Tethered Multicopter Guidance in GPS-Denied Environments Through Reinforcement Learning

Amer Al-Radaideh*, Robert Selje†and Daniel Coraspe‡
New Mexico State University, Las Cruces, New Mexico, 88003, USA

Rajdeep Dutta§, Efe Camci¶, Senthilnath Jayavelu‖, and Xiaoli Li**
*Agency for Science, Technology and Research (A*STAR), Singapore 138632*

Liang Sun††
New Mexico State University, Las Cruces, New Mexico, 88003, USA

Abstract: This paper presents a novel reinforcement learning (RL) approach for a tethered drone to follow a predefined three-dimensional trajectory in a GPS-denied environment. The adopted Q-learning strategy determines high-level actions using raw observations from the onoard accelerometers, gyros, and altimeter, which facilitates a low-level proportional-integral-derivative (PID) controller to drive the drone through the desired waypoints on a reference trajectory. The effectiveness of the proposed approach is demonstrated in a simulated environment.

I. Introduction

Multicopter unmanned aerial vehicles (UAVs), or drones, notably rely on accurate knowledge of their locations for guidance, navigation, and control. UAV localization typically counts on IMUs [1–3], the Global Positioning System (GPS) [4] (differential GPS [5]), infrared (IR) sensors [6], laser rangefinders [7, 8], and optical and vision systems [9–11]. While all of these sensing systems have successfully supported outdoor applications, extensive investment has been made to enhance the capability of self-localization for UAVs by improving the GPS infrastructure, utilizing cellular network infrastructure [12], or integrating both technologies for a wider range of applications. However, UAV self-localization in indoor environments and street canyons is still very challenging. Also, unlike ground vehicles, UAVs are constrained by their size and payload. Very limited power and flight endurance have prevented them from carrying high-end sensors for localization. This poses critical concerns to the safety of UAV operations in populated urban areas and in GPS-denied areas.

The tremendous improvement in the sensing capabilities of multicopter UAVs has endorsed the utilization of machine learning techniques and specifically the reinforcement learning (RL) in the guidance, navigation and control (GNC) field [13–16]. The authors in [17] presented a successful autonomous completion on a real remote-controlled helicopter of four aerobatic maneuvers using inverse RL. In [18] an improved adaptive RL method is proposed for UAV morphing control that learns the optimal shape-change policy at different flight conditions and it is integrated with an adaptive dynamic inversion control trajectory tracking function. In [19], The same authors proposed a Q-Learning algorithm to teach a UAV to track a ground targets for autonomous surveillance using the learned control policy of the images captured by the camera.

In outdoor environments, we assume that at any position, the drone can observe its location. If we have full information about the environment, for instance, the exact distance to the target or the locations of the obstacles, the UAV

*Ph.D. Candidate, Department of Mechanical and Aerospace Engineering, Jett Hall Rm. 104, 1040 S. Horseshoe Street, Las Cruces, NM 88003.
†MSc student, Klipsch School of Electrical and Computer Engineering, Thomas & Brown Hall, 1125 Frenger Mall, Las Cruces, NM 88003.
‡Undergraduate student, Department of Mechanical and Aerospace Engineering, Jett Hall Rm. 104, 1040 S. Horseshoe Street, Las Cruces, NM 88003.
§Scientist, Agency for Science, Technology and Research (A*STAR), 1 Fusionopolis Way, #21-01 Connexis (South Tower), Singapore 138632
¶Scientist, Agency for Science, Technology and Research (A*STAR), 1 Fusionopolis Way, #21-01 Connexis (South Tower), Singapore 138632
‖Scientist, Agency for Science, Technology and Research (A*STAR), 1 Fusionopolis Way, #21-01 Connexis (South Tower), Singapore 138632
**Principal Scientist, Agency for Science, Technology and Research (A*STAR), 1 Fusionopolis Way, #21-01 Connexis (South Tower), Singapore 138632
††Associate Professor, Department of Mechanical and Aerospace Engineering, Jett Hall Rm. 104, 1040 S. Horseshoe Street, Las Cruces, NM 88003. AIAA Senior Member.

trajectory tracking can be generated based on the model of the environment, and the problem becomes convectional. Traditional control methods, such as Sliding-mode, Lyapunov based, etc [20–22], are available to solve such problem. In many realistic cases, and specifically in our problem, localizing the drone in GPS denied environments, self localization is not possible since the environment is insufficiently known, or the data of the environment is not available or difficult to obtain. Since RL algorithms can rely only on the data obtained directly from the system, it is a natural option to consider for our tether problem. Our goal is to control the tethered multicopter to track a predefined path without prior knowledge of the cable force information nor the system dynamics. The main challenge of controlling a tethered drone comes from the interaction between the drone and the tether. The cable force variation is determined by the tether type, the thrust force of the drone motors, and drone dynamics. In our prior research work [23–25], we have been investigating a solution to self-localizing a multicopter UAV in GPS-denied environments using a tethered system. It has shown the effectiveness of the EKF by assuming the availability of a constant cable force. Without a cable force sensor, however, the cable force variation would be nontrivial to identify or model. A smart winch control mechanism would be needed to maintain the tautness of the tether. All of these challenges motivate us to develop a model-free artificially intelligent method. In this paper, we proposed to leverage the RL technique to generate a policy of controlling the tethered drone for waypoint following. The available on-board sensors that can be leveraged include the accelerometers, gyroscopes, and altimeter. Its also worth noting that since we have conducted the observability analysis in our previous research work [25], which showed that the multicopter position is observable if the tether is taut,we are confident in utilizing the RL with the same hypothesis.

To the best of our knowledge, using a RL algorithm for a UAV's guidance and navigation in unknown environments is still an open area of research. Many works often did not provide details on the practical applications of implementation of the learning algorithm on heterogeneous systems. In this paper, we provide a novel technique utilizing the Q-Table for local frame navigation to explore unknown environments. Q-Table is easier to memorize and carry along the dynamics, where it allows to carry on the knowledge in such complicated environments. Our application in local frame is extendable to cover future work. The tether drone application is considered a heterogeneous system that consists of two agents: the ground robot and the multicopter. Choosing the Q-Table based RL technique would give flexibility to expand the work to augment the work in more complicated environments (i.e. environment with moving obstacles) and accommodating new features such as (i.e. collision avoidance).The main contribution of the paper is to provide a framework for applying a Q-Table based RL algorithm to enable complicated heterogeneous systems of autonomous robot agents to navigate in unknown environments with obstacles.

The remainder of this paper is structured as follows. Key definitions of coordinate frames, the dynamics model of the tethered multicopter, and the accelerometer principle are introduced in Sections II and II.B.1, respectively. The proposed RL-based guidance policy is presented in Section III. The simulation results are presented in Section IV and Section V concludes the paper.

II. Background and Preliminaries

A. Coordinate Frames

In this section, we introduce the key coordinate frames associated with the multicopter movements, i.e., the inertial frame, the vehicle frame, and the body frame [26].

- *The inertial frame, $\mathcal{F}^i$:* The inertial coordinate system is an earth-fixed coordinate system with its origin at a pre-defined location. In this paper, this coordinate system is referred to the North-East-Down (NED) reference frame. It is common for North to be referred to as the inertial x direction, East to the y direction, and Down to the z direction.
- *The vehicle frame, $\mathcal{F}^v$:* The origin of the vehicle frame is at the center of mass of a multicopter. However, the axes of $\mathcal{F}^v$ are aligned with the axes of the inertial frame $\mathcal{F}^i$. In other words, the unit vector $\mathbf{i}^v$ points toward North, $\mathbf{j}^v$ toward East, and $\mathbf{k}^v$ toward the center of the earth.
- *The body frame, $\mathcal{F}^b$:* The body frame is obtained by rotating the vehicle frame in a right-handed rotation about $\mathbf{i}^v$ by the roll angle, ϕ and about the $\mathbf{j}^v$ axis by the pitch angle θ and about the $\mathbf{k}^v$ axis by the yaw angle ψ.

The transformation from $\mathcal{F}^v$ to $\mathcal{F}^b$ is given by

$$\mathbf{p}^b = R_v^b\,(\phi, \theta, \psi)\,\mathbf{p}^v, \tag{1}$$

where $\mathbf{p}^v$ and $\mathbf{p}^b$ are the multicopter position in the vehicle frame and body frame respectively and the transformation matrix, $R_v^b(\phi,\theta,\psi)$, is given by:

$$R_v^b(\phi,\theta,\psi) = \begin{pmatrix} \mathrm{c}_\theta \mathrm{c}_\psi & \mathrm{c}_\theta \mathrm{s}_\psi & -\mathrm{s}_\theta \\ \mathrm{s}_\phi \mathrm{s}_\theta \mathrm{c}_\psi - \mathrm{c}_\phi \mathrm{s}_\psi & \mathrm{s}_\phi \mathrm{s}_\theta \mathrm{s}_\psi + \mathrm{c}_\phi \mathrm{c}_\psi & \mathrm{s}_\phi \mathrm{c}_\theta \\ \mathrm{c}_\phi \mathrm{s}_\theta \mathrm{c}_\psi + \mathrm{s}_\phi \mathrm{s}_\psi & \mathrm{c}_\phi \mathrm{s}_\theta \mathrm{s}_\psi - \mathrm{s}_\phi \mathrm{c}_\psi & \mathrm{c}_\phi \mathrm{c}_\theta \end{pmatrix}. \tag{2}$$

where $\mathrm{c}_* \triangleq \cos(*)$ and $\mathrm{s}_* \triangleq \sin(*)$.

B. Tethered Multicopter Dynamics

In this section, we introduce the equations of motion of a multicopter tethered to a stationary station. The 6 d-o-f model for a multicopter dynamics relies on 12 states as follows.

$$\begin{pmatrix} \dot p_n \\ \dot p_e \\ \dot p_d \end{pmatrix} = R_b^v(\phi,\theta,\psi) \begin{pmatrix} u \\ v \\ w \end{pmatrix}, \tag{3}$$

$$\begin{pmatrix} \dot u \\ \dot v \\ \dot w \end{pmatrix} = \begin{pmatrix} rv - qw \\ pw - ru \\ qu - pv \end{pmatrix} + \frac{1}{m} \begin{pmatrix} f_x \\ f_y \\ f_z \end{pmatrix}, \tag{4}$$

$$\begin{pmatrix} \dot \phi \\ \dot \theta \\ \dot \psi \end{pmatrix} = \begin{pmatrix} 1 & \sin\phi \tan\theta & \cos\phi\tan\theta \\ 0 & \cos\phi & -\sin\phi \\ 0 & \frac{\sin\phi}{\cos\theta} & \frac{\cos\phi}{\cos\theta} \end{pmatrix} \begin{pmatrix} p \\ q \\ r \end{pmatrix}, \tag{5}$$

$$\begin{pmatrix} \dot p \\ \dot q \\ \dot r \end{pmatrix} = \begin{pmatrix} \frac{J_y - J_z}{J_x} qr \\ \frac{J_z - J_x}{J_y} pr \\ \frac{J_x - J_y}{J_z} pq \end{pmatrix} + \begin{pmatrix} \frac{1}{J_x}\tau_l \\ \frac{1}{J_y}\tau_m \\ \frac{1}{J_z}\tau_n \end{pmatrix}, \tag{6}$$

where $(p_n, p_e, p_d)^T \in \mathbb{R}^3$ is defined as the multicopter position in the North-East-Down (NED) inertial frame; (u, v, w) is the vehicle velocity vector in the body frame; m is the vehicle mass; (p, q, r) is the rotational velocity in the body frame; and (f_x, f_y, f_z) , (τ_l, τ_m, τ_n) are the total external forces and torques applied to the multicopter in the body frame respectively; J_x, J_y and J_z are moments of inertia of the multicopter.

1. Force and Acceleration

Commercial-of-the-shelf (COTS) Inertial Measurement Units (IMUs), such as accelerometers, are widely used in multicopter navigation tasks. From a controller design perspective, different terms associated with the output of such accelerometers need to be analyzed carefully. In view of this, the normalized kinematic accelerations and specific forces [27] are explained in the following.

Let $\eta = (u, v, w)^T$ be the linear velocity vector, $\varOmega = (p, q, r)^T$ be the rotational velocity vector of the multicopter in the body frame and $\mathbf{f}^b = \left(f_x, f_y, f_z\right)$ be the total external force vector in the body frame. Assume that the accelerometer is mounted at the center of gravity of a multicopter and define the kinematic acceleration vector $\mathbf{a}_k^b = \left(\mathbf{a}_{k,x}^b, \mathbf{a}_{k,y}^b, \mathbf{a}_{k,z}^b\right)^T$ in the body frame as

$$\mathbf{a}_k^b = \frac{\mathbf{f}^b}{mg} = \frac{\dot\eta}{g} = \frac{1}{g}\left(\frac{\partial \eta}{\partial t} + \varOmega \times \eta\right), \tag{7}$$

The constituents of $\mathbf{a}_k^b$ are given by

$$\begin{cases} \mathbf{a}_{k,x}^b = \frac{1}{g}(\dot u + qw - rv) = \frac{f_x}{mg}, \\ \mathbf{a}_{k,y}^b = \frac{1}{g}(\dot v + ru - pw) = \frac{f_y}{mg}, \\ \mathbf{a}_{k,z}^b = \frac{1}{g}(\dot w + pv - qu) = \frac{f_z}{mg}, \end{cases} \tag{8}$$

where g is the gravitational acceleration on Earth. Note that $\mathbf{a}_k^b$ and g have the same unit. The output of an accelerometer used in the multicopter autopilot is referred to as the specific force (or g-force or mass-specific force), $\mathbf{a}_{SF}^b$, which is defined as

$$\mathbf{a}_{SF}^b = \frac{\mathbf{f}^b - \mathbf{f}_g^b}{mg} = \mathbf{a}_k^b - \frac{\mathbf{f}_g^b}{mg}\ , \tag{9}$$

and the associated components are given by

$$\begin{cases} \mathbf{a}_{SF,x}^b = \mathbf{a}_{k,x}^b + \sin\theta, \\ \mathbf{a}_{SF,y}^b = \mathbf{a}_{k,y}^b - \cos\theta\sin\phi, \\ \mathbf{a}_{SF,z}^b = \mathbf{a}_{k,z}^b - \cos\theta\cos\phi. \end{cases} \tag{10}$$

External Forces on Tethered Multicopter: The total external force vector for a tethered multicopter in the body frame is given by

$$\mathbf{f}^b = \mathbf{f}_{thrust}^b + \mathbf{f}_g^b + \mathbf{f}_{cable}^b, \tag{11}$$

where $\mathbf{f}_{thrust}^b$ is the thrust force, $\mathbf{f}_g^b$ is the gravity force, and $\mathbf{f}_{cable}^b$ is the cable tension force, all in the body frame. The gravity force vector of the multicopter in the vehicle frame is: $\mathbf{f}_g^v = [0,\ 0,\ mg]^T$. Next, the gravity force in the body frame is given by

$$\mathbf{f}_g^b = R_v^b \mathbf{f}_g^v = \begin{pmatrix} -mg\sin\theta \\ mg\cos\theta\sin\phi \\ mg\cos\theta\sin\theta \end{pmatrix}. \tag{12}$$

The thrust force vector in the body frame is given by

$$\mathbf{f}_{thrust}^b = \begin{pmatrix} f_{thrust,x} \\ f_{thrust,y} \\ f_{thrust,z} \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ -(f_F + f_R + f_B + f_L) \end{pmatrix} \tag{13}$$

where subscripts F, R, B, and L denote the thrust forces provided by the front, right, back, and left motors, respectively. The individual thrust forces can be calculated using the PWM (Pulse Width Modulation) signals commanded to the motors, i.e.,

$$f_* = k_{motor} \cdot \text{pwm}_*, \tag{14}$$

where $* \in \{F, R, B, L\}$ and k_{motor} is the electric motor coefficient and pwm_* is the PWM motor control signal. The specific force can then be expressed as

$$\mathbf{a}_{SF}^b = \frac{\mathbf{f}^b - \mathbf{f}_g^b}{mg} = \mathbf{a}_k^b - \frac{\mathbf{f}_g^b}{mg} = \frac{\mathbf{f}_{thrust}^b + \mathbf{f}_{cable}^b}{mg}. \tag{15}$$

By considering the tether to be taut, $\mathbf{f}_{cable}^b$ can be deduced as: $\mathbf{f}_{cable}^b = R_v^b f_{cable}^v \times \frac{L}{\ell}$, where $L = (p_n, p_e, p_d)^T$, $\ell = \sqrt{p_n^2 + p_e^2 + p_d^2}$, and f_{cable}^v is the magnitude of the cable force, and we obtain

$$\mathbf{f}_{cable}^b = \frac{R_v^b f_{cable}^v}{\sqrt{p_n^2 + p_e^2 + p_d^2}} \left[\begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} - \begin{pmatrix} p_n \\ p_e \\ p_d \end{pmatrix} \right]. \tag{16}$$

Next, Equation (15) can be rewritten as

$$\mathbf{a}_{SF}^b = \begin{bmatrix} \mathbf{a}_{SF,x}^b \\ \mathbf{a}_{SF,y}^b \\ \mathbf{a}_{SF,z}^b \end{bmatrix} = \frac{1}{mg} \left[\begin{pmatrix} 0 \\ 0 \\ -(f_F + f_R + f_B + f_L) \end{pmatrix} - \frac{R_v^b f_{cable}^v}{\sqrt{p_n^2 + p_e^2 + p_d^2}} \begin{pmatrix} p_n \\ p_e \\ p_d \end{pmatrix} \right]. \tag{17}$$

III. Problem Statement and Solution Methodology

In a GPS-denied environment, the goal is to optimally drive a tethered multicopter through the predefined waypoints on a reference trajectory. Here, RL is adopted to steer the tethered multicopter towards the destination waypoints in minimum time. A high-level Q-learning based planner is leveraged to find the optimal route between source and destination waypoints, while a low-level PID controller is used to generate stable movements of the multicopter flying into the intermediate coordinates determined by Q-learning.

Consider a multicopter tethered to a ground robot, as shown in Figure 1. For simplicity, in this work, the ground robot is assumed to be stationary and located at the origin $(0, 0)$ of the inertial frame. Unlike a free multicopter, a tethered multicopter dynamics gets restricted since it needs to pay additional efforts to compensate the cable tension. Therefore, finding an optimal route for a tethered multicopter is a challenging problem to solve. The proposed solution method is explained in the following.

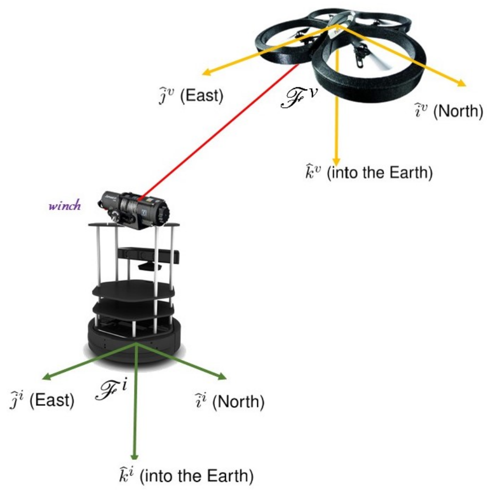

Fig. 1 Tethered multicopter to a ground robot scenario.

A. High Level Planning with Q-Learning

RL is an artificial-intelligence method that teaches an agent to tackle sequential decision-making tasks by maximizing the rewards received from its interactions with the surrounding environment. A typical sequential decision making process can be characterized by a finite and discrete time Markov Decision Process (MDP). In principle, the state transitions associated with an MDP satisfy the Markov Property [28], according to which, all the future states pursued by an agent can be fully determined by its current state and action. As a consequence, the expected cumulative reward gathered by it also becomes a function of its current state and action [29]. Precisely, an MDP can be represented by a tuple: <current state (s_t), action (a_t), next state (s_{t+1}), reward (r_t)>. The sum of the discounted rewards over a finite time-horizon (T) is called the cumulative reward or return at time t, i.e. R_t. The goal of an RL agent is to maximize the expected cumulative reward, called as the value: $V(s_t) = \mathbb{E}\left[R_t = \sum_{i=t}^{T} \gamma^{i-t} r_t\right]$ of a state s_t at time t, where $\gamma \in [0,\ 1]$ denotes a discount factor and $\mathbb{E}$ stands for the expectation operator.

In the current set up, an agent intends to learn a sequence of appropriate actions to reach a destination optimally. The associated states are the three-dimensional position coordinates (x, y and z) and the heading attitude (yaw) of the multicopter. Using these raw observations, an RL agent tries to steer the multicopter towards the optimal direction. The underlying action space is discrete and four-dimensional with possibilities: a step moving *forward, backward, left,* or *right*. Figure 2 shows the interaction between an agent and the environment in the adopted RL framework. Note that all the sequences of actions are assumed to terminate in a finite number of time-steps. This formalism gives rise to a large yet finite MDP. A Q-learning algorithm is adopted here since it has proven to be stable and efficient in solving dynamic problems with discrete action spaces [30]. Usually, a classical Q update is carried out in a sample-by-sample manner, which indicates that the value function changes every time a new transition is made. The associated iterative update rule is given by

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha\{r(s_t, a_t) + \gamma \max_{a_{t+1}} Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t)\}\,, \tag{18}$$

where s_t denotes the current state where a transition starts, a_t is the action applied onto s_t, and s_{t+1} is the resulting future state after the transition; γ is the discounting factor, and α is the learning rate.

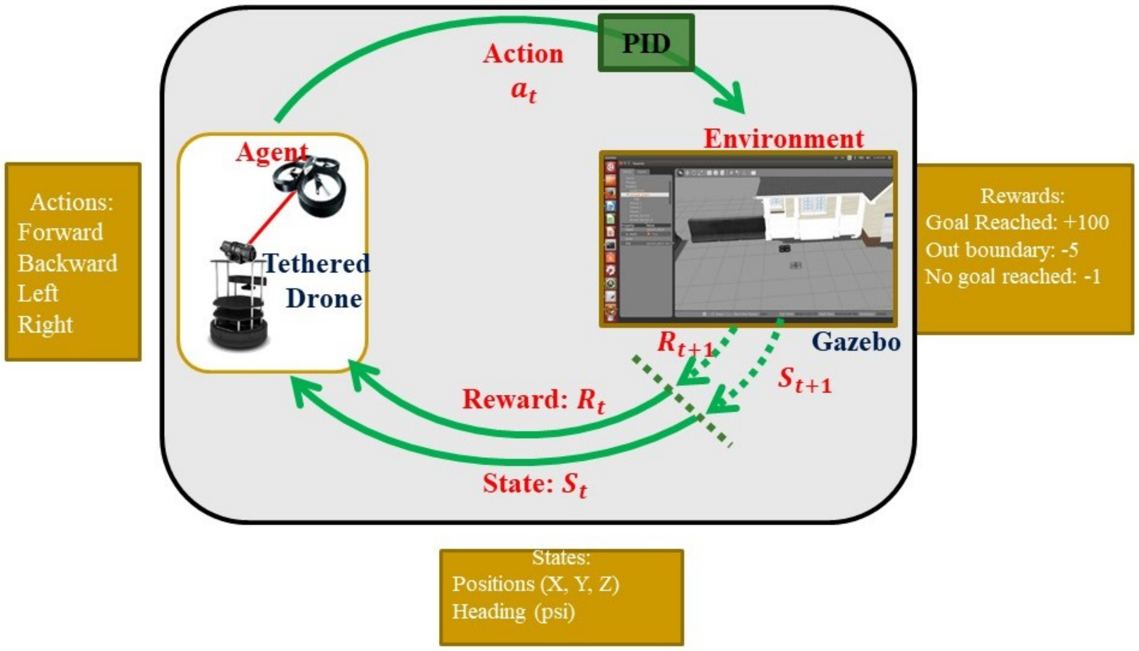

Fig. 2 The agent and the environment relationship in reinforcement learning

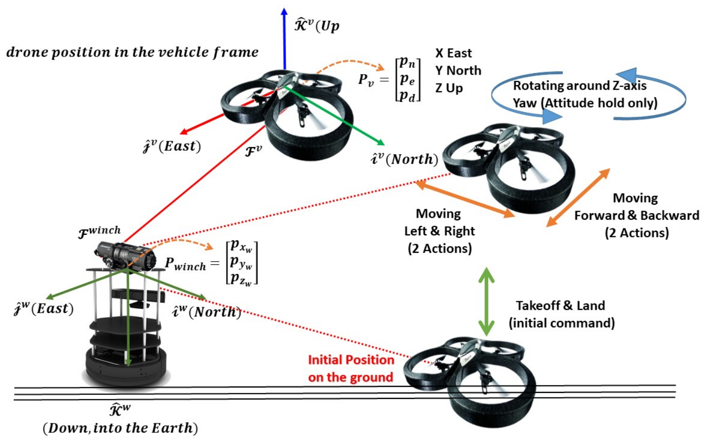

Fig. 3 Depiction of the set of actions available to the agent

1. Q-table Update

A Q-table is simply a matrix that keeps record of all the Q-values of various state-action pairs encountered while training an RL agent. The number of rows of this matrix equals to the number of states, and the number of columns is same as the number of possible actions. The associated Q-tables evolve as per the three basic steps: (i) To maintain a balance between the exploration and the exploitation of a search space, an agent chooses actions using an epsilon (ϵ)-greedy strategy. Based on the ϵ rate, the agent either takes a random action or it takes an action corresponding to the maximum Q-value of the latest Q-table. The ϵ rates are higher in the beginning episodes to promote more exploration of the environment, however, the same decreases gradually with training episodes to encourage exploitation; (ii) The agent makes a state-transition using the selected action and receives a reward; and (iii) All the Q-values of the Q-table are updated according to Equation (18) towards satisfying the Bellman optimality criteria. The updates occur after each state-transition and ends when an episode is done. Here, a *Done* episode means that the agent has reached the desired terminal state.

2. Reward Shaping

The design challenge lies in shaping an effective reward function that facilitates the RL agent to learn adaptive/optimal actions leading to the desired goal waypoint location(G). A flow chart of the logic behind the current reward scheme is depicted in Figure 4. The goals can be different based on the scenario-specific requirements. For instance, when the drone is flying freely and the tether is not taut but used for safety and power feeding, then the objective is to reach the desired Goal waypoint (G) with minimal number of actions. The rewards are assigned to meet this objective while encouraging the tethered drone to fly within a boundary (B), which is computed by the algorithm based on the set of desired waypoints, see Figure 7. The reward function for this scenario is given by

$$R = \begin{cases} \text{10 - C*Actions} & \text{if } \mathrm{s}_{t+1} \equiv \mathrm{G} \\ \text{-1} & \text{if } \mathrm{s}_{t+1} \not\equiv \mathrm{G} \ \& \equiv \mathrm{B} \\ \text{-5} & \text{if } \mathrm{s}_{t+1} \not\equiv \mathrm{G} \ \& \not\equiv \mathrm{B} \end{cases} \tag{19}$$

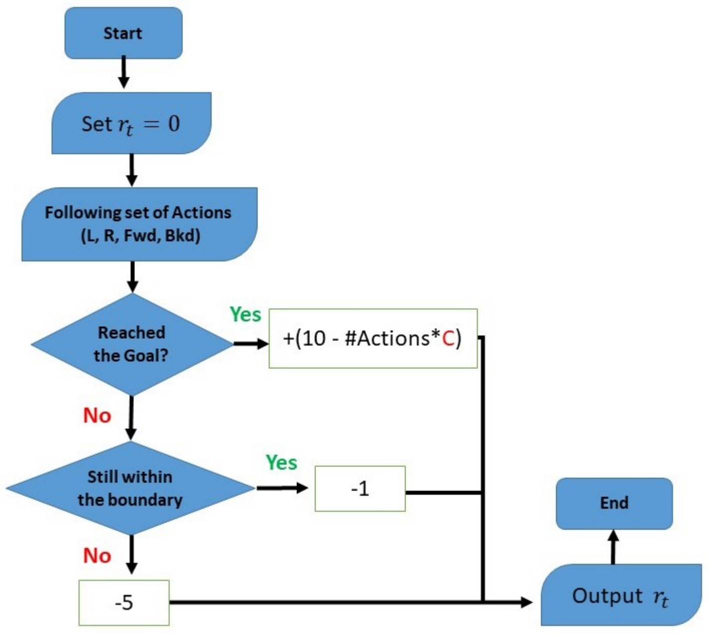

Fig. 4 Flowchart depicting the selected reward scheme for the task.

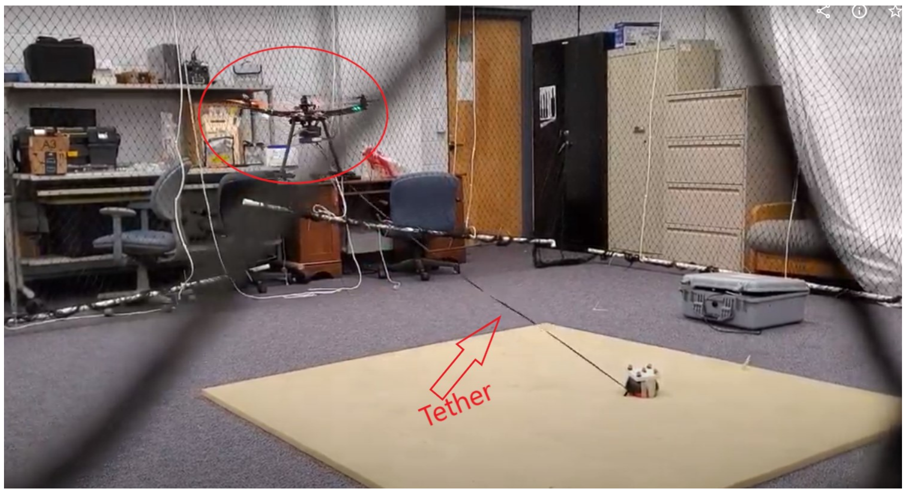

Fig. 5 Tether flight test for data collection and training

It is worth noting that the representative waypoints are picked from the inertial (global) frame while training an RL agent, and during the Q-learning phase, each pair of source (current) and destination (next) waypoints is considered in the vehicle frame, i.e. the local coordinate frames visualized in Figure 6 and Figure 7. Precisely, the coordinate of a destination waypoint is converted from the inertial frame to the vehicle frame with reference to the corresponding departed waypoint placed at the origin, as shown in Table 1. By adopting a local coordinate frame rather than a global coordinate frame, the search space for Q-learning gets reduced due to a restricted boundary. Consequently, the size of the associated Q-table becomes tractable. For example, as depicted in Figure 7, the multicopter flies within a boundary of $(3x3)$ space using the local coordinate frame; however, the same would have been a four times bigger space $(6x6)$ using the global coordinate frame.

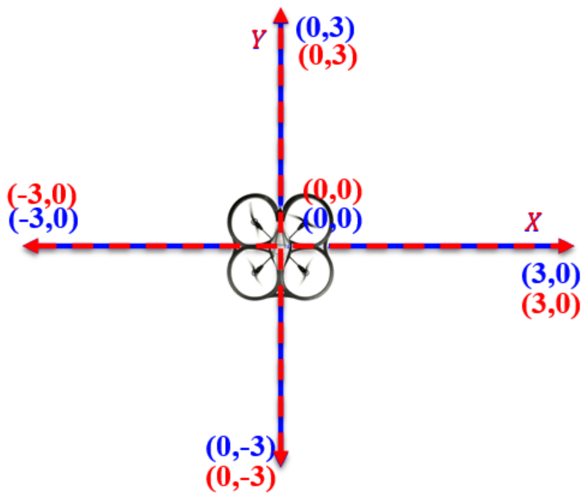

Fig. 6 Global-Local Frames:Blue-Gloabl, Red-Local

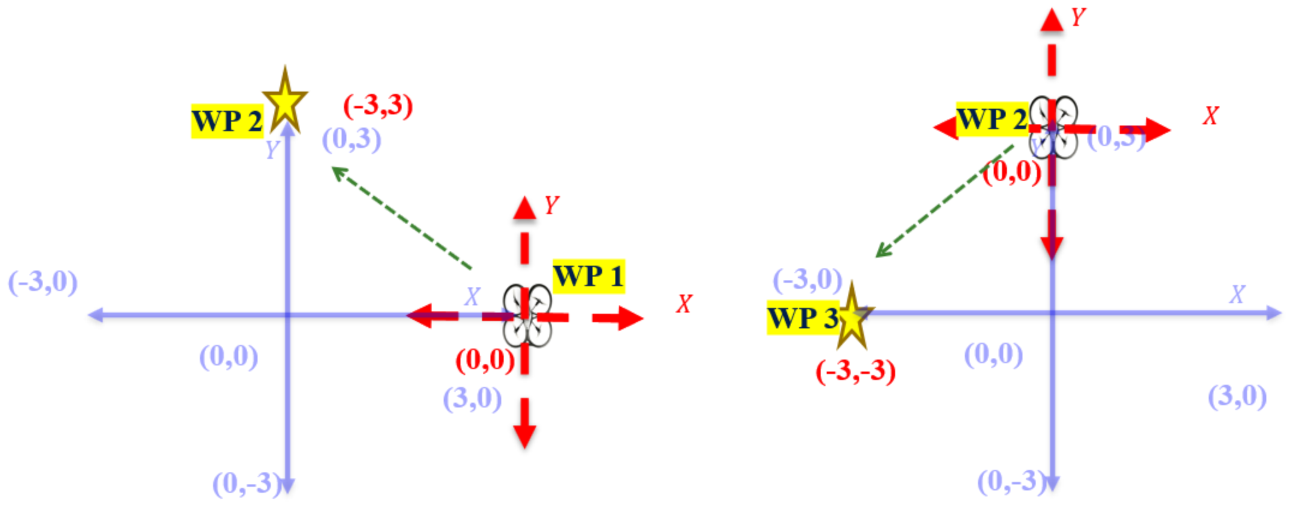

Fig. 7 Global Vs Local Coordinates

B. Low Level Control

The multicopter model in Gazebo subscribes command signals generated from the low-level PID position controller. This position controller intends to help the multicopter UAV performing an action a_k to move from current location s_t (state) to the new location s_{k+1}, or within its small neighborhood defined by an absolute distance error of d. Given that p_t is the real-time position of the multicopter, the PID controller suffices to ensure the dynamic stability of the drone.

$$u(t) = K_p e(t) + K_i \int e(t)dt + K_d \frac{de}{dt} \tag{20}$$

where the tracking error $e(t)$ is defined as:

$$e(t) = p(t) - s_{k+1} \tag{21}$$

IV. Tethered Drone Simulation Environment and Results

Simulated environment: The tethered drone simulation environment includes three softwares: *Gazebo*, *Robot Operating System (ROS)*, and *OpenAI Gym*. Gazebo is a 3D dynamic simulator that helps simulating robots in the virtual environments. To simulate a real multicopter (Parrot AR-Drone2.0 quadcopter) in *Gazebo*, we used the ROS driver: *ARdrone autonomy* [31]. This driver provides the quadcopter POSE: x, y, z and ψ, fed as state inputs to the RL agent. The adopted Q-Learning framework for waypoint navigation control problem, is shown in Figure 8. At the current POSE, the trained RL agent predicts the optimal flight direction that leads the tethered quadcopter to the destination waypoint in minimum time. Once the optimal direction is decided, then the ARdrone model in Gazebo subscribes command signals generated by the low-level PID controller to stably move a step in that direction. After completing a step-movement, the RL agent's prediction is leveraged again for the next step-movement, and this process continues until the quadcopter arrives at the desired waypoint. Algorithm 1 and Figure 8show the (Q-learning+PID) algorithm used in this paper.

Algorithm 1 PID +Q-Learning

```
Inputs:
    Learning parameters: Discount factor $\gamma$, learning rate $\alpha$, number of episode $N$
    Control Parameters: Control gains $K_p, K_i, K_d$
1: Initialize $Q_0(s,a) \leftarrow 0, \forall s_0 \in S, \forall a_0 \in A$;
2: for episode = 1, N do
3:     Acquire the Initial state $s_0$
4:     while $s_{t+1} \not\equiv$ G
5:         With probability $\epsilon$ select a random action $a_t$
6:         otherwise select $a_k$ arg max$_a$ $Q(\phi(s_k), a; \theta)$
7:         for t=0,1,2,... do
8:                 u(t)=K$_p e(t) + K_i \int e(t)dt + K_d \frac{de}{dt}$
9:
10:          until $\| e(t) = p(t) - s_{k+1} \| \leq d$
11:          Observe reward $r_{k+1}$
12:          Update Q-Table
13:          Increment time step
14:     end while
15: end for
```

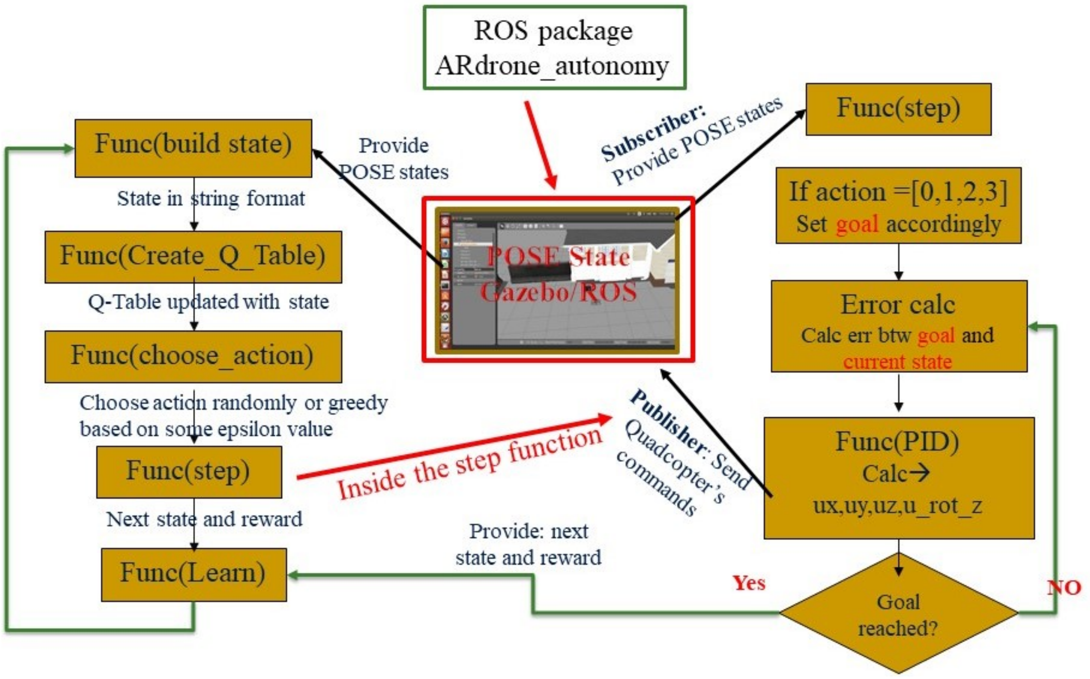
ROS package ARdrone_autonomy
Func(build state)
Provide POSE states
Subscriber: Provide POSE states
Func(step)
State in string format
If action =[0,1,2,3] Set goal accordingly
Func(Create_Q_Table)
POSE State Gazebo/ROS
Error calc Calc err btw goal and current state
Q-Table updated with state
Func(choose_action)
Choose action randomly or greedy based on some epsilon value
Publisher: Send Quadcopter's commands
Func(PID) Calc→ ux,uy,uz,u_rot_z
Func(step)
Inside the step function
Next state and reward
Provide: next state and reward
Func(Learn)
Yes
NO
Goal reached?

Fig. 8 The framework of Q-Learning for path following and waypoint navigation control

Performance evaluation: The task here to develop an efficient control strategy so that the tethered multicopter can fly through 4 predefined waypoints as fast as possible. These waypoints in the global coordinate frame are visualized in Figure 9. Table 1 shows the local coordinates conversion done in the given the global coordinates provided by the user. Note that the initial location of the drone in each waypoint navigaiton pair is $(0, 0)$. That reduces the number of possible (State-Action) pair in the Q-Table and hence having a smaller size Q-Table. The obtained results indicate that the RL agent learns effective flight directions quickly and it starts receiving high rewards after around 50 episodes only, as shown in Figure 10. The optimal high-level decisions or flight directions are learned well after 100 episodes, and the associated rewards do not improve much afterwards. When Q-learning is completed, the multicopter takes nearly 3 seconds to navigate through the 4 waypoints. Figure 12 shows the paths explored by the multicopter during the early episodes of Q-learning (blue trail) and the final path learned after episode 500 (red trail).

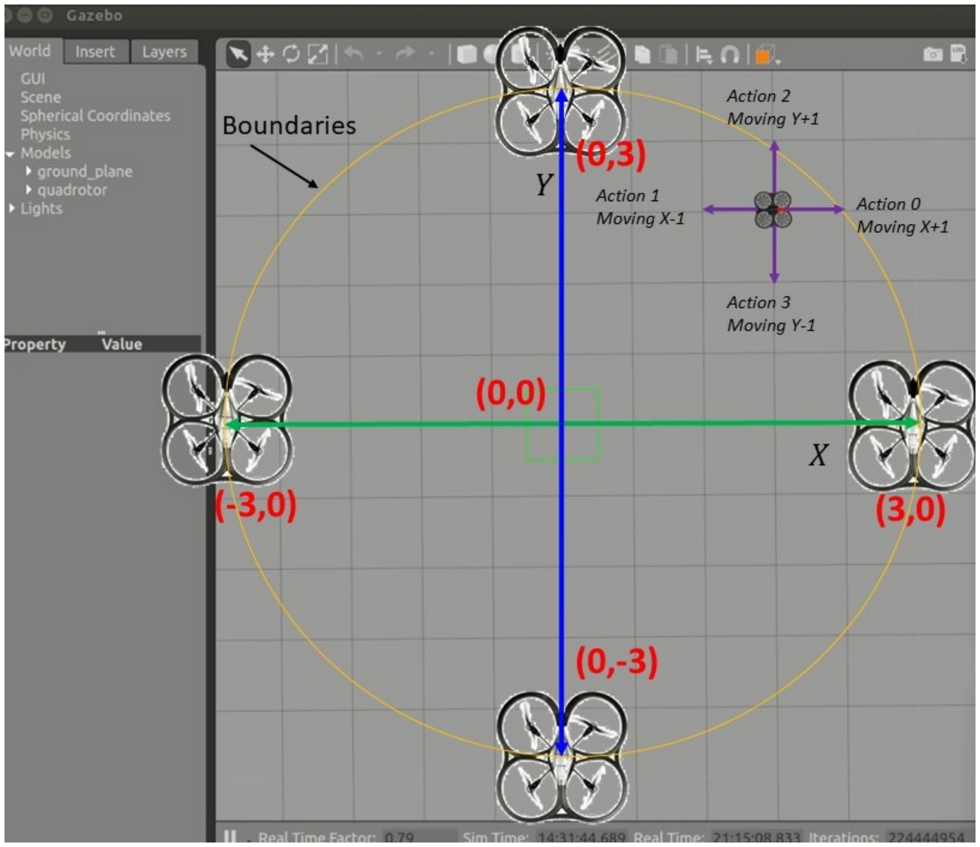

Fig. 9 Drone navigating in a closed environment, showing the actions, boundaries, sample goal waypoints.

Table 1 Waypoints coordinates in the Global and Local Frames

WP Index	Global Coordinates	Local Coordinates
$WP_0 \rightarrow WP_1$	$(0,0) \rightarrow (3,0)$	$(0,0) \rightarrow (3,0)$
$WP_1 \rightarrow WP_2$	$(3,0) \rightarrow (0,3)$	$(0,0) \rightarrow (-3,3)$
$WP_2 \rightarrow WP_3$	$(0,3) \rightarrow (-3,0)$	$(0,0) \rightarrow (-3,-3)$
$WP_3 \rightarrow WP_4$	$(-3,0) \rightarrow (0,-3)$	$(0,0) \rightarrow (3,-3)$
$WP_4 \rightarrow WP_1$	$(0,-3) \rightarrow (3,0)$	$(0,0) \rightarrow (3,3)$

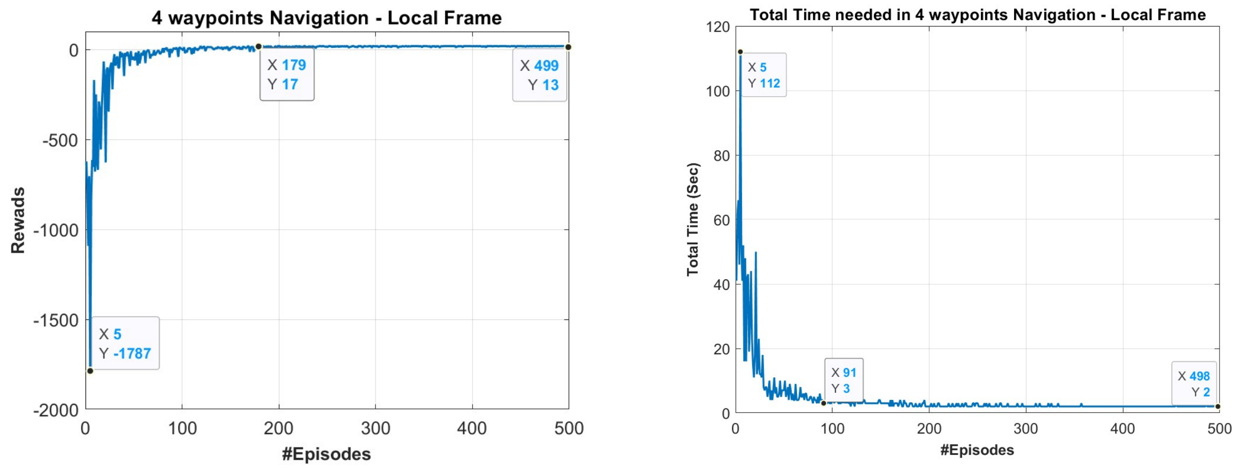

Fig. 10 Rewards and Total Time for 4 Waypoints Navigation

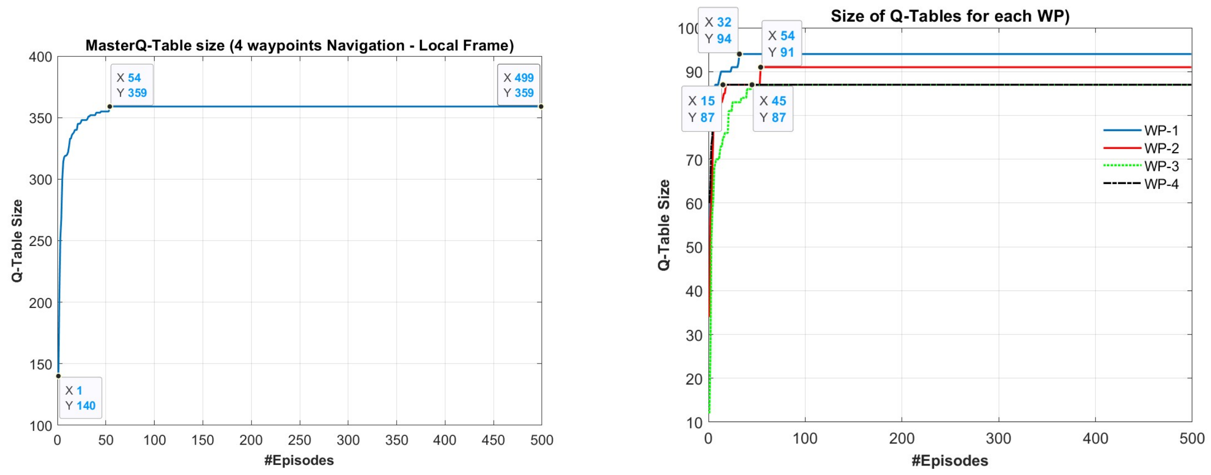

Fig. 11 Q-Table size for 4 Waypoints Navigation

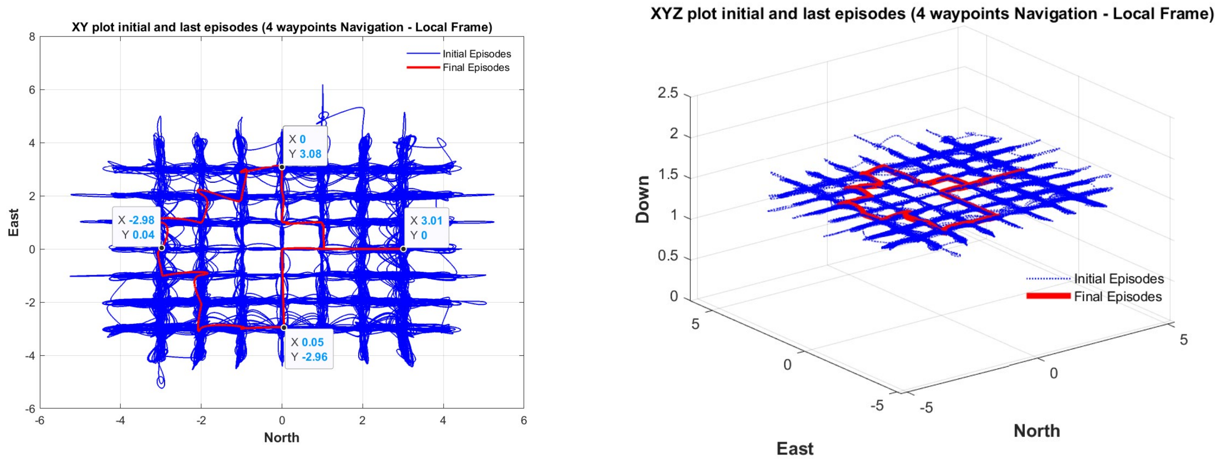

Fig. 12 2D and 3D plots for 4 Waypoints Navigation

V. Conclusion

The proposed Q-learning+PID based control technique enables a tethered drone to track a predefined trajectory in a GPS-denied environment. The implementation is carried out by leveraging the strengths of Python language and ROS-Gazebo environment. The ROS-Gazebo environment provides a realistic platform that can easily be translated into real-world applications, while Python language provides flexibility in simulating the tethered drone dynamics and sending the appropriate control commands. Simulation results justify that the tethered drone succeeds in waypoint tracking without requiring prior knowledge of the environment.

References

[1] Al-Radaideh, A., Al-Jarrah, M., and Jhemi, A., "UAV testbed building and development for research purposes at the american university of sharjah," *ISMA'10 - 7th Int. Symp. Mechatronics its Appl.*, 2010.

[2] Al-Radaideh, A., "Guidance, Control and Trajectory Tracking of Small Fixed Wing Unmanned Aerial Vehicles (UAV's)," Ph.D. thesis, 2009.

[3] Al-Radaideh, A., Al-Jarrah, M. A., Jhemi, A., and Dhaouadi, R., "ARF60 AUS-UAV modeling, system identification, guidance and control: Validation through hardware in the loop simulation," *ISMA'09 6th Int. Symp. Mechatronics its Appl. 2009.*, IEEE, 2009, pp. 1–11.

[4] Martin, P., and Salaün, E., "The true role of accelerometer feedback in quadrotor control," *Robot. Autom. (ICRA), 2010 IEEE Int. Conf.*, IEEE, 2010, pp. 1623–1629.

[5] Hoffmann, G., Huang, H., Waslander, S., and Tomlin, C., "Quadrotor helicopter flight dynamics and control: Theory and experiment," *AIAA Guid. Navig. Control Conf. Exhib.*, 2007.

[6] Escareno, J., Salazar-Cruz, S., and Lozano, R., "Embedded control of a four-rotor UAV," *Am. Control Conf. 2006*, IEEE, 2006.

[7] He, R., Prentice, S., and Roy, N., "Planning in information space for a quadrotor helicopter in a GPS-denied environment," *Proc. - IEEE Int. Conf. Robot. Autom.*, IEEE, 2008, pp. 1814–1820. doi:10.1109/ROBOT.2008.4543471.

[8] Grzonka, S., Grisetti, G., and Burgard, W., "Towards a navigation system for autonomous indoor flying," *Robot. Autom. 2009. ICRA'09. IEEE Int. Conf.*, IEEE, 2009, pp. 2878–2883.

[9] Bouabdallah, S., and Siegwart, R., "Full control of a quadrotor," *Intell. Robot. Syst. 2007. IROS 2007. IEEE/RSJ Int. Conf.*, Ieee, 2007, pp. 153–158.

[10] Guenard, N., Hamel, T., and Mahony, R., "A practical visual servo control for an unmanned aerial vehicle," *IEEE Trans. Robot.*, Vol. 24, No. 2, 2008, pp. 331–340.

[11] Kendoul, F., Fantoni, I., and Nonami, K., "Optic flow-based vision system for autonomous 3D localization and control of small aerial vehicles," *Rob. Auton. Syst.*, Vol. 57, No. 6, 2009, pp. 591–602.

[12] Xu, Q., Wang, Z., Gerber, A., and Mao, Z. M., "Cellular Data Network Infrastructure Characterization and Implication on Mobile Content Placement," *Proc. ACM SIGMETRICS Jt. Int. Conf. Meas. Model. Comput. Syst.*, 2011, pp. 317—-328.

[13] van Kampen, E.-J., Chu, Q., and Mulder, J., "Continuous adaptive critic flight control aided with approximated plant dynamics," *AIAA Guidance, Navigation, and Control Conference and Exhibit*, 2006, p. 6429.

[14] Zhou, Y., Kampen, E.-J. v., and Chu, Q., "Nonlinear adaptive flight control using incremental approximate dynamic programming and output feedback," *Journal of Guidance, Control, and Dynamics*, Vol. 40, No. 2, 2016, pp. 493–496.

[15] Zhou, Y., Van Kampen, E., and Chu, Q. P., "Incremental approximate dynamic programming for nonlinear flight control design," *Proceedings of the 3rd CEAS EuroGNC: Specialist Conference on Guidance, Navigation and Control, Toulouse, France, 13-15 April 2015*, 2015.

[16] Zhou, Y., van Kampen, E., and Chu, Q., "Incremental model based heuristic dynamic programming for nonlinear adaptive flight control," *Proceedings of the International Micro Air Vehicles Conference and Competition 2016, Beijing, China*, 2016.

[17] Abbeel, P., Coates, A., Quigley, M., and Ng, A. Y., "An application of reinforcement learning to aerobatic helicopter flight," *Advances in neural information processing systems*, Vol. 19, 2007, p. 1.

[18] Valasek, J., Doebbler, J., Tandale, M. D., and Meade, A. J., "Improved adaptive–reinforcement learning control for morphing unmanned air vehicles," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, Vol. 38, No. 4, 2008, pp. 1014–1020.

[19] Valasek, J., Kirkpatrick, K., May, J., and Harris, J., "Intelligent motion video guidance for unmanned air system ground target surveillance," *Journal of Aerospace Information Systems*, Vol. 13, No. 1, 2016, pp. 10–26.

[20] Bialy, B. J., Klotz, J., Brink, K., and Dixon, W. E., "Lyapunov-based robust adaptive control of a quadrotor UAV in the presence of modeling uncertainties," *2013 American Control Conference*, IEEE, 2013, pp. 13–18.

[21] Raj, J., Raghuwaiya, K. S., and Vanualailai, J., "Novel lyapunov-based autonomous controllers for quadrotors," *IEEE Access*, Vol. 8, 2020, pp. 47393–47406.

[22] Bouadi, H., Bouchoucha, M., and Tadjine, M., "Sliding mode control based on backstepping approach for an UAV type-quadrotor," *World Academy of Science, Engineering and Technology*, Vol. 26, No. 5, 2007, pp. 22–27.

[23] Al-Radaideh, A., and Sun, L., "Self-localization of a tethered quadcopter using inertial sensors in a GPS-denied environment," *2017 Int. Conf. Unmanned Aircr. Syst. ICUAS 2017*, 2017, pp. 271–277. doi:10.1109/ICUAS.2017.7991436.

[24] Al-Radaideh, A., and Sun, L., "Observability Analysis and Bayesian Filtering for Self-Localization of a Tethered Multicopter in GPS-Denied Environments," *International Conference on Unmanned Aircraft Systems (ICUAS)*, Atlanta, Georgia, USA, 2019, pp. 271–277.

[25] Al-Radaideh, A., and Sun, L., "Self-Localization of Tethered Drones without a Cable Force Sensor in GPS-Denied Environments," *Drones*, Vol. 5, No. 4, 2021, p. 135.

[26] Beard, R. W., and McLain, T. W., *Small unmanned aircraft: Theory and practice*, Princeton university press, 2012.

[27] Rauw, M. O., "FDC 1.2-A Simulink Toolbox for Flight Dynamics and Control Analysis," *Zeist, The Netherlands*, Vol. 1, No. 99, 2001, p. 7.

[28] Sutton, R. S., Precup, D., and Singh, S., "Between MDPs and semi-MDPs: A framework for temporal abstraction in reinforcement learning," *Artificial intelligence*, Vol. 112, No. 1-2, 1999, pp. 181–211.

[29] Siddiquee, M., Junell, J., and Van Kampen, E.-J., "Flight test of Quadcopter Guidance with Vision-Based Reinforcement Learning," *AIAA Scitech 2019 Forum*, 2019, p. 0142.

[30] Li, S., Liu, T., Zhang, C., Yeung, D., and Shen, S., "Learning unmanned aerial vehicle control for autonomous target following," *Proceedings of the International Joint Conference on Artificial Intelligence*, 2018.

[31] Monajjemi, M., et al., "ardrone autonomy: A ros driver for ardrone 1.0 & 2.0," , 2012.