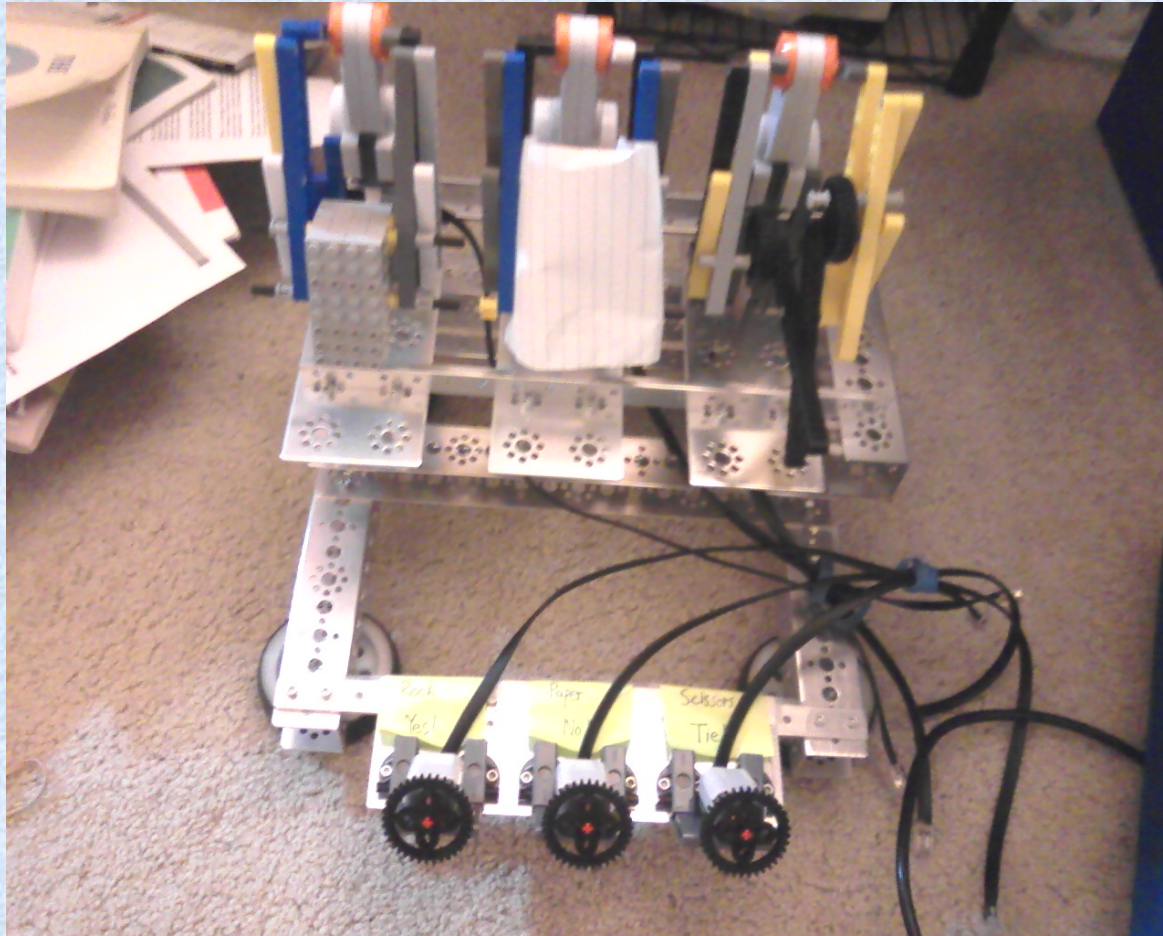


Self-Learning Rock Paper Scissors Robot



Bryant Pong

Q-Learning Algorithm

$$Q(s_t, a_t) \leftarrow \underbrace{Q(s_t, a_t)}_{\text{old value}} + \underbrace{\alpha_t(s_t, a_t)}_{\text{learning rate}} \times \left[\underbrace{R(s_t)}_{\text{reward}} + \underbrace{\gamma}_{\text{discount factor}} \underbrace{\max_{a_{t+1}} Q(s_{t+1}, a_{t+1})}_{\text{max future value}} - \underbrace{Q(s_t, a_t)}_{\text{old value}} \right]$$

Four variables needed (all variables are [0, 1]):

Alpha (learning rate): higher alpha is to 1, robot learns faster, but incorrect moves will affect the robot more

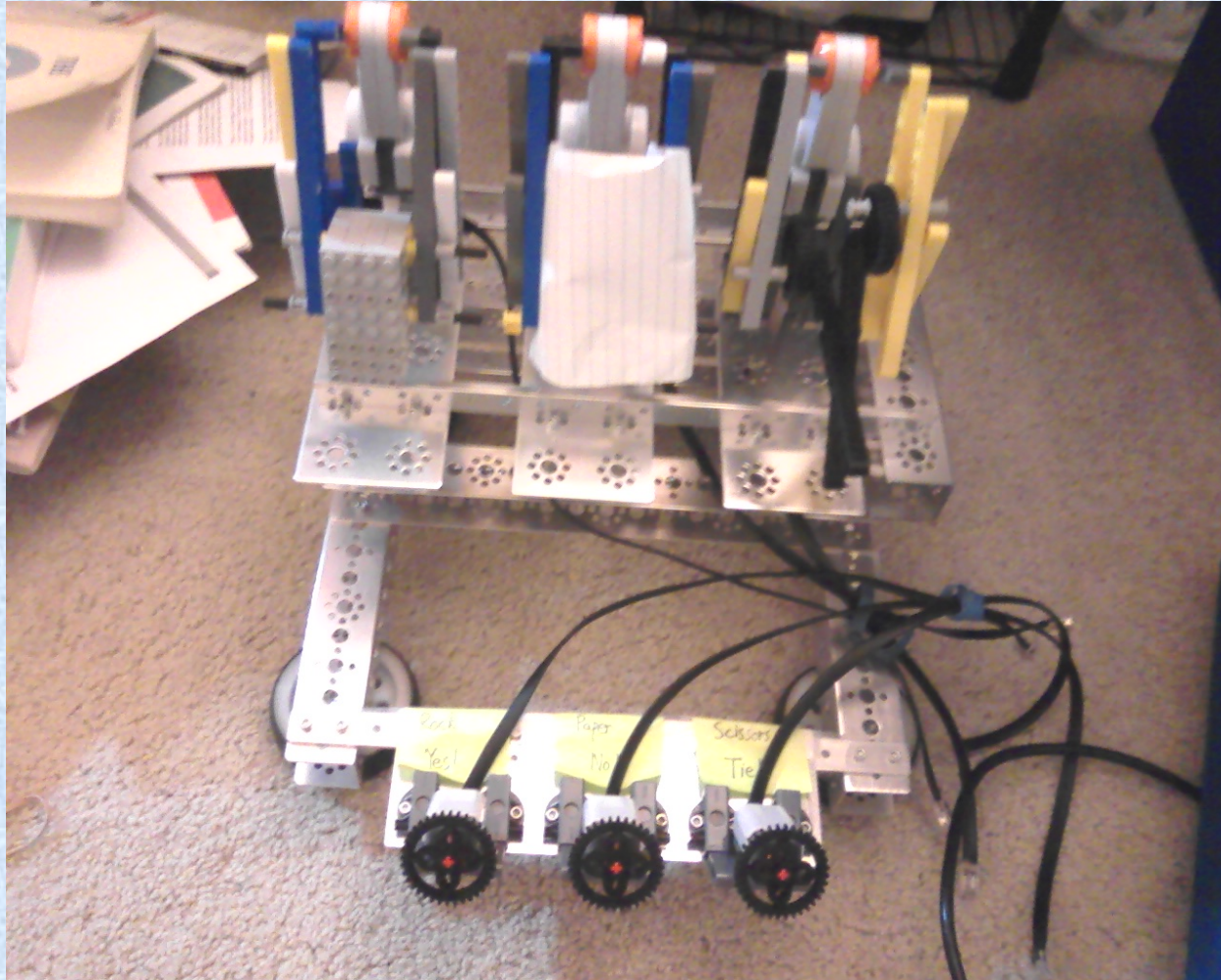
Gamma (discount factor): higher gamma is to 1, robot will use learned information rather than performing random moves

Epsilon (probability of a learned action): epsilon starts off at 0.999, decrements after robot performs a move

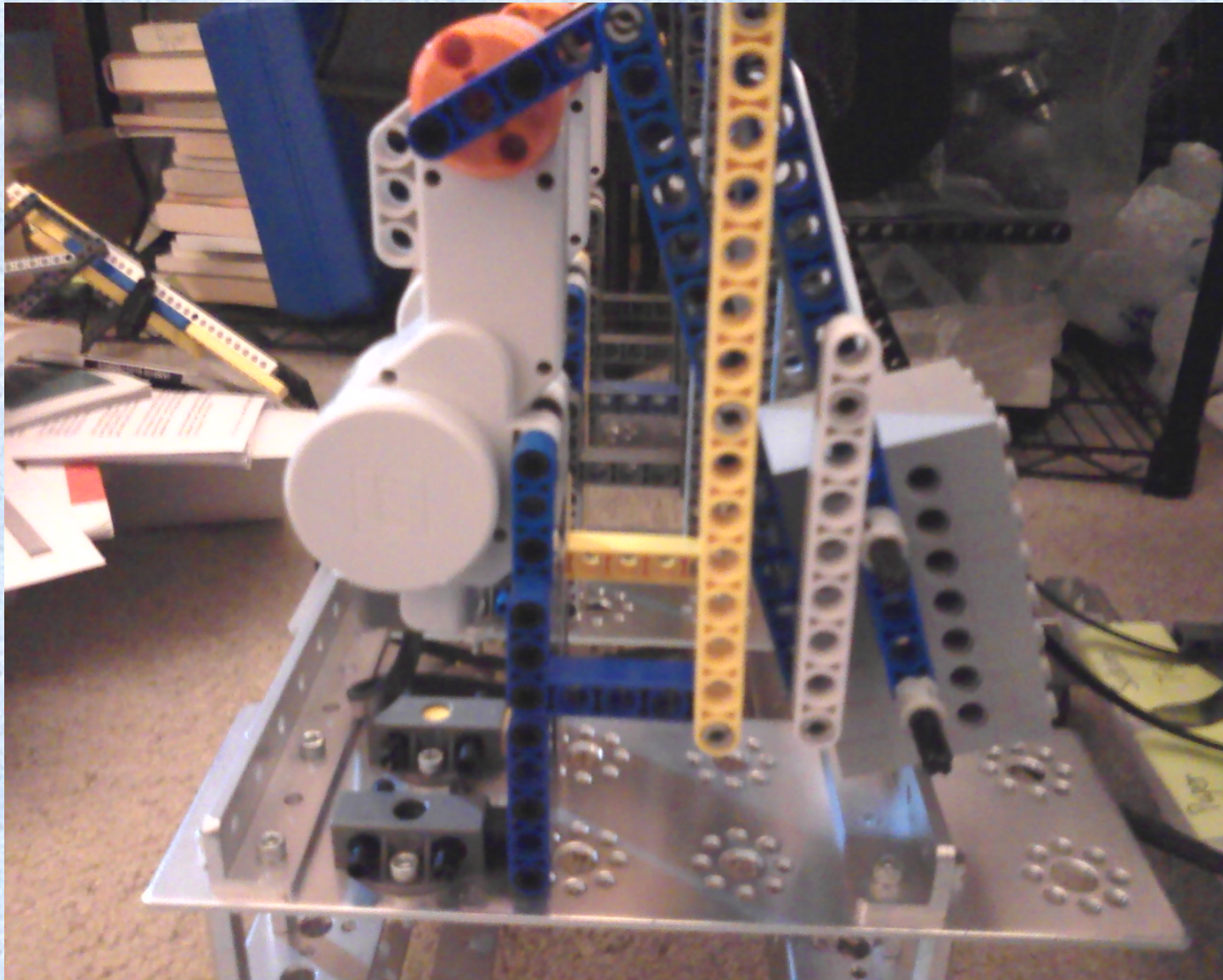
Kappa (thoroughness variable): kappa ensures that more accurate values are recorded for the future

The Robot

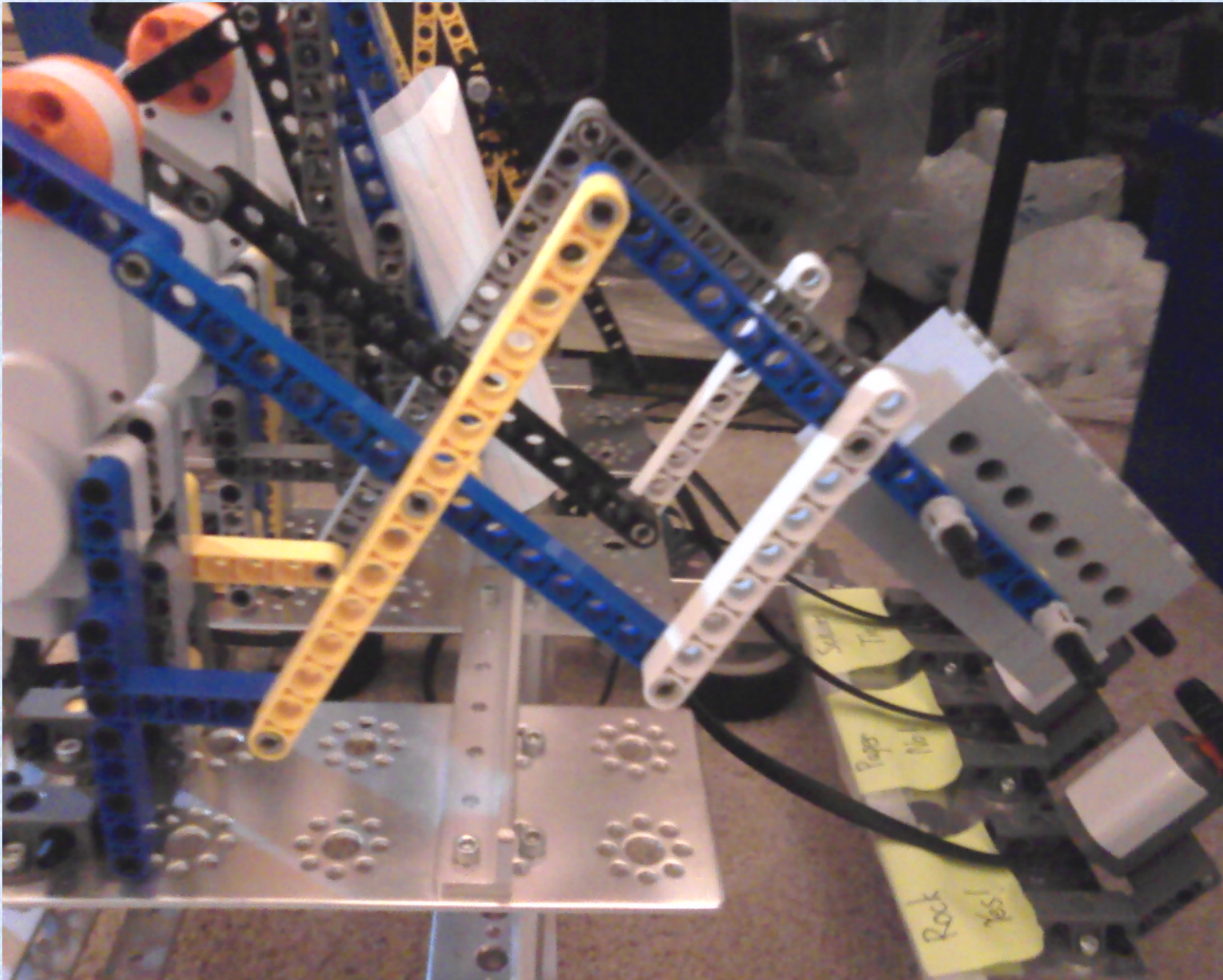
- 3 Touch Sensors/3 Motors



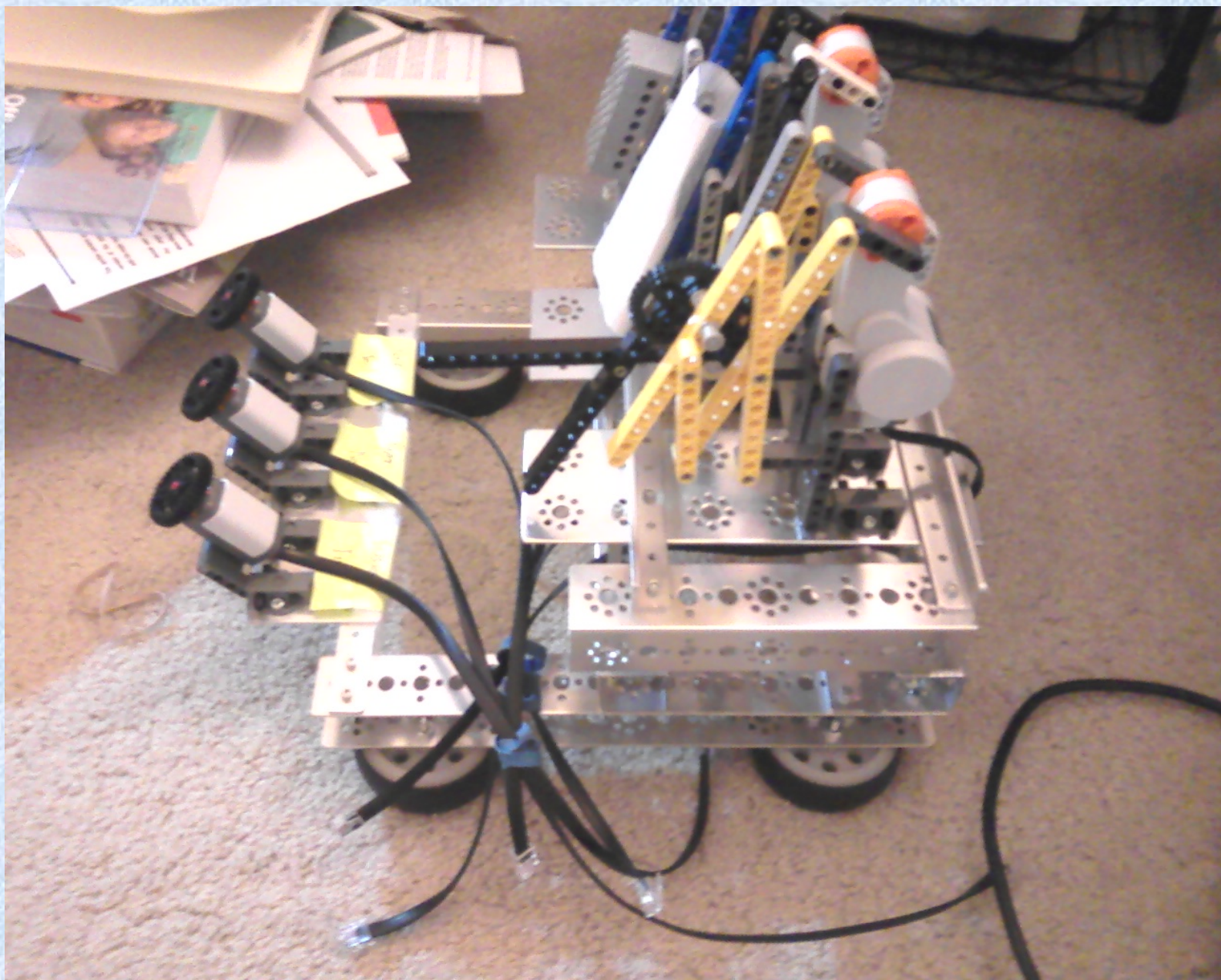




Rock (Retracted)

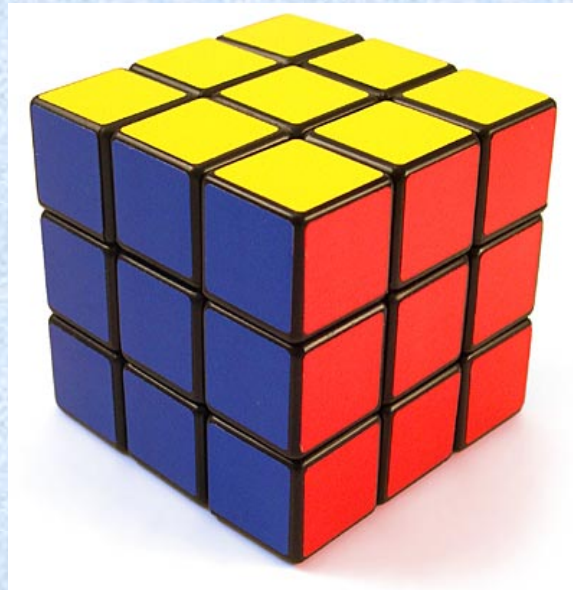


Rock (fully extended)



Decision Making

- **Markov Chains**
- 3 x 3 x 3 array (like a Rubik's cube)
 - 1st Dimension: **Move Player Made (rock, paper, scissors)**
 - 2nd Dimension: **Move Robot Made (rock, paper, scissors)**
 - 3rd Dimension: **Result (robot won, lost, or tied?)**
- Array stores probability of success for a given state



How does the robot make a decision?

```
if(norm_random() > epsilon)
{
    //Use learned information
    ClearScreen();
    TextOut(0, LCD_LINE1, "I'm thinking...");
    Wait(1000);

    //loop through all possible choices
    for(i = 0; i < CHOICES; i++)
    {
        //use the lookup table to determine what the best
        //move to make to counter the player's move
        if(qscores[initial_input][i] > max_q_score)
        {
            //find the move with the highest success rate
            //and store it
            max_q_score = qscores[initial_input][i];
            next_movement = i;
        }
    }
}
```

Reward System

```
if((i == 0 && j == 0) || (i == 0 && j == 2) || (i == 1 && j == 0) || (i == 1 && j == 1) ||  
    (i == 2 && j == 1) || (i == 2 && j == 2))  
{  
    reward = -10000;  
}  
  
//otherwise, the robot must win; reward it with a virtual point!  
else  
{  
    reward = 1;  
}
```

How does the reward system work?

- Robot gains *virtual points* when it wins; loses **10,000** points when it ties/loses
- Example: Give a kid \$20 if he can learn to fly
- Kid tries to crawl, walk, run, and jump to fly
- Eventually, kid learns that he cannot fly
- Along the way though, kid learns how to crawl, walk, run, jump

