What we accomplished:

For this iteration, we started by fixing some bugs that were present when moving a robot, so this functionality works much better now. Next, we implemented the appearance of the complex board. The complex board can now be displayed with the robots appearing on it. However, the complex board cannot be used for gameplay yet since moving a robot on the complex board has not yet been implemented.

Also, there is now a Help window containing instructions on how to move a robot and how to demonstrate a solution. This can be accessed by clicking on HELP from the opening main menu, or by clicking on the Help drop down menu and then clicking on the “How to?” option after the game has started. The LOAD and STATS options from the main menu still just show dummy frames for now.

We implemented our use case of demonstrating a human solution this iteration as well by creating a panel to the right of the simple board which handles most of this functionality. This panel displays an image of the selected target space for the current round, the timer, and the player’s current scores. This panel also includes buttons for players to select which player they are before entering a bid and a text box to enter their bid into, a bid meaning the number of moves in their solution.

We also implemented our score verification use case. This includes first checking if the current solution is valid by checking if it has more than one move, and therefore has ricocheted at least once, and then by checking if the actual number of moves made so far is less than or equal to the bid made by the player. If the solution is invalid, the robots will reset back to their original positions. Then players can bid again and another player can try. If the solution is valid, the appropriate player’s score will be incremented by 10 points and the next round will begin.

You can test the Human Solution Demonstration and Score Validation use cases by clicking on a player button, entering a bid, and then attempting to move the correct robot to the selected target space. Further instructions on this are available in the Help window.

Note: Resetting the robots to their original position after a failed solution still contains certain bugs, particularly when playing more than one round. However, the main functionality of both use cases for this iteration are working. The code will be further optimized in the next iteration.

<u>How to compile and execute the .java files:</u>

(1) Download ‘Group6RicochetRobots.zip’

(2) Ensure the file hierarchy is maintained during extraction.

(3) Open Terminal/Command Prompt.

(4) Navigate to inside of the ‘Group6RicochetRobots’ folder where .java files are located.

(5) Compile the code using: **javac *.java**

(6) To run, use the code: **java Game**

<u>How to run the .jar file in ‘Group6RicochetRobots’ folder:</u>

(1) Download ‘Group6RicochetRobotsWithJar.zip’

(2) Ensure the file hierarchy is maintained during extraction.

(3) Open Terminal/Command Prompt.

(4) Navigate to inside of the ‘Group6RicochetRobotsWithJarFile’ folder where .java files and the RicochetRobots.jar file are located.

(5) To run the .jar file, use the code: **java -jar RicochetRobots.jar**