Joshua Windsor B4008182

Assignment 2 Report

Tools Libraries & Frameworks

**For this assignment, I used full source control using a private GitHub repository.**

**1&2:** I first split both teams into individual parameter files with individual player difficulties and implemented a three-tiered difficulty system, containing 3 difficulty settings per team (6 in total). I have changed the parameter passing code to contain a single static global parameter loader, holding all the parameters for the game rules, and a non-static parameter loader for each team. At the start of the game, each of the 12 parameter files are read into an array (lines 53-60 in SoccerPitch.cpp) with a pointer to the current difficulty parameters being held in the SoccerTeam class. This enables quick switching of difficulty using the 1-3 buttons for team A and q, w and e buttons for team B.

**3a:** I added a check to the dribble execution to use the isThreatened function which checks if any players are nearby, thus triggering the kickball state (lines 687-690 in FieldPlayerStates.cpp).

**3b:** I then edited the Kickball state to contain an else on the support category to wait for support if the player is close to the opposing goal and to dribble if not (lines 648-660 in FieldPlayerStates.cpp)

**4a:** I added 2 new team states (lines 182 onward in TeamStates.cpp) winning and losing. If the team is winning, it positions its players back in a closer defensive line. If the team is winning, the players take an attacking line and wait in the opponent’s home regions.

**4b**: I added code to determine which players are closest based on their role (lines 127-144 in SoccerTeam.cpp). For attackers, they will chase the ball if it is on the opponent’s side and for defenders, they will stay back and only chase if the ball is on the team’s side of the field. I also implemented a system where the defenders will shoot forward to attackers should they come close to the middle of the pitch.

**4c:** I implemented a system to allow all defenders to chase the ball when it is on their side of the pitch, effectively surrounding any attacking players with the ball.(lines 491-501 in FieldPlayerStates.cpp)

**4d:** I implemented a Mark state which checks if any players are close to the goal (lines 892 – end on FieldPlayerStates.cpp) and sets the defending players to follow that player until the ball gets within the home region.

**5:** I added 2 new tests to the spot scheme class (lines 129 – 139 in SupportSpotCalculator.cpp) which checks if the player is in front of the current attacking player and adds score if it is. Another test I added finds the number of opponents around the spot and removes score based on this amount as the ideal spot should be far away from opponents.

**6:** I added a stamina system which drains and refuels based on the speed of the player (lines 111-128 in FieldPlayer.cpp) and a new state of ‘Fatigued’ which causes the player to stop to regain stamina (lines 360- 400 in FieldPlayerStates.cpp). It then reduces the possible power and speed the player can go depending on this level. The Fatigued state allows the player to wait and regain their stamina, while another player then comes to take over the ball.

**Extra A:** I edited the code for question 1 and 2 to enable switching during the game using the previously specified keys. I also added a system tied into question 4a which edits which parameter file is selected based on the number of goals each team has. If the team is winning, it will switch to a lower difficulty, if losing, it will increase the difficulty level and if the two teams are drawing, it will reset the difficult back to normal.

**Extra E:** I implemented a Referee class which inherits from player base with an overloaded constructor. The referee has its own state system and rotates around the ball while moving to keep distance and not interfere with the action.

**Reflection:** I think my AI overall was very well implemented but a few improvements could be made: Firstly, the system which I implemented for question 4b means that the defenders will sometimes dribble the ball up to the halfway line before leaving the ball for the opponent. To solve this, I added an overhang which makes the player pass when getting close to the line, although they still leave it should they not be able to pass. The most significant change I believe is the hemming tactics as it stops most attacks from the teams by overrunning the attacking player. In combination with the marking, the defenders are almost surrounding the player already meaning even more effective defence. I would also extend the project to have a further difficulty mode with removable features using the debug menu. I felt that the scale of this game did not benefit from the team level parameters as there is no real ‘team’ level parameters and teams with a hard team level but some easy level players seemed to just level out at medium level overall, I however, left the code in to satisfy question 2b.