Research Article

Low-Shot Wall Defect Detection for Autonomous Decoration Robots Using Deep Reinforcement Learning

Fanyu Zeng,¹ Xi Cai,¹ and Shuzhi Sam Ge¹,²

¹School of Computer Science and Engineering, Center for Robotics, University of Electronic Science and Technology of China, Chengdu 611731, China
²Department of Electrical Computer Engineering, National University of Singapore, Singapore 119077, Singapore

Correspondence should be addressed to Fanyu Zeng; zengfanyu.cs@gmail.com

Received 6 June 2020; Revised 4 August 2020; Accepted 28 August 2020; Published 21 September 2020

1. Introduction

Autonomous decoration robots are increasingly applied in the field of house decoration. Figure 1 shows our robot platform, autonomous decoration robot, which is used to decorate the walls of rough houses. The first thing of wall decoration for an autonomous decoration robot is wall defect detection.

Wall defect detection is an important function for autonomous decoration robots. Object detection methods based on deep neural networks require a large number of images with the handcrafted bounding box for training. Nonetheless, building large datasets manually is impractical, which is time-consuming and labor-intensive. In this work, we solve this issue to propose the low-shot wall defect detection algorithm using deep reinforcement learning (DRL) for autonomous decoration robots. Our algorithm first utilizes the attention proposal network (APN) to generate attention regions and applies AlexNet to extract the features of attention patches to further reduce computation. Finally, we train our method with deep reinforcement learning to learn the optimal detection policy. The experiments are implemented on a low-shot dataset in which images are collected from real decoration environments, and the experimental results show the proposed method can achieve fast convergence and learn the optimal detection policy for wall defect images.
AlexNet [13] to generate a compressed feature vector. Furthermore, we feed the feature vector into LSTM that outputs a center location of the wall defect image. In the subsequent iterations, our method continuously improves the accuracy of detection location based on the previous center location. Figure 2 shows some image samples of wall defects.

The contributions in our work can be summarized as follows:

1. Defect detection via DL requires a ground-truth dataset with the handcrafted bounding box. However, our method does not require manual annotations for wall defect images.
2. Our method extends deep reinforcement learning from classification tasks to detection tasks.

The remainder of this paper is organized as follows: in Section 2, we present the background. Then, in Section 3, the proposed method is described in detail. In Section 4, the experimental results are presented and discussed, which demonstrate the advantage of our method. Finally, the conclusions are presented in Section 5.

2. Background

2.1. Deep Reinforcement Learning. The aim of reinforcement learning is to maximize a discounted sum of rewards when an agent interacts with an environment over a number of discrete time steps [14, 15]. At each time step, the agent receives a state $s_t$ from the environment and produces an action $a_t$ according to its learned policy $\pi$. In return, the environment gives the agent a next state $s_{t+1}$ and a reward $r_{t+1}$. Reinforcement learning can be divided into three categories [14]: value-based methods, policy-based methods, and actor-critic methods. Among these categories, policy gradient methods [16] are used to compute an estimator of agent’s policy gradient by a stochastic gradient ascent algorithm and are suitable to incorporate with the deep neural network. Works on policy gradient methods have been developed, such as actor-critic (AC) [14, 17], in which the actor is a policy, and the critic is a baseline. Lillicrap et al. [18] extended DQN [15] and DPG [19] to propose deep deterministic policy gradient (DDPG), and DDPG based on AC includes 4 neural networks: current critic network, current actor network, target critic network, and target actor network. Mnih et al. [20] proposed asynchronous advantage actor-critic (A3C) that uses asynchronous training of multiple agents in parallel. In recent years, DRL algorithms have been applied to robotics [21–23].

2.2. Recurrent Attention Model. Recurrent attention model (RAM) [24] is a novel visual attention model formulated as a single recurrent neural network. This visual attention model takes a glimpse window as the input and uses an internal state of the neural network to select the next detection location and to generate control signals in a dynamic environment. Although RAM is not differentiable, the unified architecture is end-to-end from pixel inputs to actions using a policy gradient method.

2.3. Attention Proposal Network. APN [12] receives full image and iteratively generates attention regions from coarse to fine by taking the previous prediction as a reference, while a finer scale network takes as input an amplified attention region from the previous scales in a recurrent way. The learning process of the APN is trained in a weakly supervised fashion because a part-level annotation is hard to obtain. Figure 3 shows the APN architecture that consists of a pretrained VGG-19 [25] model and two-stacked fully connected layers. In addition, pretrained VGG-19 is trained on ImageNet.

3. Method

In this work, we regard the wall defect detection problem as a sequential decision process in which a goal-directed agent interacts with the environment. Our architecture, as shown in Figure 4, can be decomposed into two modules: initialization module and refinement module. The initialization module is responsible for obtaining an initial detection location that is a preliminary input for the second module. To further prove the effective detection of our method, our proposed method also classifies the wall defect images into convex ones and concave ones based on the detection results. With several recurrent iterations, the refinement module gradually refines the results of the wall defect detection.

In the initialization module, we feed an input image into the pretrained APN model to generate the initial attention region. This initial attention region becomes small and reduces computation significantly. Then, AlexNet compresses the high-dimension attention region into low-dimension feature vectors. Finally, the agent, via the policy gradient, receives the feature vectors and outputs an initial detection and classification policy for the wall defect image. The purpose of the initialization module is to calculate rough center coordinates of detection, and the refinement module receives the initial detection coordinates to implement recurrent iterations for detection improvement.

3.1. Detection Initialization. The initialization module outputs initial detection. We utilize the pretrained APN to predict a box coordinate of an attention region for a finer scale. At each step $t$, an original image $I_t$ is fed into the APN, and the APN outputs an attention region $x_t$. The representation of the attention region $x_t$ can be expressed as follows:

$$f_p(I_t) = [t_x, t_y, t_l] = l_{t-1},$$

where $t_x, t_y$ are the square’s center coordinates with respect to $x$- and $y$-axis and $t_l$ is the distance between the detection center location and its border, respectively. For ease of notation, we rewrite the location $[t_x, t_y, t_l]$ as $l_{t-1}$. $f_p$ is the neural network architecture of the APN.
To further reduce the network computation of our method, we compress image \( x_t \) to extract features by convolutional neural networks. Compared with other frequently used CNNs, such as VGG and ResNet, AlexNet has much less training parameters. Therefore, we use AlexNet \( f_n \) to extract features from \( x_t \), and \( f_n \) outputs feature vectors \( n_t = f_n(x_t) = f_n(f_p(I_t)) \) with much lower dimensionality.

Feature vectors \( n_t \) are fed into LSTM, and the input parameters of inner LSTM are shown in Figure 5. In the inner of LSTM, each hidden unit \( h_t \) has an internal state which summarizes information from environment states.

During the interaction period, the update of each hidden unit is \( h_t = f_h(h_{t-1}, n_t; \theta_h) \).

Similar to [24], as shown in Figure 6, the agent outputs a location action \( l_t \) and an environment action \( a_t \) using the internal state \( h_t \) when it interacts with the environment. In this paper, the location network \( f_l(h_t; \theta_l) \) outputs \( l_t \) for the next time step, and the environment network outputs the environment action \( a_t \) after a fixed number of time steps.

After performing an action \( a_t \), the agent receives a new visual observation \( I_{t+1} \) and a reward signal \( r_{t+1} \) from the environment. Wall defect images would be categorized into...
convex and concave types, and the reward is 1 if the agent classifies the wall defect image correctly; otherwise, the reward is set to 0. In addition, the aim of our agent is to maximize the accumulated rewards

\[ R = \sum_{t=1}^{T} r_t \]

3.2. Detection Refinement. The refinement module outputs the final detection and classification results. When the initialization module generates an initial detection location \( l_t \), we can get a new attention region \( x'_t \) around \( l_t \), and the region is fed into AlexNet for a compressed feature vector \( n'_t = f_n(x'_t) \). The LSTM agent receives \( n'_t \) and outputs classification policy \( a_{t+1} \) and detection policy \( l_{t+1} \) which generate an initial detection coordinate for its next iteration. After \( K \) recurrent iterations, the refinement module can learn optimal classification and detection policy. Furthermore, the classification policy is used to categorize wall defect images into a convex defect or concave defect, and the detection policy is used to generate a center coordinate for the final detection region.

3.3. Training Method. Similar with RAM [24], our architecture has three small neural networks: a glimpse network, a location network, and an environment network. The training goal is to learn a policy that maximizes the total rewards.

\[
J(\theta) = \mathbb{E}_p(s_t, \theta) \left[ \sum_{t=1}^{T} r_t \right] = \mathbb{E}_p(s_t, \theta)[R],
\]  

\( J(\theta) \)

\( J(\theta) \)
where $p(s_{1:T}; \theta)$ depends on the policy. It is not easy to maximize $J$ because it involves the expectation of high-dimensional relation sequence. We can regard this problem as a partially observable Markov decision process (POMDP). However, it allows us to solve this problem from the technical perspective of RL, and a sample approximation method is used to approximate the gradient:

$$
\nabla_{\theta}J \approx \frac{1}{M} \sum_{t=1}^{T} \sum_{i=1}^{M} \nabla_{\theta} \log \pi(u'_{i,t} | s'_{i,1:T}; \theta) R_{i}. \tag{3}
$$

3.4 Neural Network Architecture. As shown in Figure 4, the initialization module is a pretrained APN which includes pretrained VGG-19 layers on ImageNet and two fully connected layers. The APN parameters are frozen, and the APN outputs a $200 \times 200 \times 3$ attention region from a $640 \times 480 \times 3$ RGB defect image. In addition, we feed the attention region into AlexNet to generate a 4000-d feature vector, and the vector is further passed through 512-LSTM to produce an initial classification policy and a center coordinate of initial detection. The refinement module has a similar neural network architecture as the initialization module except the APN.

4. Experiments

Experiments are implemented in PyTorch and trained on Nvidia GeForce GTX 1080Ti GPU. We evaluate our method on the low-shot dataset of the wall defect images collected from real decoration environments.

4.1 Experimental Configuration. We collected 317 RGB images for a low-shot dataset from real decoration environments by our autonomous decoration robot and split the images into the training dataset with 254 images and test dataset with 63 images, respectively. For ease of training, all the images are resized into the same size $640 \times 480$, and the distance between the detection center and its border $t_{j}$ is set to 100. Therefore, the size of the attention region from the APN is $200 \times 200$. In addition, proper $K$ recurrence is set to 2 for high accuracy and short training time. We trained this neural network based on a shared RMSProp optimizer with learning rate $7 \times 10^{-4}$.

4.2 Experimental Results and Analysis. We took five wall defect images as examples and gave qualitative and quantitative analysis to demonstrate the effectiveness of our method. As shown in Figure 7, the first column is original images, the second column is initial wall defect detections, and the third column is refined detections. Figure 7 represents the qualitative experimental results. Each original image is fed into the proposed neural network, and the first column is the output of the APN which
generates the initial detection center. Though several images (Img-000, Img-082, Img-167, and Img-288) output the rough detection image, the refinement module still generates more accurate detections after recurrent iterations. In addition, APN even gives wrong initialization, such as Img-237, whose initial detection does not include the wall defect, and our method also maintains correct detection. The qualitative experimental results show the agent can learn optimal detection policy via the low-shot dataset.

To further prove the effective detection of our method, we classified images of wall defect after detection into convex ones and concave ones. Figure 8 shows the quantitative experimental results. The classification loss decreases rapidly, as shown in Figure 8(a), and it fast reaches stability with low value at 200 episodes. In Figure 8(b), the classification accuracy curve reaches 90% when the agent is trained at about 100 episodes. The training process begins to achieve convergence at 150 episodes, and the average classification accuracy is 98.25% that is calculated between 150 episodes and the end episode. The quantitative experimental results show that our agent can learn the optimal wall defect classification policy via the low-shot dataset, which further proves the effectiveness of our detection method.

According to the qualitative and quantitative analysis, training of our method can achieve fast convergence using deep reinforcement learning via the low-shot dataset. Moreover, the agent can learn optimal detection, while the training dataset is low-shot. In addition, the trained parameter model is 452M which is small, and our method is real-time in practice.

5. Conclusion

Wall defect detection is an important function for autonomous decoration robots. However, object detection methods via deep learning require a large number of image datasets annotated with the ground-truth bounding box. It is impractical to collect enough images and to handcraft bounding boxes for wall defect detection from real decoration environments. To address this issue, we proposed low-shot wall defect detection using deep reinforcement learning in this paper. We first utilized the attention proposal network to generate attention regions and applied AlexNet to extract features of attention patches to reduce the computation. Then, we used deep reinforcement learning to train the proposed method successfully. The proposed method can reach fast convergence via the low-shot dataset and learn optimal detection policy for wall defect images for autonomous decoration robots.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This research was funded by the National Natural Science Foundation of China (U1813202 and 61773093), National Key R&D Program of China (2018YFC0831800), Research Programs of Sichuan Science and Technology Department (17ZDYF3184), and Important Science and Technology Innovation Projects in Chengdu182 (2018-YF08-00039-GX).

References


