Article

Robust Quadrotor Control through Reinforcement Learning with Disturbance Compensation

Chen-Huan Pi, Wei-Yuan Ye and Stone Cheng *

Department of Mechanical Engineering, National Yang Ming Chiao Tung University, Hsinchu City 30010, Taiwan; john40532.me6008@g2.nctu.edu.tw (C.-H.P); s10030789.me035@g2.nctu.edu.tw (W.-Y.Y.)
* Correspondence: stonecheng@mail.nctu.edu.tw

Abstract: In this paper, a novel control strategy is presented for reinforcement learning with disturbance compensation to solve the problem of quadrotor positioning under external disturbance. The proposed control scheme applies a trained neural-network-based reinforcement learning agent to control the quadrotor, and its output is directly mapped to four actuators in an end-to-end manner. The proposed control scheme constructs a disturbance observer to estimate the external forces exerted on the three axes of the quadrotor, such as wind gusts in an outdoor environment. By introducing an interference compensator into the neural network control agent, the tracking accuracy and robustness were significantly increased in indoor and outdoor experiments. The experimental results indicate that the proposed control strategy is highly robust to external disturbances. In the experiments, compensation improved control accuracy and reduced positioning error by 75%. To the best of our knowledge, this study is the first to achieve quadrotor positioning control through low-level reinforcement learning by using a global positioning system in an outdoor environment.

Keywords: external disturbance; quadrotor; reinforcement learning

1. Introduction

A quadrotor is an underactuated, nonlinear coupled system. Because quadrotors have various applications, researchers have long been focusing on the problems of attitude stabilization and trajectory tracking in quadrotors. Many control methods are used for quadrotors. Proportional–integral–derivative (PID) controllers are widely used in consumer quadrotor products and is often treated as a baseline controller for comparison with other controllers [1]. In practice, the tuning of the PID controller’s gain often requires expertise, and the gain is selected intuitively by trial and error. Advanced control strategies using model-based methods have been applied to improve the flight performance of quadrotors. Methods such as feedback linearization [2], model predictive control (MPC) [3,4], robust control [5], sliding mode control (SMC) [6–8], and adaptive control [9,10] have been applied to optimize the flight performance of quadrotors. However, the performance and the robustness of the aforementioned strategies are highly related to the accuracy of the manually developed dynamic model.

During outdoor flight, quadrotors are susceptible to wind gust, which affects the flight performance or even leads to system instability [3]. Although quadrotors are sensitive to external disturbances [11], designers of most controllers have not accounted for this problem. Some active disturbance rejection methods have been proposed to estimate disturbances, and these methods perform well in cases of a sustained disturbance. Chovancova et al. [12] designed proportional–derivative (PD), linear quadratic regulator (LQR), and backstepping controllers for position tracking and compared their performance in a simulation. A disturbance observer with a position estimator was designed to improve controller positioning performance, which was evaluated when external disturbance was applied in simulations. The active disturbance rejection control (ADRC) algorithm treats the total disturbance as a new state variable and estimates it through an extended state observer (ESO). Moreover,
the ADRC algorithm does not require the exact mathematical model of the overall system to be known. Therefore, this algorithm has become an attractive technique for the flight control of quadrotor unmanned aerial vehicles (UAVs) [13,14]. Yang et al. proposed the use of ADRC and PD control in a dual closed-loop control framework [15]. An ESO was used to estimate the perturbations of gust wind as dynamic disturbances in the inner loop control. A quadrotor flight controller with a sliding mode disturbance observer (SMC-SMDO) was used in [16]. The SMC-SMDO is robust to external disturbances and model uncertainties without the use of high control gain. Chen et al. [17] constructed a nonlinear disturbance observer that considers external disturbances from wind model uncertainties separately from the controller and compensates for the negative effects of the disturbances. In [18], a nonlinear observer based on an unscented Kalman filter was developed for estimating the external force and torque. This estimator reacted to a wide variety of disturbances in the experiment conducted in [18].

Reinforcement learning (RL) has solved many complicated quadrotor control problems in many studies. RL outperforms other optimization approaches and does not require a pre-defined controller structure, which limits the performance of an agent. In [19], a quadrotor with a deep neural network (DNN)-based controller was proposed for following trails in an unstructured outdoor environment. In [20], RL and MPC were used to enable a quadrotor to navigate unknown environments. MPC enables vehicle control, whereas RL is used to guide a quadrotor through complex environments. In addition to high-level planning and navigation problems, RL control has been used for achieving robust attitude and position control [21,22]. The control policy generated through the RL training of a neural network achieves low-level stabilization and position control and the policy can control the quadrotor directly from the quadrotor state inputs to four motor outputs. The aforementioned studies have implemented their proposed control strategies in simulations and real environments. Although quadrotors with RL controller exhibit stability under disturbance, the control policy cannot eliminate the steady-state error caused by wind or modeling error and the performance of the controller can be improved. In [23], an integral compensator was used to enhance tracking accuracy. The effect of this compensator on the tracking accuracy of the controller was verified by introducing a constant horizontal wind that flowed parallel to the ground in a simulation. Although the aforementioned integral compensator can eliminate the steady-state error, it slows down the controller response and has a large overshoot.

This paper presents a unique disturbance compensation RL (DCRL) framework that includes a disturbance compensator and an RL controller. The external disturbance observer in this framework is based on the work of [24]. The rest of this paper is organized as follows. Section 2 introduces the dynamic model of a quadrotor and the basics of RL. Section 3 describes the proposed DCRL control strategy. Section 4 describes the training and implementation of the proposed DCRL strategy in a quadrotor experiment in indoor and outdoor environments. Finally, Section 5 concludes the paper.

2. Preliminary Information

This section briefly introduces the dynamic model of a quadrotor, the basics of RL and the use of RL in solving the quadrotor control problem.

2.1. Quadrotor Dynamic Model

In this paper, we assume that a quadrotor is a rigid and symmetrical body whose center of gravity coincides with its geometric center.

The vector \( \mathbf{x} = [x, y, z]^T \) denotes the position of the quadrotor in an inertial frame. The translation dynamics of the quadrotor can be expressed as follows:

\[
\dot{x} = \mathbf{v} \\
\dot{v} = m^{-1} \left\{ \sum_{i=1}^{4} T_i \mathbf{b}_z + F_{ext} \right\} - g \mathbf{1}_z,
\]
where \( m \) and \( g \) are the mass of the quadrotor and the acceleration due to gravity, respectively; \( \mathbf{R} = [\mathbf{b}_x \mathbf{b}_y \mathbf{b}_z] \in SO(3) \) is the rotation matrix, which is used to transform a coordinate from the body-fixed reference frame to the inertial reference frame; and \( T_i \) is the thrust generated from motors and applied on the \( z \)-axis of the body frame \( \mathbf{b}_z \). Figure 1 displays the order of motors placement. Finally, the vector \( \mathbf{F}_{ext} \) represents the external disturbance force accounting for all other forces acting on the quadrotor.

For the rotation dynamics of system, we use a quaternion representation of quadrotor attitude to avoid gimbal lock and ensure better computational efficiency.

\[
\dot{q} = 0.5 \cdot q \otimes \begin{bmatrix} 0 \\ \Omega \end{bmatrix}
\]

\[
\dot{\Omega} = J^{-1}(\mu - \Omega \times J \Omega),
\]

where \( \dot{q} = [q_w, q_x, q_y, q_z]^T \) is the normed quaternion attitude vector, \( \otimes \) is the quaternion multiplication. \( \Omega \) is the angular velocity of body-frame, \( \mu \) is the control moment vector and \( J \) is the matrix of vehicle moment of inertia tensor. The rotation transformation between the quaternion \( q \) to the rotation matrix \( R \) can be expressed as follows:

\[
R = \begin{bmatrix}
1 - 2(q_y^2 + q_z^2) & 2(q_xq_y - q_zq_w) & 2(q_xq_z + q_yq_w) \\
2(q_xq_y + q_zq_w) & 1 - 2(q_x^2 + q_z^2) & 2(q_yq_z - q_xq_w) \\
2(q_xq_z - q_yq_w) & 2(q_yq_z + q_xq_w) & 1 - 2(q_y^2 + q_z^2)
\end{bmatrix}
\]

Each thrust from the propeller axis is assumed to be aligned perfectly with the \( z \)-axis. The force \( T_i \) and motor moment \( \mu \) produced at a motor spinning speed of \( \omega_i \) can be expressed as follows:

\[
T_i = c_f \omega_i^2
\]

\[
\mu = \frac{1}{\sqrt{2}} \begin{bmatrix}
-lc_f & lc_f & lc_f & -lc_f \\
-lc_f & lc_f & -lc_f & lc_f \\
lc_f & lc_f & lc_f & lc_f \\
-c_d & c_d & c_d & -c_d
\end{bmatrix} \begin{bmatrix}
\omega_1 \\
\omega_2 \\
\omega_3 \\
\omega_4
\end{bmatrix}^2
\]

where \( i = 1, 2, 3, 4 \). \( \Omega_i \) is the speed of motors; \( l \) is the arm length of the quadrotor; and \( c_f \), \( c_d \) are the coefficients of the generated force and \( z \)-axis moment, respectively.

![Figure 1. Body-fixed frame of the quadrotor and motor placement.](image)

The developed dynamic model is based on the following assumptions: (a) the quadrotor structure is rigid, (b) the center of mass of the quadrotor and the rotor thrusts are in the same plane, and (c) blade flapping and aerodynamics can be ignored.

### 2.2. Reinforcement Learning

The standard RL framework comprises a learning agent interacting with the environment according to a Markov decision process (MDP). A state transition has a Markov
where the probability of this transition is independent of its history. The MDP involves solving decision problems with the Markov property, and RL theories are based on the MDP. The standard MDP is defined by the tuple \((S, A, P^\pi_s, r, \rho_0, \gamma)\), where \(S\) is the state space, \(A\) is the action space, \(P^\pi_s : S \times A \times S \rightarrow \mathbb{R}^+\) is the transition probability density of the environment, \(r : S \times A \rightarrow \mathbb{R}\) is the reward function, \(\rho_0 : S \rightarrow \mathbb{R}^+\) is the distribution of the initial state \(s_0\), and \(\gamma \in (0, 1)\) is the discount factor. In modern deep RL conducted using neural networks, the agent selects an action at each time step according to the policy \(\pi(a|s; \theta) = Pr(a|s; \theta)\), where \(\theta \in \mathbb{R}^{N_{\theta}}\) is the weight of the neural network.

The goal of the MDP is to find a policy \(\pi(a \in A|s)\) that can maximize the cumulative discounted reward.

\[
\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t).
\]

A state-dependent value function \(V^\pi\) that measures the expected discounted reward with respect to \(\pi\) can be defined as follows:

\[
V^\pi(s) = \mathbb{E} \left[ \sum_{t=0}^{l} \gamma^t r_{t+1} \mid s_t = s, \pi \right].
\]

The state-action-dependent value function can be defined as follows:

\[
Q^\pi(s, a) = \mathbb{E} \left[ \sum_{t=0}^{l} \gamma^t r_{t+1} \mid s_t = s, a_t = a, \pi \right].
\]

The advantage function can be defined as follows:

\[
A^\pi(s, a) = Q^\pi(s, a) - V^\pi(s),
\]

where \(A^\pi\) is the difference between the expected value when selecting some specific action \(a\). The advantage function can be used to determine whether the selected action is suitable with respect to policy \(\pi\). Many basic RL algorithms, such as the policy gradient method [25], off-policy actor–critic algorithm [26], and trust region policy optimization [27] can be used to optimize a policy. To maximize the expected reward function \(V^\pi(s)\), the neural network parameterized policy \(\pi = \pi(a|s; \theta)\) is adjusted as follows:

\[
\theta \leftarrow \theta + \alpha \sum_{s \in S, a \in A} \rho^\pi(s) Q^\pi(s, a) \frac{\partial \pi(a|s)}{\partial \theta},
\]

where

\[
\rho^\pi(s) = \sum_{t \geq 0} \gamma^t Pr(s_t = s|t, a \sim \pi)
\]

is the state occurrence probability and \(\alpha > 0\) is the size of the learning step. Equation (9) is an expression for the policy gradient [28]. By using the state distribution \(\rho\) and state-action value function \(Q\), a policy can be improved without any environmental information. The state distribution \(\rho^\pi(s)\) depends on the policy \(\pi\), which indicates that it must be re-estimated when the policy is changed. In [26], the policy gradient was analyzed by replacing the original policy \(\pi\) with another policy \(\mu\); therefore, (9) had the following form in [26]:

\[
\sum_{s \in S, a \in A} \rho^\mu(s) Q^\pi(s, a) \frac{\partial \pi(a|s)}{\partial \theta},
\]

Equation (11) can still maximize \(V^\pi\) with distinct policy gradient strategies.

To solve the nonlinear dynamic control problem for a quadrotor, we used the proximal policy optimization algorithm (PPO) [29] and off-policy training method [22] to train the actor and critic functions. The following inequalities are valid for the actor function:
\[ V^\pi(s) - V^\mu(s) \geq 0, \forall (s,a) \in \mathcal{S} \times \mathcal{A}, \quad (12) \]

if

\[ [\pi(a|s) - \mu(a|s)]A^\mu(s,a) > 0 \quad (13a) \]

or

\[ [\pi(a|s) - \mu(a|s)]A^\pi(s,a) > 0. \quad (13b) \]

Therefore, for a policy search iteration, (13) provides improvement criteria for the action policy under a certain state.

3. Disturbance Observation and Control Strategy

Figure 2 depicts a block diagram of the quadrotor control with the DCRL framework. The proposed DCRL strategy enhances the RL control policy with external disturbance observer and compensator to strengthen the system robustness. The compensation algorithm estimates the external forces and adjusts the input command for the RL controller. The RL controller then changes the motor thrusts accordingly. In Figure 2, the observer takes the attitude \( q_f \) and acceleration \( a_f \) as input, and outputs the estimated external disturbance. \( F_{\text{ext}}, \hat{F}_{\text{ext}} \) are the external disturbance and the external disturbance estimated by the observer. The disturbance compensator calculates the \( q_{\text{comp}} \) from quadrotor attitude \( q_f \) and \( \hat{F}_{\text{ext}} \). The original RL actor was trained to hover at the original point by receiving the state \( s \) which contains the position, velocity, attitude and angular velocity of the quadrotor, and output four motors thrust follows the policy \( \alpha(s) \). In DCRL, to make the quadrotor hover at the reference position \( x_{\text{ref}} \), the original point can be shifted with an off-set of reference command and as the input of position \( x_{\text{dev}} \) to the RL actor. The DCRL generates thrust command with the sum of RL actor output and compensation force.

![Figure 2. The structures of quadrotor control. (a) Block diagram of the quadrotor control structure with reinforcement learning control and external disturbance compensator. (b) The neural network in the reinforcement learning (RL) actor.](image-url)
The RL controller in the DCRL was trained to recover and hover at the original point under an ideal simulation environment, while the external disturbances were not considered in RL control policy training. Several reasons exist for not adding external disturbances in RL training. First, the RL controller sometimes has superior performance to traditional methods if the simulation environment is highly similar to a real-world controller model. However, such a model has numerous uncertainties and is highly difficult to reproduce in a simulator. In this study, the sensor noise was one of the uncertain factors because each inertial measurement unit (IMU) sensor on the flight computer had different physical characteristics. Second, the sensor noise does not follow the assumption of the MDP in RL theory; thus, the final performance of the trained policy cannot be guaranteed to be suitable. Finally, the aforementioned traditional external disturbance estimation methods have been demonstrated to be effective. Therefore, we focused on eliminating known disturbances by using an RL controller with a traditional observation method for achieving a superior positioning performance in this study.

3.1. Disturbance Observer

In general, by rearranging the terms in (1), the external force may be calculated directly from the acceleration information as follows:

\[
\hat{F}_{ext} = mR^T a^b - \sum_{i=1}^{4} T_i b_z + mg_i z,
\]

where the thrust forces are only applied on the z-axis of the quadrotor in the body frame and \( a^b \) is the acceleration measured by the onboard IMU sensor. The parameter \( a^b \) includes the gravitational acceleration.

A low-pass filter (LPF) with cut-off frequency at 30 Hz is used to reduce the effects of noise caused by rotor spinning vibrations or the IMU. The thrust and acceleration are transformed to the inertial reference frame prior to filtering. The reason of this pre-processing is that the external force in the inertial reference frame \( \hat{F}_{ext} \) is assumed to have a lower rate of change than do be slow-changing relative to the LPF dynamics.

3.2. Disturbance Compensator

When an external disturbance \( F_{ext} \) is acting on the quadrotor (Figure 3), this disturbance generates a translational acceleration vector \( a_{ext} \). For disturbance compensation, a new compensation thrust vector \( g_{fc} \) is defined. This vector combines \( a_{ext} \) and the gravitational acceleration vector \( g_i z \) and can be expressed as follows:

\[
i_{zc} = a_{ext} + g_i z \quad \text{and} \quad g_{fc} = g + a_{ext} \quad \text{or} \quad g_{fc} = g + a,
\]

which only considers the hovering situation without an acceleration command from the trajectory tracking reference. The normalized vector \( g_{fc} \) is then used to formulate a new coordinate frame (force compensation frame) with a rotation matrix \( R_{ci} \) relative to the inertial frame.

![Figure 3. Force compensation frame.](image-url)
In three-dimensional space, any rotation coordinate system about a fixed point is equivalent to a single rotation by an angle $\theta$ about a fixed axis (called the Euler axis) that passes through the fixed point.

To obtain the rotation matrix $R_{ci}$ from $i_z$ to $g_{fc}$ represent in quaternion with the following equations can be used:

$$v = [0, g_{fc}]$$

$$q = [\cos(\frac{\theta}{2}), \sin(\frac{\theta}{2})u]$$

$$v' = qvq^{-1} = qvq^{-1} = i_z,$$

where $v$ is the original quaternion vector, $u$ is the unit vector of the rotation axis, $v'$ is the rotated quaternion vector, $q$ is the rotation vector between $v$ and $v'$, and $\theta$ is the rotation angle.

By substituting $q_{ci}$ into (3), $R_{ci}$ can be determined as follows:

$$v' = q_{ci}vq_{ci}^{-1} = q_{ci}vq_{ci}^{-1} = R_{ci}v.$$  \hspace{1cm} (17)

The aforementioned equation is equivalent to Rodrigues’ rotation formula. After obtaining the rotation matrix $R_{ci}$, the quadrotor attitude in the force compensation frame $R_{cb}$ can be calculated using the following equation:

$$R_{cb} = R_{ci}R_{ib}. \hspace{1cm} (18)$$

After obtained the corrected coordinate for compensation, the magnitude of thrust of motors with compensation is

$$T_{ci} = \hat{f}_{ext} \cdot b_z + a_i(s), \hspace{1cm} (19)$$

where $a_i(s)$ is the $i$-th motor action output of neural network which would be specified in following section. By modifying the quadrotor attitude in the compensation frame $R_{cb}$ and using this attitude as the input state of the RL controller, the controller can generate corresponding motor thrust to maintain the target attitude and therefore eliminate the disturbance acting on the quadrotor.

4. Experiments

In this section, we introduce our training method for a low-level quadrotor control policy. The RL controller receives the information on the quadrotor state (position, velocity, attitude, and angular velocity) from sensors and directly outputs the control commands of four rotors. The training was first performed and tested in a simulator. The quadrotor simulator was established using Python according to the dynamic model described in Section 2.1 for training and verifying the flight performance.

After verifying that the RL control policy was trained successfully, we transported the controller into our DCRL structure and performed a real flight with the quadrotor. To implement the proposed DCRL control algorithm in this study, PixRacer flight controller hardware was developed and implemented using Simulink. The DCRL control strategy was examined in an indoor environment by performing fixed-point hovering under an external wind disturbance. Then, the quadrotor was set to track a square trajectory in an outdoor experiment. The position and velocity of the quadrotor were obtained using an OptiTrack motion capture system on the ground station computer. These data were transmitted through Wi-Fi to the onboard PixRacer flight controller within 10 m range in the indoor experiment. The physical parameters of the quadrotor platform are presented...
in Table 1. For outdoor trajectory tracking, position information was only obtained from an onboard global positioning system (GPS) sensor.

Table 1. Physical parameters of the quadrotor.

| Weight (g) | \( I_{xx}, I_{yy}, I_{zz} \) (kgm²) |
|------------|----------------------------------|
| 665        | 0.0023, 0.0025, 0.0037            |

4.1. RL Controller Training

In the RL training process, we followed the dynamic equations in Section 2.1 and constructed a simulation environment in Python to generate training data. In the simulation environment, the state space of the MDP comprised the position, velocity, attitude, and angular velocity of the quadrotor. Moreover, the four motors thrust outputs were chosen as the action space. The training process follows the work in [22] using two processes, one for data collection and another for value and policy network update. The update is based on off-policy training, and the main difference between on-policy is the on-policy only uses the collected data once and be cleaned up after each time neural network updates. On the contrary in the off-policy training, the collection thread keeps generating the trajectory data and neural network updating thread can reuse the data from collection which accelerates the learning process.

In data collection process, the quadrotor was randomly launched in a 2 m cubic space with random states. The training data, which comprised the quadrotor states, action, probability, and reward, were recorded in each episode which contained 200 steps in two seconds flight, and then saved as a single data trajectory in a memory buffer. The normalized reward function for evaluating the current state of the quadrotor is as follows:

\[
    r = -(0.002\|e_q\| + 0.002\|e_p\| + 0.002\|a\|),
\]

where \( e_q \) is the vehicle angle error, \( e_p \) is the vehicle position error, and \( a \) is the motor thrust command for constraining the energy cost.

When the number of data trajectories in memory buffer exceeded 10, the training process starts. In training process, trajectories were randomly sampled from the memory buffer. The advantage and value functions were defined recursively and calculated in reverse direction which depend on the future time \( t+1 \). The functions were estimated according to

\[
    A^\text{trace}_{t} = A_{t} + \gamma \min(1, \frac{\pi_{t+1}}{\mu_{t+1}}) A^\text{trace}_{t+1},
\]

and

\[
    V^\text{trace}_{t} = V_{t} + \min(1, \frac{\pi_{t}}{\mu_{t}}) A^\text{trace}_{t}.
\]

With the two equations above, we use stochastic gradient descent to optimize the objectives as follows:

\[
    \text{maximize } L_{\text{policy}} = \sum_{(s,a) \in T} \min \left[ \left( \frac{\pi(a|s)}{\mu(a|s)} - 1 \right) A^\text{trace}, e|A^\text{trace}| \right]
\]

\[
    \text{minimize } L_{\text{value}} = \frac{1}{|T|} \sum_{(s,a) \in T} (V(s) - V^\text{trace})^2.
\]
To approximate the function $\pi(a|s)$ for proposing actions and $V(s)$ for predicting the state value, two neural network were formulate as following equations, where stochastic Gaussian policy was used for the actor network:

$$\theta(s) = h_{32}^1 \circ h_{32}^0(s)$$

(24)

$$\alpha(s) = \sin \circ y_2^1 \circ \theta(s)$$

(25)

$$\sigma(s) = \frac{1}{2} + \frac{1}{2} \cos \circ y_2^1 \circ \theta(s)$$

(26)

$$\pi(a|s) = \frac{1}{\sqrt{2\pi\sigma^2(s)}} e^{-\frac{(a-\alpha(s))^2}{2\sigma^2(s)}}.$$  

(27)

The state value function can be approximated as follows:

$$V(s) = y_2^1 \circ h_{128}^1 \circ h_{0}^{128}(s),$$

(28)

where $h_j^i$ and $y_j^i$ are the $i$th hidden layer and output layer of neural networks with width $j$, $\circ$ is the fully connected activation function. In both the actor and critic networks, the input state $s$ is the quadrotor's position, velocity, attitude and angular velocity. The sin and cos functions are used to constrain the output range. A rectified linear unit (ReLU) is used as the activation function due to its characteristic of fast calculation and easy implementation in a microcontroller unit. When implementing the RL controller in a quadrotor flight computer system, only (25) is used to control the quadrotor.

To apply the developed RL controller in an outdoor environment, we extracted the parameters of a trained neural network and loaded them into a Simulink model. The input state of position was limited to the same finite range as that adopted in the training environment to prevent an untrained condition from occurring when using the developed RL controller with a GPS in outdoor environments. With the successful training of the external disturbance observer and RL neural network controller, the DCRL control policy was transferred to the PixRacer flight computer in real quadrotors to replace the original PID controller.

4.2. Results of the Indoor Experiment

In the indoor experiment, we put an electric fan to simulate a constant wind disturbance (Figure 4). We used a self-made quadrotor with a flight control board and GPS mounted in the plane. An OptiTrack motion capture system provides reliable state information, and a multisensor fusion framework in the flight computer fuses the measurement from the onboard IMU and the motion capture data to compensate for the time delay and low update frequency of the OptiTrack system. We compared the position errors between RL with and without compensation under wind disturbance. The measured wind speed was 3.6 m/s at the center of the $x$-axis of the quadrotor. Figure 5 displays the position error histogram. The mean errors of the original RL and DCRL controllers were 8.4 and 2 cm, respectively, which reduced the hovering error by 75%. Figure 6 presents the estimated position error and external disturbance force for a 30-s flight. Video clips of the indoor experiment can be found at https://youtu.be/RtAoiljZTSI (accessed on 5 April 2021).

4.3. Results of the Outdoor Experiment

After verifying the DCRL control algorithm under motion capture accurate position and velocity measurement and relative steady wind perturbation in a laboratory environment, the quadrotor was moved outdoors and a GPS was used to obtain position feedback. The maximum wind speed was measured to be 4.2 m/s by an anemometer. The position trajectory was a 10-m$^2$ square with a constant height and a velocity of 1 m/s. The results of the outdoor experiment are shown in Figure 7, and the estimated external forces acting on the $x$-axis and $y$-axis are presented in Figure 8. Table 2 summarizes the position errors in the indoor and outdoor experiments. In the outdoor experiment, a position waypoint was
used as a reference for trajectory tracking without a velocity command. Thus, the quadrotor had to maintain a certain position error to obtain the moving velocity for following the waypoint. The tracking errors may have also been caused by the 2.5-m horizontal position accuracy and 10-Hz update rate of the adopted GPS sensor. However, the experimental results still indicate that the DCRL structure can reduce the quadrotor positioning error.

Figure 4. Experimental setup for indoor fixed-position hovering. An electric fan was used for simulating a constant wind disturbance with a speed of 3.6 m/s which was applied along the x-axis of the quadrotor.

Figure 5. The position error histogram distribution of indoor fix-position hover experiment. The mean error of the original RL controller and the disturbance compensation RL (DCRL) is 8.4 cm and is 2 cm respectively.
Figure 6. (a) The indoor fix-position hover experiment setup. An electric fan was used for simulating a constant wind disturbance with wind speed 3.6 m/s applied onto quadrotor x-axis. (b) The position error along x-axis. (c) The estimation of external force on x-axis.

Figure 7. The position of tracking a 10 m square trajectory in an outdoor experiment. The GPS sensor was used for position information measurement.
Figure 8. The estimation of external forces on x and y-axis in outdoor tracking experiment.

Table 2. Root mean square errors along the three quadrotor axes in the indoor and outdoor experiments.

|          | $e_x$ RMSE (m) | $e_y$ RMSE (m) | $e_z$ RMSE (m) |
|----------|----------------|----------------|----------------|
| Indoor   | RL 0.08        | 0.04           | 0.02           |
|          | DCRL 0.02      | 0.06           | 0.02           |
| Outdoor  | RL 0.80        | 0.46           | 0.17           |
|          | DCRL 0.45      | 0.41           | 0.07           |

5. Conclusions

In this paper, an RL control structure with external force compensation and an external force disturbance observer is proposed for quadrotors. The DCRL controller can reduce the effects of wind gust on quadrotors in fixed-position hovering and trajectory tracking tasks and improve their flight performance. In the outdoor experiments, compared with the original RL control algorithm, the proposed control strategy reduced the fixed-position hovering error by 75%. To the best of our knowledge, this study is the first to use a low-level RL controller with a GPS in an outdoor environment to eliminate the external disturbance acting on flying quadrotors.

Author Contributions: Conceptualization, C.-H.P.; Methodology, C.-H.P.; Project administration, S.C.; Software, C.-H.P. and W.-Y.Y.; Supervision, S.C.; Validation, C.-H.P. and W.-Y.Y.; Writing—original draft, C.-H.P. and W.-Y.Y.; Writing—review & editing, S.C. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data presented in this study are available on request from the corresponding author.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Bouabdallah, S.; Notth, A.; Siegwart, R. PID vs. LQ control techniques applied to an indoor micro quadrotor. In Proceedings of the 2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (IEEE Cat. No.04CH37566), Sendai, Japan, 28 September–2 October 2004; Volume 3, pp. 2451–2456.

2. Lee, D.; Kim, H.J.; Sastry, S.S. Feedback linearization vs. adaptive sliding mode control for a quadrotor helicopter. *Int. J. Control. Autom. Syst.* **2009**, *7*, 419–428. [CrossRef]

3. Alexis, K.; Nikolakopoulos, G.; Tzes, A. Switching model predictive attitude control for a quadrotor helicopter subject to atmospheric disturbances. *Control Eng. Pract.* **2011**, *19*, 1195–1207. [CrossRef]

4. Alexis, K.; Nikolakopoulos, G.; Tzes, A. Model predictive quadrotor control: Attitude, altitude and position experimental studies. *IET Control Theory Appl.* **2012**, *6*, 1812–1827. [CrossRef]
5. Lee, T. Robust Adaptive Attitude Tracking on $\text{SO}(3)$ With an Application to a Quadrotor UAV. *IEEE Trans. Control Syst. Technol.* 2013, 21, 1924–1930.

6. Wang, H.; Ye, X.; Tian, Y.; Zheng, G.; Christov, N. Model-free–based terminal SMC of quadrotor attitude and position. *IEEE Trans. Aerosp. Electron. Syst.* 2016, 52, 2519–2528. [CrossRef]

7. Xu, B. Composite Learning Finite-Time Control With Application to Quadrotors. *IEEE Trans. Syst. Man Cybern. Syst.* 2018, 48, 1806–1815. [CrossRef]

8. Xu, R.; Özgüner, Ü. Sliding Mode Control of a Quadrotor Helicopter. In Proceedings of the 45th IEEE Conference on Decision and Control, San Diego, CA, USA, 13–15 December 2006; pp. 4957–4962.

9. Dydek, Z.T.; Annaswamy, A.M.; Lavretsky, E. Adaptive Control of Quadrotor UAVs: A Design Trade Study with Flight Evaluations. *IEEE Trans. Control Syst. Technol.* 2013, 21, 1400–1406. [CrossRef]

10. Zou, Y.; Meng, Z. Immersion and Invariance-Based Adaptive Controller for Quadrotor Systems. *IEEE Trans. Syst. Man Cybern. Syst.* 2019, 49, 2288–2297. [CrossRef]

11. Liu, H.; Zhao, W.; Zuo, Z.; Zhong, Y. Robust control for quadrotors with multiple time-varying uncertainties and delays. *IEEE Trans. Ind. Electron.* 2016, 64, 1303–1312. [CrossRef]

12. Chovancova, A.; Fico, T.; Hubinsky, P.; Duchon, F. Comparison of various quaternion-based control methods applied to quadrotor with disturbance observer and position estimator. *Robot. Auton. Syst.* 2016, 79, 87–98. [CrossRef]

13. Zhang, Y.; Chen, Z.; Zhang, X.; Sun, Q.; Sun, M. A novel control scheme for quadrotor UAV based upon active disturbance rejection control. *Aerosp. Sci. Technol.* 2018, 79, 601–609. [CrossRef]

14. Xu, L.X.; Ma, H.J.; Guo, D.; Xie, A.H.; Song, D.L. Backstepping sliding-mode and cascade active disturbance rejection control for a quadrotor UAV. *IEEE/ASME Trans. Mechatronics* 2020, 25, 2743–2753. [CrossRef]

15. Yang, H.; Cheng, L.; Xia, Y.; Yuan, Y. Active Disturbance Rejection Attitude Control for a Dual Closed-Loop Quadrotor Under Gust Wind. *IEEE Trans. Control Syst. Technol.* 2018, 26, 1400–1405. [CrossRef]

16. Besnard, L.; Shtessel, Y.; Landrum, B. Quadrotor vehicle control via sliding mode controller driven by sliding mode disturbance observer. *J. Frankl. Inst.* 2012, 349, 658–684. [CrossRef]

17. Shen, F.; Lei, W.; Zhang, K.; Tao, G.; Jiang, B. A novel nonlinear resilient control for a quadrotor UAV via backstepping control and nonlinear disturbance observer. *Nonlinear Dyn.* 2016, 85, 1281–1295. [CrossRef]

18. McKinnon, C.D.; Schoellig, A.P. Unscented external force and torque estimation for quadrotors. In Proceedings of the 2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Daejeon, Korea, 9–14 October 2016; pp. 5651–5657.

19. Smolyanskiy, N.; Kamesev, A.; Smith, J.; Birchfield, S. Toward low-flying autonomous MAV trail navigation using deep neural networks for environmental awareness. In Proceedings of the 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Vancouver, BC, Canada, 24–28 September 2017; pp. 4241–4247.

20. Greenwood, C.; Richards, A. Reinforcement learning and model predictive control for robust embedded quadrotor guidance and control. *Aeron. Robot.* 2019, 1–13. [CrossRef]

21. Hwangbo, J.; Sa, I.; Siegwart, R.; Hutter, M. Control of a quadrotor with reinforcement learning. *IEEE Robot. Autom. Lett.* 2017, 2, 2096–2103. [CrossRef]

22. Pi, C.H.; Hu, K.C.; Cheng, S.; Wu, I.C. Low-level autonomous control and tracking of quadrotor using reinforcement learning. *Control Eng. Pract.* 2020, 95, 104222. [CrossRef]

23. Wang, Y.; Sun, J.; He, H.; Sun, C. Deterministic Policy Gradient With Integral Compensator for Robust Quadrotor Control. *IEEE Trans. Syst. Man Cybern. Syst.* 2020, 50, 3713–3725. [CrossRef]

24. Tomić, T.; Ott, C.; Haddadin, S. External wrench estimation, collision detection, and reflex reaction for flying robots. *IEEE Trans. Robot.* 2017, 33, 1467–1482. [CrossRef]

25. Sutton, R.S.; McAllester, D.A.; Singh, S.P.; Mansour, Y. Policy gradient methods for reinforcement learning with function approximation. In *Advances in Neural Information Processing Systems*; ACM: New York, NY, USA, 2000; pp. 1057–1063.

26. Degris, T.; White, M.; Sutton, R.S. Off-Policy Actor-Critic. arXiv 2012, arXiv:1205.4839.

27. Schulman, J.; Levine, S.; Abbeel, P.; Jordan, M.; Moritz, P. Trust region policy optimization. In Proceedings of the International Conference on Machine Learning, Lille, France, 6–11 July 2015; pp. 1889–1897.

28. Sutton, R.S.; Barto, A.G. Reinforcement Learning: An Introduction. *IEEE Trans. Neural Netw.* 1988, 16, 285–286. [CrossRef]

29. Schulman, J.; Wolski, F.; Dhariwal, P.; Radford, A.; Klimov, O. Proximal policy optimization algorithms. arXiv 2017, arXiv:1707.06347.