AI BEHAVIOURS

Explanation of the problem
Develop a way for the game to manage AI behaviours based on proximity of relevant agents.
Submit a solution taking into account performances and scalability of the entire system.

PROPOSED SOLUTION

Environment
The walkable surface is a *NavMeshSurface* that must be baked at the start.
Entities must be in the *Entity* layer to exclude them from the baking of the walkable surface. They can have *Player* or *Enemy* tag.

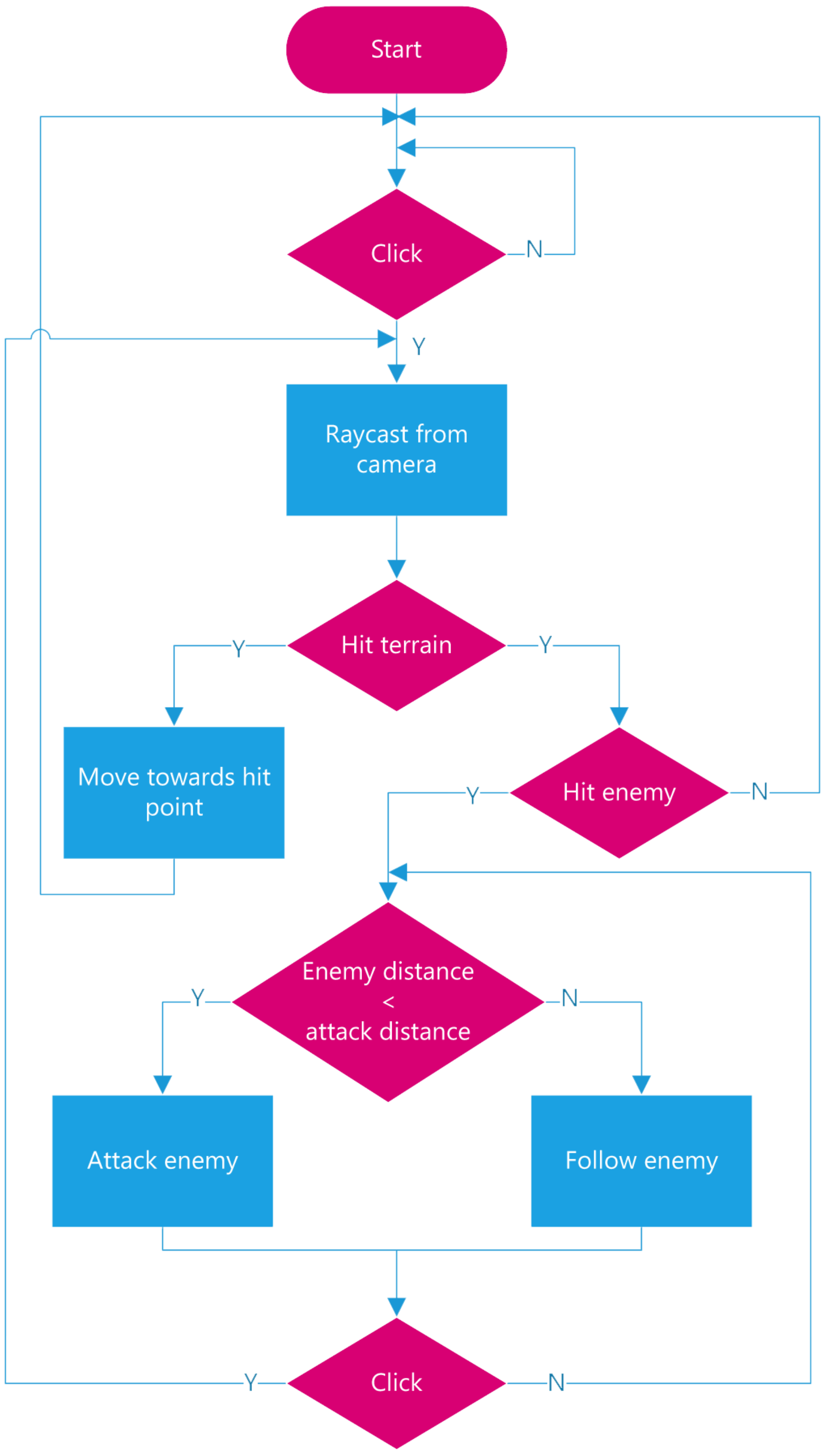
Entities
Player and enemies' scripts inherit the *Entity* class, which define their stats in terms of:
- Life
- Mana
- Damages
- Attack speed
- Attack distance
- Shield
- Critical hit probability

Entities belong to *NavMeshAgent* class and are therefore defined by:
- Type (*Humanoid* or custom)
- Movement speed
- Angular speed
- Acceleration, whose value needs to be high in order to produce a constant speed
- Stopping distance (this value is changed at runtime according to the target type: terrain or entity)

FUNCTIONING

Player

The player can be moved by clicking the right mouse button on the walkable surface or on the enemies.
When the user clicks on the walkable surface, the player simply starts following the assigned target. If the user clicks on the enemy, the player starts following and attacking his target once he is within his attack distance.

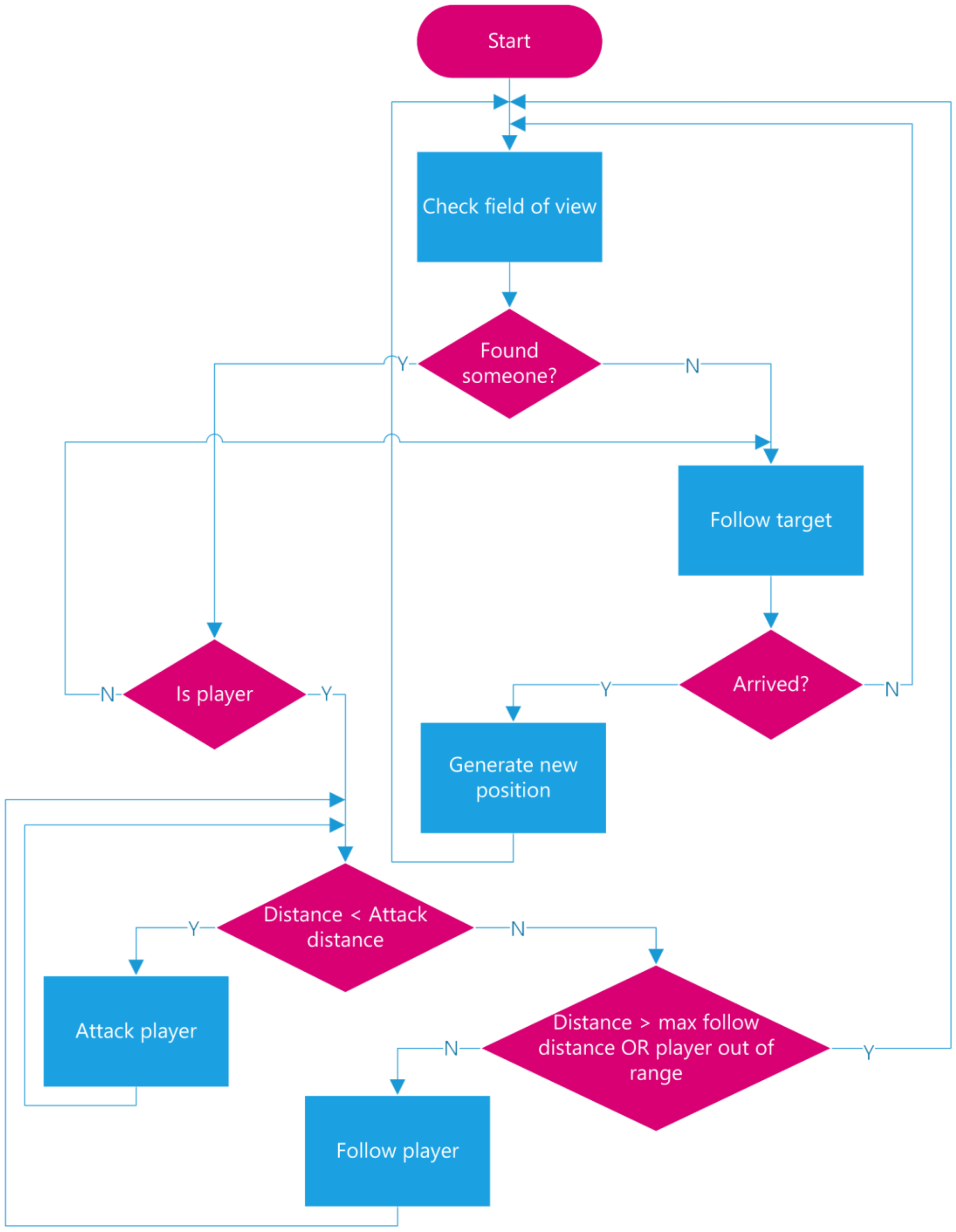


Enemies

NPCs have a specified wandering area and a cone of vision, defined by *range*, *distance* and *viewing angle* values.

When a player entered the enemy walkable area, he can be heard by him according to the values of the sound range. Making sounds near the enemy will put him into alert state: he will start searching around the area of the heard sound.

After a few seconds, if the player is hidden or out of his cone of vision, the enemy will start wandering again. If the enemy sees the player or is attacked by him, he will fix his target on him and will start following and attacking him, once he arrives at his attack distance. If the player goes out of the enemy range of vision, hearing and moving, the enemy will restore his stats and start wandering again in his area.



POSSIBLE IMPLEMENTATIONS

Attack move

One of the future implementations could be the possibility for the user to click with another mouse button and use the *attack move* function, which works in the following way:
When the user clicks on a walkable surface, the player starts following the position until it finds an enemy entity. If an enemy is found, he starts attacking it and, once killed, starts moving again towards the target.

Audio behaviours

The enemies could be able to feel sounds around them and interact with the environment according to the noises made by player movement, abilities or objects. For example, the player could distract the enemy by throwing objects near him.

Abilities

Player and enemies are considered primary entities, which means that they could be endued of certain abilities (heal, increased speed, increased damages, teleport, …) to make the combat system more interesting and difficult.

AI combat system

It could be useful to develop a combat system for the AIs in order to make it possible for the NPCs to dodge, parry, escape or foresee player attacks and movements instead of just following and attacking their targets.

Minor entities

We could consider to implement other types of NPCs in the game, for example, ally and enemy minions.

Gold and experience

Killing enemy entities and finding objects will increase the experience of the player and unlock features necessary to go on with the history or finish a mission.
These actions will give gold to the player who can spend it to in the stores of the game.

OPTIMIZATION

A good way to optimize the performances of the game could be exploitation of the Unity 2019.3 C# Job System to implement multithreading. This will make it possible to get a faster pathfinding and a better network management, using background threads and having less lags and freezes in the game.
However, this implementation will probably bring up concurrency problems.
We will therefore need a well-structured developing workflow to overcome this difficulty.