Contents lists available at ScienceDirect

Robotics and Autonomous Systems

journal homepage: www.elsevier.com/locate/robot

A saliency-based reinforcement learning approach for a UAV to avoid flying obstacles

Zhaowei Ma, Chang Wang, Yifeng Niu *, Xiangke Wang, Lincheng Shen

College of Mechatronic Engineering and Automation, National University of Defense Technology, Changsha 410073, China

HIGHLIGHTS

- We propose a novel framework in which a UAV can learn to avoid flying obstacles.
- We propose an improved saliency detection method based on convolution neural networks.
- We propose an actor–critic reinforcement learning algorithm to control UAV in continuous spaces.

ARTICLE INFO

Article history:
Received 15 February 2017
Received in revised form 12 September 2017
Accepted 13 October 2017
Available online 14 November 2017

Keywords:
UAV
Flying obstacle avoidance
Convolution neural networks based saliency detection
Reinforcement learning

ABSTRACT

Obstacle avoidance is a necessary behavior to guarantee the safety of an unmanned aerial vehicle (UAV). However, it is a challenge for the UAV to detect and avoid high-speed flying obstacles such as other UAVs or birds. In this paper, we propose a generic framework that integrates an autonomous obstacle detection module and a reinforcement learning (RL) module to develop reactive obstacle avoidance behavior for a UAV. In the obstacle detection module, we design a saliency detection algorithm using deep convolution neural networks (CNNs) to extract monocular visual cues. The algorithm imitates human's visual detection system, and it can accurately estimate the location of obstacles in the field of view (FOV). The RL module uses an actor–critic structure that chooses the RBF neural network to approximate the value function and control policy in continuous state and action spaces. We have tested the effectiveness of the proposed learning framework in a semi-physical experiment. The results show that the proposed saliency detection algorithm performs better than state-of-the-art, and the RL algorithm can learn the avoidance behavior from the manual experiences.

1. Introduction

Unmanned aerial vehicles (UAVs) have been widely used in many domains such as aerial photography, search and rescue, anti-terrorist action, and agricultural surveillance. However, avoiding flying obstacles remains a big challenge for integrating UAVs into the national airspace system (NAS) with reliable and safe operations. Manual control may reduce the risk of UAV collisions under certain circumstances, but it requires substantial training and imposes significant cognitive loads on human operators, especially when a swarm of UAVs are controlled. In addition, it is difficult for the human operators to find obstacles at high altitude or in low visibility using the human visual system (HVS). Furthermore, both the UAV and the flying obstacles are moving in real-time and the dynamics of the flying obstacles are not always available. Therefore, it is necessary to develop an autonomous obstacle detection and avoidance method that can handle dynamic environments and learn from the human operators.

In the literature, methods combined with a low-level reactive layer have proved effective in solving the obstacle avoidance problem [1–6]. In the low-level reactive layer, the biological inspired behavior-based mode (BBM) [1] maps from perception to behavior directly, which is reactive to the changing environment and runs fast. The BBM mainly mimics how flying insects avoid the obstacle. However, these methods are typically based on specific environment models which are difficult to acquire [4–6]. In contrast, we take a model-free robot learning approach to develop adaptive obstacle avoidance behavior. Specifically, we choose the actor–critic (AC) reinforcement learning (RL) framework which is a machine learning technique based on trial-and-error mechanisms, thus improving the performance by getting feedback from the environment. It does not require knowledge of the environment and supports online learning.

Another challenge for avoiding obstacles for UAVs is to detect the obstacles. Much research have been done on automatic obstacles detection using computer vision methods. Among them,

* Corresponding author.
E-mail address: niuyifeng@nudt.edu.cn (Y. Niu).

https://doi.org/10.1016/j.robot.2017.10.009
0921-8890/© 2017 Elsevier B.V. All rights reserved.

optical flow is a bio-inspired approach that many flying insects use to sense the collision [7]. However, an embedded optical flow is limited in its resolution, providing only general guidance about obstacles. An estimated depth map using multiple images can provide obstacle distance information [8]. However, these algorithms require handcrafted features, which decreases the generalization ability of handling unknown environments. In this paper, we choose the saliency-based approach to detect the obstacles using a monocular forward-looking camera. This method is inspired by the human saliency system, which is a neural cognitive reaction controlled by human brains. And a lot of works have been done to develop this method [9–13]. Unlike their methods where mid-level filters are handcrafted, the saliency detection method based on deep convolution neural network (CNN) in our framework is automatically and jointly learned in a discriminative manner, because the deep networks mimic the functions of neocortex in the human brain as a hierarchy of filters and nonlinear operations, which improves the robustness of the algorithm.

The main contributions of this paper are as follows:

(1) We propose a novel framework in which a UAV can learn to avoid flying obstacles. This framework integrates an autonomous obstacle detection module and a reinforcement learning (RL) module to develop reactive obstacle avoidance behavior for a UAV. By using this framework, the UAVs can take the right obstacle avoidance action at the right time based on the environmental states.
(2) We propose an improved saliency detection method based on convolution neural networks. By using the deep neural networks, the obstacle detection algorithm has more generic application without handcrafted features. Furthermore, we apply the Kalman filter (KF) to predict the moving trajectory of flying obstacles.
(3) We propose an actor–critic reinforcement learning algorithm to control the UAV in continuous spaces. In this paper, we use RL to construct a mapping from the environmental states to the avoidance action. We choose the radial basis function (RBF) neural network in an actor–critic reinforcement learning module. In this algorithm, a reward function is specially constructed according to the obstacle state, which is used to guide and update the actor and critic the fitting model.

The remainder of this paper is organized as follows. In Section 2, we briefly review related work on the saliency detection methods and vision-based reinforcement learning methods. In Section 3, we introduce the UAV obstacle avoidance task and propose the saliency-based learning framework. A saliency detection algorithm based on CNN is proposed in Section 4. In Section 5, the reinforcement learning framework shows how to map the visual states to the reference angles based on the RBF neural network. Section 6 presents the offline validation experiments about this learning framework by using the simulated environment and a control hardware unit. The saliency detection algorithm is also validated using the databases and images from the simulating system. Finally, we conclude and discuss the future work in Section 7.

2. Related work

We give a brief overview about saliency detection methods and the vision-based RL method in this section.

The saliency detection method is inspired by the human vision system (HVS). Visual saliency shows the region that attracts human attention. The HVS is able to identify salient objects even in a complex scene by exploiting the inherent visual attention mechanisms. The first proposed saliency model computes feature maps with luminance, color, and orientation using a center–surround operator across different scales [14]. Then it performs normalization and summation to generate the saliency map. There are two kinds of methods to compute visual saliency: the bottom-up and the top-down methods [9–13]. Generally, the bottom-up models are based on a center–surround scheme, computing a master saliency map by various types of handcrafted low-level features such as color, intensity, and texture. The top-down methods need high-level knowledge to obtain the location of a salient object. However, handcrafted features are not robust enough for challenging scenes and to lose detailed, important and useful information.

Vision-based reinforcement learning (RL) methods have been applied in a variety of robotic tasks. For example, behaviors of other soccer robots are recognized for developing cooperative action policies [15–18]. Q-learning has been used for solving the simultaneous localization and mapping (SLAM) problem [19]. A deep unsupervised convolution network has been developed on the TORCS simulator [20]. In [21], neuro-evolution is combined with an unsupervised sensory pre-processor or compressor that is trained on images generated from the environment by the population of evolving recurrent neural network controllers. Reaching and pushing actions are learned to manipulate a box [22]. An intrinsically motivated RL approach has been proposed for active learning of object affordances and manipulation skills in continuous state and action spaces [23]. Similarly, a model-free actor–critic algorithm based on the deterministic policy gradient can handle continuous actions [24].

However, all these methods have limitations in solving the flying obstacle avoidance problem for UAVs, because the flying obstacles can be fast, small and blurred, which makes them hard to detect and track. Besides, learning control policies are also challenging in high-dimensional visual spaces and continuous action spaces. To address these challenges, we propose a novel method that integrates saliency-based object detection with the RL-based action selection. Our research is related to the saliency detection [9–13], but we have done this work using the CNN-based method.

3. Task description and the RL framework

3.1. Flying obstacle avoidance

In this paper, we discuss the problem of avoiding a single flying obstacle for UAV using a single camera. As shown in Fig. 1, the obstacle can be another aircraft or a bird. As the relative distance between the UAV and the obstacle changes, the size and location of the obstacle will change accordingly in the captured image, as shown in Fig. 1. The task is to detect the obstacle and observe its states, then select actions for the UAV to avoid the obstacle.

At the time step t, the UAV captures an image I_t with the size of $m = w \times h \times c$, where w is the width of the image, h is the height and c, is the channel. Denote by $s_t = \{u_t, v_t, O_t\}$ the state vector of the obstacle, where (u_t, v_t) is the center coordinates of the obstacle in I_t, and O_t is the number of pixels occupied by the obstacle. Then, we choose the reference turning angles $\begin{bmatrix}\psi_t & \phi_t\end{bmatrix}$as the action vector a_t for the UAV, where ψ_t is the yaw angle and ϕ_t is the roll angle. The goal state of avoiding the obstacle is described as $s_{t\_final} = \left\{u_{t\_final}, v_{t\_final}, 0\right\}$, where $u_{t\_final} \in \left\{-\frac{w}{2}, \frac{w}{2}\right\}$ and $v_{t\_final} \in \left[-\frac{h}{2}, \frac{h}{2}\right]$. The final state means that the flying obstacle is located at the margin of the image coordinate system.

3.2. Vision-based reinforcement learning framework

Based on the obstacle avoidance task description, we propose a saliency-based reinforcement learning framework as shown in

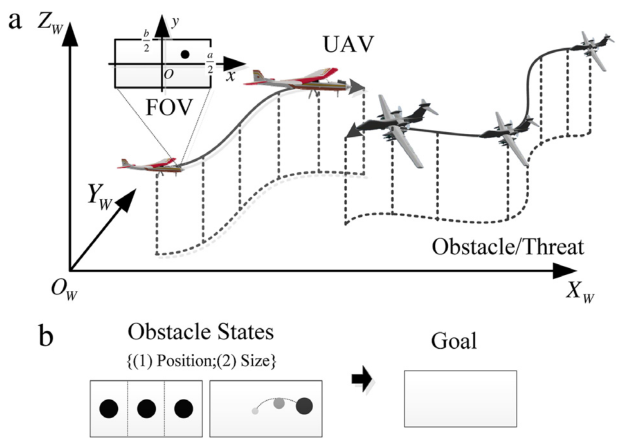


Fig. 1. The flying obstacle avoidance task and the state description.

Fig. 2. The framework mainly consists of three modules: an obstacle detection module, a reinforcement learning module and a UAV module.

(1) There are two main sub-modules in the obstacle detection module. The image data flow from the airborne forward-looking camera is sent to the saliency detection algorithm to extract the obstacle's location in the captured image. The detection algorithm may cause errors about the obstacle's location in the image. So we use the Kalman filter (KF) for the state modification in this module. We will further discuss the state modification model in Section 4.

(2) The reinforcement learning module receives s_t from the obstacle detection module and evaluates the environmental state using the reward function. The reward function consists of expected errors and transferred errors. The expected error lies in the difference between the current observed state and the final state, and the transferred error is in the difference between the current observed state and the next state. We choose the neural networks to approximate the actor and critic in this module. The temporal difference is calculated from the reward and used to tune the parameters of the actor–critic neural networks. The reference angle signal is the output of the RL module. By training and constantly correcting the control strategy, the RL module can learn the optimal control strategy for the UAV to avoid obstacles. The detailed algorithm will be given in Section 5.

(3) The UAV model adjusts the position and attitude of the UAV based on the reference angles, thus changes the captured image and the state s_t. We choose the lateral maneuver behavior as the UAV's controlling mode. In general, horizontal lateral stability is controlled by the rudder and aileron. Coordinated turn maneuver at fixed height is the best strategy for avoidance [25].

4. CNN-based saliency detection

Deep convolution neural networks (CNNs) have been applied to computer vision tasks such as handwritten digit classification and face detection. To obtain features more robust than handcrafted ones for salient object detection, CNNs have achieved better results than the previous state-of-the-art. CNNs have been widely used for the end-to-end learning of the image, where "end-to-end" means that the deep network only needs to be run on the input image to produce a complete saliency map with the same pixel resolution as the input image [26].

4.1. A brief introduction of the CNN

A CNN typically contains three types of layers, i.e. a convolution layer, a sub-sampling layer and a fully connected layer [26]. Denote by I^{l-1} the input data of the lth layer and I^l the output data, where $l \in \{1, \ldots, L\}$. The convolution layer is as follows:

$$I_j^l = sigm\left(\sum_{i=1} I_i^{l-1} \otimes k_{ij}^l + b_j^l\right) \tag{1}$$

where k_{ij}^l is the convolution kernel and b_j^l is the bias term, $sigm$ is the activation term using the sigmoid function, and $\otimes$ denotes a convolution operation. The convolution operation is performed first with convolution kernels. The bias term b_j^l is added to the resulting feature maps, in which an activation operation is performed subsequently.

The sub-sampling layer is usually used to select the dominant features over non-overlapping square windows per feature map as follows:

$$I_j^l = downsample\left(I_i^{l-1}, m\right) + b_j^l \tag{2}$$

where b_j^l is a bias term; and $downsample$ performs the average sub-sampling operation in each $m \times m$ region that does not overlap with each other.

The hierarchical feature extraction architecture consists of several convolution layers and sub-sampling layers. The final layer is

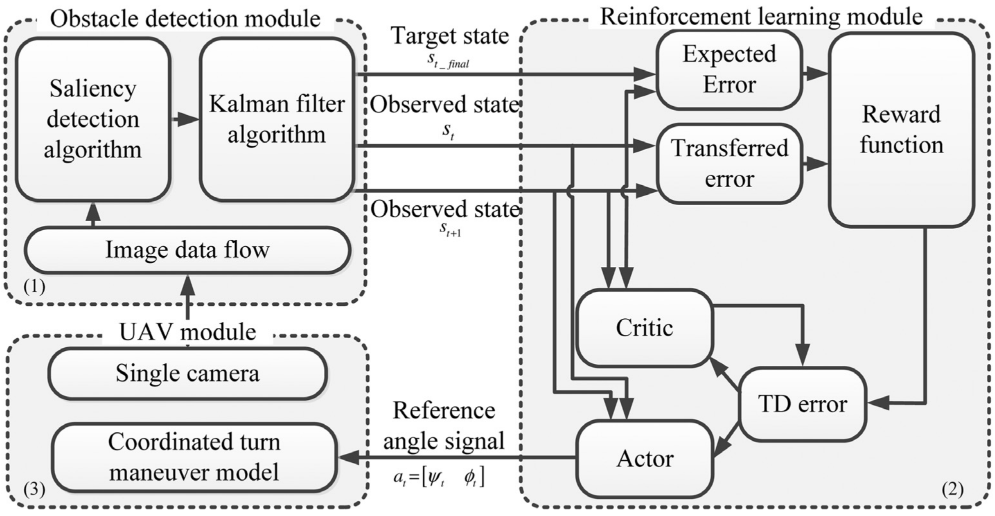


Fig. 2. A saliency-based reinforcement learning framework for developing UAV obstacle avoidance behavior.

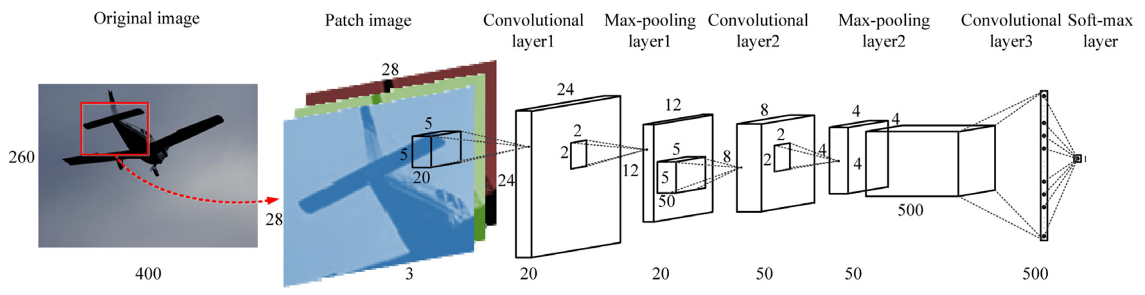


Fig. 3. The architecture of the proposed CNN-based model.

combined into a feature vector by several fully connected layers. The last classification layer is usually a softmax layer with the same number of neurons with the number of output classes.

4.2. CNN-based saliency detection

Inspired by the hierarchical feature extraction architecture, we propose a novel CNN architecture with six layers to learn the bottom features and predict the saliency regions of an image (see Fig. 3). The input is an image and the output is the saliency map. The original size of the image I is $w \times h \times c$, where c is the channel size of the image. And the size of the saliency map is $w \times h$. The network contains six learned layers: three convolution layers, two sub-sampling layers, and a fully connected layer. The goal of saliency detection is to distinguish the salient region from the background region. We consider the saliency detection task as a binary classification problem. For a patch image $I_p \in I$, the label is *label* $\in \{0, 1\}$ where 1 represents the salient region and 0 represents the non-salient region.

(1) Preprocessing of training Data

Given a group of training images and their saliency maps, we need to construct the labeled training database. In the database, the patch images are randomly sampled from the original images based on the saliency values in the ground truth maps. We randomly select 50 patch images in each original image. The patch images are chosen as the input of the CNN model, centered at the salient or non-salient locations with corresponding binary classification labels. The patch images are sampled to 28 × 28 and separated into color channels R, G and B (see Fig. 3). In order to alleviate the over-fitting problem and reduce the variance of the data samples, each input patch image is preprocessed by:

$$I_p' = \frac{I_p - mean\left(I_{p\_database}\right)}{\max\left(I_{p\_database}\right) - \min\left(I_{p\_database}\right)} \tag{3}$$

where $I_{p\_database}$ is the whole patch image database; *mean* is the mean operation; max is the function to find the maximum pixel value; and min is the function to find the minimum pixel value.

(2) Architecture

Three convolution layers and two sub-sampling layers are stacked one by one to construct the whole hierarchical feature extraction architecture (see Fig. 3). The network size is I[28 × 28 × 3] − C[24 × 24 × 20] − P[12 × 12 × 20] − C[8 × 8 × 50] − P[4 × 4 × 50] − FC[500)] − O[1]. In other words, C1 is the first convolution layer and it has 20 channels. Each channel in I1 contributes to all 20 channels in C2. Therefore, overall 60 (3×20) kernels are applied into the first convolution layer. The size of a kernel is 5×5 which leads to 1520 trainable parameters in total (20 comes from the biases with one for each channel). Each kernel represents a feature between two corresponding channels. The sigmoid function has the phenomenon of saturation and gradient disappearance. The rectified linear unit (ReLU) is used to enhance the training speed in the convolution layer. Finally, the output of the last maxout layer is the saliency prediction label. The output layer has only one channel. We use a softmax layer with one neuron to execute the binary classification. The output value represents the probability of the input belonging to the positive class. And it is used to present the salient level.

(3) Training process

The patch images with corresponding labels are trained using the CNN model with a large number of weights. The weights in all layers are initialized from a normal Gaussian distribution with zero mean and a standard deviation of 0.01, with biases initialized to 0.1. During training, we apply the delta rule back-propagation algorithm to train our model based on the minimum errors between the predicted labels and the ground truth labels in the last layer. When the training errors converge, the trained network can be used to predict the saliency class for the patch images.

(4) Salient region detection

The trained feature extraction architecture is applied to the patch image $I_p \in I$. So in order to acquire a complete saliency map with the same pixel resolution as the input image, a sliding window with one-pixel step is applied to the whole image I. In this paper, the obstacle is a solid object rather than a ring-shaped obstacle. In order to obtain a precise state vector $s_t = \{u_t, v_t, O_t\}$ of the obstacle, we use the following average method to get the obstacle center image coordinates (u_t, v_t) based on the saliency map.

$$\begin{cases} u_t = \dfrac{1}{O_t} \displaystyle\sum_{i=1}^{m} p_{x\_i} \\ v_t = \dfrac{1}{O_t} \displaystyle\sum_{j=1}^{n} p_{y\_j} \end{cases} \tag{4}$$

where O_t is the number of nonzero pixels in the saliency map and $\left(p_{x\_i}, p_{y\_j}\right)$ is the image coordinates of nonzero pixels.

In order to reduce detection errors, we construct a linear state prediction model to predict the obstacle's state s_{t+1} at the time step $t+1$ based on the current state s_t:

$$\begin{cases} x_{t+1} = f_t x_t + \upsilon_t \\ y_{t+1} = h_{t+1} x_{t+1} \end{cases} \tag{5}$$

where $x_t = \begin{bmatrix} u & v & O & \dot{u} & \dot{v} & \dot{O} \end{bmatrix}_{t'}$ is the state vector of the model, $\begin{bmatrix} \dot{u} & \dot{v} & \dot{O} \end{bmatrix}_t$ is the derivative and represents the variation trend, $f_t = \begin{bmatrix} I_{3\times3} & \Delta t_{3\times3} \\ 0 & I_{3\times3} \end{bmatrix}$ is the system matrix, υ_t is the zero-mean Gaussian noise, and $h_{t+1} = \begin{bmatrix} I_{3\times3} & 0_{3\times3} \end{bmatrix}$ is the output matrix.

We use a Kalman filter (KF) for the state modification based on the predicted output y_{t+1}at the time step $t+1$. The modified state is:

$$\hat{x}_{t+1} = x_{t+1}^- + K_{t+1}\left(y_{t+1} - h_{t+1}x_{t+1}^-\right) \tag{6}$$

where $K_{t+1} = P_{t+1}^- h_{t+1}^T \left(h_{t+1} P_{t+1}^- h_{t+1}^T + R_{t+1}\right)^{-1}$ is the Kalman filter gain matrix, R_{t+1} is the noise covariance matrix of the output equation, $P_{t+1}^{-} = f_t P_t^{+} f_t^{T} + Q_t$, $P_{t+1}^{+} = (1 - K_{t+1} h_{t+1}) P_{t+1}^{-}$, Q and R are the noise covariance matrix of the state and the output, respectively.

5. The saliency-based RL for obstacle avoidance

5.1. Actor–critic learning architecture

We choose the actor–critic (AC) architecture to get the optimal control policy. In the architecture, there are three important sub-modules, the actor, the critic, and the reward function. The critic learns to predict the value of each state and computes the Temporal Difference (TD) error, which is used by the actor to output optimal actions that will maximize the accumulated future rewards. The input of the critic is the state vector and the output is the estimated value. The nonlinear approximate function of the critic is:

$$\hat{V}(s_t) = f(s_t, W_t) \tag{7}$$

where W_t is a parameter vector at the time step t. Similarly, a nonlinear function is used to approximate the actor. The input of the actor is also the state vector and the output is the reference angle of the UAV. The nonlinear approximate function of the actor is:

$$a(t) = f(s_t, \Theta_t) \tag{8}$$

where Θ_t is a parameter vector at the time step t. The input of the actor and the critic modules both have the state s_t, but they have different parameter vectors. In order to reduce the storage space of the learning system and avoid the repeated computation for the output of hidden layer nodes, we use a forward RBF neural network to approximate the actor and critic. This means that the hidden layer is shared by both the actor and the critic.

The whole structure is described as follows.

(1) Input layer

The input of the neural network is the state vector $s_t = \{u_t, v_t, O_t\}$. Each node is an element in the vector.

(2) Hidden layer

The forward hidden layer uses the RBF to map the input layer to the hidden layer. We use the Gaussian function as the base function of the hidden layer where $\varphi_i(t)$ is the output of each hidden layer.

$$\varphi_i(t) = \exp\left\{-\frac{[s(t) - c_i(t)]^2}{2\sigma_i^2}\right\}, i = 1, \ldots, N\_g \tag{9}$$

in which $c_i = \begin{bmatrix} c_{1i} & c_{2i} & c_{3i} \end{bmatrix}^T$ is the central vector of the ith node, σ_i is the standardized constant of the ith node, and N_g is the hidden neuron number. The output range of the hidden layer is $[0, 1]$.

(3) Output layer

The output layer contains nodes for both the actor and the critic. The actor $\hat{a}(t) = \begin{bmatrix} \hat{a}_1(t) & \cdots & \hat{a}_{N\_a}(t) \end{bmatrix}^T$ is as follows

$$\hat{a}_j(t) = \sum_{n=1}^{N\_g} \omega_{jn}(t)\varphi_n(t), j = 1, \ldots, N\_a \tag{10}$$

in which N_a is the number of the control variable, $\omega_{jn}(t)$ is the weight factor from the jth control node to the nth node of the hidden layer at the time step t. And the value function $V(s_t)$ is as follows

$$V(s_t) = \sum_{n=1}^{N\_g} \upsilon_n(t)\varphi_n(t) \tag{11}$$

in which $\upsilon_n(t)$ is the weight factor from the nth node to the value function $V(s_t)$ of the hidden layer.

(4) Reward function

In this paper, the reward function is mainly used for evaluating the changes of obstacle's relative position in the FOV, which leads the UAV to perform the avoidance action. We choose the following reward function:

$$r_t(\hat{s}_t, \hat{s}_{t+1}) = \left\|\hat{s}_{t+1} - \hat{s}_t\right\|_2 + \left\|\hat{s}_t - s_{final}\right\|_2 \tag{12}$$

where the first term describes the cost between two measured states and the second term describes the cost occurring from the current state to the goal state.

(5) Temporal difference learning

When the state transfers from s_t to s_{t+1}, the TD error is calculated as follows

$$\delta_t = r_t + \gamma\hat{V}(s_{t+1}) - \hat{V}(s_t) \tag{13}$$

in which γ is the discount factor, and r_t is the reward function defined in (12).

We use the TD learning algorithm based on the nonlinear function to approximate and update the actor and critic, i.e. the parameters of the RBF neural network. In the RBF neural network, the updated weighted factor of the critic network is

$$\upsilon_n(t+1) = \upsilon_n(t) + \alpha_c\delta_t\varphi_n(t) \tag{14}$$

in which α_c is the learning rate of the critic network.

In the process of learning, we use Gaussian probability distribution to determine the actual control output for action exploration:

$$p\left[\hat{a}_j(t)\right] = \frac{1}{\sqrt{2\pi}\sigma_t}\exp\left[-\frac{\hat{a}_j(t)^2}{2\sigma_t^2}\right] \tag{15}$$

in which $u_j(t)$ is the actual control variable and the variance is computed by,

$$\sigma_t = \frac{1}{1 + \exp[2V(s_t)]} \tag{16}$$

So the actual control variables $a(t) = \begin{bmatrix} a_1(t) & \cdots & a_{N\_a}(t) \end{bmatrix}^T$ of the actor are calculated as

$$a(t) = \hat{a}(t) + p\left[\hat{a}_j(t)\right] \tag{17}$$

The updated weight factor of the actor network is defined by,

$$\omega_{jn}(t+1) = \omega_{jn}(t) + \alpha_a\delta_t\frac{a_j(t) - \hat{a}_j(t)}{\sigma_t}\varphi_n(t) \tag{18}$$

in which α_a is the learning factor of the actor network.

5.2. Obstacle avoidance control algorithm for the UAV

Coordinated turn maneuver at fixed height is the best strategy for realizing avoidance [25]. We assume some conditions to realize this maneuver. (1) The roll angle and yaw rate are normal values in the steady state. (2) The lifting speed and the sideslip angle are zero in the steady state. In order to analyze the force of the UAV in the avoidance process, we assume that the pitching angle $\theta = 0$. As shown in Fig. 4, there is the following equivalence relationship for the balancing force when the UAV performs the turning behavior.

In the horizontal direction, the centripetal force is defined by,

$$F_{lift}\sin\phi = \frac{mV^2}{R} \tag{19}$$

In the direction of gravity,

$$F_{lift}\cos\phi = mg \tag{20}$$

Based on Eq. (19) and Eq. (20), we can get the turning radius:

$$R = \frac{V^2}{g\tan\phi} \tag{21}$$

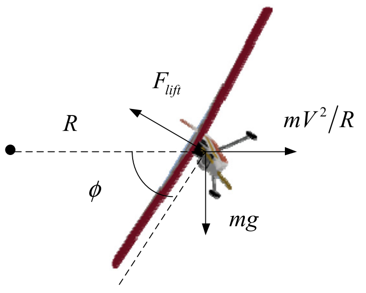


Fig. 4. UAV force analysis in the coordinated turn maneuver.

The time for completing a circle turning is computed by:

$$t = \frac{2\pi R}{V} = \frac{2\pi V}{g\tan\phi} \tag{22}$$

So the turning rate is defined by,

$$\dot{\psi} = \frac{2\pi}{t} = \frac{g\tan\phi}{V} \tag{23}$$

When realizing a coordinated turn, the UAV needs to tune the roll angle ϕ and the pitching angle θ in order to maintain a constant height and a constant yaw angular rate. Based on the Eq. (20), the height of the UAV will decrease as the roll angle ϕ increases. We also need to control the elevator δ_e to maintain the constant height. So in order to achieve coordinated turn maneuver at a constant height, we need to control the aileron δ_a, elevator δ_e and rudder δ_r based on the analysis. The control law between the angle and the control variables can be described as follows.

$$\begin{cases} \delta_a = I_{\phi}\phi + I_{\phi}\phi + I_{\int\phi}\displaystyle\int\left(\phi - \phi_g\right)dt \\ \delta_r = K_{\psi}\psi + K_{\psi}\left(\psi - \psi_g\right) - K_{\beta}\displaystyle\int\beta dt \\ \delta_e = M\left|\phi\right| \end{cases} \tag{24}$$

in which $I_{\dot{\phi}}$, I_{ϕ}, $I_{\int\phi}$, $K_{\dot{\psi}}$, K_{ψ}, K_{β} and M are some related parameters, ϕ_g and ψ_g are the reference angles. In order to alleviate the effect of the sideslip angle β, we add an integral item for the β in the Eq. (24). So if the reference angles ϕ_g and ψ_g are given in the Eq. (24), the UAV can realize the avoidance behavior. Based on the analysis, we choose the angles ψ and ϕ as the output of the network.

The whole algorithm using the actor–critic architecture for UAV avoidance is described as follows.

6. Experiments

6.1. Experiment setup

This paper uses a semi-physical simulation platform to validate the proposed algorithm and the platform as shown in Fig. 5. The platform consists of three main parts: the simulation software, ground control station, and flight control module. The commercial aircraft simulation software *Xplane* provides a variety of aircraft models and can simulate the aircraft's flight environment. In addition, there are different viewpoint patterns in the *Xplane*. This paper adopts the forward view pattern to simulate the airborne forward-looking camera. The captured images are used to extract the obstacles' state vector in the FOV. The ground control station is mainly used for data communication with the UAV simulation platform and the flight control module. Meanwhile, the UAV flight status data and the image sensor data are recorded in the control station. The proposed CNN model is implemented using *Matlab*,

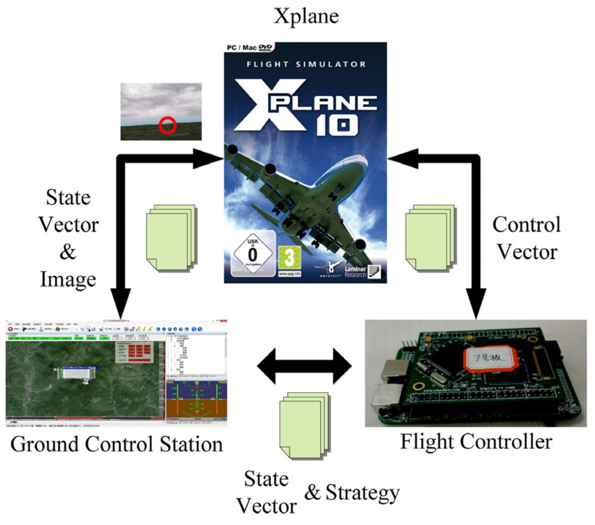


Fig. 5. Semi-physical experimental system: Xplane, ground control station, and flight controller.

which runs on a workstation with 3.3 GHz CPUs, 16 GB memory, a 64-bit Windows server, and additionally a GTX 970 GPU for acceleration. The CNN routine is based on the *matcovnet* library[1] . According to the flight data and the state of the obstacle from the ground station, the flight control module is mainly used for driving the rudder to control the UAV's flight. The hardware-in-the-loop platform also supports the multi-UAV flight platform control task. So we simulate another airplane as the moving obstacle and use the wired network for data communication between the modules.

6.2. CNN-based Saliency detection experiments

To demonstrate the effectiveness of the proposed CNN-based saliency detection method, we compare it with the six state-of-the-art models, including SeR [14], SIM [9], SR [10], SS_LAB [11], SS_RGB [11], and SUN [12]. These selected methods have been proposed in recent years and their codes or calculated saliency maps are publicly available. We select two benchmark databases (MSRA10k and ECSSD) to evaluate the performance of the proposed method. In total, we choose 1000 images from the two databases as training samples in our network. The model is initialized with random weights.

In this part, we will provide saliency detection results on the selected two databases. Similarly, we also use PR-curves and F_β to evaluate the saliency and segmentation performance on two databases. Fig. 6 shows that our model is the best among the tested algorithms based on the PR-curves. Fig. 7 shows the F_β value of the different saliency and segmentation methods, from which we can see that the proposed saliency detection method gives the better F_β value than other algorithms.

Some processed visual comparison results selected from two databases are also given in Fig. 8(a). Our method performs better than other methods in a variety of challenging cases, e.g., against cluttered background, and low contrast between the object and background. On the other hand, we apply the trained CNN on some simulated image (see Fig. 8(b)) from the forward-looking camera. The simulated images are not in the training database. The processing results show that our proposed method is also effective in dealing with the untrained images and has good generalization

1 http://www.vlfeat.org/matconvnet/.

Input: The initial weight factor $v_n(0), \omega_{jn}(0)$ of the critic and the actor network, the learning factor α_c, α_a, the discount factor γ, the central vector c, the standardization constant σ, and the initial state vector s_0 from the saliency detection algorithm;

Output: UAV lateral turn angle $[\psi_t \quad \phi_t]$;

Process:

Step 1. Initialize the parameters;

Step 2. Based on Eq.(5) and Eq.(6), get the obstacle predicted state s_{t+1}^{-} and measured state vector s_{t+1} from the saliency detection algorithm to acquire the predicted state $\hat{s}_{t+1}$, $t = 0$, ;

Step 3. Calculate the output of the critic and actor based on Eq.(10), Eq.(11)and Eq.(12);

Step 4. Calculate the reward value r_{t+1} of the system and temporal difference δ_t based on Eq.(13) and Eq.(14);

Step 5. Respectively update the network weights of the actor and critic based on δ_t; if $\sum_{n=1}^{q}\left|v_n(t)-v_n(t-1)\right| > \tau_v$ and $\sum_{n=1}^{q}\left|\omega_{jn}(t)-\omega_{jn}(t-1)\right| > \tau_\omega$, $t \leftarrow t+1$, apply the output on the UAV and go to **Step 2** ;else break the algorithm ;

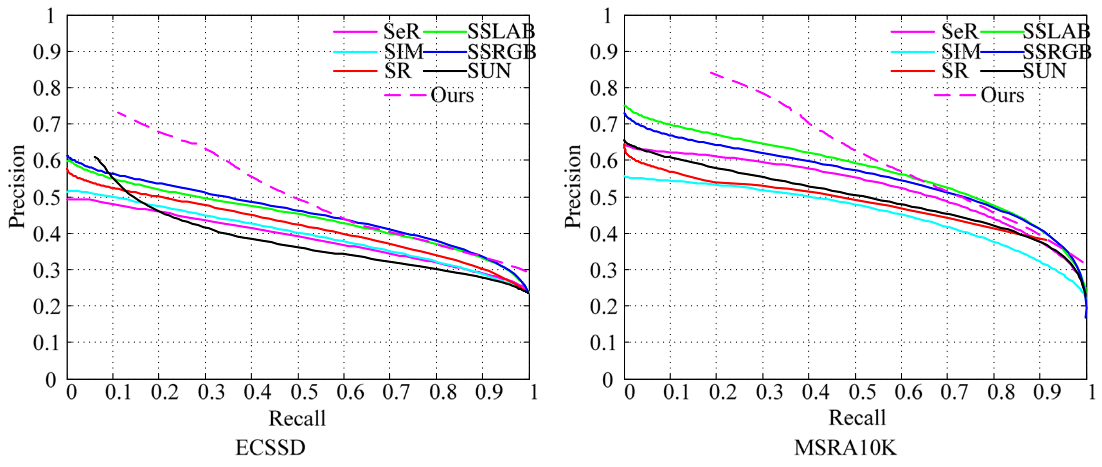


Fig. 6. The comparison results of PR-curves for ECSSD\MSRA10K databases by different saliency detection methods.

ability. Although there are many salient regions in the simulated images, it is easier to detect the another UAV. The salient region of the UAV becomes larger as the distance decreases. And the salient results can be used as the description of the obstacle in the sky.

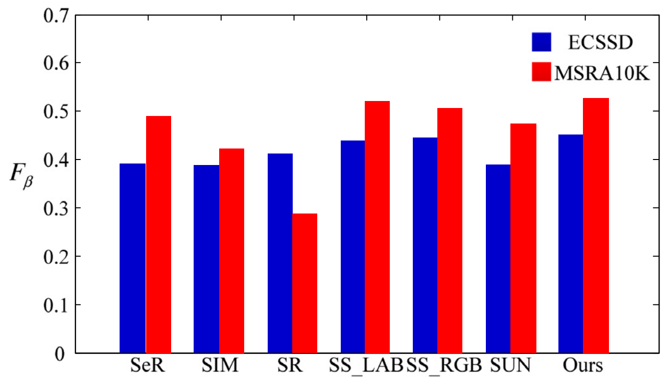


Fig. 7. The comparison results of F_β for ECSSD\MSRA10K databases by different saliency detection methods.

6.3. Simulation experiment

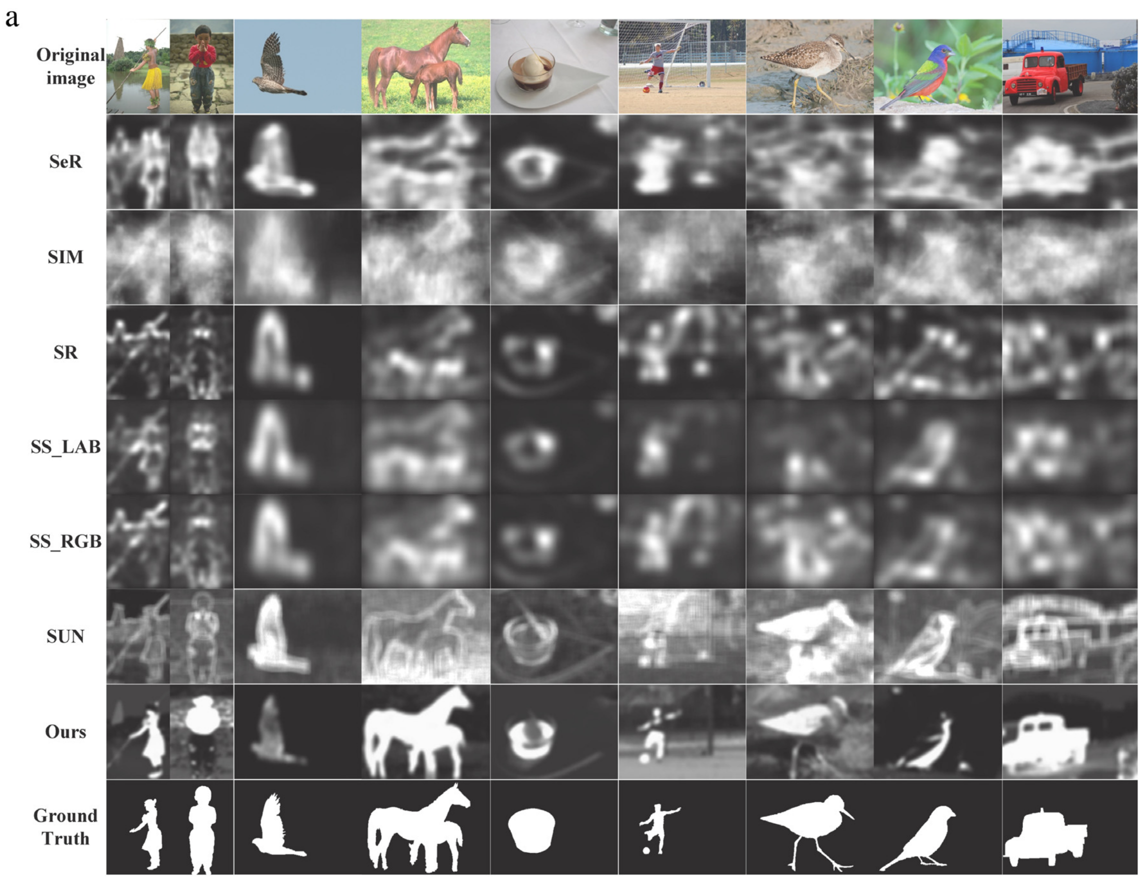
According to the above analysis on the framework, it can be concluded that a precise state is important in the whole framework. And the smooth trajectory of the obstacle object can decrease the control difficulty. However, the detection result may include some noise. So we need to verify the effectiveness of the prediction algorithm using a KF. A simulated moving point with three kinds of trajectories is designed in this paper. We assume that the UAV and obstacles move at constant speeds and the expansion rate $\frac{dO_t}{dt} = \rho$ is a constant value. We use the UAV's own coordinate system as the reference, and the following cases are studied:

CASE 1 Obstacles or threats fly to the positive direction in the center of the UAV's FOV, i.e. the state $s_t = \{0, 0, O_0 + \rho\Delta t\}$;

CASE 2 Obstacles or threats fly to the right or left side at a relative constant height, i.e. the state $s_t = \{u_0 + \mu\Delta t, v, O_0 + \rho\Delta t\}$;

CASE 3 Obstacles or threats fly in random directions towards the UAV, i.e. the state $s_t = \{u_0 + \eta(t)\Delta t, v_0 + \upsilon(t)t, O_0 + \rho\Delta t\}$;

As shown in Fig. 9 for the above three kinds of cases, this paper introduces the state prediction experiments respectively. Given a set of observation experiments with Gaussian noise, the experiments shows the real coordinates change. The initial state of the obstacle that appears in the FOV is $s_0 = \{50, 45, 1\}$. It can be seen from the figures that the predicted state $s_t = \{u_t, v_t, O_t\}$ gradually converges to the reference state. The measured state



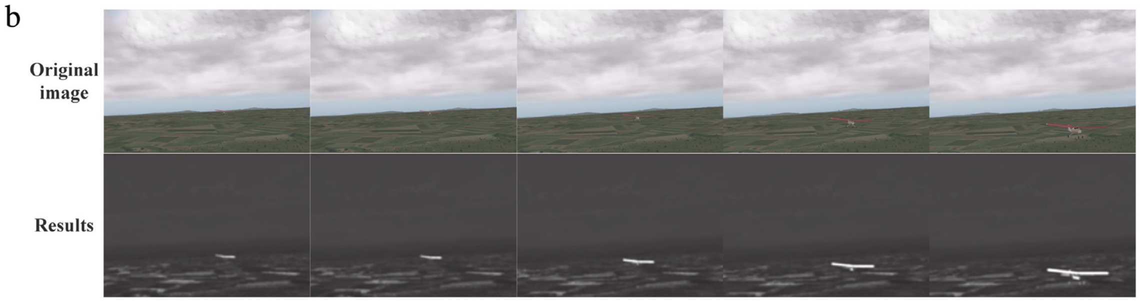


Fig. 8. Visual saliency detection results. (a) results of saliency maps generated from state-of-the-art methods, compared with our CNN-based method. The ground truth is shown in the last row; (b) saliency detection results of the UAV image database using the proposed CNN-based method.

with loud noise has influence on the predicted state by using the KF at the initial running time. Especially, the obstacle moves in the FOV with a nonlinear changing trend in Fig. 9(3-a/b/c). The Fig. 9(3) shows the more regular trend rather than the measured disordered state.

6.4. Semi-physical obstacle avoidance experiment

In this experiment, the semi-physical system provides the UAV flight status and the image of the FOV in the UAV. In order to obtain the training data, the obstacle avoidance process is controlled by a human operator. The recorded UAV attitude data and related images are used for offline training and testing data of the proposed actor–critic RL algorithm. When the obstacle appears in the FOV, we use the saliency detection algorithm to get the state vector of the obstacle. Based on the Kalman filter algorithm, the state vector of the other plane is shown in Fig. 10. The moving obstacle gradually flies towards the UAV with a "right–left– right" moving pattern in the FOV.

Based on the framework of the AC, the algorithm depends on many parameters. In this paper, the parameters are initialized as follows: the learning rates $\alpha_c = 0.01$, $\alpha_a = 0.02$, discount factor $\gamma = 0.1$, and hidden neuron number $q = 5$. As we mainly focus on the algorithm verification offline, the center vector is initialized by the mean value of each input vector and the standardization constant is set by the mean covariance value of the input vector. The output of the network is the yaw and roll angle. Initial angles of the UAV are as follows: roll angle $\phi_0 = -3°$ and yaw angle $\psi_0 = 2.5°$. As shown in Fig. 10, the reference angle is the practical artificial control results to avoid the obstacle and the experimental results are the output of the AC network. It can be seen from the results that the basic network can realize mapping from the image to the control angles. Once the main control angle is obtained based on the algorithm, the rudder control value can be calculated based

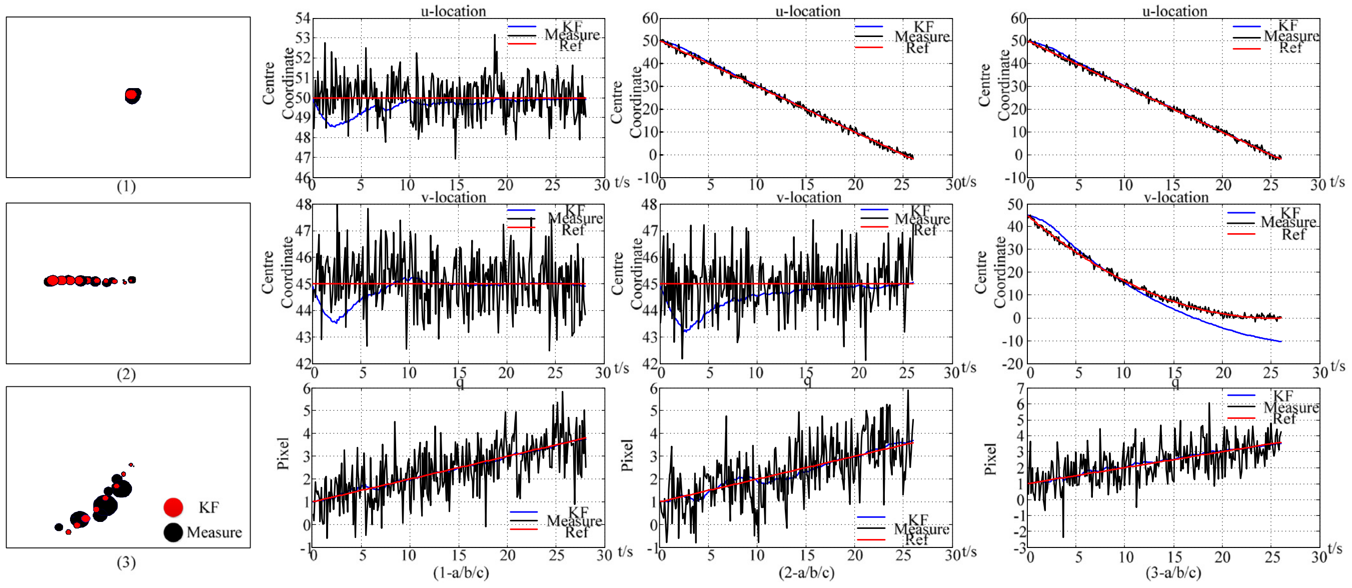


Fig. 9. Obstacle state prediction simulation experiment in three different cases.

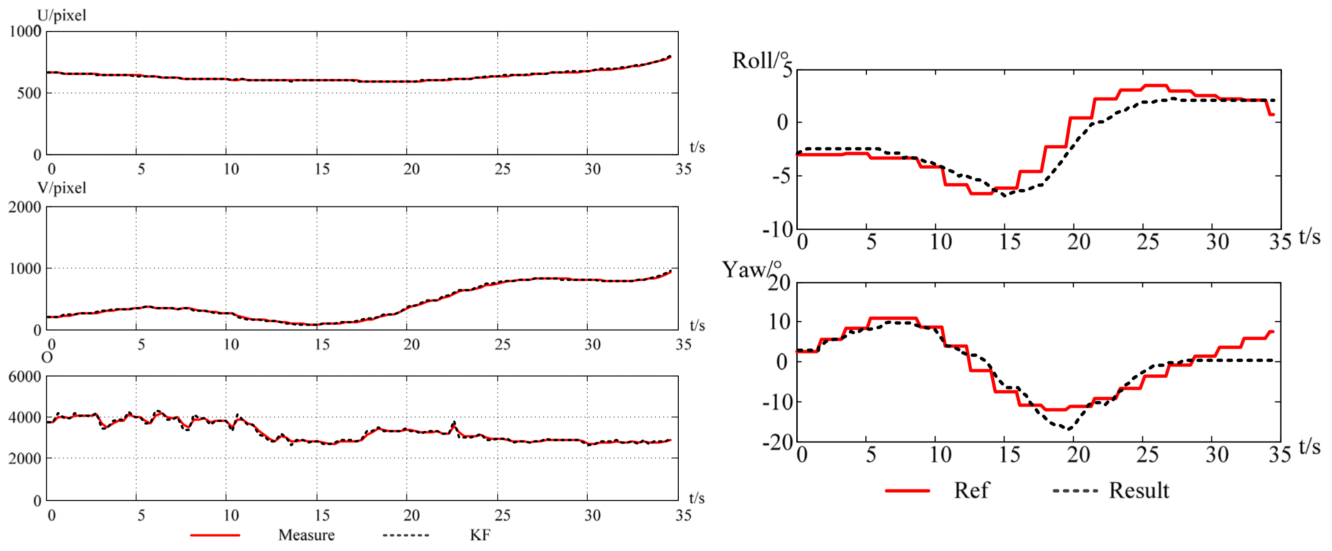


Fig. 10. Obstacle state prediction input of the architecture using the image data from the semi-physical system and the UAV control reference angle output for avoiding the moving obstacle.

on Eq. (24). On the other hand, the reward value defined by Eq. (12) converges in the learning process in Fig. 11.

However, the difference between the changes of the input vector and the output is relatively large, because there are many tuning parameters in this algorithm and the influence of the parameters is coupled with each other. And some parameters are manually tuned. So there are some errors between the artificial control angles and the output of the algorithm.

7. Concluding remarks

Obstacle avoidance is an essential capability for a UAV to fly safely. Due to the high speed movement of the UAV as well as the flying obstacle, we propose a reactive UAV control method that maps from perception to action directly. Considering the changing environment and the payload of the UAV, we propose a learning framework that integrates the object detection and reinforcement learning method. In the framework, we propose the predicted model for the movement process of an obstacle in the FOV and analyze the obstacle avoidance status. In order to acquire the state's description of the obstacle, we propose a saliency detection method based on the deep convolution neural networks. The CNN-based saliency detection method shows higher robustness than the handcrafted features. Based on the saliency detection, we design a reinforcement learning algorithm using an actor–critic module. The RL module chooses the RBF neural network to approximate the state and controller in the continuous spaces. In this way, we construct the mapping from the obstacle state to the UAV's action. In this paper, the effectiveness of the proposed framework has been verified using the hardware-in-the-loop simulation system. The saliency detection method based on deep features is verified and compared with state-of-the-art using the standard databases in the saliency detection area. On the other hand, our database from the semi-physical system also shows better detection results. The

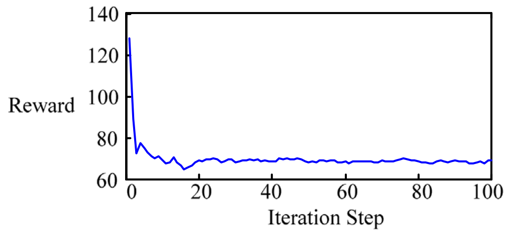


Fig. 11. The reward value in the iterative process.

whole framework is verified by using large numbers of training image data and controlling data of the UAV.

However, there are several limitations in the current work. The whole framework is tested offline by using the collecting data from the semi-physical system. We choose three kinds of cases as a reference for basic research in order to realize the reinforcement learning framework based on vision. However, the three kinds of cases are not specific in reality. More complicated cases need to be analyzed based on the real flight environment. Although the patch-based saliency method increases the accuracy of object detection, it decreases the running time of the whole framework. It is necessary to develop the data online parallel processing technology using GPU acceleration. On the other hand, the framework provides a good way to integrate learning method into processing the high-dimensional raw image data and the continuous controlling signal. However, the chosen deep CNN and RBF neural network require much tuning of parameters in this paper. It is necessary to develop the optimal parameters selection method. With the development of the airborne processing module with light weight and high performance, we believe that it increases the possibility of applying this framework to the UAV.

Acknowledgment

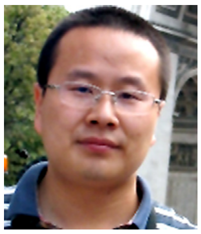
We also gratefully acknowledge the support of the National Natural Science Foundation of China (61403410).

References

[1] R.A. Brooks, Intelligence without representation, Artificial Intelligence 47 (1) (1991) 139–159.
[2] X. Yu, Y. Zhang, Sense and avoid technologies with applications to unmanned aircraft systems: Review and prospects, Prog. Aerosp. Sci. 74 (2015) 152–166.
[3] M. Mujahed, D. Fischer, B. Mertsching, Tangential gap flow (TGF) navigation: A new reactive obstacle avoidance approach for highly cluttered environments, Robot. Auton. Syst. 84 (2016) 15–30.
[4] H. Yu, R. Sharma, R.W. Beard, et al., Observability-based local path planning and obstacle avoidance using bearing-only measurements, Robot. Auton. Syst. 61 (12) (2013) 1392–1405.
[5] W.H. Huang, B.R. Fajen, J.R. Fink, et al., Visual navigation and obstacle avoidance using a steering potential function, Robot. Auton. Syst. 54 (4) (2006) 288–299.
[6] J.C. Zufferey, A. Beyeler, D. Floreano, Autonomous flight at low altitude with vision-based collision avoidance and GPS-based path following, IEEE International Conference on Robotics and Automation, ICRA, 2010, pp. 3329–3334.
[7] P. Fallavollita, F. Cimini, M. Balsi, et al., A new bio-inspired decision chain for UAV sense-and-avoid applications, in: International Society for Optics and Photonics, 2012, pp. 84541Z–84541Z-8.
[8] H. Oleynikova, D. Honegger, M. Pollefeys, Reactive avoidance using embedded stereo vision for MAV flight, in: IEEE International Conference on Robotics and Automation, ICRA, 2015, pp. 50–56.
[9] H.J. Seo, P. Milanfar, Static and space–time visual saliency detection by self-resemblance, J. Vis. 9 (12) (2009) 15–15.
[10] N. Murray, M. Vanrell, X. Otazu, et al., Saliency estimation using a non-parametric low-level vision model, in: IEEE Conference on omputer Vision and Pattern Recognition CVPR, 2011, pp. 433–440.
[11] X. Hou, L. Zhang, Saliency detection: A spectral residual approach, IEEE Conference on Computer Vision and Pattern Recognition, 2007, pp. 1–8.
[12] X. Hou, J. Harel, C. Koch, Image signature: Highlighting sparse salient regions, IEEE Trans. Pattern Anal. Mach. Intell. 34 (1) (2012) 194–201.
[13] L. Zhang, M.H. Tong, T.K. Marks, et al., SUN: A Bayesian framework for saliency using natural statistics, J. Vis. 8 (7) (2008) 32–32.
[14] L. Itti, C. Koch, E. Niebur, A model of saliency-based visual attention for rapid scene analysis, TPAMI 20 (11) (1998) 1254–1259.
[15] Eiji Uchibe, et al., Vision-based reinforcement learning for robocup: Towards real robot competition, in: Proceeding of IROS, vol. 96, 1996.
[16] M. Asada, E. Uchibe, K. Hosoda, Cooperative behavior acquisition for mobile robots in dynamically changing real worlds via vision-based reinforcement learning and development, Artificial Intelligence 110 (2) (1999) 275–292.
[17] M. Asada, S. Noda, S. Tawaratsumida, et al., Vision-based reinforcement learning for purposive behavior acquisition, in: IEEE International Conference on Robotics and Automation, vol. 1, 1995, pp. 146–153.
[18] M. Asada, E. Uchibe, S. Noda, et al., Coordination of multiple behaviors acquired by a vision-based reinforcement learning, in: Proceedings of the IEEE/RSJ/GI International Conference on IROS, vol. 2, 1994, pp. 917–924.
[19] S. Wen, X. Chen, C. Ma, et al., The Q-learning obstacle avoidance algorithm based on EKF-SLAM for NAO autonomous walking under unknown environments, Robot. Auton. Syst. 72 (2015) 29–36.
[20] J. Koutník, J. Schmidhuber, F. Gomez, Evolving deep unsupervised convolutional networks for vision-based reinforcement learning, in: Proceedings of the 2014 conference on Genetic and Evolutionary Computation, 2014, pp. 541–548.
[21] G. Cuccu, M. Luciw, J. Schmidhuber, et al., Intrinsically motivated neuroevolution for vision-based reinforcement learning, in: IEEE International Conference on Development and Learning, vol. 2, 2011, pp. 1–7.
[22] K. Shibata, M. Iida, Acquisition of box pushing by direct-vision-based reinforcement learning, in: IEEE Annual Conference on SICE, vol. 3, 2003, pp. 2322–2327.
[23] C. Wang, K.V. Hindriks, R. Babuska, Active learning of affordances for robot use of household objects, in: 14th IEEE-RAS International Conference on Humanoid Robots (Humanoids), 2014, pp. 566–572.
[24] T.P. Lillicrap, J.J. Hunt, A. Pritzel, Continuous control with deep reinforcement learning, Under review as a conference paper at ICLR, 2016.
[25] US Department of Transportation, Airplane Flying Handbook, FAA-H-8083-3A, 2004.
[26] A. Krizhevsky, I. Sutskever, G.E. Hinton, Imagenet classification with deep convolutional neural networks, in: Advances in Neural Information Processing Systems, 2012, pp. 1097–1105.

Zhaowei Ma is currently pursuing his Ph.D. degree in the control science and engineering from National University of Defense Technology. He received his B.S. degree (outstanding graduates) in automation from Xidian University, Xian, China, in 2011. In 2013, he got the M.S. degree in control science and engineering from National University of Defense Technology, Changsha, China. His research interests are in the fields of deep learning and reinforcement learning.

Chang Wang is now a lecturer in the College of Mechatronic Engineering and Automation at National University of Defense Technology. He received the B.S. and M.S. degrees from National University of Defense Technology, China, in 2007 and 2009. He received the Ph.D. degree from the Delft University of Technology in 2015. His research interests include learning control and robotic systems.


Yifeng Niu (S'07-M'10) received the Ph.D. degree in robotics and automation from the National University of Defense Technology (NUDT), Changsha, China, in 2007. He now is an associate professor of the College of Mechatronic Engineering and Automation. He has published more than 40 technical papers in refereed international journals and academic conference proceedings. His research interests include image processing, learning control, and artificial intelligence.

Xiangke Wang received the B.S., M.S., and Ph.D. degrees from National University of Defense Technology, China, in 2004, 2006 and 2012, respectively. From 2012, he is with the College of Mechatronic Engineering and Automation, National University of Defense Technology, China, and currently he is an associate professor. He was a visiting Ph.D. student at the Research School of Engineering, Australian National University, supported by the China Scholarship Council from September 2009 to September 2011. His current research interests include coordination control of multiple UAVs, nonlinear control and robot soccer.

Lincheng Shen received the B.E., M.S., and Ph.D. degrees in automatic control from the National University of Defense Technology (NUDT), Changsha, China, in 1986, 1989, and 1994, respectively. In 1989, he joined the Department of Automatic Control, NUDT, where he is currently a Full Professor and serves as the Dean of the College of Mechatronic Engineering and Automation. He has published more than 100 technical papers in refereed international journals and academic conferences proceedings. His research interests include mission planning, autonomous and cooperative control for unmanned systems, biomimetic robotics, and intelligent control.