**Training Method**

The robot is trained to have two separate **skills**—a turning skill and a moving skill. The turning skill model predicts whether turning clockwise or counterclockwise is a good idea. The moving skill model predicts whether moving forward is a good idea. Both skills are trained using a combination of Reinforcement Learning ideology as well as standard classification model algorithms. Labels for training are either 1 or -1—good or bad.

In this simulated Matlab environment, there exists two separate objects. One is a **Robot** object that has a floating point (x, y) coordinate as well as a pointing angle (0⁰ - 360⁰) that can be turned clockwise or counterclockwise. The Robot can also move a given distance in the direction of its current pointing angle. The other object is a **Destination** object that only has a floating point (x, y) coordinate. The goal of this is to get the Robot to navigate itself to the Destination using Machine Learning.

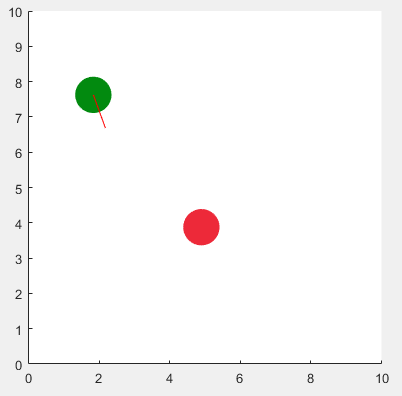


Figure : Simulated Matlab environment showing Robot (green) and Destination (red)

The turning skill is trained by generating 1000+ random locations for both the Robot and Destination as well as initial pointing angles. The computer calculates the angle between the Robot and the Destination—the difference between the Robots current pointing direction compared to where the Destination is. The robot then turns either clockwise or counterclockwise. If that turn reduces the angle between the Robot and the Destination, it is labeled 1—“good”. Otherwise, it is labeled -1—“bad”. The vector between the Robot and the Destination, the Robot’s current pointing angle, the current angle between, and the turn direction (CW/CCW), and the good/bad label are all record and a Decision Tree Classification algorithm is used to train a model to determine whether a CW/CCW turn is good or bad.

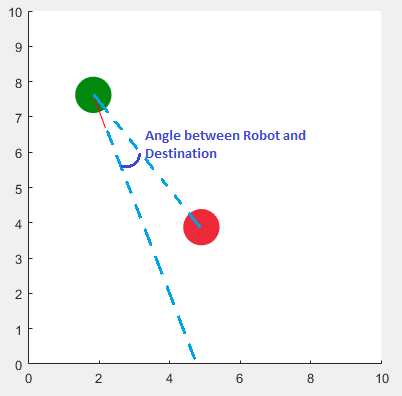


Figure : Depiction of what the "angle between" the Robot and Destination is

The training of the movement skill is essentially the same except it is trained to determine whether moving the Robot forward will reduce the distance between the Robot and the Destination, as opposed to reducing the angle between. The vector between the Robot and the Destination, the Robot’s current pointing angle, and the good/bad label are all record and a Decision Tree Classification algorithm is used to train a model.

**Using the Skills**

To *use* the skills, the computer creates a features vector for each possible action: turning clockwise, turning counterclockwise, and moving forward. It then asks both skill models if any of these decisions are good decisions to make given the current environment. If an action is deemed a *good* decision by the models, then the computer executes that action. After every action, the computer updates the feature vectors for the new environment and checks potential decisions again. This loop happens again and again until either the Robot makes it to the Destination or until the Robot gets stuck.

**Skill Hybridization?**

Currently, the two skills are working towards the same goal—getting the Robot to the Destination—however, they aren’t exactly aware of one another. I wonder if a valuable, further step would be to experiment with **hybridizing** the two skills together—to create another model whose skill is to understand the best way to get the separate skills to work together in sync.