SETU Robocode 2025 Strategy

Authors: Emma Brennan

Sample Robot Analysis

“Tracker” – Aggressively charges the opponent and fires, obliterates most of the sample bots.

“Walls” – Crawls along the outskirts, pointing its scanner inwards and fires whenever it spots the opponent.

“Crazy” – Moves in alternating curves and fires when the enemy happens to be scanned.

“SpinBot” – Turns in a tight radius and fires on sight.

In a 1v1, the “Tracker” sample robot can defeat most of the other sample bots, except for the “Walls” – which defeats the Tracker robot consistently.

The addition of a Sentry to the battle complicates its strategy, as it will encounter the sentry bot along the outskirts of the arena.

Ideally, we want a bot that combines the manoeuvrability of Walls with the lethality of Tracker and can dodge the Sentry.

Scanner Strategy

Once we’ve scanned the sentry and saved its location, we don’t need to point the scanner at it again. So, the scanner should be reserved to keeping track of the opponent.

Since the scanner can move 45 degrees per turn, the most efficient scan would be 1 turn of 45 degrees to locate the opponent. Identifying whether the opponent is to the left or right of the scanner before moving it could save 2 turns: the 45 degrees back to the starting location, and the next turn to scan the other direction 45 degrees.

The disadvantage of having a bot in the Sentry radius is the requirement to move and dodge Sentry and opponent bullets, which reduces the opportunities for scanning.

Sentry Strategy

Staying in the middle 200x200 pixels of the arena nullifies the sentry threat but opens difficulties when duelling the opponent. For one, the increased risk of collisions, and two, the limited reaction time for dodging bullets. (Note that the tight turn radius of the SpinBot is able to dodge bullets at an extremely close distance.) There is a hope that the opponent will be silly enough to leave the safe zone and take fatal damage from the Sentry, but this is a strategy based on underestimating the opponent.

After testing, it appears viable to dodge the Sentry. The linear targeting assumes that the robot will move consistently, and breaking up the movement pattern every 100 pixels or so will seamlessly dodge most bullets. The wall-crawling robot is most likely to be hit if:

1. It is heading straight for the wall at the start of the round over a long distance
2. It picks a wall that puts it at an acute angle near the sentry.

Aiming Strategy

1. Fire-on-sight
2. Linear targeting (predictive)
3. Fire-on-sight + random variation

In tests thus far, the fire-on-straight strategy works well at hitting the opponent. It’s simple to implement and has a decent chance of hitting the opponent if:

1. The enemy is close or medium range (halfway across the arena)
2. The enemy does not move consistently in a perpendicular direction to our bot’s turret.

At some point, the opponents stand still to briefly scan, or fire, and these are usually opportunities for them to get hit by a fire-on-sight bullet. If the enemy is oscillating in one area, or moving in line with our turret, they are also likely to get hit. Inefficient scanning by the enemy also gives opportunities to hit them.

Implementing Linear targeting is much more complicated than Fire-on-sight. If the enemy robot can dodge the Sentry bullets, then it is possible that it can also dodge our own Linear targeting.

It is therefore important that any Linear targeting accounts for robots that consistently change direction before the predictive bullet could hit them. **Perhaps by tracking the accuracy of the predictive shots, and reducing the predictive future distance as the bullets are missed** could reduce the risk.

Distance-based strategy choice, or every x shot uses a different strategy could also introduce some variation into our bot’s firing strategy, to reasonably adapt to as many situations as possible.

Movement Strategy

Staying still in one place for too long usually ends with your robot being shot into scrap metal. Moving too much outside the Sentry radius results in a barrage of bullets headed for your bot. Moving inside the safe zone potentially turns it into a game of bumper cars.

**Perpendicular** or **circular** movement appears to be the most effective for dodging.

Effective Sentry dodging relies on distance and perpendicularity; the more perpendicular, the greater the dodge chance. Oscillating the movement results along with this results in a foil for linear targeting.

Implementation

The current strategy is to head directly for a wall, then turn parallel to it. Move in small increments in an oscillating motion while firing on sight. The trick is to time the oscillations so that the robot doesn’t run into the bullets fired by Sentry. To advance this strategy, the robot should move to a wall that is far from the sentry, and also not collide with the enemy on the way there.

Other considerations

Get Others() – find out how many participating robots and check that there is only 1 sentry bot. If there are 2 sentry bots we don’t want to leave the safe area, unless they spawned in an ideal location.

Survival and Shot Accuracy

Reduce the number of risky shots taken based on the battle scenario. (Eg. Long distance, opponent is moving a lot).

Record our hits and misses count, and change strategy based on this data.

Track the enemy health and compare to our health to see who is dealing more damage over time.