In the arena class, we need to create a function called CheckForEntityProximity. This function takes two entities parameters and one EventProximity. One of these entities is a robot. Firstly, we need to find the distance between these two entities and check if they are in the range of each other or no. However, until now I do not know how the field of view will help me to determine the direction of the robot entity. If the entity is not in the range, then the proximity does not happen. The other situation is when it is in the range. The proximity happens, the parent sensor will call Accept function which pass the two entity and the event inside the Accdpt function all scenario will check and call the Accept function for each entity(which is passed for Arena).

If a robot – home base, the robot will go forward to the home base.

If a robot – player, go away.

If robot – robot, no action is taken

If robot – frozen robot, the robot will go forward to the frozen one

If a robot – wall, go away.

If a robot – recharge station, go away.

If a robot – obstacles, go away.

If robot – super bot, no action is taken

To go forward to an entity, we need to check the distance between the two entities if it is increasing or decreasing. If it is increasing, we need to go back to the original place and change the heading angle a little bit and go forward until the distance became 0.

After that EventCollision happens, so we will call CheckForEntityCollision function for all entities. And sensor will call Accept and pass the two entities and the event. The robot can not collision with the entities except:

If robot – player, the robot will freeze, if the robot already frozen no action is taken

If robot – frozen robot, the robot will go forward to the frozen one

If robot – super bot, no action is taken

If frozen robot – super bot, robot will unfreeze

The other entities collision is:

Home base – recharge station, bounce off

Home base – obstacles, bounce off

Home base – robot, the robot will be a super bot

Home base – player, bounce off

Home base – wall, bounce off

Home base – super bot, bounce off

Player – super bot, the player will freeze for a period of time

Player – recharge station, bounce off

Player – obstacles, bounce off

Player – wall, bounce off

Super bot – recharge station, bounce off

Super bot – obstacles, bounce off

Super bot – wall, bounce off

Recharge station, obstacles is immobile entities.

When the robot is frozen, it will start sending signals to the other robot to release him. So, we will create a function called CheckForEntityDistressCall. This function takes three parameters, two robots and one EventDistressCall. Firstly, we need to check the situation of movement of each robot. If they are moving or they are stop, we do not need to do anything. However, if one of them is moving and one of them is stopped, we need to find the distance between them and then check in they are in the same range. If they are in the same rage, the sensor will call the Accept function and pass the two robot and the event.The mobile entity should move forward to the freeze robot and check the distance between them if it increases and it decreases as we did in the CheckForEntityProximity function until the EventCollision happens. After that, the robot will turn on and start moving.

For transforming the robot to SuperBot I need to call CheckForEntityCollision(robot\_, home\_base\_, &ec, robot\_->collision\_delta()). Then create a variables to save the location and heading angle and then will call robot\_.~Robot() and create SupperBot at the same location of the dead robot with a new color.

There are some points, I do not know how I will deal with such as how the field of view will help me to determine the heading angle. Another thing, does the distress sensor and proximity works at the same time? I am not sure of the changing the Robot to SupperBot will work.