Robot: PewPewFSM

PewPewFSM is a finite state machine based robot. It has 3 main states, FindTarget, Brawl, and Ranged.

**States:**

**FindTarget-**

The FindTarget state of the robot is determined sole on the robot’s CurrentTarget variable, if CurrentTarget is null the robot will spin radar 360-degrees and search for a target, as it scans if the target is friendly it will skip and continue looking for an enemy robot. Once an enemy is located CurrentTarget is updated.

**Brawl-**

Brawl state is the robots main attack state, the robot will aggressively pursue the current target, actively firing and chasing it down in an attempt to ram the target.

**Ranged-**

The Ranged state occurs under the conditions that the robot falls below 40 energy and the enemy has greater energy than the robot. If the robot is under 40 but the enemy has less energy the state will be Brawl, as its obviously working why change it. Ranged uses a form of Antigravity (From the Robocode.net site) to avoid the enemy. Instead of calculating every x and y point on each antigravity calculation it only needs to grab the enemy x and y then perform the force vector calculation. This is slightly different from the version I looked at that recalculated every x and y each time including static x and y coordinates like walls. The wall x and y coordinates are all calculated before the match starts and saved in an array, the enemy coordinates are updated on each enemy scan.

**Normal operation regardless of state:**

Regardless of the current state there are some operations that will always perform the same.

**Update-**

The Update function is the basic controller for the FSM. It is called constantly and used to determine the state of the robot. It is based on a switch where depending on the current state it will call the appropriate functions for FindTarget, Brawl, and Ranged.

**OnScannedRobot-**

The OnScannedRobot function first determines if the scanned target is friendly or not. If the target is friendly it immediately leaves the function as there is nothing to do. If the target is an enemy, it checks to see if it is the current target, if it is the target it will update the current targets stats, take aim, fire, and if we are in ranged mode it will get the enemy’s x and y coordinate to use in the antigravity calculation. Aiming is built into this function, it uses basic geometry to estimate where the target will be based on its current bearing, bullet speed, distance to target. The Fire command will only be called if the gun is cool, this is to prevent late fire where the target would clearly not be where calculated.

**Dodge-**

The dodge function is called every time the enemy is “detected” to have fired. This is determined by calculating the difference in the enemy energy over time, if the change is 3 energy or less it is assumed that the enemy fired. In the case the enemy fires the robot will take action change direction in an attempt to avoid the enemy fire.

**TargetLock-**

The target lock function is a basic flip flop design where the radar flip flops back and forth in direction going past the enemy by a small margin before switching direction and doing the same. In the even the robot somehow loses lock on the enemy it will revert to a 360-degree rotation scanning until it find the current target again and then returning to flip flopping.

**OnHitRobot-**

On collision with another robot, it immediately checks to see if the collision object is the enemy, if so it immediately fires in that direction, then backs up 50. If the State is Brawl it will attempt to ram again, if Ranged it will attempt to move away.

**Team Based options**

As this was originally a team-based assignment, I did include some team-based functions. These functions are there simply to coordinate targeting and focus fire. If there is no current target, the first scanned target will be broadcast, all robots will receive that broadcast and set that enemy to the current target assuring they all have the same target, then they will proceed to attack that target based on their own individual current states.