An Environmental Characterization Method for Monocular Vision Obstacle of Rotor Aircraft

Li Biao¹, Wu Wenzhen¹, Yang Xiaohui², Ji Desheng³, Lu Xiaoyu⁴, Cao Yasheng³, Wang Xi⁵a*
¹China Coal Science and Technology Research Institute Co., Ltd., Dongcheng District, Beijing, 100013, China
²China Coal Research Institute, Dongcheng District, Beijing, 100013, China
³Beijing Aerospace Xinfeng Machinery Equipment Co., Ltd., No.52 Yongding Road, Haidian District, Beijing, 100039, China
⁴Department of information and communication, Army Armored Forces Academy, Fengtai District, Beijing, 100072, China
⁵xi-wang16@mails.tsinghua.edu.cn  *13810600312@126.com

Abstract. Combining the structure of coding and anti-coding in the field of deep learning, we explore a method to represent the complex environment of four rotor aircraft. First, we use this structure to explore the process of dimensionality reduction and reconstruction of high-dimensional image data. Second, combining this structure, we use the concept of transposition convolution to propose a network structure for depth distance estimation using monocular vision images. Finally, we get the direct physical feature information used to describe complex environment.

1. Introduction
It’s difficulty to use simple target detection method to achieve environmental characterization, while high-dimensional image input cannot be used as a direct description of the state of the environment. High-dimensional data have more information than low dimensional data. However, the direct operation of high-dimensional data in practical applications will lead to a series of problems, and high-dimensional data will not only increase the computational complexity[1-5]. Moreover, redundant information will bring corresponding errors. For learning problems, high data and redundant information will reduce learning rate. Therefore, it is necessary to extract some key direct feature information that can represent the near earth environment information from high-dimensional data, realize dimensionality reduction from high-dimensional data to low dimensional data, reduce the dimension representation ability of environmental representation, but without losing its representational ability[4,6-8].

2. Our approach
We combine the structure of encoder and decoder in the field of deep learning to explore a representation method suitable for the complex environment of four rotor airborne.
2.1. Encoder and decoder structure

Automatic encoder (Autoencoder) [10] is an unsupervised learning algorithm. It is a three layer basic network structure. For network input, the purpose of the automatic encoder is to obtain an identical wBF x function, so that the output of the function is close to the input, as shown in Figure 1.

![Figure 1. Schematic diagram of automatic encoder network structure.](image)

Suppose that the number of samples is \( n \) Sample set\( \{(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)\} \). For a single example\( (x, y) \). The cost function is

\[
J(W, b, x, y) = \frac{1}{2} |f_{W,b}(x) - y|^2
\]

and the cost function for the whole sample set is:

\[
J(W, b) = \frac{1}{n} \sum_{i=1}^{n} J(W, b, x, y) + \frac{\lambda}{2} \sum_{l=1}^{s_l} \sum_{j=1}^{s_{l+1}} (W_{ij}^{(l)})^2
\]

(1)

The first item is the mean square deviation and the second is the regularization term. The purpose is to prevent the over fitting of the algorithm, which is mainly used for the relative proportion of the two errors before and after the equalization. After obtaining the reconstruction error of the encoder, the purpose of the backpropagation algorithm is to use the cost error, \( J(W, b) \) Update network weight \( W \) And offset \( b \). First, we need to take every parameter. \( W_{ij}^{(l)} \) and \( b_i^{(l)} \). Initialization is a very small random value approaching 0, and then update the weight by iteration of gradient descent method. \( \alpha \) is the learning efficiency.

\[
W_{ij}^{(l)} = W_{ij}^{(l)} - \alpha \frac{\partial}{\partial W_{ij}^{(l)}} J(W, b)
\]

(2)

\[
b_i^{(l)} = b_i^{(l)} - \alpha \frac{\partial}{\partial b_i^{(l)}} J(W, b)
\]

(3)

According to the basic concept of automatic encoder, we use the automatic encoder to construct the deep self-coding network.

2.2. Transpose convolution

Transposition convolution usually contains many volumes of CNN[11], followed by the pooled layer for reducing the input dimension. According to the description of the general convolutional network, the low dimensional feature extraction of high-dimensional vectors is realized through the convolution and the down sampling process. However, in some constructive deep learning problems, such as depth estimation, semantic segmentation, etc.

The final result often needs to preserve spatial information and reconstruct the low dimensional vector to the high-dimensional vector. When the input vector dimension is lower than the output vector dimension, the neural network is equivalent to a reverse encoder. For convolution networks, we need to
consider how to realize the reverse operation of the convolution network. According to the definition of convolution operation, if we consider only the operation of one dimension, the input of the convolution layer is \( I \in \mathbb{R}^{I \times H \times W} \). The filter convolution kernel is defined as \( k \in \mathbb{R}^{K \times H \times W} \). Output is \( y \in \mathbb{R}^{O \times H' \times W'} \). After the convolution operator, the output is:

\[
y_{i'j'} = \sum_{i''j''} k_{i''j'j''} x_{i''j''} + b_{i''j''}
\]

Make input as \( I \in \mathbb{R}^{U \times H \times W} \), then the output is convoluted.

\[
y_{u \times H' \times W'} = C_{u \times H' \times W'} y_{u \times H' \times W'}
\]

\( C_{u \times H' \times W'} \) is a sparse matrix. For example, \( I \in \mathbb{R}^{1 \times H \times W} \), and the convolution kernel is \( k \in \mathbb{R}^{3 \times 3} \), then we get:

\[
y_{4 \times 1} = C_{4 \times 16} y_{16 \times 1}
\]

The backpropagation operation of convolution networks is defined as:

\[
\frac{\partial \text{Loss}}{\partial l} = \sum_i \frac{\partial \text{Loss}}{\partial y_i} l = \sum_i \frac{\partial \text{Loss}}{\partial y_i} C_{i,j} = \frac{\partial \text{Loss}}{\partial y} C_{*,j} = C^{T}_{*,j} \frac{\partial \text{Loss}}{\partial y}
\]

According to mathematical description, deconvolution is easy to cause misunderstanding. Transposed Convolution is a more suitable method. Transposition convolution general process is as follows: first, the input is anti pooled and then applied to convolution operation. Reverse pool is used to extend the input by tracking the location information of the activation value, usually 0, so as to get a sparse picture. The convolution operation is done again, as shown in Figure 2.

![Figure 2. convolution and transposition convolution sketch map, (1) convolution process (2) transposition convolution process.](image)

### 3. Design of algorithm

#### 3.1. Design of deep encoder and decoder network

According to the basic concept of automatic encoders, we can use the automatic encoders to construct deep coding network. The essence of ordinary automatic encoders is to learn an equal function, that is, the output of input and reconstruction is equal. The disadvantage of the representation of this equality function is that when the test samples and training samples do not conform to a contract distribution, that is, the difference is large, and the effect is not good. The processing of deep autoencoder (DAE) has made some progress in this aspect. The algorithm process of deep encoder includes two parts: greedy training by layer by layer hidden layer and training with the whole network optimization.

Layer by layer single hidden layer greedy training. The training stage mainly adopts the traditional Autoencoder training method. The main idea of the greedy algorithm is to train one layer in the network at a time, that is to say, first training a network containing only one hidden layer. Only after the end of the network training is it started to train a network with two hidden layers, and so on. \( l - 1 \) Layer is fixed and then added. The design of the network structure is the key to the training process. The training process mainly realizes the initialization of the DAE network structure. The network optimization
training process, also known as the Finetuning process, mainly optimizes a complete network structure, and its training process can be either an unsupervised process or a supervised process.

In summary, the network structure training algorithm mainly consists of three main steps: network initialization (including initialization of network structure parameters and input training data set), pre-training of deep network structure, and optimization of deep network structure. The deep network constructed by automatic encoder is shown in Figure 3, which is a multilayer neural network structure, usually odd layer structure. Total network structure \(2L - 1\) Layer. The network is a symmetrical structure. They are called No.\(i\) number of nodes in the symmetric layer is the same, but the weight parameters of nodes are not necessarily the same.

![Figure 3. Deep network constructed by automatic encoder](image)

3.2. Algorithm of deep coding anti-coding network

According to the main three steps of the deep coding network, we get the framework of the depth coding algorithm as follows:

Input: original size library \(I \in \mathbb{R}^{H \times W \times D}\) Test image \(I_{test} \in \mathbb{R}^{H \times W \times D}\) Network layer \(n\) Symmetric network structure \([n_1 \ n_2 \ ... \ n_l \ ... \ n_2 \ n_1]\) among \(n_l, l = 1, \cdots, L\). For the first time \(l\) number of network nodes is in the layer.

The learning factor is \(\alpha\) Training error \(\varepsilon\) Maximum iteration steps \(T_{max}\) Activation function \(\sigma(\cdot)\)

Output: coding anti coding network;

Algorithm flow:

- Network Initialization
  - Step 1.1 random initialization of weight coefficients of each layer \(W_0, b_i\);
  - Step 1.2 data preprocessing: input data \(I \in \mathbb{R}^{H \times W \times D}\) Do normalization.

- Network Pre-training
  - Step 2.1 makes the input and output value of the first level network as follows: \(A^1_n = I^1_n, n = 1, \cdots, N\)
  - While \(e_{train} \leq \varepsilon\) or \(T_{train} \leq T_{max}\)
    - Step 2.2 for the first \(l\) Layer network input value \(A^l_n\) Forward propagation operation: \(\tilde{A}^l_n = \sigma(\sigma(A^l_n * W^l_{en} + b^l_{en}) * W^l_{den} + b^l_{den})\);
    - Step 2.3 calculate reconfiguration error \(e_{train}\) Back propagation operation, update/layer weight \(b^l_{en}, W^l_{den}, b^l_{den}\)
      - Step 2.4 \(l = l + 1\);
      - Step 2.5 update/layer network input output value \(l = l + 1\);

- Network Optimization Training
  - While \(e_{train} \leq \varepsilon\) or \(T_{train} \leq T_{max}\)
Step 3.1 gives input values for the entire pre-training network. $A^l_n$. Forward propagation operation is obtained.

Step 3.2 calculate reconfiguration error $E_{\text{train}}$. Back propagation operation, update $W_{\text{en}}^l$, $b_{\text{en}}^l$, and $W_{\text{den}}^l$, $b_{\text{den}}^l$.

End

From algorithm, we can see that based on the traditional neural network, with the whole picture as input, each pixel as a single neuron node, the network structure is relatively simple and easy to implement, but it does not take into account the spatial relationship of adjacent pixels, and ignores the local details. The characteristics of the central pixel can be represented by the characteristics of the surrounding pixels.

4. Experimental and results

4.1. Database and preprocessing

This section uses the general database NYU depth dataset V2 [12, 13] in depth distance estimation. The dataset contains 2284 images. The original data in the database is a picture of 640*480, and the depth image is also a picture of 640*480. Since the output image of the network is a picture of 79*59, we need to prune the database pictures. In this paper, we use direct down-sampling to reduce the size of images to get training data.

4.2. Experimental platform

In order to facilitate development, we chose TensorFlow as the learning and training platform of depth estimation network. TensorFlow is for machine learning and deep neural network research, but the versatility of the system makes it widely used in other computing areas. TensorFlow is a data flow graph. The open source software library for numerical computation. The experimental verification conditions are: the computer is configured as Intel Core i7 processor, and the graphics card is Nvidia Geforce RTX2080.

4.3. Algorithm comparison index

In order to compare the network structure proposed in this paper with the published algorithm, we define several common indicators in the field of depth distance estimation as follows:

4.3.1. Average Relative Loss. It describes the dispersion degree of the corresponding sample mean sampling distribution and measures the sampling error size of the corresponding sample relative.

$$REL = \frac{1}{\text{Num}} \sum_i \frac{|y_i^* - y_i|}{y_i^*}$$

(8)

4.3.2. Root Mean Square. It is a commonly used measure of the difference between measured values, and its value is usually the quantity predicted by the model or the estimated quantity observed. The root mean square deviation represents the sample standard deviation of the difference between the predicted value and the observed value.

$$RMS = \sqrt{\frac{1}{\text{Num}} \sum_i (y_i^* - y_i)^2}$$

(9)

4.3.3. Logarithmic spatial mean error. The vision of man responds to the logarithm of image lightness.

$$\text{Log} = \frac{1}{\text{Num}} \sum_i | \log y_i^* - \log y_i |$$

(10)

4.3.4. Accuracy of a threshold. The proportion of pixel value to the total number of pixels.

$$\max \left( \frac{y_i^*}{y_i}, \frac{y_i}{y_i^*} \right) = \delta < \text{threshold}$$

(11)
Where $y_i^*$ and $y_i$ are the true depth value and predicted depth value of pixel I respectively, and $\delta$ is the proportion of the number of pixels in all test images.

4.4. Results
We use the NYU dataset to estimate the depth distance of two different scale networks respectively. The maximum training step of each training network is 300 steps, each step has 1000 batch, and the following training error curve results are obtained. When the two networks are trained, we first train the single scale network, and apply the trained single scale convolution network structure to the multi-scale network. The random initialization of parameters in all network structures is avoided. When training multi-scale networks, the multi-scale network is trained on the basis of the original single scale network, and in multi-scale network training process.

Table 1. Comparison results of depth estimation indices for NYU datasets

| Index | Karsch | Ladicky | David  | Algorithm in this paper |
|-------|--------|---------|--------|-------------------------|
|       | Coarse | Coarse+fine | Coarse | Coarse+fine |
| REL   | 0.350  | 0.332   | 0.228  | 0.215     | 0.212     |
| RMS   | 0.232  | 0.838   | 0.871  | 0.907     | 0.863     |
| Log   | 0.289  | 0.239   | 0.383  | 0.285     | 0.257     |
| $\delta$ | 0.598  | 0.542   | 0.619  | 0.611     | 0.689     |

The original single scale network structure parameters remain unchanged. The output of the single scale network is one dimensional input of the multi-scale network. It only needs to iteratively update the increased network structure. Our results show a partial depth image estimated by single scale and multi-scale deep convolution network. From the result, we can see that the results of the multi-scale network can better reflect some small details of the depth map. The difference of object contour is more obvious, while the result graph of single scale network estimation is blurred on object contour. From the error curve, multi-scale network can gradually outperform single scale network, the convergence error of multi-scale network is about 2.44, the convergence error of single scale network is about 3.49, and the convergence of error is reduced by 43%.

According to the evaluation index, we choose the algorithm in this field to compare. The obtained algorithm index is shown in the table. The algorithm listed in this paper is the algorithm listed in [14], [15] and [16]. From the algorithm index, we can see that for the NYU dataset, the depth map of single scale network structure prediction in this paper is better than the single scale network result in the optimal David algorithm (coarse). The network has increased by an average of 7.1%. The REL index increased by 7%. The RMS index increased by 0.9%. The Log index increased by 9%. The $\delta$ index increased by 11.4%.

5. Summary
In view of the more complex ground environment, we propose to use the coding inverse coding structure to characterize the environmental characteristics. First, we propose a deep coding algorithm based on the improved coding and anti coding network, which can reconstruct the contour information and apply to the simple environment representation. Based on this, a depth distance estimation algorithm based on deep convolution network is proposed. The depth distance between the original image and convolution neural network convolution of different scales is obtained. The effectiveness of the algorithm is verified by general database. This algorithm has low computational complexity and is suitable for UAV's environmental perception.
Acknowledgments
Funding from the Science and Technology Innovation Project of Tiandi Technology Co., Ltd(2018-TD-QN002) is gratefully acknowledged.

References
[1] R. W. OSBORNE, I., et al. Design of an Adaptive Passive Collision Warning System for UAVs. IEEE Transactions on Aerospace and Electronic Systems, 47(3): 2169-2189.
[2] Lai, J., et al. Airborne Vision-Based Collision-Detection System. Journal of Field Robotics, 28(2): 137-157.
[3] Carnie, R., et al. Image Processing Algorithms for UAV “Sense and Avoid” [C]. 2006 IEEE International Conference on Robotics and Automation, 2848-2853.
[4] Cho S, Huh S, Shim D H, et al. Vision-based detection and tracking of airborne obstacles in a cluttered environment. Journal of Intelligent and Robotic Systems, 2013, 69(1-4): 475-488.
[5] Mejias L, McNamara S, Lai J, et al. Vision-based detection and tracking of aerial targets for UAV collision avoidance. 2010 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2010: 87-92.
[6] Dey D, Shankar K S, Zeng S, et al. Vision and learning for deliberative monocular cluttered flight. Field and Service Robotics, 2016: 391-409.
[7] Daftry S, Zeng S, Khan A, et al. Robust monocular flight in cluttered outdoor environments. arXiv preprint arXiv:1604.04779, 2016.
[8] Barry A J, Tedrake R. Pushbroom stereo for high-speed navigation in cluttered environment. 2015 IEEE International Conference on Robotics and automation, 2015: 3046-3052.
[9] Smolyanskiy N, Kamenev A, Smith J, et al. Toward low-flying autonomous MAV trail navigation using deep neural networks for environmental awareness. arXiv preprint arXiv:1705.02550, 2017.
[10] Wang, Wei, et al. "Generalized Autoencoder: A Neural Network Framework for Dimensionality Reduction." IEEE Conference on Computer Vision & Pattern Recognition Workshops IEEE, 2014.
[11] Roska, T., and L. O. Chua. "The CNN universal machine: an analogic array computer." (2015).
[12] https://cs.nyu.edu/~silberman/datasets/nyu_depth_v2.html
[13] Silberman, Nathan, et al. "Indoor Segmentation and Support Inference from RGBD Images." Proceedings of the 12th European conference on Computer Vision - Volume Part V Springer, Berlin, Heidelberg, 2012.
[14] Karsch, K., Liu, C., & Kang, S. B. Depth extraction from video using non-parametric sampling. European Conference on Computer Vision. 2012: 775-788.
[15] Ladicky, Lubor, Jianbo Shi, and Marc Pollefeys. Pulling things out of perspective. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2014.
[16] Eigen D, Puhrsch C, Fergus R. Depth map prediction from a single image using a multi-scale deep network. Advances in neural information processing systems, 2014: 2366-2374.