Detailed Description

Consider a very simple game where there are two teams: "*Terrorists*" and "*Counter-Terrorists*". The aim of the terrorists is to go to a special pre-determined site called as "*Bomb*". The aim of the counter-terrorists is to ensure that none of the terrorists can go to the site called "*Bomb*". Because playing this game manually is very boring, you have an AI Engine that automatically plays this game for you. There are three kinds of players in both teams possible:

- *AggressivePlayers*, who tend to run very fast and not observe around. Their energy level reduces by 2 at every step of move. They need to be hit twice to be dead.
- *CautiousPlayers*, who tend to go very slow and observe around. Their energy level reduces by 1 at every step of move. They need to be hit once to be dead.
- *BlindPlayer*, who very quickly run to do whatever they are doing, and do not observe around at all. Their energy level reduces by 3 at every step. They need to be hit 5 times to be dead.

Each player of counter-terrorists selects an opponent player and goes to kill the same. A player can have any of three strategies. It is assumed that everyone knows each other's position.

- Go to nearest terrorist
- Go to a random terrorist
- Go to a terrorist 'ahead' in the map

A terrorist has all the strategies of the counter-terrorists, with one additional strategy: Go to *bomb*. Any number of terrorists can select any counter-terrorist and vice versa. A terrorist can be the target of any number of counter-terrorist and vice versa. A bomb may be aimed by any number of terrorists. The strategies are constant, however the selected opponent will change as the players move. So the nearest terrorist strategy followed by a player will remain as it is, however the specific terrorist will change as the terrorists move around.

The game is sequential in nature. All players make a move one after the other. Hence the order in which the players move can be critical to the game. The order may be *circular* (one chance to every player in the same order by which they entered the arena), *by energy level* (most fit player moves first), *by success* (the player who killed the maximum opponents moves first). However, first a terrorist moves, then a counter-terrorist, then a terrorist and so on, till both the sets expire, and a new turn starts. There is a single order.

The logic may be implemented using a sorted array, unsorted array, sorted linked list, or unsorted linked list.

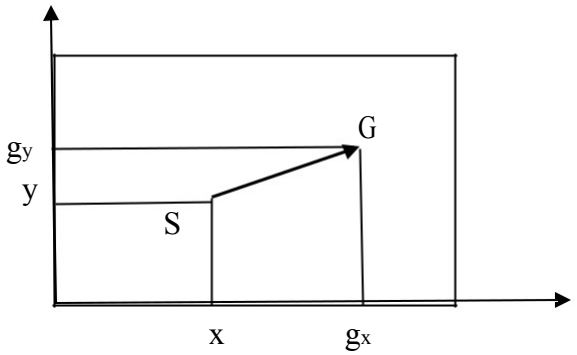
Assume function '*AImove'* exists for each player of every type that takes the positions of all players and goal, and makes a move as per some computation. Also assume that a function *site* exists for every player, that returns the player currently in the line of sight, and a *fire* action on the player may kill the other player. If the other player is an opponent, the opponent must be killed.

The class *InputHandler* takes the number of two opponents (terrorists and counter-terrorists) as input, along with their types (aggressive, cautious, blind) and selection strategies (nearest, random, ahead, bomb). The class also takes the playing order for the two teams (circular, energy level, success) and the mechanism by which the storage will happen (array, linked list, sorted/uunsorted). Everything is stored in the class *Environment* that has all information related to the game. The class *GameEngine* can access environment and has a function *play* to play the game. The function has a loop till the game completes, selects a player, computes the goal as per strategy, computes the move using AI logic given, fires (if necessary), and updates the changes in the environment (success metric, positon, and killing of player).

Mathematical Explaination of Algo

AIMove // Line 427

Every player has a position in X axis, a position in Y axis and an orientation. The orientation is an angle state governs where the person is looking at (θ).



Suppose the current position is S(x,y). The player is moving at a speed s towards a goal at G(g_x,g_y). The position at the next time step is given by:

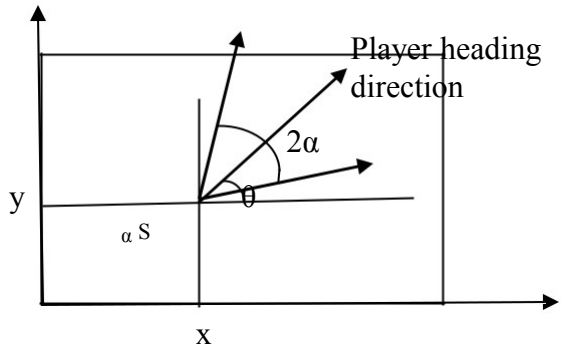
$(x',y')=(x,y)+s(g_x-x,g_y-y)/sqrt((g_x-x)^2+(g_y-y)^2)$

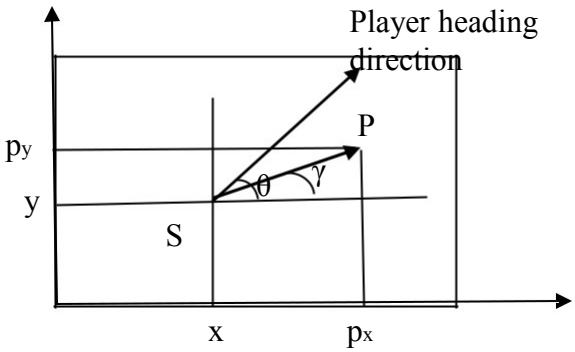
The proof is simple. Consider the vector G-S, that is (g_x-x,g_y-y). A unit vector in the same direction is (g_x-x,g_y-y)/sqrt($(g_x-x)^2+(g_y-y)^2$), while a vector at a distance of s from S is given by the formula.

When the player walks towards the specified direction his/her orientation changes to atan2(g_y-y, g_x-x)

site() //Line 399

The function checks if two players are in line of sight to each other. Assume each player can see α radians around the current orientation (θ). The person is facing at an angle of θ, while can look around the angle of ± α from the current orientation. The angle α is different for different players. The angular range of view of the person is hence in the range θ- α to θ+α.





The angle subtended by a new person at (p_x,p_y) is γ=atan2(p_y-y, p_x-x). The person is in line of site if θ- α < γ < θ+α. However since angles have a circular property the inequality cannot be directly used. The angle between the heading direction and line SP is given by θ-γ. Hence for angular coverage, cos(θ-γ)>cos(α).

The 'ahead' is simply taking α=60 degrees.