Trevor Aten and Jordan Smith

Clobberin' Time's Team Design

Strategy:

Our robot's strategy was very simple: don't shoot a teammate robot. To make this happen, our robots check the names on the event scan with the known names of the team members. If a scanned robot has one of these names, it will be ignored by the scanning robot and the scan will continue to find enemy robots.

Implementation:

Inside of both robots' OnScannedRobot event handlers, there is an if statement that decides if the robot that has been scanned is a robot of the same team using the name property from the scanned event.

Successes:

The robots do not go into their attacking behavior for team robots and therefore do not shoot directly at one another. Additionally, TrevBot's scans ignore team members and WreckItRalph is able to quickly find the next enemy to kill when the currently targeted one is destroyed.

Team communication and involvement was excellent. Both team members agreed on a goal that the robots should meet and we made that happen. Even with being fully remote, communication was easy and painless as both members were responsive to each other. Both team members made this project a priority and helped each other with any problems that arose.

Failures:

TrevBot:

TrevBot gets interrupted if it is locked onto another player and a team member passes the radar scan. This interruption causes it to go back into the SitAndScan state and requires TrevBot to relocate its enemy target. TrevBot isn't always accurate, so when it isn't, the missed shots will hit team bots. TrevBot has no collision detection with a robot of its own kind. So if TrevBot was to hit a bot on its own team, it would still continuously run into the bot. TrevBot does not have an input to return to the SitAndScan state when the bot it is currently attacking dies. This means that when the bot that TrevBot is currently attacking dies, TrevBot continues to move straight and shoot, eventually hitting a wall and continuously running into it.

WreckItRalph:

WreckItRalph has an elusive bug that causes the robot to sometimes suddenly lose track of the enemy robot, even if the enemy is right in front of it. Another failing is that while TrevBot's scans will pass right over team players, WreckItRalph will scan a robot and lock on, and only fire at it if it's an enemy. This issue was discovered very late in development, and fixing this issue would've needed a complete redesign of core elements of WreckItRalph's architecture, so it is unfortunate that the time constraints were such that this bug couldn't be fixed in time.

Despite these failures, we believe that because the two bots unique failures are not present in both bots, such as TrevBot's handling of scanning over team members and WreckItRalph's handling of finding the next enemy when the current one dies, that they are a good pairing for team play.

Trevor's Reflection:

I learned how to implement a finite state machine in software. I have had experience in the past creating finite state machines, but this was the first time I had to do it outside of hardware. This was my first time working with managing events in a system and this was a good learning experience to understand how events are handled. I did feel like my design was really structured to beat failbot, rather than any general robot, so I'm curious to see how that turns out for my robot. I feel like watching my robot perform against other robots will give me an idea of how to make my robot better going forward.

Jordan's Reflection:

I learned a lot about the usefulness and power of finite state machines while working on this project, even if they were a bit tough to code. I think that this project taught me a lot about how to structure an AI for a variety of situations, as it's not possible for me to know what kinds of techniques and tactics will be used against my robot, so I had to plan for the unexpected. I think that if I were to do this assignment again, I would approach it first from a team perspective in order to plan for things like ignoring ScannedRobotEvents from team robots without having to redo my FSM's architecture.