In Q-Learning, there’s a Q matrix that provides a particular state with an optimal action for that state. This matrix can be written as Q(s,a) which is a state-action pair.

In our task, the target is launched in a 2D plane and follows some ballistic trajectory. A short while after that, a catch stimulus attempts to intercept the target.

For our purposes, a state can be the x,y coordinate of the catch stimulus. Actions can be as simple as – Up, Down, Left, Right, DR, UR, DL, UL. At each state, a particular action is taken.

We can also update the state to include the target properties to make the program generalizable. For example, the state can be the difference in position between the target and the catch stimulus. So x & y can be either positive or negative.

An extension of the state variables can include a “velocity” component in the future so that the catch stimulus can accelerate or decelerate to intercept the target.

The Q(s,a) doesn’t have to to be very massive in size as we had initially thought. Perhaps just 3 dimensions (diffX, diffY, action)

**Next steps**: How to update the Q matrix based on rewards.

The reward will initially be delivered when the catch stimulus intercepts the target. As the training progresses, the rewards will propagate back through the state-actions to provide reinforcement to each action.

Things to think about

Our learning should be able to deal with many different types of trajectories.