Simple RL Learning Roadmap - Week by Week

🎯 Your Goal: Get your robot to follow objects using RL

Week 1: Learn RL Basics (No Code Yet!)

Day 1-2: Watch These Videos (Total: 3 hours)

1. **Deep RL Course Unit 1** - What is RL? (1 hour)
2. **Deep RL Course Unit 2** - Q-Learning basics (2 hours)

Day 3-4: Read This Book Chapter

Book: "Reinforcement Learning: An Introduction" by Sutton & Barto

- **Chapter 1: Introduction** (free PDF: http://incompleteideas.net/book/RLbook2020.pdf)
- Skip math, focus on concepts

Day 5-7: Hands-On Practice

Follow Gymnasium's "Training an Agent" tutorial EXACTLY:

1. Go to: https://gymnasium.farama.org/tutorials/training_agents/
2. Copy-paste their code
3. Run it step by step
4. Play with Blackjack environment

python

```python
# Just run this first - don't modify anything
import gymnasium as gym
from collections import defaultdict
import numpy as np

env = gym.make('Blackjack-v1', render_mode='human')
observation, info = env.reset()
print(f"Initial observation: {observation}")
```

Week 2: Practice with Simple Environments

Day 1-3: Master CartPole

Follow this exact tutorial:

- **Stable Baselines3 Getting Started**

python

```
# Install first
pip install stable-baselines3[extra]

# Then run this exact code
import gymnasium as gym
from stable_baselines3 import PPO

env = gym.make("CartPole-v1", render_mode="human")
model = PPO("MlpPolicy", env, verbose=1)
model.learn(total_timesteps=10000)

# Test your trained agent
obs, info = env.reset()
for i in range(1000):
    action, _states = model.predict(obs, deterministic=True)
    obs, reward, done, truncated, info = env.step(action)
    if done or truncated:
        obs, info = env.reset()
env.close()
```

Day 4-7: Try Different Environments

Same code, just change the environment:

- `LunarLander-v2`
- `MountainCar-v0`
- `Acrobot-v1`

Goal: See how the same algorithm works on different problems

Week 3: Create Your Robot Environment

Day 1-2: Setup Your Python Environment

bash

```
pip install gymnasium stable-baselines3[extra] websocket-client
```

Day 3-5: Build Basic Robot Environment

Create this file: `robot_env.py`

python

```python
import gymnasium as gym
from gymnasium import spaces
import numpy as np
import websocket
import json
import time

class RobotEnv(gym.Env):
    def __init__(self):
        # Actions: left wheel speed, right wheel speed (-1 to 1)
        self.action_space = spaces.Box(low=-1, high=1, shape=(2,))

        # Observations: 4 distance sensors + 4 IR sensors
        self.observation_space = spaces.Box(low=0, high=2000, shape=(8,))

        # Connect to your robot
        self.ws = websocket.WebSocket()
        self.robot_data = {}

    def reset(self, seed=None):
        # Connect to robot
        try:
            self.ws.connect("ws://192.168.4.1/ws")
        except:
            print("Can't connect to robot - using fake data")

        # Stop robot
        self.send_action([0, 0])
        time.sleep(0.1)

        # Get initial observation
        obs = self.get_observation()
        return obs, {}
```

```
def step(self, action):
    # Send action to robot
    self.send_action(action)

    # Wait a bit
    time.sleep(0.1)

    # Get new observation
    obs = self.get_observation()

    # Calculate reward (simple version)
    reward = self.calculate_reward(obs)

    # Episode ends if too close to wall or IR sensor triggered
    done = obs[0] < 100 or obs[1] < 100 or obs[2] < 100 or obs[3] < 100

    return obs, reward, done, False, {}

def send_action(self, action):
    # Convert action to robot commands
    left_speed = action[0] * 30  # Your max speed
    right_speed = action[1] * 30

    # Calculate x, y for your joystick format
    forward = (left_speed + right_speed) / 2
    turn = (right_speed - left_speed) / 2

    x = turn / 30
    y = -forward / 30

    command = {"type": "control", "x": x, "y": y}
```

```
        try:
            self.ws.send(json.dumps(command))
        except:
            pass  # Robot not connected

    def get_observation(self):
        # Get sensor data from robot
        # For now, return fake data - you'll connect this to your WebSocket
        return np.array([500, 500, 500, 500, 0, 0, 0, 0], dtype=np.float32)

    def calculate_reward(self, obs):
        # Simple reward: stay away from walls but not too far
        min_distance = min(obs[0:4])  # Minimum ToF reading

        if min_distance < 100:
            return -10  # Too close to wall
        elif 200 < min_distance < 800:
            return 1    # Good distance
        else:
            return -1   # Too far
```

Day 6-7: Test Your Environment

python

```python
# Test your environment
from robot_env import RobotEnv

env = RobotEnv()
obs, info = env.reset()
print(f"Initial observation: {obs}")

for i in range(10):
    action = env.action_space.sample()  # Random action
    obs, reward, done, truncated, info = env.step(action)
    print(f"Step {i}: obs={obs}, reward={reward}")
    if done:
        break
```

Week 4: Train Your Robot

Day 1-3: Connect Real Robot Data

Modify your `get_observation()` method to parse real sensor data from your ESP32

Day 4-7: Train Your First Robot Policy

python

```python
from stable_baselines3 import PPO
from robot_env import RobotEnv

# Create environment
env = RobotEnv()

# Train model
model = PPO("MlpPolicy", env, verbose=1)
model.learn(total_timesteps=5000)

# Test trained model
obs, info = env.reset()
for i in range(100):
    action, _states = model.predict(obs, deterministic=True)
    obs, reward, done, truncated, info = env.step(action)
    if done:
        obs, info = env.reset()
```

📚 Resources to Use (In Order)

Week 1 Resources:

1. **Hugging Face Deep RL Course** - Units 1-2 only
2. **Sutton & Barto Book** - Chapter 1 only

Week 2 Resources:

1. **Stable Baselines3 Quickstart**
2. **Gymnasium Basic Usage**

Week 3-4 Resources:

1. **Custom Environment Guide**
2. **Your ESP32 code** (for WebSocket communication)

🎯 Success Metrics

Week 1: You understand what RL is and can run Blackjack example **Week 2:** You can train CartPole and see it working **Week 3:** Your robot environment runs without errors **Week 4:** Your robot moves based on RL policy (even if not perfect)

🚨 Important Rules

1. **Don't read everything** - Follow only the specific tutorials mentioned
2. **Don't modify code initially** - Copy-paste exactly first
3. **One week at a time** - Don't jump ahead
4. **Test each step** - Make sure it works before moving on
5. **Keep it simple** - Don't add fancy features yet

Start with Week 1 today. Don't look at Week 2 until you finish Week 1.