MTP

Arshdeep Singh
2019csm1001
Supervisor : Dr. Shashi Shekhar Jha

Topic

Multi-UAVs (Unmanned Aerial vehicles) Safe Navigation through Multi Agent Reinforcement Learning

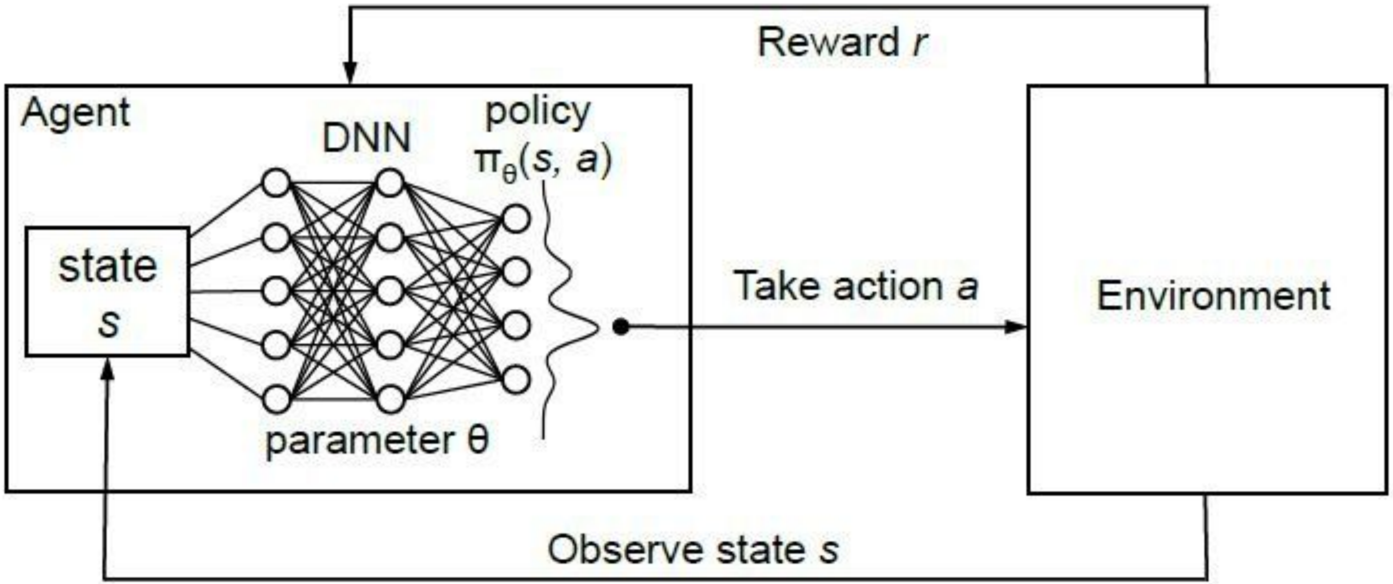
Multi-UAV Applications

1. Mapping and Surveying
2. Crop Monitoring
3. Emergency Response

Formulation

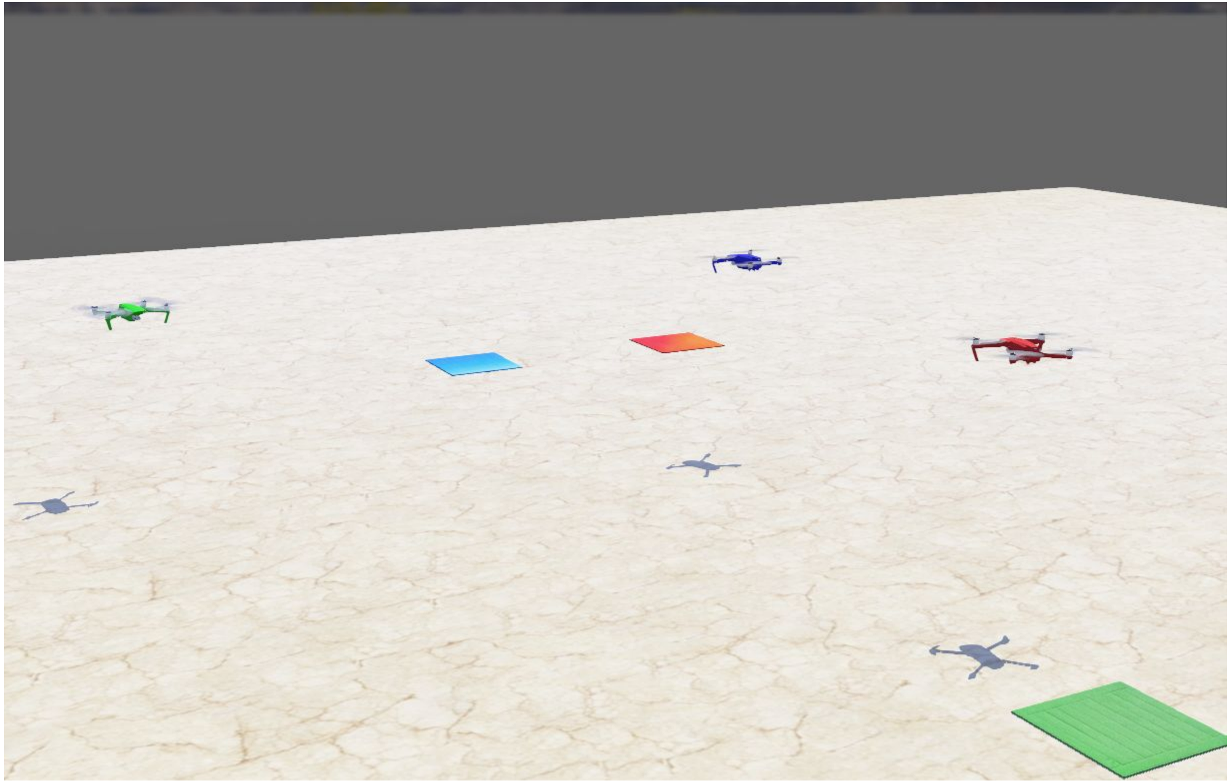
1. Multi Agent Reinforcement Learning is when we are considering various intelligent agents interacting with an environment
2. Multi Agent Systems is still an under-explored area of reinforcement learning
3. Multi Agent scenario
   a. Cooperative
   b. Competitive
   c. Mixture of both

RL world



RL feedback loop[1]

Continued



Formulation

1. Actor-Critic Methods
2. Each agent is modelled as an Actor
   a. Updates its policy as suggested by the Critic
3. The Critic estimates the values of the state
   a. It guides the actor to prefer good actions over bad ones

MADDPG (Lowe *et al.*)[2]

1. Multi Agent Deep Deterministic Policy Gradient
2. Centralized Critic network
3. Decentralized execution (multiple independent actors)
4. Deterministic Policy function

Architecture

- An actor network for using observations for *deterministic* actions
- An identical target actor network for training stability
- A critic network that uses *joint* states action pairs to estimate Q-values

Issues

- No obstacle avoidance : static, dynamic
- No safety guarantees
- Large state space configuration
- Techniques :
    - Safety Controllers[3]
    - Control Barrier functions[4]

Safe Reinforcement Learning

1. Learning optimal policies while satisfying Safety Constraints
2. Safe RL will help in transitioning the control to physical systems
3. A major challenge in using reinforcement learning is safety - control policies learned using reinforcement learning typically do not provide any safety guarantees.

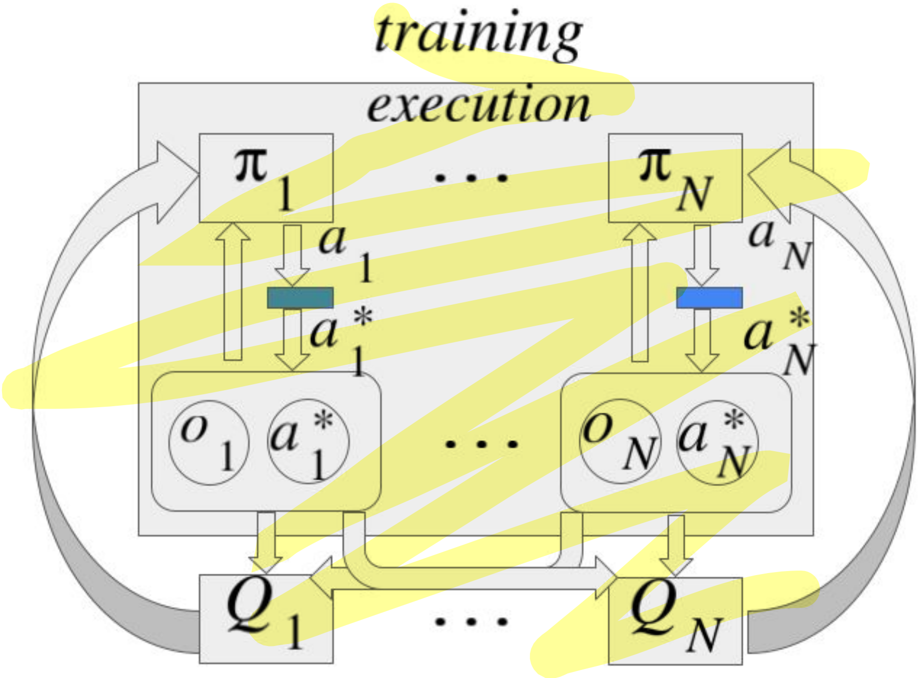
Safety Layer (Dalal *et al.*)[3]

1. Modifications to original policy
2. Learning immediate constraint functions
3. After Safety Layer is trained, safe actions *a** will be used by Agents(UAVs):

$$a_i^{\,*} = arg\,\min_{a|i} \frac{1}{2} \parallel a_i - \mu_{i|\theta}(o) \parallel^2$$

$$c_k(o_i, a) + g(o, w)^T a_i \leq C_k, \forall k \in K$$

Safe-MADDPG



Safe-MADDPG

STEP 1:

while *epochs* $\leq$ *TOTAL EPOCHS* **do**
  while *step* $\leq$ *TOTAL SAMPLE STEPS* **do**
    Reset Environment(state = $o_i, \forall i \in N$);
    Calculate constraint values c_i;
    Execute actions $(a_1, ..., a_N)$ and observe reward r_i and new obs o_i' for each agent $\forall i \in N$;
    Calculate constraint values ${c_i}^{next}$;
    Store $(a_i, o_i, c_i, {c_i}^{next})$ in replay buffer D;
  end

Safety Network update:

  while *update via mini-batch* **do**
    Sample a mini-batch for each agent $i \in N$: $(a_i, o_i, c_i, {c_i}^{next})$;
    Observe safety layer : $g_i(o_i, w_i)$;
    Predict next constraint values;
    ${c_i}^{next.predicted} = c_i + g_i(o_i, w_i)a_i$;
    Calculate loss and update safety layer weights;
    $L^k = \parallel {c_i}^{next} - {c_i}^{next.predicted} \parallel^2, \forall k \in K$
  end
end

Safe-MADDPG

STEP 2:

while *episode number* $\leq$ *TOTAL EPISODES* **do**

Initialize exploration noise;

Reset environment (state = o_i, $\forall i \in N$);

while *timestep(t)* $\leq$ *TOTAL EPISODE LENGTH* **do**

For each agent i, select action $a_i = \mu_{i|\theta}(o_i) + \mathcal{N}_t$ w.r.t. the current policy and exploration;

calculate lagrange multipliers : $\lambda_i{}^* = \frac{g(o_i,w_i)^T \mu_{i|\theta}(o_i) + c_i(o_i) - C_i}{g(o_i,w_i)^T g(o_i,w_i)}$;

correct each agent's action as : $a_i{}^* = \mu_{i|\theta}(s) - \lambda_i{}^* g(o_i, w_i)$;

Execute corrected actions $(a_1{}^*, ..., a_N{}^*)$ and observe reward r_i and new obs o'_i for agent i, $\forall i \in N$;

Store $(o_i, a_i{}^*, r_i, o'_i)$ in replay buffer D;

$o_i \leftarrow o'_i$;

while *agent i = 1 to N* **do**

Sampling of a randomized minibatch of M samples (o^j, a^j, r^j, o^j) from D;

set $y_i{}^j =$ $r_i{}^j + \gamma Q_i{}^{\mu'}(o'^{\,j}_i, a'_1, ..., a'_N)|_{a'_i = \mu'_{i|\theta}(o_i{}^j)}$;

Update of the critic by minimization of the loss $L(\theta_c) =$ $\frac{1}{M}\sum_j (y_i{}^j - Q_i{}^{\mu}(o_i{}^j, a_1{}^j, ..., a_N{}^j))^2$;

Update of the actor by the use of sampled policy gradient;

$\nabla_{\theta_i} J \approx \frac{1}{M}\sum_j \nabla_{\theta_i}\mu_i(o_i{}^j)$

$\nabla_{a_i} Q_i{}^{\mu}(x_i{}^j, a_1{}^j, ..., a_N{}^j)|_{a_i = \mu_i(o_i{}^j)}$;

end

Update of the target network parameters for each agent i;

$\theta'_i \leftarrow \tau\theta_i + (1-\tau)\theta'_i$;

end

end

Environments

1. Multiple UAVs and their Target locations
2. Quadcopter: DJI Mavic
3. Multi-Agent Particle Environments(MAPE) by OpenAI (2D)
4. Webots Simulation by Cyberbotics (3D)
   a. Real world physics engine
   b. Deepbots[3]: Deepbots is a simple framework which is used as middleware between the Webots Simulator and Reinforcement Learning algorithm
   c. Emitter-Receiver scheme
   d. Combined robot-supervisor controller scheme

Continued..

- Observation Space(UAVs and Targets):

  Position of each entity: $P_i : [{p_i}^x, {p_i}^y, {p_i}^z]$

  Vel. of each UAV: $V_i : [{l_i}^x, {l_i}^y, {l_i}^z, {v_i}^x, {v_i}^y, {v_i}^z]$

  Concatenated Tuple $\{P^i, V^i, P^i_{Target}, \bigcup_{\forall j \in N; j \neq i} P^j\}$

- Action Space:

  For each UAV: $a_i : \{pitch_i, yaw_i\}$

- Reward Structure:

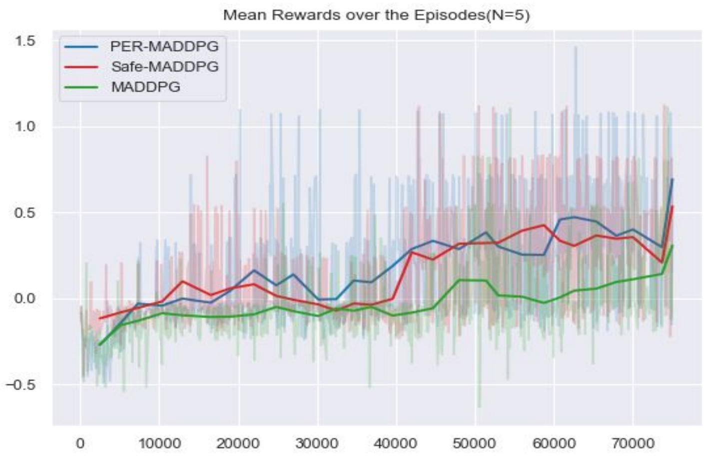
  Distance, Angular and Collision Penalty

Reward Value	Conditions
-4	if the angle between agent and its target is $>$ 1.5 Radians
-2	if angle $<$ 1.5 Radians and $curr\_dist > prev\_dist$
-2	if angle $<$ 0.1 Radians and $curr\_dist > prev\_dist$
$+5$	if angle $<$ 1.5 Radians and $curr\_dist < prev\_dist$
$+10$	if angle $<$ 0.1 Radians and $curr\_dist < prev\_dist$
-10	if there is a collision

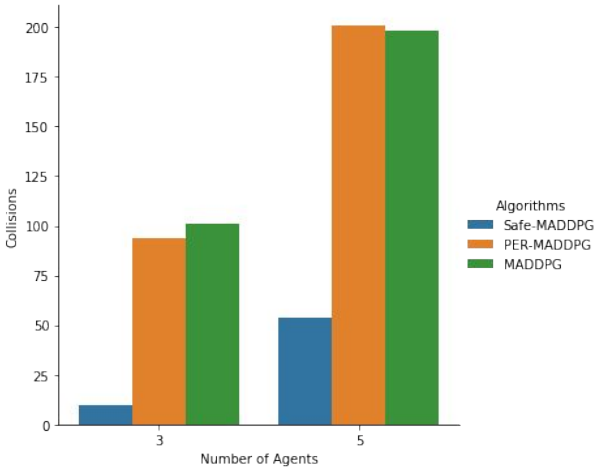
Results of 2D MAPE environment

Evaluation

- Mean Rewards

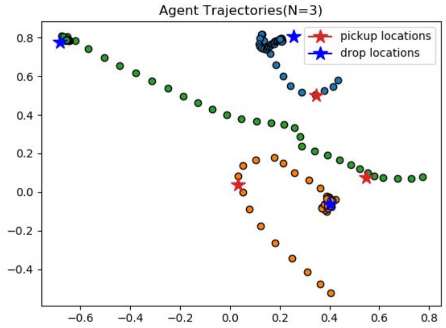


- Testing final Policies

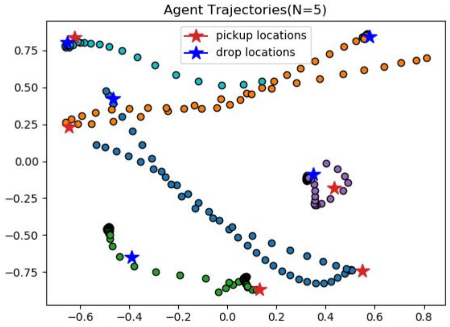


Trajectory Plotting(N=3) and (N=5)

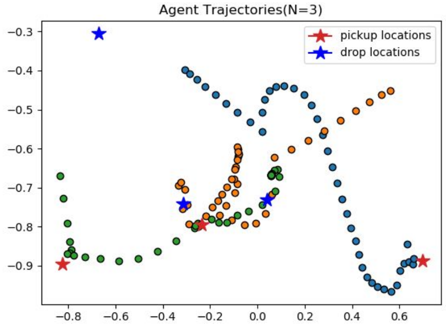
Max. reward : 56.19



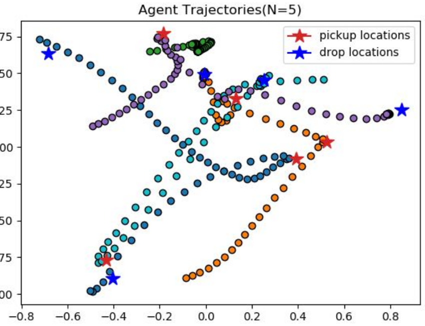
Max. reward : 88.02



Min. reward : 11.02

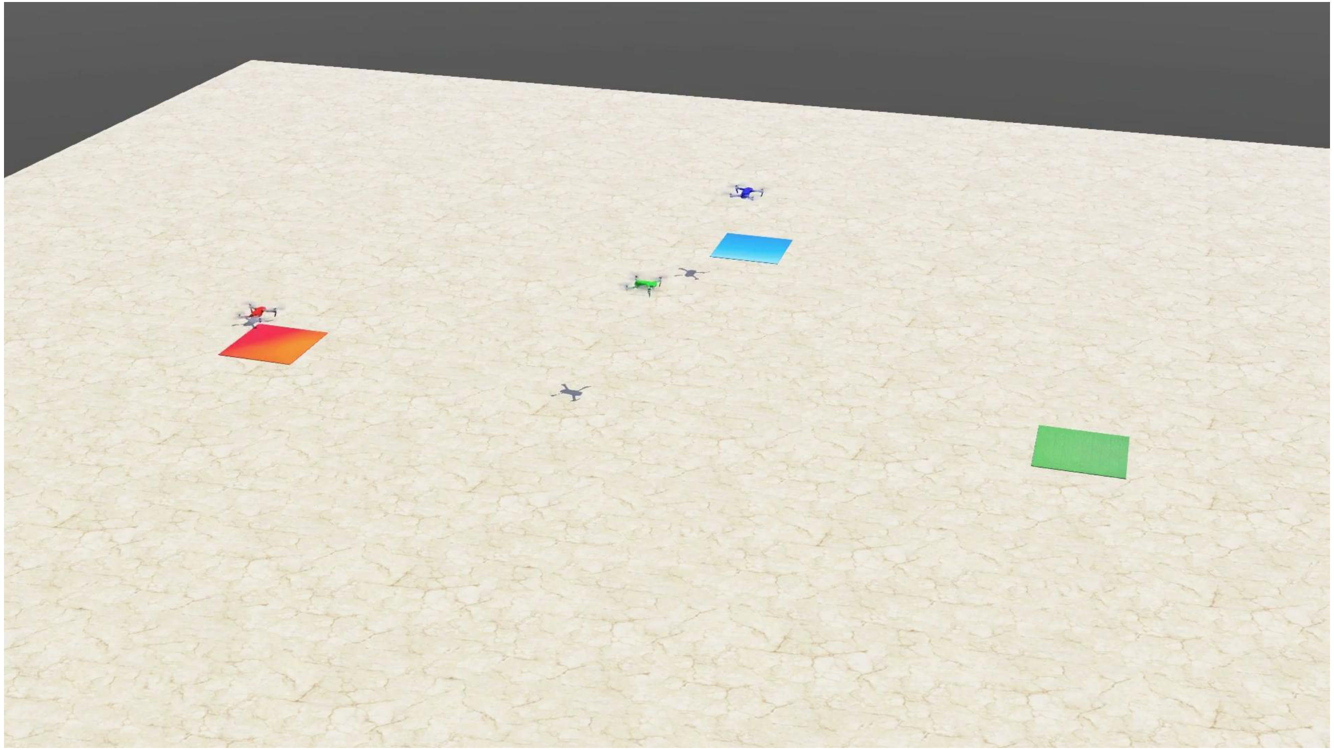


Min. reward : 24.11



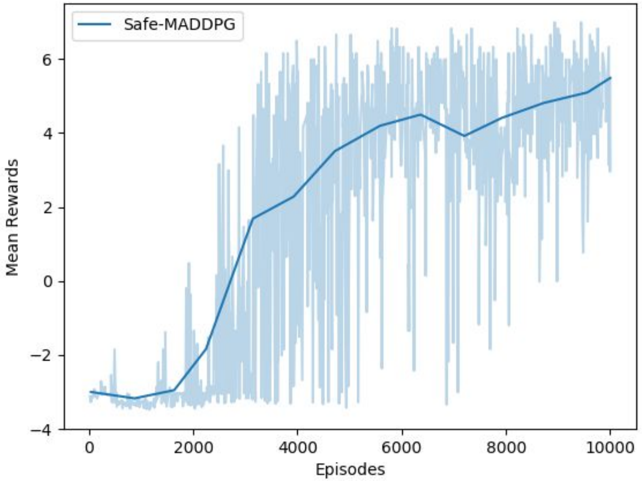
Results of 3D Webots environment

3-Agent Scenario

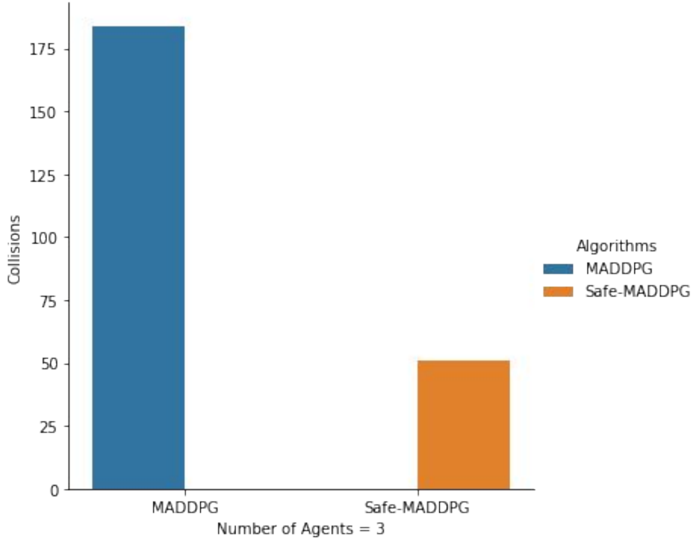


Evaluation

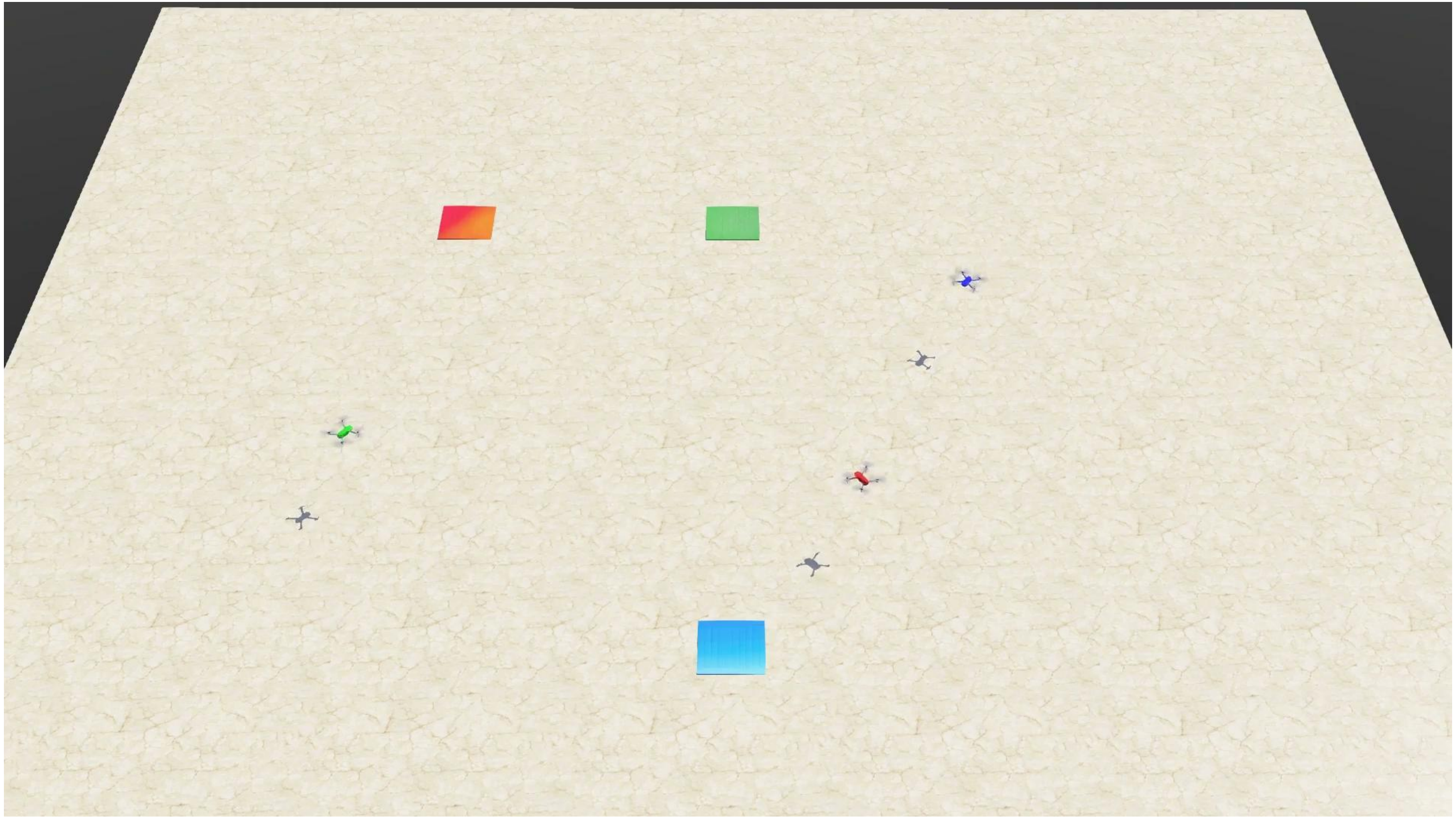
- Mean Rewards



- Testing Final Policies



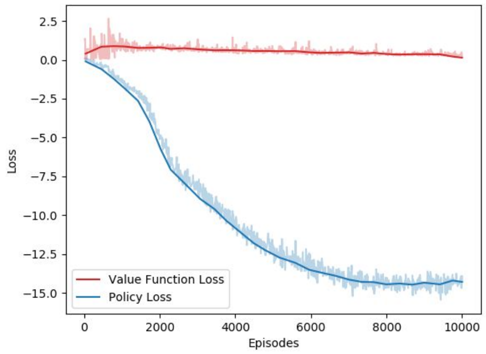
Behaviour-MADDDPG



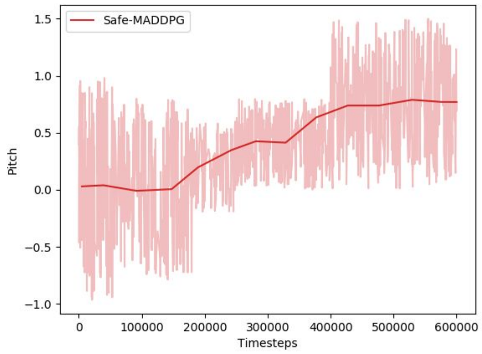
Behaviour-SafeMADDPG

Evaluation Continued..

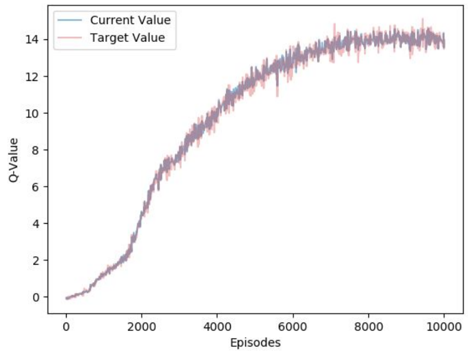
Value & Policy Loss



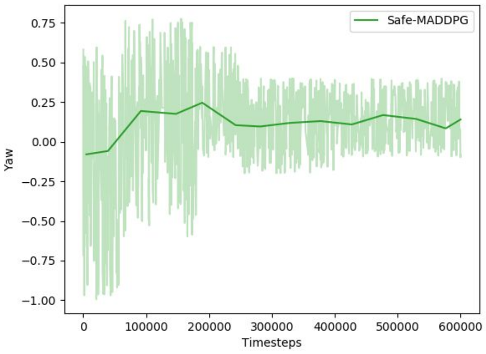
Pitch variation



Current - Target Q-Value



Yaw variation



Reviewer Comments

- It is not very clear what is observations, what is state space and what is action space. Why if an agent is taking an action a_i based on the observation o_i, then state of every other agent is changing. What is \theta_i?
- On page 3, the student mentions that "the bias is also corrected in the loss function during the training". What kind of bias are we talking about?
- In safe reinforcement learning, what is C_k? how easy or difficult in general it is to specify the constraints?
- If the constraints are already enforced, what is the use of the safety layer?
- In algorithm 1, where is L^k used in rest of the algorithm?

Continued..

- I understand that the "safety" aspect of the MADDPG the central novelty claim of the paper. This "safety" aspect should be clearly defined and illustrated right in the beginning sections of the paper.

References

1. https://www.researchgate.net/figure/Deep-Reinforcement-Learning-13_fig2_335109551
2. https://arxiv.org/abs/1706.02275
3. https://arxiv.org/abs/1801.08757
4. https://arxiv.org/abs/1903.08792