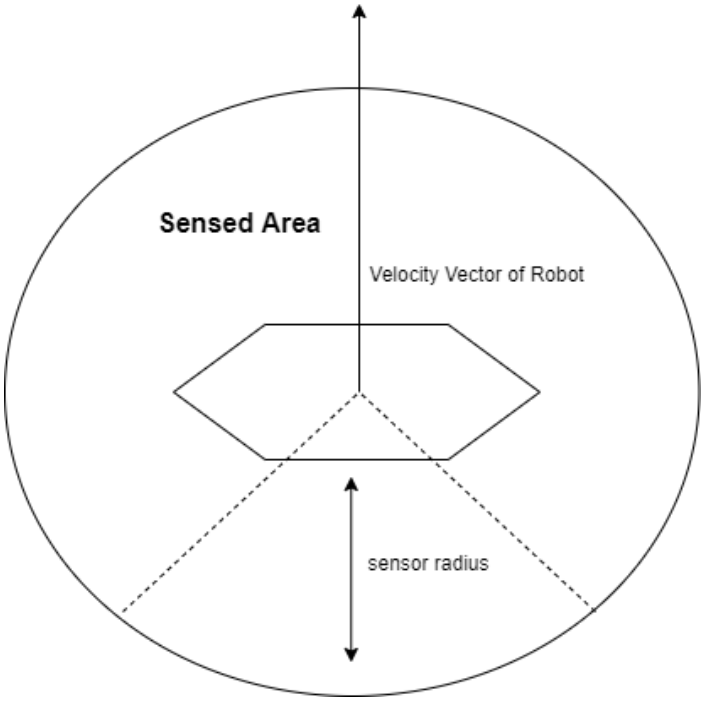

Fig. 1: How robots and their censors are designed and implemented in Box2D

Each robot does certain operations every time step. Robots select their actions according to the action hierarchy. I used a layered architecture for determining the behavior of robots. First of all, since I programmed a realistic system, there are certain forces acting on the robot. At first robot, tries to keep its orientation if a force exerted on it results in a torque. Robot starts applying torques to control the angular velocity and stops in desired orientation. Secondly, if robot wants to stay in an exact position, it tries to keep its position if external forces changes its location. Thirdly, if robot wants to move in one direction, it always try to keep its direction despite the existence of external forces. And on the top layer, boids algorithm which I have modified runs for each robot. In my implementation, for demonstration, robots are randomly changing their directions. I tried to achieve bird flock type movements by tweaking the algorithm constants and managed to do that.So far,I have modified boids algorithm and used it in my project.In this algorithm, each robot has a neighbourhood which it senses through its sensors and in order to determine their velocities in the next time step, they are checking other robots in their neighbourhood. Algorithm has three key concepts which are separation,alignment and cohesion. In algorithm 1, I have explained how I implemented boids in the project. After finishing my algorithm, I tested it and saw that it was working like the examples that I have seen on the internet. I uploaded a video to show my progress in implementing the algorithm. In the video, robots turn red if they sense other robots nearby. Normally, as explained in Fig.1, robots cannot see the opposite direction of their velocity vectors to some degree. By exploiting this blind spot, robots which are in front of the swarm can navigate others to the destination they desire. In my video, robots are randomly patrolling a circular area. There are 100 robots on the scene and they are introduced to the patrolling area from different places. At first 3-4 robots move together, but as the time goes, they all start moving together in a swarm and eventualy it converges into a one big swarm. Time to convergence varies according to the variables such as patrolling area size,sensor radius,maximum speed, maximum drag force or constants in the modified boids algorithm.

Algorithm 1 Boids Algorithm That I've implemented

procedure COHESION ▷ Calculates vector to the average position of neighbors
  $iterator \leftarrow 0$
  $neighborRobots \leftarrow giveRobotsInNeighborhood()$
  $center \leftarrow (0, 0)$
  for $i < neighborRobots.size()$ **do**
    $center \leftarrow center + neighborRobots.get(i).getPosition()$
  end for
  $center \leftarrow center/neighborRobots.size()$
  $result \leftarrow center - getCurrentPosition()$
  $result \leftarrow result/100$ ▷ Move 1 percent of the way towards center
  return $result$
end procedure
procedure SEPARATION
  $result \leftarrow (0, 0)$
  $neighborRobots \leftarrow giveRobotsInNeighborhood()$
  for $i < neighborRobots.size()$ **do**
    $vec \leftarrow neighborRobots.get(i).getPosition() - getCurrentPosition()$
    $distance \leftarrow vec.length()$
    $vec \leftarrow vec * \frac{1}{distance^3} * 10000$ ▷ If robot is closer then separation gets stronger
    if $n < 2.5$ **then**
      $result \leftarrow result - vec$
    end if
  end for
  return $result$
end procedure
procedure ALIGNMENT
  $result \leftarrow (0, 0)$
  $neighborRobots \leftarrow giveRobotsInNeighborhood()$
  for $i < neighborRobots.size()$ **do**
    $result \leftarrow result + neighborRobots.get(i).getLinearVelocity()$
  end for
  $result \leftarrow result/neighborRobots.size()$ ▷ take average velocity
  $result \leftarrow result * 16$ ▷ 16 constant produced better results for me
  return $result$
end procedure
procedure BOIDSGIVEDESIREDVELOCITY
  $currentVelocity \leftarrow getVelocity()$ ▷ Take current velocity of robot
  $v1 \leftarrow cohesion()$ ▷ v1 towards the center of neighbors
  $v2 \leftarrow separation()$ ▷ v2 towards opposite direction of neighbors if they are so close
  $v3 \leftarrow alignment()$ ▷ v3 aligns direction of robot to the average of neighbors
  $desiredVelocity \leftarrow v1 + v2 + v3$
  return $desiredVelocity$
end procedure

IV. Conclusions and Future Plan

To sum up, since I started the project in the beginning of the semester, chronologically, I looked up for physics engine and a multimedia library and learned how to make projects with them. Designed a multi agent simulation with object oriented properties and integrated Box2D engine and SFML to the simulation. Then, I wrote low level procedures of robots such as stable orientation and moving directions. All of these motions are done with forces and torques in order to make simulation more realistically. And then I implemented sensors of robots by learning how to add collision listeners in Box2D. After testing the sensors and seeing they work fine, I started implemented boids algorithm and tested it with different parameters. For now, I implemented core behavior of swarm robots. In the future, I will extend the rules in the boids algorithm and add goals to robots such as tracking the missiles and estimating its trajectory, refueling if they are out of fuel, trying to change direction of missiles or explode the missile. After that, I will add two teams. One of them will be attackers the other one will be the defenders. Attackers will fire missiles to the base of defender team and defenders will try to intercept missiles and protect their bases. I might turn this simulation into a game and test defenders abilities with firing missiles by getting user input.

References

[1] "Box2D site," https://box2d.org/, accessed: 2020-04-20.

[2] "Simple and Fast Multimedia Library wikipediaexplanation," https://en.wikipedia.org/wiki/Simple_and_Fast_Multimedia_Library, accessed: 2020-04-20.

[3] C. W. Reynolds, "Flocks, herds and schools: A distributed behavioral model," in *Proceedings of the 14th annual conference on Computer graphics and interactive techniques*, 1987, pp. 25–34.

[4] "Boids wikipediaexplanation," https://en.wikipedia.org/wiki/Boids, accessed: 2020-04-20.