Hindawi
Mathematical Problems in Engineering
Volume 2019, Article ID 9602460, 11 pages
https://doi.org/10.1155/2019/9602460

Research Article

A Novel Motion-Intelligence-Based Control Algorithm for Object Tracking by Controlling PAN-Tilt Automatically

Jianbing Yang, Zhiyong Tang, Zhongcai Pei, and Xiao Song

School of Automation Science and Electrical Engineering, Beihang University (BUAA), China

Correspondence should be addressed to Zhongcai Pei; peizc@buaa.edu.cn

Received 13 November 2018; Revised 16 March 2019; Accepted 1 April 2019; Published 17 April 2019

Academic Editor: Gustavo Scaglia

As computer vision develops, pan-tilt platform visual systems are able to track moving target over static camera systems. In this paper, a novel motion-intelligence-based control algorithm for object tracking by controlling pan-tilt platform has been proposed. The algorithm includes the motion control model based on angular speed and the intelligent control algorithm based on reinforcement learning (RL). The motion control model converts deviation between the center point of the tracked target and the center point of the frame to angular speed of pan-tilt platform. It can keep position of the tracked object in the center of the frame automatically. The intelligent control algorithm based on reinforcement learning can reduce the error between the ideal value and the actual value when the pan-tilt platform moves. The two blocks work together to make the pan-tilt platform track a dynamic object more stably and the experiment result shows that both the tracking accuracy and robustness are improved.

1. Introduction

In recent years, visual object tracking has been used in many fields such as video surveillance [1], unmanned assistance systems [2], and autonomous robotics [3]. Visual object tracking with pan-tilt platform can be employed as active surveillance, trajectory measured, and so on. However, how to make the pan-tilt platform control camera move smoothly and accurately to tracking object is a challenge. Reference [4] proposed a PID control algorithm based on the displacement between the tracked object in image and the center of image to control the pan-tilt platform. Reference [5] proposed a control algorithm with two ADRC controllers based rotation angle of the horizontal and vertical direction. Reference [6] proposed a control methods, namely, lead-lag compensator and PID control, which using black box model. Although the control algorithms mentioned above can control the pan-tilt platform to track the target, there will still be overshooting, lag, and vibration.

In fact, the camera installed in the pan-tilt platform follows the target, which makes the azimuth axis and pitching axis of the pan-tilt platform follow the target's angular speed when the target rotates around the pan-tilt platform. Therefore, we proposed a motion control model based on angular speed for pan-tilt platform. In order to eliminate the overshooting, lag, and vibration, we still need a control algorithm for the motion control model.

Since the advent of Alpha Go, reinforcement learning has gradually become the focus and attracted more and more scholars to study it. In robotic applications, [7] presents an implementation of RL enabling the learning process for multiple robotic tasks with minimal per-task tuning. Reference [8] makes robotic vehicles capable of Socially Aware Motion Planning with Deep Reinforcement Learning. Reference [9] makes mobile robot navigation with inverse reinforcement learning. Reference [10] uses Reinforcement Learning to design UAV controller. The pan-tilt platform motion can be formulated as a sequential decision process based on Markov stochastic model and reinforcement learning (RL) is a class of machine learning methods for solving sequential decision-making problems with unknown state-transition dynamics [11]. As such, we proposed an intelligent control algorithm based on reinforcement learning.

The outline of this paper is organized as follows. Section 2 discusses the composition of visual tracking system. Section 3 presents the design details of motion control model based

Figure 1: Left: visual tracking system structure; middle-left: visual pan-tilt platform; middle-right: axis control system; right: graphics processing computer.

on angular speed. Section 4 presents the intelligent control algorithm based on reinforcement learning. Section 5 draws the conclusion.

2. Composition of Visual Tracking System

The visual tracking system consists of visual pan-tilt platform, axis control system, and graphics processing computer, which is shown in Figure 1. The visual pan-tilt platform consists of the azimuth axis, the pitching axis, the tracking camera, and the high-speed recording camera (in this case, the visual pan-tilt platform is used to track parachute and we need to record the parachute posture; thus there is a high-speed recording camera in the pan-tilt platform). The axis control system consists of two motor drivers and a programmable Multiaxis Controller. Tracking camera is used to capture the target and send frame to graphics processing computer. The graphics processing computer processes the frame to generate the control instructions and the send them to axis control system and then the axis control system controls the azimuth axis and the pitching axis to yaw and pitch, respectively. The target will be kept at the center of the frame with the yawing and pitching of the pan-tilt platform in order to achieve real-time tracking.

Hardware configuration:

(1) Tracking camera: PointGray (GS3-U3-41C6C-C), resolution: 2048 × 2048, Pixel Size: 5.5μm

(2) Lens: NIKON (Focal length = 50 mm)

(3) Motor: Kollmorgen (KBMS-17H01-A00 and KBMS-25H01-A00)

(4) Motor driver: Kollmorgen (AKD-P00606-NBEC-0000 and AKD-P01206-NBEC-0000)

(5) Programmable Multi-Axis Controller: Beckhoff (CX5130-0125)

(6) Graphics processing computer: CPU is INTEL 7700 K, 32 GB RAM, two GPU: Gigabyte GV-N1080Ti, Linux system.

3. Motion Control Model Based on Angular Speed for Pan-Tilt Platform

In this section, we will at first explain the relationship between deviation and rotation angle of the pan-tilt platform. Then we will present a mathematical motion control model based on angular speed for pan-tilt platform.

3.1. Relationship between Deviation and Rotation Angle. In this article, particle tracking algorithm [12–15] is used to track the moving target on the frame. The particle tracking algorithm is not the focus of this paper, so it will not be described in detail.

When the moving target is captured by the tracking camera, the tracking algorithm will generate the tracking box of the moving target in frame, as shown in Figure 2. If the moving target is not in the center of the frame, there will be two deviations (Δx and Δy) between the center point of the target and the center point of the frame in x-axis and y-axis, respectively. The Δx and Δy are the number of pixels. We already know the pixel size (see Section 2) of the tracking camera, so we can calculate the distance between the center of target and the center of the mos plane on the camera mos plane. The equations are shown in (1) and (2).

$$\Delta P_x = \Delta x \times s \tag{1}$$

$$\Delta P_y = \Delta y \times s \tag{2}$$

where ΔP_x and ΔP_y represent the distance deviations in x-axis and y-axis on the camera mos plane, separately. s represents the pixel size.

Through the image-forming principle, we can convert the Δx and Δy to the deviation angles θ_x and θ_y, shown in Figure 3. There is a point P in the space, and it is projected onto the mos plane of the camera as point P'. Through the geometric relationship, we can get the deviation angle θ_x between the line connecting point P' and focal point and the camera's central axis (the z-axis) in the x-axis direction. Meanwhile the deviation angle θ_y between the line connecting point P' and focal point and the camera's central

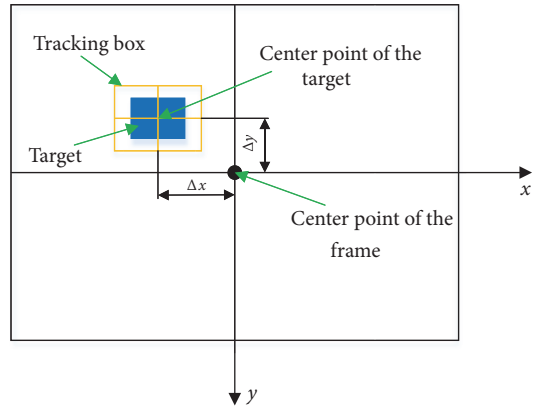


Figure 2: The deviations between the center point of the target and the center point of the frame (unit: pixel).

axis in the y-axis direction can also be obtained; the equations are shown as (3) and (4).

$$\theta_x = \arctan \frac{\Delta P_x}{f} \tag{3}$$

$$\theta_y = \arctan \frac{\Delta P_y}{f} \tag{4}$$

where f is the focal length. Due to the fact that we are using a fixed-focus lens, f is a known constant and $f = 50$ mm.

3.2. Mathematical Model of Motion Control. In order to drive the movement of the pan-tilt platform to keep position of the tracked object in the center of the frame automatically, the most commonly used control methods for pan-tilt platform are to calculate θ_x and θ_y, and then let the azimuth axis and the pitch axis of the pan-tilt platform rotate the angles θ_x and θ_y, respectively.

In fact, the camera installed in the pan-tilt platform follows the target, which makes the azimuth axis and pitching axis of the pan-tilt platform follow the target's angular speed when the target rotates around the pan-tilt platform. If we can calculate the angular speed of the target and make the angular speed of the pan-tilt platform follow the target, we can also keep position of the tracked object in the center of the frame. Therefore, we proposed a motion control model based on angular speed for pan-tilt platform.

For ease of analysis, we separate the control of pan-tilt platform into the control of the azimuth axis and the pitch axis, because the motion of the azimuth axis and the pitch axis is independent. We will introduce the mathematical model of motion control for azimuth axis; the mathematical model for pitch axis is the same.

The movement of the target relative to the pan-tilt platform is the movement of the target on the frame. We can get the previous angle θ_x^{t-1} and the current angle θ_x^t of the target on the frame, as shown in Figure 4. Then, the current angular speed ω_r^t of the relative motion can be obtained. The equation is shown as

$$\omega_r^t = \frac{\left(\theta_x^t - \theta_x^{t-1}\right)}{\Delta t} \tag{5}$$

where Δt is the cycle of the image processing algorithm. ω_r^t can also be expressed as

$$\omega_r^t = \omega_{tar}^t - \omega_{tur}^t \tag{6}$$

where ω_{tar}^t is the current angular speed of the target which rotates around azimuth axis and ω_{tur}^t is the current angular speed of azimuth axis. ω_{tur}^t can be obtained from motor driver of azimuth motor and it is a known constant.

According to (6), we can get the current angle speed of the target. The equation is shown as

$$\omega_{tar}^t = \omega_r^t + \omega_{tur}^t \tag{7}$$

Our goal is to calculate the next angle speed instruction ω_{tur}^{t+1} for the azimuth axis and make the tracked object in the center of the frame. Therefore, the ideal position of the target at the next moment is at the center of the image. In Figure 4, the next position is point O. When the target is at the point O, the angle θ_x^{t+1} between the line connecting point O and focal point and the camera's central axis in the x-axis direction is 0°. The next angular speed ω_r^{t+1} of the target relative to the azimuth axis is shown as

$$\omega_r^{t+1} = \frac{\left(\theta_x^{t+1} - \theta_x^t\right)}{\Delta t} = \frac{\left(0 - \theta_x^t\right)}{\Delta t} \tag{8}$$

ω_r^{t+1} can also be expressed as

$$\omega_r^{t+1} = \omega_{tar}^{t+1} - \omega_{tur}^{t+1} \tag{9}$$

So, we can get ω_{tur}^{t+1}; the equation is shown as

$$\omega_{tur}^{t+1} = \omega_{tar}^{t+1} - \omega_r^{t+1} \tag{10}$$

where ω_{tar}^{t+1} is the angle speed of target at the next moment. We cannot accurately predict ω_{tar}^{t+1}, but the Δt ($\Delta t = 30$ms) is

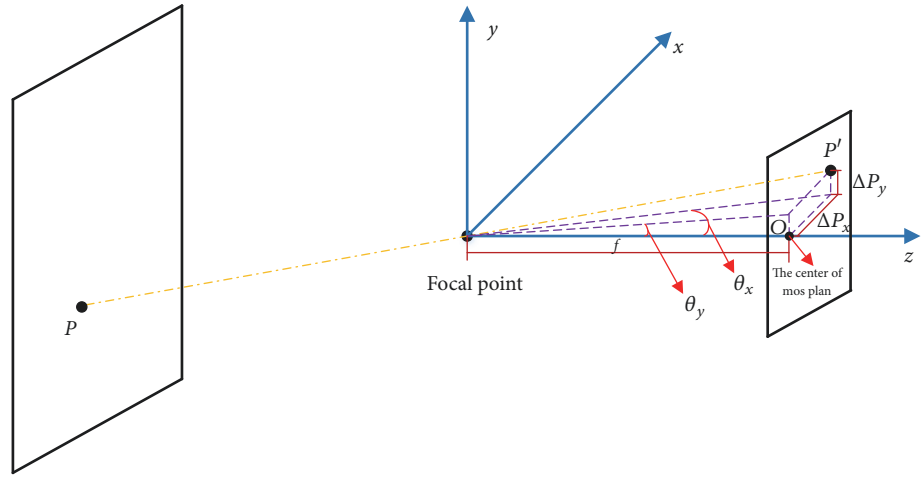


Figure 3: Image-forming principle.

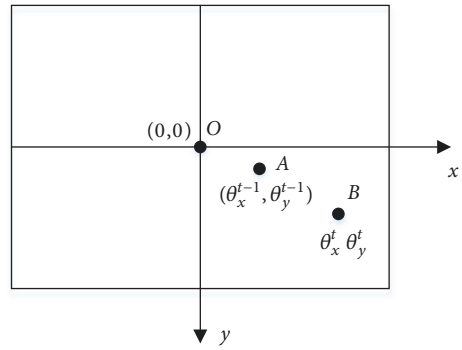


Figure 4: Angles θ_x and θ_y in angle plane for previous position A and current position B.

short enough; hence we consider $\omega_{tar}^{t+1} = \omega_{tar}^{t}$. So, (10) can be reformulated as

$$\omega_{tur}^{t+1} = \omega_{tar}^{t} - \omega_{r}^{t+1} \tag{11}$$

Finally, ω_{tur}^{t+1} can be expressed as

$$\omega_{tur}^{t+1} = \omega_{r}^{t} + \omega_{tur}^{t} - \omega_{r}^{t+1} = \frac{\left(\theta_x^t - \theta_x^{t-1}\right)}{\Delta t} + \omega_{tur}^{t} + \frac{\theta_x^t}{\Delta t} \tag{12}$$

Equation (12) is the motion control model based on angular speed which is proposed in this paper. The algorithm is shown as Algorithm 1.

In order to verify motion control model based on angular speed, we conducted related experiments. The experiment scenarios are shown in Table 1.

With Algorithm 1, the pan-tilt platform keeps the pedestrian's face at the center of the image in *Experiment 1*, shown as Figure 5. This proves that the motion control model based on angular speed we proposed is correct. However, from Figure 5, we find that the pan-tilt platform vibrates back and forth around the center of the frame.

In order to tackle this problem, we multiply (8) by a coefficient $\gamma(\gamma \epsilon (0, 1])$, shown as (13). The reason is that it can reduce the angular speed of the target relative to the pan-tilt platform. As we can see from Figure 5(a)), the speed of the pedestrian moving toward the center of the image will be reduced on the image. Finally, (12) can be expressed as (14).

$$\omega_{r}^{t+1} = \gamma \times \frac{\left(0 - \theta_x^t\right)}{\Delta t} \tag{13}$$

$$\omega_{tur}^{t+1} = \omega_{r}^{t} + \omega_{tur}^{t} + \frac{\theta_x^t}{\Delta t} = \frac{\left(\theta_x^t - \theta_x^{t-1}\right)}{\Delta t} + \omega_{tur}^{t} + \gamma \times \frac{\theta_x^t}{\Delta t} \tag{14}$$

We conducted the second experiment to verify whether the addition of coefficient γ can solve the vibration problem. In *Experiment 2*, we set $\gamma = 0.1$. The result of *Experiment 2* is shown as Figure 6. Comparing Figure 6 with Figure 5(b)), we find that after adding coefficient γ, the amplitude of $\boldsymbol{\omega_{tur}^{t+1}}$ is significantly reduced and is almost equal to 0. The vibration has been greatly reduced.

Experiment 2 also verified that the pan-tilt platform can effectively keep the stationary target at the center of the image with the motion control model based on angular speed which we proposed. Next, we will use the pan-tilt platform to track the moving object in *Experiment 3*.

In *Experiment 3*, we also set $\gamma = 0.1$. The result is shown as Figure 7. As the target begins to move, the pan-tilt platform has a significant lag in tracking the UAV and the response of the pan-tilt platform is not fast enough, causing the target to fly out of the camera's field of view. The reason is that the γ is

Input Capture the adjacent two frames from camera: $\boldsymbol{f}_{t-1}$, $\boldsymbol{f}_t$
Output ω_{tur}^{t+1}
(1) Initialization particle tracking algorithm with the target
(2) **While** true **do**
(3) Find the position of the target on frame $\boldsymbol{f}_{t-1}$, $\boldsymbol{f}_t$ with particle tracking algorithm
(4) Get θ_x^{t-1} and θ_x^t with equation (3)
(5) Get ω_r^t and ω_r^{t+1} with equation (5) and (8), respectively
(6) Get ω_{tur}^t from motor driver of azimuth motor
(7) Get ω_{tur}^{t+1} with equation (12)
(8) Send the instructions ω_{tur}^{t+1} to motor driver of azimuth motor

Algorithm 1: Algorithm of motion control model based on angular speed.

Table 1: Experiment scenarios.

Experiment	Description of experiment scenarios
1	Let the pedestrian stay in the camera's field of view and stay still; then select the pedestrian's face as the initialization of the particle tracking algorithm. The distance between pedestrian and pan-tilt platform is 100 m.
2	*Experiment 2* is the same as *Experiment 1*.
3	Let the UAV (Unmanned Aerial Vehicle) in the camera's field of view and stay still; then select the UAV as the initialization of the particle tracking algorithm and then make the UAV move at *1 m/s*. The distance between UAV and pan-tilt platform is 100 m.
4	*Experiment 4* is the same as *Experiment 3*.
5	Let the UAV (Unmanned Aerial Vehicle) in the camera's field of view and stay still, then select the UAV as the initialization of the particle tracking algorithm and then make the UAV move at *2 m/s*. The distance between UAV and pan-tilt platform is 100 m.

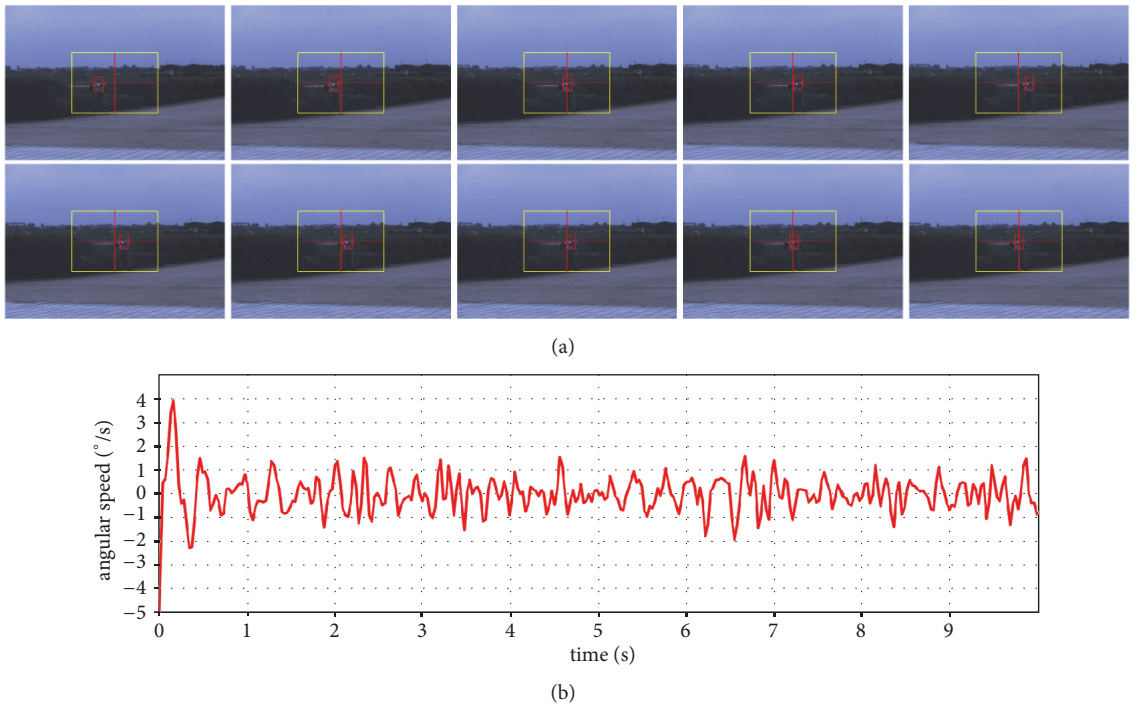


Figure 5: Pan-tilt platform tracking result. (a) The pictures of the tracking of the still target by pan-tilt platform. (b) The angle speed instruction ($\boldsymbol{\omega}_{tur}^{t+1}$) for the azimuth axis.

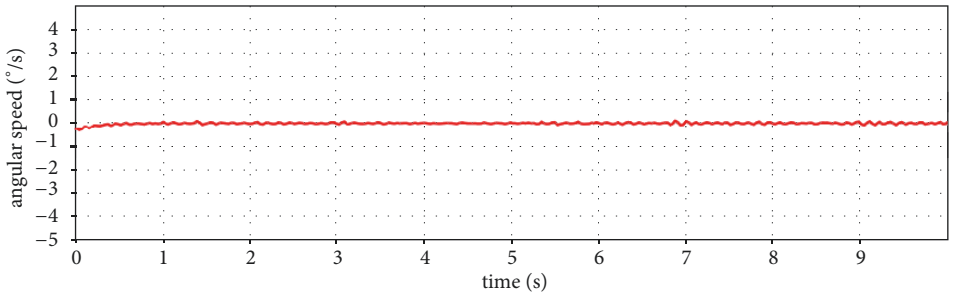


FIGURE 6: The angle speed instruction (ω_{tur}^{t+1}) for the azimuth axis.

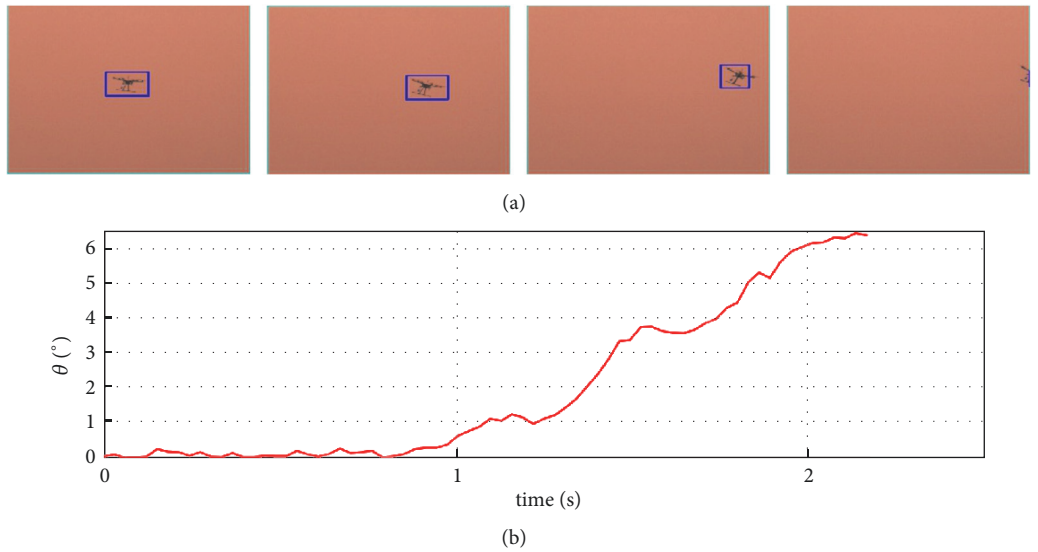


FIGURE 7: Pan-tilt platform tracking result. (a) The pictures of the tracking of the moving target by pan-tilt platform; (b) the deviation angle θ (see equation (3)).

too small, resulting in the angular speed of the UAV relative to the pan-tilt platform is too small, so that the target cannot be tracked in time.

In *Experiment 4*, we selected a series of γ ($\gamma = 0.1, 0.2, 0.3, 0.4, 0.5, 0.6$) to conduct experiments separately. The results of *Experiment 4* are shown in Figure 8. We have found that increasing γ has a significant improvement in hysteresis. When γ reached 0.4, we found that a good result was achieved. However, when γ exceeded 0.4, the pan-tilt platform still shakes.

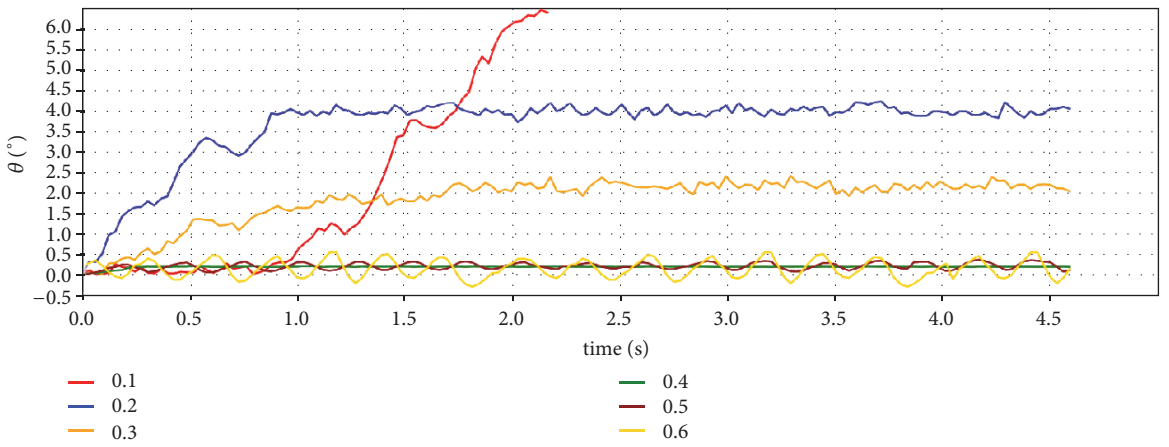
In *Experiment 5*, we increased the speed of the UAV and conduct experiment with the optimal $\gamma = 0.4$ obtained in *Experiment 4*. We find that there is still a hysteresis. Therefore, we increased γ and conducted experiments. The results of *Experiment 5* are shown as Figure 9. When γ reached 0.6, a good result was achieved. However, when γ exceeds 0.6, the pan-tilt platform still shakes.

Through the above experiments, we find that under different motion states of the target, we need different γ to keep the target at the center of the frame and make the pan-tilt platform move smoothly. Therefore, we propose an intelligent control algorithm based on reinforcement learning for γ.

4. Intelligent Control Algorithm Based on Reinforcement Learning

As time goes by, the moving target will be in different states and the pan-tilt platform needs optimal γ to keep the target at the center of the frame and move smoothly. The fact that the pan-tilt platform continuously obtains optimal γ can be formulated as a sequential decision process based on Markov stochastic model. Meanwhile, reinforcement learning (RL) is a class of machine learning methods for solving sequential decision-making problems with unknown state-transition dynamics [11]. Normally, Markov decision process (MDP) can be defined by a tuple $M = (S, a, P, R, \alpha)$, in which S is the state space, a is the action space, P is the state-transition model, R is the reward function, and α is a discount factor.

When pan-tilt platform stays in a certain state, it obtains the optimal γ by learning the process of trial through the reinforcement learning algorithm. However, it is infeasible to let the pan-tilt platform continuously explore online to learn an optimal γ because the pan-tilt platform will lose the target with a bad γ when the target is out of the camera's field of view. To tackle this problem, we apply data-driven reinforcement learning that the real-life data is collected by tracking object



(a)

(b)

FIGURE 8: Pan-tilt platform tracking result with different γ. (a) The deviation angle θ (see equation (3)). (b) The pictures of the tracking of the moving target by pan-tilt platform, from top to down; γ are 0.1, 0.2, 0.3, 0.4, 0.5, 0.6.

in different moving conditions with the pan-tilt platform and then is used to train the proposed Q-network in order to output optimal γ.

In this section, we will design variables of reinforcement learning and then design experiments and collect data according to the requirements of S and a. Finally, we show how to use the collected data to train the proposed Q-network. For the control of azimuth axis, the proposed intelligent control algorithm based on reinforcement learning is as follows. The method for pitch axis is the same.

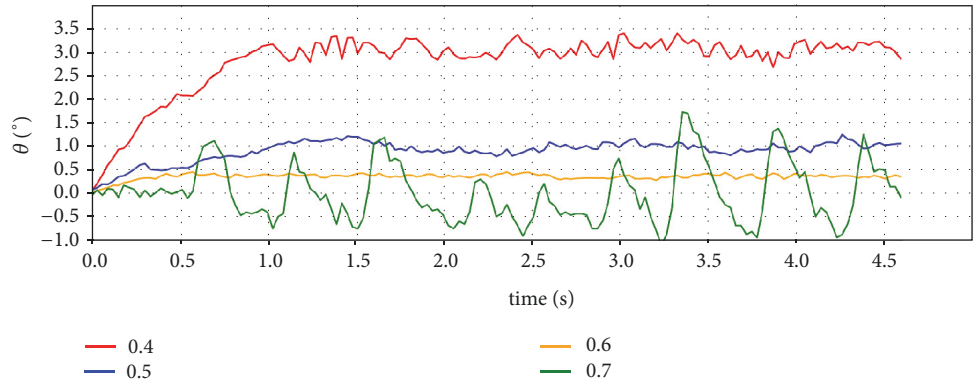


Figure 9: The deviation angle θ (see equation (3)).

4.1. Variable Design of Reinforcement Learning

4.1.1. Design of State Space. Since the feedback of the control pan-tilt platform is derived from the frame directly, we consider that θ_x^t (see equation (3)), ω_r^t (see equation (5)), and β_x^t (see equation (15)) are the most direct factors of γ.

$$\beta_x^t = \frac{\omega_r^t - \omega_r^{t-1}}{\Delta t} \tag{15}$$

where β_x^t is current angular acceleration of the target relative to the pan-tilt platform.

Therefore, the state is designed as follows.

$$S = \left\{\theta_x^t, \omega_r^t, \beta_x^t\right\} \tag{16}$$

4.1.2. Design of Action Space. Our goal is to output the optimal value of γ, so the action space is the range of γ. Obviously, it is desirable if we can design an action space of all possibilities of γ. However, the computer cannot compute a continuous variable. In (13), $\gamma \in (0, 1]$. Therefore, for action space of γ, we divide 0 to 1 into 10 values at the step of 0.1. The action space of γ is $[0.1, 0.2, \ldots, 1.0]$.

4.1.3. Design Experiments and Collect Data. In order to output the optimal γ, we need to collect as many states as possible for each action to make Q-network training better. According to this principle, for each action, we conduct 10 sets of experiments and these experimental scenarios are as follows.

The distance between UAV and pan-tilt platform is 100 m. Select the UAV as the initialization of the particle tracking algorithm and then make the UAV move at 10 different speeds $(1\text{m/s}, \ldots, 10\text{m/s})$ but the motion control model based on angular speed with same γ. Note that the speed at $1\,\text{m/s}, \ldots, 10\text{m/s}$ does not mean that the UAV moves at this speed at a constant speed. It means that the UAV first accelerates to this speed and then keeps moving at a constant speed for a while before it decelerates to 0 m/s. The purpose of this is to get more target motion states because our action space has 10 actions with 10 sets of experiments under each action, and, finally, we have done a total of 100 sets of experiments.

Since the state is $S = \{\theta_x^t, \omega_r^t, \beta_x^t\}$, we store the obtained θ_x^t, ω_r^t, and β_x^t in chronological order in ".txt" file. When we finish all the experiments, we will get the ".txt" files that store the experimental data shown Table 2. The content format in the ".txt" file is as follows:

$$DataSet = \begin{bmatrix} \left\{\theta_x^0, \omega_r^0, \beta_x^0\right\}, \\ \vdots \\ \left\{\theta_x^{t-1}, \omega_r^{t-1}, \beta_x^{t-1}\right\}, \\ \left\{\theta_x^t, \omega_r^t, \beta_x^t\right\} \end{bmatrix} \tag{17}$$

4.1.4. Design of Reward Function and Q-Value Function. To compute Q-value of the RL network, we need to firstly design the reward function for γ. When ω_{tur}^{t+1} is sent to the motor driver of azimuth motor, the azimuth axis will move and make target arrive at θ_x^{t+1} during Δt. To evaluate the reward of γ, the distance between the center of the target and the center of the frame is an important factor. When the target is very close to the center of the frame, we think γ is good, so it will be given a higher reward; otherwise it will be a lower reward. The reward function is shown in

$$R = \begin{cases} 100, & \left|\theta_x^{t+1}\right| \le \theta_{thr} \\ -10 \times \left|\theta_x^{t+1}\right|, & else \end{cases} \tag{18}$$

where R is the reward of γ. θ_{thr} is the threshold of the θ_x^{t+1}. θ_x^{t+1} is the deviation angle at the next moment. If $|\theta_x^{t+1}|$ is in the range of θ_{thr}, γ can be regarded as an optimal action and a positive reward value 100 is given. Otherwise, the reward value is negative. Meanwhile, the greater the $|\theta_x^{t+1}|$ is, the smaller the reward value is, and R is -10 times of $|\theta_x^{t+1}|$. In this paper, we set $\theta_{thr} = 0.5$ because when $\theta_{thr} = 0.5$, the target is almost at the center of the image.

Moreover, we use the one step Q-Learning for proposed intelligent control algorithm based on reinforcement learning approach. The transition rule of Q-Learning is shown in

$$\begin{aligned} Q(S_t, a_t) &= R_t + \alpha \bullet \max_{a_{t+1}} \{Q(S_{t+1}, a_{t+1})\} \\ &= \max \mathbf{E}\left[R_t + \alpha R_{t+1} + \alpha^2 R_{t+2} + \cdots\right] \end{aligned} \tag{19}$$

Table 2: Experimental data.

$v(m/s)$	γ									
	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0
1	Data-v1-1	Data-v1-2	Data-v1-3	Data-v1-4	Data-v1-5	Data-v1-6	Data-v1-7	Data-v1-8	Data-v1-9	Data-v1-10
2	Data-v2-1	Data-v2-2	Data-v2-3	Data-v2-4	Data-v2-5	Data-v2-6	Data-v2-7	Data-v2-8	Data-v2-9	Data-v2-10
⋮						⋮				
10	Data-v10-1	Data-v10-2	Data-v10-3	Data-v10-4	Data-v10-5	Data-v10-6	Data-v10-7	Data-v10-8	Data-v10-9	Data-v10-10

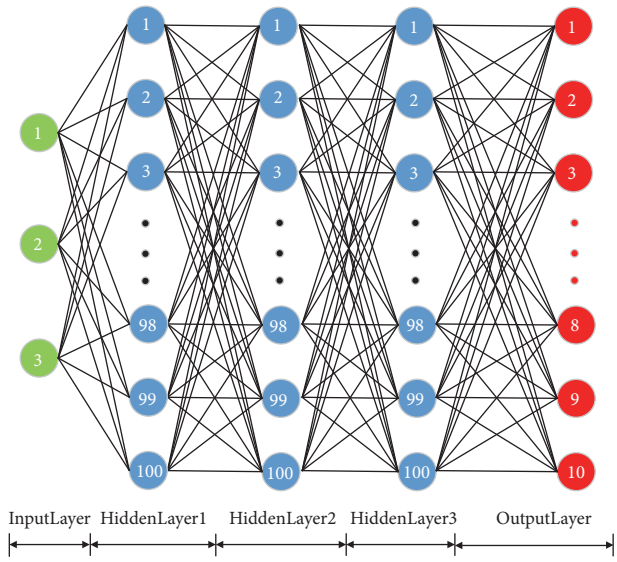


Figure 10: Structure of the Q-value network.

where S_{t+1} corresponds to the next state after carrying action a_t while in state S_t, a_{t+1} is the action of state S_{t+1}. α is the discount factors and we use $\alpha = 0.9$ in this paper.

4.2. Design and Training of Q-Value Network

4.2.1. Design of Q-Value Network. We propose to design a neural network to learn Q-Values of the 10 actions and a fully connected neural network with ReLU nonlinearities is employed to parametrize the Q-Value function. The structure of Q-Value network is shown as Figure 10.

The Q-Value network with 3 hidden layers and the numbers of hidden neurons are all 100. In input layer, there are 3 neurons corresponding to the state elements $S = \{\theta_x^t, \omega_r^t, \beta_x^t\}$. In output layer, there are 10 neurons corresponding to the Q-Values of the 10 actions. The input and output of the Q-Value network are shown as Tables 3 and 4.

4.2.2. Training Method for Q-Value Network

(1) State-Transition and Getting Experience. Unlike online exploration learning, we do not use a random selection of an action to transfer to the next state, but to transfer the state of agent (pan-tilt platform) under the specified action. Therefore, the state-transition model is that the current state transfer to next state with the same γ which we chose during an experiment.

Table 3: The inputs of the Q-value Network in Figure 10.

Layer	No.	Meaning	Unit
Input	1	$S[1], \theta_x^t$, see equation (3)	°
	2	$S[2], \omega_r^t$, see equation (5)	°/s
	3	$S[3], \beta_x^t$, see equation (15)	$°/s^2$

Before training the Q-Value network, we first need to get experience of agent. In collected experimental data, we can get the current state S_t, the next state S_{t+1}, the reward R_t which get from (18), and the current action a_t(a_t is γ which we chose during an experiment). We store S_t, a_t, R_t, S_{t+1} as a tuple and the agent's experience will be expressed as $Exp = \{S_t, a_t, R_t, S_{t+1}\}$. We put all Exp into the memory pool D.

$$D = [Exp_1, Exp_2, \ldots,] \tag{20}$$

There are about 900,000 experiences in the memory pool D.

(2) Memory Replay for Q-Value Computation. In order to train our Q-Value network, the key idea is experience replay [16]. During learning, we sample in the memory pool D randomly to get a bath of experiences. We use S_t as input of the Q-Value network which is shown in Figure 10 and get current state Q-Value: $Q(S_t, a_t)$. We clone the Q-Value network to obtain the target Q-Value network for every 100 steps. We use S_{t+1} as input of the target Q-Value network and get next

Get the memory pool D
Initialize Q-value network with random θ
For epochs = 1, 1000,000 **do**
    Sample random from memory pool to get 50 samples
    Perform a gradient descent step on equation (21) respect to the network parameters θ
    Every 100 steps, clone the Q-value network to obtain the target network Q-value network
End For

Algorithm 2: Memory replay for Q-value network.

Table 4: The output of Q-value network in Figure 10.

Layer	No.	Meaning	Unit
Output of Q-Value Network	1-10	10 Q-Values computed with equations (18) and (19)	-

state Q-Value: $\widehat{Q}(S_{t+1}, a_{t+1})$. Then we apply Q-Learning (see (19)) to update the current state Q-Value: $\widehat{Q}(S_t, a_t)$. So, we get $\widehat{Q}(S_t, a_t) = R_t + \gamma \bullet \max_{a_{t+1}}\{\widehat{Q}(S_{t+1}, a_{t+1})\}$. The loss function is as follows:

$$L(\theta) = \left(Q(S_t, a_t) - \widehat{Q}(S_t, a_t)\right)^2 \tag{21}$$

where θ are the parameters (weights and bias) of the network.

Since the sample collected by the agent to explore the environment is a time series, there is continuity between the samples. We use a random sampling strategy to get training batch in the memory pool. The algorithm for training Q-value network is presented in Algorithm 2.

(3) Network Training Details. The main learning method of our Q-value network is Backpropagation (BP) [17]. This procedure is repeated numerous times until the predefined upper limit of a certain number of steps. Other key training parameters and functions are as follows.

(1) Optimization algorithm: stochastic gradient descent.

(2) Batch size and the number of training steps: 50, 1000,000.

(3) Learning rate: 0.00001. To make gradient descent have a good trade-off of accuracy and performance, it is desirable to set the learning rate to a suitable value.

(4) Activation function is Rectified Linear Unit (ReLU): $\mathrm{ReLU}(x) = \max(x, 0)$.

(5) Loss function $L(\theta)$ is using the mean-square error (MSE) method, i.e., $L(\theta) = \sum_k (\widehat{y_k} - f(x_k, \theta))^2$. Here θ is the networks parameter set, i.e., $\theta = \{w^1, b^1, w^2, b^2, \ldots, w^L, b^L\}$. w is the weight and b is the bias.

4.3. Experiment. Select the UAV as the initialization of the particle tracking algorithm and then make the UAV move at different speed (1m/s, $\cdots$,10m/s). The experiment results are shown as Figure 11.

From Figure 11, we found the tracked target under different motion states, the deviation angle θ in the range of $[-0.5, 0.5]$ which means the tracked target almost in the center of the frame. This shows that our proposed algorithm has better accuracy and robustness.

5. Conclusion

This paper presents a novel pan-tilt platform control method which consists of two parts. One is a motion control model based on angular speed and the other is an intelligent control algorithm based on reinforcement learning. To the best of our knowledge, this is the first work to use the angular speed to control visual pan-tilt platform to track target. Meanwhile, in order to reduce the error between the ideal value and the actual value when the pan-tilt platform moves, we implement an intelligent control algorithm based on reinforcement by learning in a data-driven manner, which is different from traditional knowledge-driven methods, such as PID. The experiment results show that the proposed methods can be applied to track a real-time dynamic target. Moreover, the tracking accuracy and robustness are improved and the stability of the system is ensured.

Due to the limitations of our experimental conditions, the experimental data we collected does not contain all the motion states of the target. In the future work, we will focus on improving the response frequency of the pan-tilt platform, doing more experiments to track high-speed targets, collecting more data and designing deeper Q-Value networks to output optimal γ.

Data Availability

(1) The [.txt] data used to support the findings of this study have been deposited in the [Partially data collected by experiments_datas] repository (https://github.com/blueskyM01/experiment-data-of-tracking/tree/master/Partially%20data%20collected%20by%20experiments_datas). As the project is funded by AVIC AEROSPACE LIFE-SUPPORT INDUSTRIES, LTD and has been used for commercial purposes. It only provides partial data to prevent other people from using the complete data for other commercial purposes. (2) The [.jpg] data used to support the findings of this study have been deposited in the [Partially data collected by experiments_

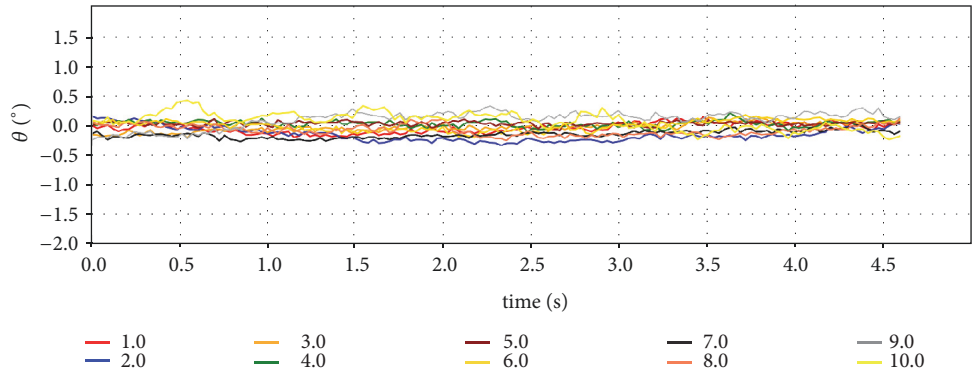


Figure 11: The deviation angle θ (see equation (3)).

photos] repository (https://github.com/blueskyM01/experiment-data-of-tracking/tree/master/Partially%20data%20collected%20by%20experiments_photos). As the project is funded by AVIC AEROSPACE LIFE-SUPPORT INDUSTRIES, LTD and has been used for commercial purposes. It only provides partial data to prevent other people from using the complete data for other commercial purposes. (3) The [Fig.] data used to support the findings of this study are included within the article. (4) The [Code] data used to support the findings of this study are currently under embargo while the research findings are commercialized. Requests for data, [3 years] after publication of this article, will be considered by the corresponding author.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported by a grant from the Major State Basic Research Development Program of China (973 Program) (no. 2016YFC0802703).

References

[1] K. Mehmood, M. Mrak, J. Calic, and A. Kondoz, "Object tracking in surveillance videos using compressed domain features from scalable bit-streams," *Signal Processing: Image Communication*, vol. 24, no. 10, pp. 814–824, 2009.

[2] C. Yu, J. Cai, and Q. Chen, "Multi-resolution visual fiducial and assistant navigation system for unmanned aerial vehicle landing," *Aerospace Science and Technology*, vol. 67, pp. 249–256, 2017.

[3] Y. Motai, S. Kumar Jha, and D. Kruse, "Human tracking from a mobile agent: optical flow and Kalman filter arbitration," *Signal Processing: Image Communication*, vol. 27, no. 1, pp. 83–95, 2012.

[4] B. Zhang, J. Huang, and J. Lin, "A novel control algorithm for object tracking by controlling PAN/TILT automatically," in *Proceedings of the 2010 2nd International Conference on Education Technology and Computer (ICETC)*, pp. V1-596–V1-602, Shanghai, China, June 2010.

[5] H. Chen, X. Zhao, and M. Tan, "A novel pan-tilt camera control approach for visual tracking," in *Proceedings of the 2014 11th World Congress on Intelligent Control and Automation, WCICA 2014*, pp. 2860–2865, China, July 2014.

[6] S. R. Yosafat, C. Machbub, and E. M. I. Hidayat, "Design and implementation of Pan-Tilt control for face tracking," in *Proceedings of the 7th IEEE International Conference on System Engineering and Technology, ICSET 2017*, pp. 217–222, Malaysia, October 2017.

[7] A. Martínez-Tenor, J. A. Fernández-Madrigal, A. Cruz-Martín, and J. González-Jiménez, "Towards a common implementation of reinforcement learning for multiple robotic tasks," *Expert Systems with Applications*, vol. 100, no. 15, pp. 246–259, 2018.

[8] Y. F. Chen, M. Everett, M. Liu, and J. P. How, "Socially aware motion planning with deep reinforcement learning," in *Proceedings of the 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 1343–1350, Vancouver, BC, September 2017.

[9] H. Kretzschmar, M. Spies, C. Sprunk, and W. Burgard, "Socially compliant mobile robot navigation via inverse reinforcement learning," *International Journal of Robotics Research*, vol. 35, no. 11, pp. 1352–1370, 2016.

[10] J. A. Bagnell and J. G. Schneider, "Autonomous helicopter control using reinforcement learning policy search methods," *Proceedings - IEEE International Conference on Robotics and Automation*, vol. 2, pp. 1615–1620, 2001.

[11] R. S. Sutton and A. G. Barto, *Introduction to Reinforcement Learning*, MIT Press, Cambridge, MA, USA, 1st edition, 1998.

[12] M. Firouznia, K. Faez, H. Amindavar, and J. A. Koupaei, "Chaotic particle filter for visual object tracking," *Journal of Visual Communication and Image Representation*, vol. 53, pp. 1–12, 2018.

[13] X. Qian, L. Han, Y. Wang, and M. Ding, "Deep learning assisted robust visual tracking with adaptive particle filtering," *Signal Processing: Image Communication*, vol. 60, pp. 183–192, 2018.

[14] W. Li, P. Wang, and H. Qiao, "Top-down visual attention integrated particle filter for robust object tracking," *Signal Processing: Image Communication*, vol. 43, pp. 28–41, 2016.

[15] I. A. Iswanto and B. Li, "Visual Object Tracking Based on Mean-shift and Particle-Kalman Filter," in *Proceedings of the 2nd International Conference on Computer Science and Computational Intelligence, ICCSCI 2017*, pp. 587–595, Indonesia, October 2017.

[16] V. Mnih, K. Kavukcuoglu, D. Silver et al., "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, 2015.

[17] X. Song and J. Sun, "OpenPTDS," in *Proceedings of AsiaSim*, 2018.