Tiltrotors Position Tracking Controller Design Using Deep Reinforcement Learning

Yujia Huo$^{1,2,3,*}$, Yiping Li$^{2,3}$ and Xisheng Feng$^{2,3}$

$^1$State Key Laboratory of Robotics, Shenyang Institute of Automation, Chinese Academy of Sciences, Shenyang, 110016, China
$^2$Institutes for Robotics and Intelligent Manufacturing, Chinese Academy of Sciences, Shenyang 110169, China
$^3$University of Chinese Academy of Sciences, Beijing, 100049, China

*E-mail: huoyujia@sia.cn

Abstract. In this paper, a quad-tiltrotors air-water trans-domain robot is introduced. The nonlinear dynamic behaviours with uncertainties require a robust controller for multi-tasks. For this robot, controllers are designed using deep reinforcement learning method solving position and attitude control when operating as a UAV in the air. A ROS combining Gazebo simulation platform is designed to train the robot. The simulation results show the tiltrotors robot gets capabilities of spots tracking as a quad-rotors, and trajectory tracking as both the quad-rotors and tiltrotors.

Keywords – deep reinforcement learning, tiltrotors robot, trans-domain robot, position tracking

1. INTRODUCTION

Unmanned vehicles have rapidly extended their fields of application for decades, including unmanned aerial vehicles(UAVs) and unmanned underwater vehicles(UUVs). These two kinds of unmanned vehicles each advances differently. UAVs in the air have a fast speed of motivation when compared with UUVs operated underwater but have weak stealthiness of both observable performance and acoustic performance because of the high rate of rotors rotation. To improve environmental adaptabilities of UAVs and UUVs to expand their application scenarios and to merge the benefits, scientists and engineers on each side have focused on developing aquatic–aerial amphibious vehicles with a trans-domain capability operating in the air and underwater[1]. The trans-domain conception will significantly increase the range of sensing, communication performance, and mission scenarios available to the vehicles.

These trans-domain unmanned vehicles or trans-domain robots should at least need the capabilities of flight in the air as fixed-wing airplanes or multicopters, and the skills of diving underwater like UUVs do. Also, some trans-domain robots have the capability of surfing on the water surface as normal operation mode or just as a transition mode when taking off or landing. Moreover, the robots’ cross-domain transition between air and water should be repeatedly and bidirectional. Thus, seaplanes and submarine-launched UAVs aren’t considered to be with the trans-domain conception.

Due to the uncertainties of environmental conditions from ocean current and wind, as well as the uncertainties of robots’ model with nonlinearity, suitable motion controllers for tasks of attitude steady
control or position tracking are required for autonomous operation. Some institutions have achievements in autonomous control in the two kinds of fluid media. Maria et al. presented an octocopter able to operate in the air and underwater with a PD controller [2]. W. Weisler et al. demonstrated a fixed-wing cross-domain robot operating in both domains [3].

Unlike the trans-domain robots mentioned, in this paper, we present a tiltrotors trans-domain robot with four ducted propeller thrusts and wings foldable into the vehicle body. This robot has three operation modes, the underwater mode, the quadrotors mode with constant tilt thrust angles, and the tiltrotors mode with variable tilt thrust angles and wings providing the lift force in the air. The classical PID method was used for a tiltrotor UAVs during the VTOL and transition from vertical flight to horizontal flight [4]. MPC controller was designed with an unmanned quad-tiltrotors [5].

For our trans-domain robot with tiltrotor shape and more complex kinetics, classical methods hard to get excellent performance, neither functional or costly with time. Within the growth of artificial intelligence, reinforcement learning (RL), particular deep reinforcement learning (DRL) when combined with deep neural networks, has achieved several exciting works such as playing games, and this type of method interacting with the environment and learning from experience also performs well in robotic fields. Hwangbo from Ethz used reinforcement learning to control a quadrotors’ position and attitude steady from a hand throw [6]. In the underwater area, deep reinforcement learning has been applied for depth control [7] and position tracking [8].

In this paper, the trans-domain quad-tiltrotors is only considered simulated in the air condition operating in quadrotors mode and quad-tiltrotor mode. In Section 2, the kinetic model will be presented. DRL method controller explanation will be given in Section 3. In Section 4, simulation of both two flight modes and the transitions between modes are shown, and consequently, some conclusions will be presented in Section 5.

2. KINETICS OF TRANS-DOMAIN QUAD-TILTROTORS ROBOT

The reinforcement learning method learns from interacting with the environment repeatedly and gets an optimal control policy. However, great times of training the robot physically in the real world cost a lot. It is necessary to simulate the train tasks before deployment. In this section, the kinetics model of the trans-domain quad-tiltrotors robot is presented, and this model will be used for designing the controller and RL simulation training.

Trans-domain quad-tiltrotors robot has four ducted propeller thrusts (Figure 1) with variable angels $\delta_i=1,2,3,4$ and $\delta_i \in [0, \pi/2]$. When the angles of propeller thrusts are 0, the robot behaves like a simple quad-rotors UAV, and the propeller thrusts provide the lift (Figure 2).

![Figure 1 The Trans-domain Quad-Tiltrotors robot](image1)

![Figure 2 Tiltrotor propeller angle](image2)

In Figure 1, $G$ is the center mass of the robot, $x$ points to the bow, $y$ points to the starboard, and $z$ is determined by the right-hand rule. The body coordinate is fixed relative to the fuselage and rotates as the aircraft rotates. The attitude angles $\psi, \theta, \phi$ represent yaw, pitch, and roll, respectively [5]. The robot mainly effects from the force of four rotors $T_i=1,2,3,4$, gravity $mg$, wing force $F_W$. The wing...
force $F_w$ can be decoupled into three orthogonal forces, the lift of wings $F_{lift}$, side force of wings $F_{sideforce}$ and air drag of wings $F_{drag}$. And in the same way, moments of the robot are gotten.

2.1 Force and moment provided by the tilt propeller thrusts

The force provided by thrusts in the Body-Fixed coordinates Frame can be explained in Equation (1) [9], [10], where $T_i$ presents the thrust force of the rotor $i$, and $T_i = C_i \rho \omega_i^2$ with the rotor thrust coefficient $C_i$. For the moments $L_{x,f}$, $L_{y,f}$ and $L_{z,f}$ show the rotor position vector component of the rotor in front of the robot along with the axes $x$, $y$, $z$ respectively. And $L_{x,b}$, $L_{y,b}$, $L_{z,b}$ in the same way for the rotors behind the robot. Moreover, the rotating propellers with gyroscopic effects can be described as $M_{G,i} = C_i \rho \omega_i^2 \Omega$.

\[
\begin{align*}
F_{T,x} &= T_i \cos \delta_1 + T_2 \cos \delta_2 + T_3 \cos \delta_3 + T_4 \cos \delta_4 \\
F_{T,y} &= T_i \sin \delta_1 + T_2 \sin \delta_2 + T_3 \sin \delta_3 + T_4 \sin \delta_4 \\
M_{T,x} &= T_i L_{x,f} + T_2 L_{x,b} + T_3 L_{x,b} + T_4 L_{x,b} + \left[ \sin \delta_1 M_{q,1} - \sin \delta_2 M_{q,2} + \sin \delta_3 M_{q,3} - \sin \delta_4 M_{q,4} \right] \\
M_{T,y} &= T_i L_{y,f} + T_2 L_{y,b} + T_3 L_{y,b} + T_4 L_{y,b} + \left[ \cos \delta_1 M_{q,1} - \cos \delta_2 M_{q,2} + \cos \delta_3 M_{q,3} - \cos \delta_4 M_{q,4} \right] \\
M_{T,z} &= T_i L_{z,f} + T_2 L_{z,b} + T_3 L_{z,b} + T_4 L_{z,b} + \left[ \sin \delta_1 M_{q,1} - \sin \delta_2 M_{q,2} + \sin \delta_3 M_{q,3} - \sin \delta_4 M_{q,4} \right]
\end{align*}
\]  

2.2 Force and moment provided by the fixed-wings

The force provided by the fixed-wings can be decoupled into three orthogonal forces, the lift of wings $F_{lift}$, side force of wings $F_{sideforce}$ and air drag of wings $F_{drag}$. The force and moment effect of the fixed wings can be rewritten as:

\[
F_w = \begin{bmatrix}
F_{w,x} \\
F_{w,y} \\
F_{w,z}
\end{bmatrix} = \begin{bmatrix}
C_D Q S \\
C_Y Q S \\
C_L Q S
\end{bmatrix},
M_w = \begin{bmatrix}
M_{w,x} \\
M_{w,y} \\
M_{w,z}
\end{bmatrix} = \begin{bmatrix}
C_{Q Sl} \\
C_{m Sl} \\
C_{n Sl}
\end{bmatrix}
\]  

Dynamic pressure $Q = \rho V^2 / 2$, where $\rho$ presents the air density, $V = \sqrt{v_x^2 + v_y^2}$ represents the vector sum of $x$ axis and $y$ axis velocity. In addition, let $S$ be the area of the wing, $C_D$, $C_Y$, $C_L$ are the drag coefficient, side force coefficient and the lift coefficient, and $C_i$, $C_m$, $C_n$ means the moment coefficient respectively [11].

2.3 Force and moment effect on the robot

Thus, in the World Frame, the force and moment effect on robot can be rewritten as follows:

\[
\begin{align*}
F_x &= F_{w,x} + F_i \sin \theta & M_x &= (M_i + M_u) \cos \theta \\
F_y &= F_{w,y} + F_i \cos \theta \sin \phi & M_y &= (M_i + M_u) \sin \theta \sin \phi \\
F_z &= F_{w,z} - F_i \cos \theta \cos \phi + mg & M_z &= (M_i + M_u) \sin \theta \cos \phi
\end{align*}
\]

where $m$ is the mass of the robot and $g = 9.8 m/s^2$.

3. ROBOT POSITION TRACKING CONTROLLER DESIGN

The trans-domain quad-tiltrotors robot has multiple operation conditions and needs controller suitable for these operation conditions with different media and operation modes. But it is hard to design only one controller to fix the whole control problem. So, an acceptable way is to create departed controllers
for each condition. And in this way, the learned control policy may get void of being ruined when other condition’s control policies are under training.

For our recent robot, three sub-controllers are designed for three modes, the quad-rotors mode, the quad-tiltrotors mode with fixed wings and the underwater mode(Figure 3). The others mean it could have more operation mode in the future, for example, now the robot takes off/lands as a helicopter-like quad-rotors, but may have the capability of surfing taking-off on the surface.

3.1 Deep reinforcement learning
To solve the uncertainties of the robotics kinetics modeling and the environmental disturbs, DRL method controllers are designed. The reinforcement learning method can learn autonomously by exploring the environment space and evaluating the action policy that has been made. The processes of making action policies can be described as a Markov Decision Process(MDP), within a tuple: \( \{S, A, P, R\} \), where \( S \) is the state space, \( A \) the action space, \( P \) the state transition probability, and \( R \) the reward. Therefore, the goal of MDP is to find an action policy to maximize the long-term cumulative cost value[12]. The policy gradient is an approach to solve reinforcement learning problems. It maximizes the expected total reward by repeatedly estimating the gradient

\[
g := \nabla_{\theta} E \left[ \sum_{t=0}^{\infty} r_t \right].
\]

Using the advantage function \( A(s_t, a_t) \) as the rewards, the gradient can be described in Equation (4). With the state-value \( V^\pi(s_t) \) and the action-value \( Q^\pi(s_t, a_t) \), the advantage function

\[
A(s_t, a_t) = Q^\pi(s_t, a_t) - V^\pi(s_t).
\]

3.2 Proximal policy optimization reinforcement learning(PPO)
The PPO[13] method is sort of an improved algorithm from the TRPO(Trust Region Policy Optimization)[14]. When considering TRPO, the problem can be described as follows:

\[
\text{maximize } \frac{1}{N} \sum_{t=1}^{N} \pi(a_t | s_t) A_{\pi_{old}}(s_t, a_t)
\]

And the problem improved by PPO with Adaptive KL Penalty can be described as follows:

\[
\text{maximize } \frac{1}{N} \sum_{t=1}^{N} \pi(a_t | s_t) A_{\pi_{old}}(s_t, a_t) - \beta KL[\pi_{\theta_{new}}(\cdot|s_t), \pi_{\theta}(\cdot|s_t)]
\]

where \( \beta \) controls the weight of the penalty. It penalizes the objective if the new policy is different from the old policy. we can dynamically adjust \( \beta \) with \( d \) and a threshold \( d_{targ} \), and \( d \) can be described as:

\[
d = \frac{1}{N} \sum_{t=1}^{N} [KL(\pi_{\theta_{old}}(\cdot|s_t), \pi_{\theta}(\cdot|s_t))]
\]
Algorithm1

for $i \in \{1, \ldots, N\}$ do

run policy $\pi_\theta$ for $T$ timesteps, get feedback $\{s_i, a_i, r_i\}$ and $A_i = \sum_{t'} y^{t'-t}r_i - V(s_t)$

$\pi_{old} \leftarrow \pi_\theta$

for $j \in \{1, \ldots, M\}$ do

$J(\theta) = \sum_{t=1}^{T} \frac{\pi_\theta(s_t | s_{t-1}) A_i}{\pi_{old}(a_t | s_t)} - KL[\pi_{old} | \pi_\theta] \text{ and update } \theta \text{ by gradient method with } J(\theta)$

end for

for $j \in \{1, \ldots, B\}$ do

$L(\phi) = -\sum_{t=1}^{T} \sum_{i} y^{t'-t}r_i - V(s_t))^2 \text{ and update } \phi \text{ by gradient method with } L(\phi)$

end for

if $d < d_{\text{arg}}$ then $\delta, \beta \leftarrow \beta / \alpha$

end for

if $d > d_{\text{arg}} \times \delta, \beta \leftarrow \beta \times \alpha$

end for

3.3 Reinforcement learning for point tracking control

For the point tracking tasks, the desired position of the robot can be described as a three-dimensional vector $p_{\text{target}}$, where $p_{\text{target}} = [x_{\text{target}}, y_{\text{target}}, z_{\text{target}}]^T$. The point-tracking problem is to make the robot move to a desired point and remain with a steady state of position, attitude, linear speed and angular speed. As only the quad-rotors has the hover capability with zero linear velocities, the point tracking tasks are only for quad-rotors operation mode. The actions are the rotation rate of four propeller thrusts. A quadratic reward function at each step is designed:

$$r(t) = -c_1 \| p_t - p_{\text{target}} \|^2 - c_2 \| \Omega \|^2 - c_3 \| V_t \|^2 - c_4 \| a_{\text{target}} \|^2 - c_5 \| a_t \|^2$$

where, $p_t$ present the robot’s current position vector $[x_t, y_t, z_t]^T$, and L2 norm means as follows:

$$\| p_t - p_{\text{target}} \|^2 = (x_t - x_{\text{target}})^2 + (y_t - y_{\text{target}})^2 + (z_t - z_{\text{target}})^2$$

So in the same way with the L2 norm and the vector, $\Omega$ means the current attitude angle vector $[\psi_t, \theta_t, \psi_{\text{heading}}]^T$, $V_t$ the linear velocity vector $[u_t, v_t, w_t]^T$, $\omega_t$ the angular velocity $[p_t, q_t, r_t]^T$, and the $a_t$ means the current action of thrust $i$ at the time of $t$. $c_1, c_2, c_3, c_4, c_5$ weights for each vector. The goal is of the RL algorithm is to find an optimal action policy, that under this policy, the robot can move to a desired point autonomously and stay steady at that position.

3.4 Reinforcement learning for quad-rotors trajectory tracking control

For the quad-rotors trajectory tracking tasks, $l = [[x_1, y_1, z_1]^T, [x_2, y_2, z_2]^T, \ldots, [x_l, y_l, z_l]^T, \ldots]$ , the actions are the rotation rate of four propeller thrusts. The reward function is designed as follows:

$$r(t) = -c_1 \| \psi_{\text{heading}} - \psi_{\text{target}} \|^2 - c_2 \| \psi_t \|^2 - c_3 \| V_t \|^2 - c_4 \| z_t - z_{\text{target}} \|^2 - c_5 \| a_{\text{target}} \|^2 - c_6 \| a_t \|^2$$

In this task, the velocity of the robot is not considered, and $\psi_{\text{heading}}$ is the attitude $\psi$, yaw of the robot. $\psi_t$ is the angle of the velocity in the horizontal plane:

$$\psi_t = \arctan \frac{V_t}{u_t}$$
And proposing
\[ \psi_i^V = \arctan \frac{y_{i+1} - y_i}{x_{i+1} - x_i} \] (13)

In the reward function, \( z_i \) is the hover-height/depth of the robot. While \( \theta_i, \phi_i \) are the pitch and roll of the robot, respectively.

3.5 Reinforcement learning for quad-tiltrotors trajectory tracking control

For the quad-tiltrotors trajectory tracking task, the propeller angles vary from 0 to \( \pi / 2 \), and the action is the tuple. And the reward function is designed as follows:
\[ r(t) = -c_1 \| \psi_i - \psi_{i, \text{target}} \|^2 - c_2 \| z_i - z_{i, \text{target}} \|^2 - c_3 \| \theta_i \|^2 - c_4 \| \phi_i \|^2 - c_5 \| \psi_i \|^2 - c_6 \| \omega_i \|^2 - c_7 \| a_i \|^2 \] (14)

Where, \( a_i \) is the element of the tuple of actions. When comparing the reward functions (11) and (14), we suppose that \( \psi_i^{\text{heading}} \approx \psi_i^V \) with a little drift angle.

4. ROBOT POSITION TRACKING SIMULATION

It is often costly when the robot interacts with the environment repeatedly using the reinforcement learning method. Thus, simulations are necessary before physical deploying the controller to the robot. In this section, ROS, combined with Gazebo, is designed to implement a platform for controllers’ training of the trans-domain quad-tiltrotors with the method of deep reinforcement learning, and the kinetics parameters are listed in Table 1. And the simulation of reinforcement learning is code in Python2.7 with ROS and Python3.5 with Tensorflow1.14 on Ubuntu 16.04 system.

| parameter | unit | value | parameter | unit | value |
|-----------|------|-------|-----------|------|-------|
| \( m \)   | kg   | 33    | \( L_{c,f} \) | m    | 0.063 |
| \( L_{x,f} \) | m   | 0.717 | \( L_{x,b} \) | m   | 0.109 |
| \( L_{y,f} \) | m   | 0.472 | \( L_{y,b} \) | m   | 0.265 |
| \( L_{z,b} \) | m   | 0.474 | \( \rho \)    | kg \cdot m^{-3} | 1.293 |

4.1 Quad-rotors position control simulation

In this part, the robot is operated as a quad-rotors with fixed propeller angles. Usually, the fixed propeller angles are set as zeros. Thus, the robot’s action simplified to the rotation rate tuple-four thrusts. The quad-rotors control tasks can be mainly separate as one position tracking and trajectory tracking. While the robot tracks the goals, attitudes are also under control. For the one-point tracking task, the robot takes off from \([0,0,0]\) and achieves the desired point, we set the desired goal at \([1,1,1]\).

Figure 4-8 show the point tracking task, the robot arrives at the goal \([1,1,1]\) in 2 seconds and remains. Meanwhile, the robot gets steady linear velocities, angular velocities and attitudes. For trajectory tracking task, as mentioned in Sec. 3.4, the continuous trajectory is described as several waypoints. In the horizontal plane, the line of sight method is adopted to design the reward function. However, in the vertical plane, due to the quad-rotors dynamics model, the attack angle always exists. The control in the vertical plane uses the height error rather than the angle error like in the horizontal plane.

We design a trajectory described as lemniscate of Huygens in Equation (15), the robot takes off and begins to track this trajectory.
\[ \begin{align*}
  x &= 5 \cos(\varphi) \\
  y &= 5 \sin(2\varphi) \\
  z &= 15 - 5 \cos(\varphi)
\end{align*} \] (15)
Figuer 4-13 shows the task that the robot is operated as a quad-rotors and tracks a trajectory. The robot takes off from land at $[0,0,0]$, and begins to track the trajectory at the same time. In Figure 10, the robot slows down to avoid falling due to tough attitude angles when the robot turns rapidly.

4.2 Tiltrotors position control simulation
In this part, the robot is operated as a quad-tiltrotors with variable propeller angles. Some traditional tiltrotors, attitude control is mainly provided by the elevators and rudders like fixed-wing planes.
Unlike these tiltrotors, our tiltrotors controls the attitude by changing the propeller angles and rotation rates at the same time. The control action of the tiltrotors is a tuple $[a_1, a_2, a_3, a_4, \delta_1, \delta_2, \delta_3, \delta_4]$. 

Figure 14 Tiltrotors trajectory tracking  
Figure 15 Tiltrotors linear velocity  

For this task, we design a similar trajectory lemniscate of Huygens in Equation (16).

$$\begin{align*}
x &= 15 \cos(\rho) \\
y &= 10 \sin(2\rho) \\
z &= 15 - \cos(\rho) / 10
\end{align*}$$  

Figure 14-19 depict the response of the robot in the tiltrotors mode operation. The robot takes off from $[0,0,0]$, and begins to track the trajectory by controlling the variable propeller angles and the thrust rotation rates. The robot gets higher speed and lowers attitude angle when compared with the quad-rotors mode. As the fixed-wing provides lift force, the thrusts perform efficiently with lower rotation rates. Due to the roll angle gets tough when turning, less lift force in the vertical plane is not enough. To avoid falling down, propeller angles are controlled to provide the differential angle and consequently provides force compensating the lost lift force in Figure 19. 

Figure 16 Tiltrotors angular velocity  
Figure 17 Tiltrotors attitude  
Figure 18 Tiltrotors thrust rotation rate  
Figure 19 Tiltrotors propeller angle  

5. CONCLUSION

The nonlinear model dynamics of a quad–tiltrotors UAV type were extracted, and a ROS combined with Gazebo simulation platform of the multi-mode flight envelope was developed. A control algorithm has been learned using reinforcement learning and fitted value iteration using the nonlinear dynamic model of the quad-tiltrotors. The effectiveness of the proposed controller in tackling the complexity of the conversion task, were demonstrated via simulation studies.
References

[1] X. Yang, T. Wang, J. Liang, G. Yao, and M. Liu, “Survey on the novel hybrid aquatic–aerial amphibious aircraft: Aquatic unmanned aerial vehicle (AquaUAV),” Prog. Aerosp. Sci., vol. 74, pp. 131–151, Apr. 2015.

[2] M. M. Maia, P. Soni, and F. J. Diez-Garias, “Demonstration of an Aerial and Submersible Vehicle Capable of Flight and Underwater Navigation with Seamless Air-Water Transition,” p. 9.

[3] W. Weisler, W. Stewart, M. B. Anderson, K. J. Peters, A. Gopalarathnam, and M. Bryant, “Testing and Characterization of a Fixed Wing Cross-Domain Unmanned Vehicle Operating in Aerial and Underwater Environments,” IEEE J. Ocean. Eng., vol. 43, no. 4, pp. 969–982, Oct. 2018.

[4] Z. Kong and Q. Lu, “Mathematical Modeling and Modal Switching Control of a Novel Tiltrotor UAV,” J. Robot., vol. 2018, pp. 1–12, Aug. 2018.

[5] C. Papachristos, K. Alexis, and A. Tzes, “Hybrid model predictive flight mode conversion control of unmanned Quad-TiltRotors,” in 2013 European Control Conference (ECC), Zurich, 2013, pp. 1793–1798.

[6] J. Hwangbo, I. Sa, R. Siegwart, and M. Hutter, “Control of a Quadrotor with Reinforcement Learning,” IEEE Robot. Autom. Lett., vol. 2, no. 4, pp. 2096–2103, Oct. 2017.

[7] H. Wu, S. Song, K. You, and C. Wu, “Depth Control of Model-Free AUVs via Reinforcement Learning,” IEEE Trans. Syst. Man Cybern. Syst., pp. 1–12, 2018.

[8] I. Carlucho, M. D. Paula, S. Wang, B. V. Menna, Y. R. Petillot, and G. G. Acosta, “AUV Position Tracking Control Using End-to-End Deep Reinforcement Learning,” in OCEANS 2018 MTS/IEEE Charleston, 2018, pp. 1–8.

[9] M. Miller and J. Narkiewicz, “TILTROTOR MODELLING FOR SIMULATION IN VARIOUS FLIGHT CONDITIONS,” p. 26.

[10] K. M. Kleinhesselein, “STABILITY AND CONTROL MODELING OF TILTROTOR AIRCRAFT,” p. 255.

[11] J. D. Anderson, “Fundamentals of Aerodynamics,” p. 1131, 2009.

[12] R. S. Sutton and A. G. Barto, Reinforcement Learning: An Introduction. 1988.

[13] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, “Proximal Policy Optimization Algorithms,” ArXiv170706347 Cs, Jul. 2017.

[14] J. Schulman, S. Levine, P. Moritz, M. I. Jordan, and P. Abbeel, “Trust Region Policy Optimization,” ArXiv150205477 Cs, Feb. 2015.