SIMULATED DRONE NAVIGATION THROUGH REINFORCEMENT LEARNING

Chaitanya Manem,[1,*] John Paul Reddy Gopidi,[1,†] and Rohit Varre[1,‡]

[1] *School of Physics and Astronomy, University of Nottingham, Nottingham, NG7 2RD, UK*

(Dated: January 19, 2023)

Drones are widely used around the world for different tasks, which are manually operated more often than not and are subject to specific operation, it is necessary at times for them to be automated to perform important tasks. This project aims to model one of those important tasks which is for the drone to reach the target points within the given time in the best possible and efficient way through Reinforcement Learning. The project is navigation of a drone in a 2D simulated environment.

I. INTRODUCTION

In the increasing usage of consumer electronics to boost the productivity of different tasks building an object like drone have become quite common for doing tasks such as film making etc. All such drones navigate around the environment generally through a flight controller. Here the drone must take into account different factors of the environment that influence the flight quality such as acceleration due to gravity, air drag etc. while the flight controller might be taking only some of the factors of the environment and could be limited to the task it has been designed for. This paper is about making the simulation of that task, incorporating the different factors of the real world environment through Reinforcement Learning. The aim of the project is to find the fastest possible path i.e. the optimal path to reach the target within the least possible time.

II. METHOD DEVELOPMENT

The simulation environment is a continuous space therefore having the infinitely many states. The solution is approached with Model-free methods as calculating the weights for each states will become untenable. So giving a reward signal to the heuristic method that can be maximised by learning from agent interacting with the environment is an appropriate method for navigating the environment. Reinforcement Learning task through the structure of Markov Decision Process is given as:.

A. Agent

Since the project is operating the drone towards the target, the agent for the task is Drone. Role of the Drone is to set the thrust values in order to navigate around the environment. The drone has two propellers which produce thrusts, T_1(from left propeller) and T_2(from Right propeller). Since it is 2D Navigation here we are only considering the pitch of the drone over yaw and roll actions which is responsible for driving of the drone to horizontal or vertical motion. Each propeller is capable of producing a maximum thrust of 1N (Newton) force perpendicular to the plane of the drone.

- T_1 = Thrust or force from the left propeller (Initially $T_1 = 0.5$N)
- T_2 = Thrust or force from the right propeller (Initially $T_2 = 0.5$N)
- θ = Pitch of the drone. (Initially $\theta = 10$ degrees)

B. Environment

Environment is the 2D continuous space where the boundaries are customisable i.e. they are user defined and it is a fully observable environment with each point has its coordinates(location) with respect to the environment. Since, this is the simulation to the real world the environment is considered to have a factor, which is acceleration due to gravity ($g = 9.8m/sec^2$). Therefore, at any point in the environment the drone with certain mass (m), experiences a force, $F = mg$, downwards.

C. State

The state of the environment is defined by the relative position with respect to the target, pitch, velocity and acceleration (both linear and angular components) of the agent at any particular time step in the event of flight.

D. Action

The drone has to take actions or interact with environment to propagate to the next states. At any time step the action taken by the drone is the magnitudes of the thrust by the two propellers.

* psxcm4@nottingham.ac.uk
† psxjg13@nottingham.ac.uk
‡ ppxrv1@nottingham.ac.uk

1. Vertical Motion

The movement of the drone is upwards when the resultant force exceeds the force due to weight of the drone. Drone moves upwards when, $T_1 + T_2 > mg$ by some constant ϵ.

2. Horizontal Motion

For the drone to move in the horizontal direction it needs to rotate, such that the thrust vectors are now at an angle θ with the upwards vertical which perturbs the equilibrium thrust T_{eq} by an amount γ, such that $T_1 = T_{eq} + \gamma$, $T_2 = T_{eq} - \gamma$. This will result in a resultant torque $\tau \propto 2 * \gamma$

E. Reward Function

The return G_t with discounted factor $\gamma = 1$, hence the task is episodic with the end state when the target is hit. Since the reward function is a vital equation which determines how the flight should occur in order to lessen the travel time, various methods are proposed.

- Reward Function A:

$$(Cost)c(s) = \begin{cases} +1 & \text{towards the target} \\ -1 & \text{otherwise.} \\ +2 & \text{On reaching the target.} \\ Optimiser & \text{Vanilla Gradient} \end{cases} \tag{1}$$

- Reward Function B:

$$c(s) = \begin{cases} +1 & \text{towards the target} \\ -1 & \text{otherwise.} \\ +2 & \text{On reaching the target.} \\ Optimiser & \text{Adam Optimiser} \end{cases} \tag{2}$$

- Reward Function C:

$$c(s) = \begin{cases} -(RelativeDistance) & \text{towards the target} \\ +100 & \text{On reaching the target.} \\ Optimiser & \text{Adam Optimiser} \end{cases} \tag{3}$$

- Reward Function D:

$$c(s) = \begin{cases} -1 & \text{For every time step} \\ +1000 & \text{On reaching the target} \\ -480 * (distanceleft) & \text{For last uncompleted target} \\ Optimiser & \text{Adam Optimiser} \end{cases} \tag{4}$$

III. METHOD FORMULATION

A. Training

As the state space is continuous and it is not feasible to calculate the value function for each state. Instead, the agent is trained by interacting with the environment and gaining experience using bootstrapping. Optimal parameters are estimated by running multiple trajectories (epochs) and updating the parameters by parameterised the policy gradients. Gradient ascent methods are incorporated to estimate the optimal policy.

B. Parameter Tuning

There are four intrinsic parameters in the heuristic approach that are tuned to improve the flight of the drone they are:

- kx: The distance parameter from the target.
- ky: The pitch parameter from the target.
- absolute pitch delta: Change in pitch.
- absolute thrust delta: Change in thrust.

C. Gradient Calculation

Gradients are calculated using numerical approximation methods which involves

- Starting with default values for all parameters, return is calculated by interacting with the environment.
- Each parameter is changed by a value of 10^{-4} keeping other parameters constant.
- The corresponding change in the return is calculated by interacting with the environment.
- The gradient with respect to the parameter can be approximated as change in the return over change in the corresponding parameter and is calculated by the equation:

$$\frac{\partial G}{\partial \omega_i} = \frac{G(s;\omega_i) - G(s^{'};\omega_i + \omega^{-4})}{10^{-4}} \tag{5}$$

where:

$G(s;\omega_i)$**:** is the return at state s.

$G(s^{'};\omega_i + \omega^{-4})$**:** is the return for the next state $s^{'}$.

$\partial\omega_i = 10^{-4}$**:** is small change in the parameter.

$i = 1, 2, 3, 4$**:** Since there are four parameters.

1. Vanilla Gradient Ascent

- The gradient is calculated for each parameter with respect to the change in reward.
- Vanilla Gradient ascent is performed to find the parameters to maximise the return by optimising the parameters.
- The learning rate alpha used is 1×10^{-4}
- The parameter updation equation is:

$$\omega_i = \omega_i + \alpha \frac{\partial G}{\partial \omega_i} \tag{6}$$

- Exploding and vanishing gradients problem has been observed during the optimisation, which can be inferred from the figure 1.

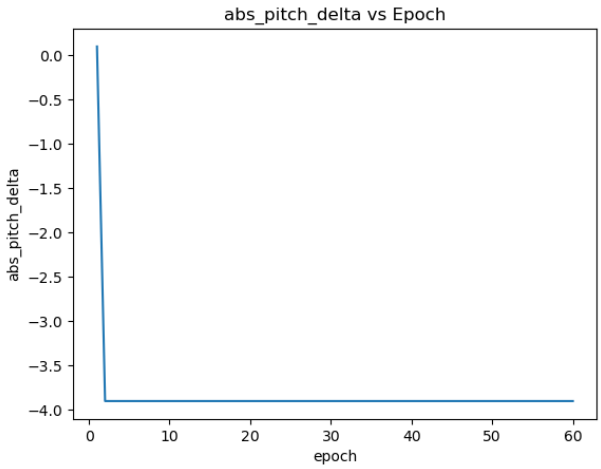

FIG. 1. parameter absolute pitch delta vs Epoch.

Here it is observed the gradient for the first epoch is too large so that the parameter absolute pitch delta change from 0.1 to -4.0 resulting no further update.

2. Adam optimiser with Gradient

- The resultant gradients are calculated by incorporating the past gradient momentum and RMSprop to deal with exploding and vanishing gradients. The hyper parameters of the Adam Optimiser are:

$$\beta_1 = 0.9; \quad \beta_2 = 0.999; \quad \epsilon = 1 \times 10^{-8} \tag{7}$$

The updating procedure of the parameters is as below:

$$\omega_t = \omega_{t-1} + \alpha \frac{V_w}{\sqrt{S_w + \epsilon}} \tag{8}$$

$$V_w = \beta_1 V_{w-1} + (1 - \beta_1) g_{w_{t-1}} \tag{9}$$

$$S_w = \beta_2 S_{w-1} + (1 - \beta_2) g^2_{w_{t-1}} \tag{10}$$

where:

β_1, β_2**:** are constant hyper parameters defined in Adam Optimiser[1]

ϵ**:** is a small constant used to avoid Zero division error.

$g_{\omega_{t-1}}$**:** is the gradient at previous time step $t-1$ of the performance measure equation.

V_ω**:** is the momentum factor contains the weights of the previous gradients.

S_ω**:** is the RMS prop component [1] used to penalise the exploding gradients.

IV. METHOD EVALUATION

1. Reward Function A

- Based on reward function A using the vanilla gradient ascent the drone is not trying to improve the path as the gradient are exploding once the drone reached around -3000 units reward and the learning is halted.

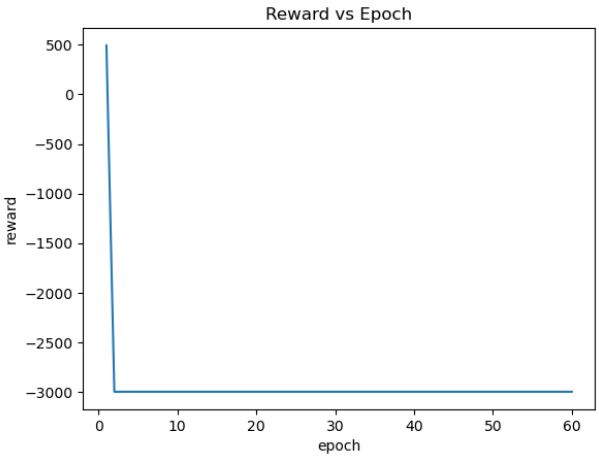

FIG. 2. Reward Function A vs Epoch

- The intrinsic parameters kx, ky, absolute pitch delta and absolute thrust delta are not changing with respect to reward since the reward is constant therefore the gradient with respect to each parameter is zero.

2. *Reward Function B*

- The reward has improved gradually until a steep fall near epoch 10 and improved almost linearly with ranging more than +1000 reward points. After 40 epochs the drone is not improving with the number of epochs even when the gradients are calculated.

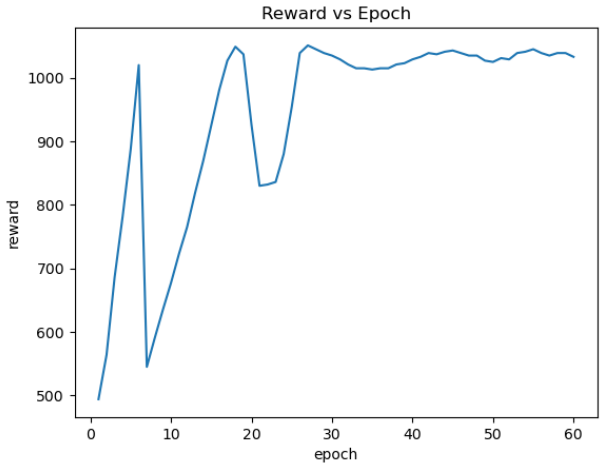

FIG. 3. Reward Function B vs Epoch

- The intrinsic parameters kx, ky, absolute pitch delta and absolute thrust delta all have changed with respect to reward and are updated with adam optimiser gradient ascent unlike in reward function A where we observed exploding and vanishing gradients.

3. *Reward Function C*

- The reward function considering the negative distance as the factor and +100 points for reaching the target has obtained the maximum reward around 10 epochs and after running 60 epochs the reward has declined using the Adam optimiser gradient ascent.

4. *Reward Function D*

- The reward function considers both time and distance. It is slightly improved until 100 epochs but declined steeply after 100 epochs.

- The change in reward function D with respect to the parameters of the drone are visualised by the graphs below. It is observed that reward function is not completely convex with change in the parameters. Though some improvement is gained.

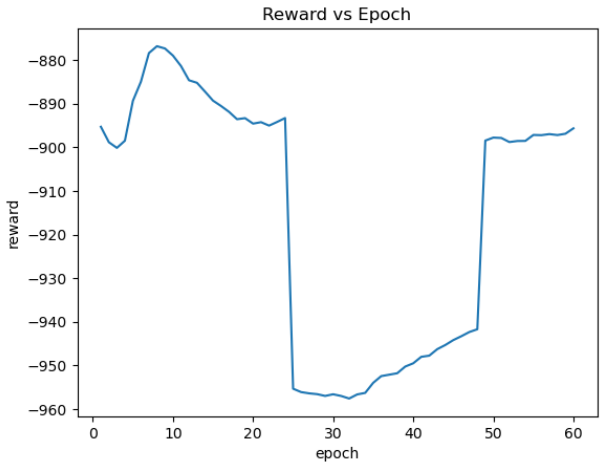

FIG. 4. Reward Function C vs Epoch

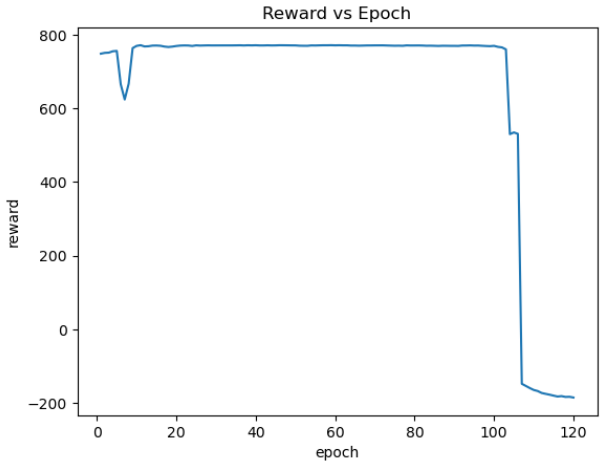

FIG. 5. Reward Function D vs Epoch

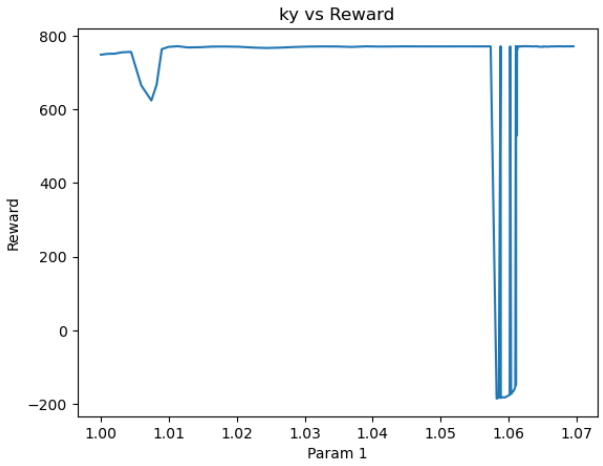

FIG. 6. Parameter ky vs reward

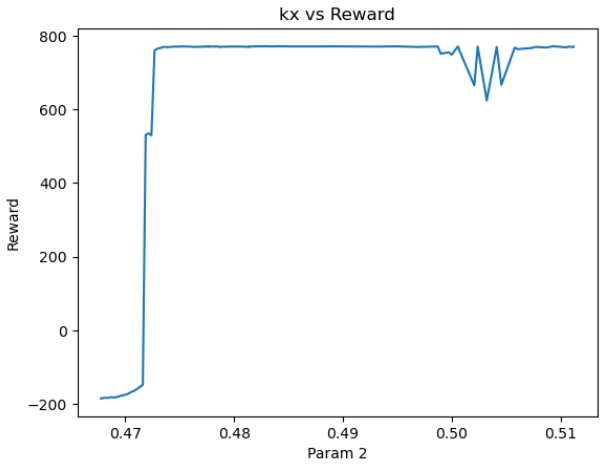

FIG. 7. Parameter kx vs reward

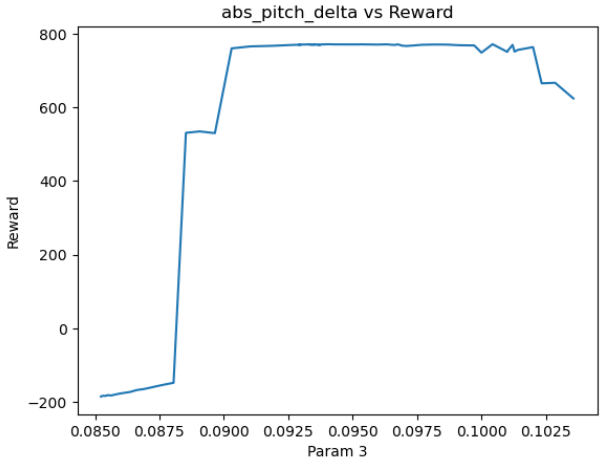

FIG. 8. Parameter Absolute pitch delta vs reward

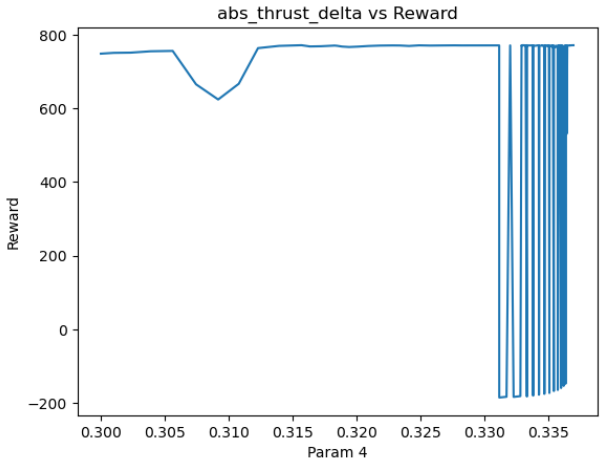

FIG. 9. Parameter Absolute thrust delta vs reward

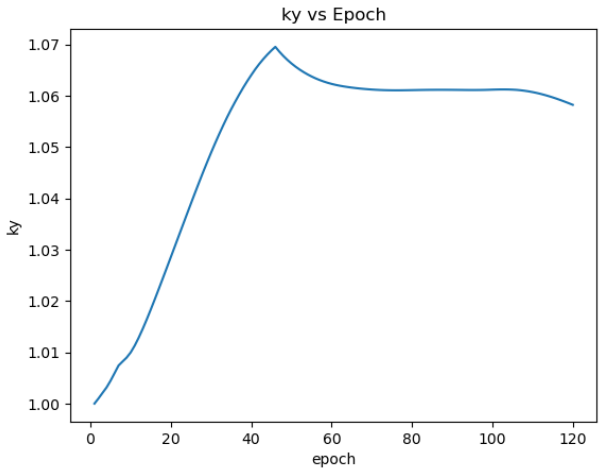

FIG. 10. Parameter ky vs epoch

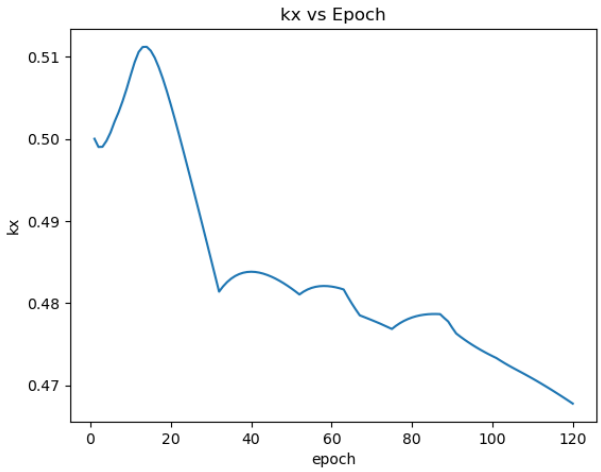

FIG. 11. Parameter kx vs epoch

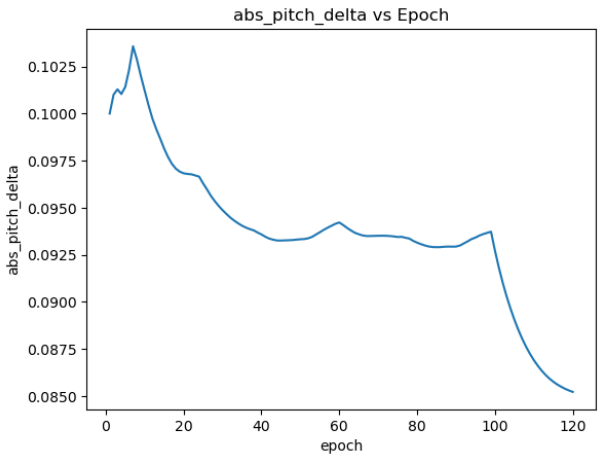

FIG. 12. Parameter absolute pitch delta vs epoch

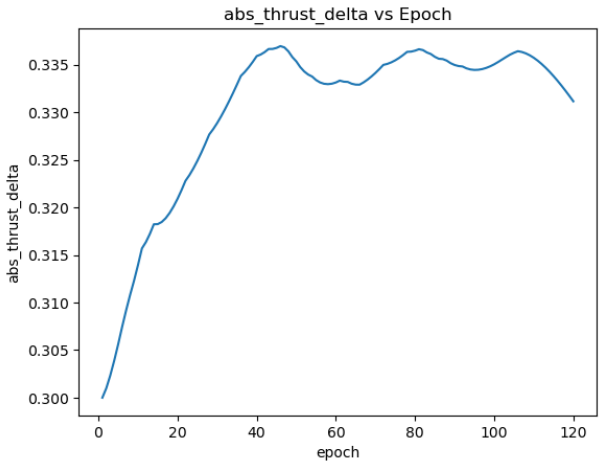

FIG. 13. Parameter Absolute Thrust delta vs epoch

V. RESULTS

With old parameters the drone was able to hit four targets and reach 255.38 units away from the 5th target point. While with the new optimised parameters drone could hit four targets and reach 232.40 units away from the 5th target. Which means with improved parameters drone could travel faster and reach 22.98 units closer to the 5th target than the given parameters.

The flight of the drone has been improved slightly when compared to the given heuristic, it covered more distance in less time towards the target, with the reward function D incorporating time and distance factor has received a maximum positive reward of 771.5968 with the change in parameters by gradient calculation based on Adam Optimiser [1].

TABLE I. Change in Reward with time taken based on relative distance from the 5th target.

epoch	reward	hits	dist_left (units)
1	748.611909	4	255.388091
2	750.806263	4	253.193737
3	751.388647	4	252.611353
4	755.1149	4	248.8851
5	756.077908	4	247.922092
59	771.596817	4	232.403183s

TABLE II. Change in parameters with epochs.

epoch	ky	kx	absolute pitch delta	absolute thrust delta
1	1	0.5	0.1	0.3
2	1.001	0.499	0.101	0.301
3	1.002075	0.499022	0.101285	0.30234
4	1.003134	0.499747	0.101035	0.3039
5	1.004449	0.500768	0.101411	0.305617
59	1.062539	0.482056	0.094149	0.333005

TABLE III. Change in Gradients of parameters ky, kx, absolute pitch delta, absolute thrust delta with Epochs respectively.

epoch	Grad 1	Grad 2	Grad 3	Grad 4
1	21.031827	-50.597908	5879.691818	1331.829706
2	4.088305	47.062522	-3364.263115	1239.437666
3	2.061095	69.842616	-3652.665332	1222.106422
4	11.652496	47.012862	5218.387838	1272.70625
5	10.746852	68.324664	8523.873718	1248.768695
59	35.968759	-181.656414	-4258.338649	1070.454817

VI. CONCLUSIONS

This 2D Drone simulation has a continuous environment space and it is solved through model free methods. As the initial step heuristic approach with parameters is used. Designed reward functions to estimate the quality of flight. Different optimisation algorithms along with gradient ascent are used to tune the parameters, by interacting with the environment and measuring the quality through reward function. Adam optimiser is found to be the best optimiser for this heuristic approach. With this approach small improvement in the speed of Drone is observed. With the new parameters the drone has reached 23 units closer towards the 5^{th} target than with the given parameters. Further to this approach policy gradients with deep learning can be directly used to learn the optimal policy that can give thrust values given drone state.

[1] D. P. Kingma and J. Ba, Adam: A method for stochastic optimization (2014).