Time image sequence self-encoding statistics to improve visual odometer

Chuang Zeng *, Hongyang Yu

UESTC, Chengdu, China

*Corresponding author e-mail: hyyu@uestc.edu.cn

Abstract. Visual odometers are essential in SLAM applications is very important in the application of SLAM, and it is a test for visual odometer of plastering robot. The chaos of the construction site and the difficulty of extracting feature points on the wall have always been a bottleneck restricting the application of SLAM robots. In this paper, based on time series images, a neural network is trained. According to the real-time sequence scene prediction feature extraction algorithm parameters, the feature operator is extracted according to the predicted value. Then the feature operator is subjected to self-encoding dimension reduction and denoising, and finally the feature point is performed. Match. The experiment verifies that in the process of real-time visual feature detection, real-time correction of relevant feature extraction parameters by time series self-encoding statistics can improve the accuracy of feature extraction and matching.

1. Introduction

With the full penetration of artificial intelligence applications, in the era of Industry 4.0, devices that have been self-organizing, self-adaptive, and self-adaptive have become hot spots. People are paying more and more attention to the autonomy and intelligence of robots. Autonomous mobile robots have rapidly emerged in fields such as autopilot, drones, and unmanned retail, and have gradually become a part of this world, playing a new role in the era of intelligence. Mobile robot positioning and map creation (SLAM, simultaneous localization and mapping) [1] are hot research issues that are moving toward high levels of intelligence. In an unknown environment, robots need to explore and observe themselves to estimate their own movements, understand the environment and determine their location, and create an environmental map. Further through the establishment of a good environment map, the use of deep learning technology to recognize the surrounding things, navigation, obstacle avoidance, real-time decision-making.

The uncertainty of SLAM measurement mainly comes from sensor noise or technical limitations. A single sensor cannot obtain robust feature extraction in various application environments. For example, in a complex urban environment, a lidar can only obtain stable edge features in a stable environment region, and an unstable region like off-road environment. Lidar extraction of stable geometric features will be very difficult. Visual sensors, on the other hand, can acquire a large number of features in a large number of textures in unstable areas of the environment. Vision sensors undoubtedly enhance the certainty of measurement. Computer three-dimensional vision undoubtedly increases the intelligence and accuracy of the entire complex engineering system. As a result,
computer efficiency problems have emerged, and more and more experts have begun to study the computational efficiency and robustness of the SLAM technology that incorporates vision. This article focuses on the time series self-coding statistical algorithm to improve the visual odometer.

2. Research content overview

First of all, we describe the points of discovery of the problem. The performance of the number of feature points of objects at different light intensity and different angles is actually inconsistent, although today's feature extraction methods usually use a normalization to avoid this problem, but this is from a mathematical perspective. Reduce this interference on the extraction. Later, there will be specific experimental data to prove the relevant arguments, that is, we cannot rely solely on mathematical changes to solve problems such as light scale textures. We also need to compensate for the parameters from the front end of the extracted data.

This paper studies the extraction processes such as SIFT[2], SURF[3], ORB[4], etc., and finds out the influence of algorithm related parameters on feature extraction. In the SIFT algorithm, the Gaussian effect coefficient sigma and the edge filter threshold edgeThreshold are two factors that affect the number of image pyramid layers and the number of feature points extracted, and often take an empirical value. For the threshold of the Hessian matrix in the SURF algorithm and the fixed selection of image piles, this improvement to the SIFT can achieve real-time results, but the scene versatility or the robustness of multiple scenes is actually greatly reduced. In the ORB algorithm, the number of extracted feature points can be fixed, and a more concise binary description is used in the description. However, the scale space of the core is still limited, which obviously cannot be applied in high-complexity scenarios.

This paper also studies the method of feature matching screening BF[5], FLANN[6], RANSAC[7]. Through a series of comparisons, there are also cases where the parametric interference algorithm converges. For example, the search step length in the FLANN algorithm affects the filtering speed of the algorithm. BF is a violent match, its time complexity is often the highest, and RANSAC has an iterative k value, this value is also a fixed value, but some scenes can not just for real-time, but sacrifice the number of iterations, some scenes need to be high Accuracy can improve the accuracy of subsequent SLAM map creation.

For this reason, this paper proposes a temporal image sequence self-encoding network to improve the problem of curing parameters related to the visual odometer. Use the time series network to learn the changes of each scene, and timely feedback to the visual odometer through scene changes. Finally, the visual odometer extracts parameters through the feedback correction feature of the time series network. The algorithm has been optimized in the application scene of visual odometry of construction plastering robots.

3. Algorithm Overview

During the movement process of building plastering robots, color images and depth images are continuously acquired through the visual odometer. The pose relationship can be obtained by calculating the geometric relationship between the images. Using the depth data to match the point cloud, a three-dimensional map can be created. So how to use a visual odometer to capture images is particularly critical. The core idea of this paper is the temporal image sequence segmentation parameter, that is, the parameter value of the feature extraction algorithm that should be needed to predict the next temporal image sequence through the image sequence over a period of time. The temporal image sequence neural network is used to correct the feature extraction parameters in the image in real time so as to achieve better real-time performance and robustness.

The temporal image sequence neural network is inspired by the adjustment of the human eye to light. When the light intensity is strong, the pupil of the human eye becomes smaller, and in a dark environment, the pupil of the human eye becomes larger. People will naturally adjust this parameter, but the visual odometer does not have such an adjustment function. After careful analysis of biology, we found that humans' adjustment from one environment to another and the human eye also has a
process. This establishes that computers can also use certain methods to adjust activities to achieve the maximum in this environment. Great visual effect.

Figure 1. Parameter impact feature matching.

3.1. Time Image Sequence Network
For the temporal image sequence, we first make an explanation. The time series here is not a traditional continuous time, but a kind of generalization. It is mainly divided into the following three types:
1. Image sequences with different grayscale distributions in successive time periods.
2. Different image sequences in gray-scale distribution at discrete intervals.
3. The continuous change sequence of images under different conditions. The conditions here may be different lighting conditions and different observation angles.

By acquiring a time image sequence network, during the operation of the robot visual odometer, after a certain period of time data acquisition, the situation of future data can be acquired according to this data collection, and the visual odometer can be fed back in real time to modify the corresponding parameters in time. First of all, this article defines a time series network. The input of the network is a time-series grayscale image, and the output corresponds to the real-time parameter prediction value.

Define the entire input and output functions of the network middle layer as

\[ y_t = \alpha(ax_t + by_{t-1} + c) \]  \hspace{1cm} (1)

\( x_t \) is the input time \( t \) image, \( y_t \) is different feature extraction algorithm required parameters output. Note that the time series image here is a kind of change image defined by us. This kind of change exists in every moment of the actual scene.

Figure 2. Time Image Sequence Neural Network.

Define the entire input and output functions of the network middle layer as

\[ y_t = \alpha(ax_t + by_{t-1} + c) \]  \hspace{1cm} (1)

\( x_t \) is the input time \( t \) image, \( y_t \) is different feature extraction algorithm required parameters output. Note that the time series image here is a kind of change image defined by us. This kind of change exists in every moment of the actual scene.
Define the upper and lower layers of the network. The upper layer corresponds to the real-time output value:

\[ o_t = \beta y_t + c \]  \hspace{1cm} (2)

In training the neural network will often give a certain weight, and loss function, so we get the time series visual odometer expression formula:

\[ o_t = \alpha(a) + L(x) \]  \hspace{1cm} (3)

For this network training method, we can use the gradient descent method[8]. Because of the local independence of the image, in order to speed up the training, we divide the image into 3x3, that is, \( x_t \) image input is a 3x3 block. At the same time, the output \( y_t \) quantity is the relevant parameter for feature extraction, and it can also increase the information feedback compensation coefficient such as contrast and image enhancement.

![Figure 3. Block diagram.](image)

Therefore, the input variables and output variables are extended into the time image sequence neural network as follows:

\[
y_t = \alpha(a) \begin{bmatrix} x_t^1 \\ x_t^2 \\ \vdots \\ x_t^9 \end{bmatrix} + b y_{t-1} + c \]  \hspace{1cm} (4)

\[
\begin{bmatrix} y_t^1 \\ y_t^2 \\ \vdots \\ y_t^k \end{bmatrix} = \alpha(a) \begin{bmatrix} x_t^1 \\ x_t^2 \\ \vdots \\ x_t^9 \end{bmatrix} + b \begin{bmatrix} y_{t-1}^1 \\ y_{t-1}^2 \\ \vdots \\ y_{t-1}^k \end{bmatrix} + c \]  \hspace{1cm} (5)

\( \alpha \) is the activation function, using the characteristics of the activation function. Here we use softmax[9] as our activation function we will obtain the weight coefficient of the image sequence of the entire time and use the weight coefficient to form a non-linear relationship such as the change of the gray scale of the time series, the contrast condition and the feature extraction coefficient.

According to the above, we have established a model of a time-series neural network. We need to establish a training model for the target. Therefore, we define the gradient of the middle-layer at any
time to obtain our training parameter model. Define the middle layer transfer function as $\sigma_t$, The gradient function is $\frac{\partial L}{\partial \beta}$.

$$\sigma_t = \frac{\partial L}{\partial y'_t}$$

$$\frac{\partial L}{\partial \beta} = \sum_t \sum_i \frac{\partial L}{\partial y'_t} \frac{\partial y'_t}{\partial \beta}$$

$$= \sum_t \sum_i \text{diag}(1-y'_t)\sigma_t(x_t)^T$$

With the gradient function we can use commonly used deep learning tools such as tensorflow for network learning, time-series neural networks, and feature prediction parameters for real-time prediction.

### 3.2. Time series self-encoding network

We can obtain real-time feature extraction parameters through the time image sequence network. That is, we can now obtain a set of feature extraction solutions that belong to a specific scene according to different scenarios. Therefore, it is easy to obtain more feature points than the previous algorithm and the appropriate number of feature points, and obtaining more points means that there are more point interference points. For this reason, we use a time-series denoising self-encoding network to improve the accuracy of our existing feature point extraction.

Here, we directly use the denoising self-encoding[10], and put the denoising self-encoding into our time-series neural network. At the beginning our input image is a 3x3 block image, and at the same time the original self-encoding network becomes Self-encoded networks with temporal properties. The main difference is that the point estimate becomes a time series time score estimate. At the same time, different self-encoding sets are obtained in different scenarios. Therefore, we can use the self-encoded

![Time series self-coding](image_url)
set of the plastering site scenes flexibly to the architectural plastering site, and at the same time, obtain a set of denoising solutions suitable for special scenes.

According to our changes, the denoising self-coding minimization target suitable for our time image sequence network is obtained.

\[
E(x'_{in}, g(f(x'_{in}))) + \Omega(y', x'_{out})
\]

\[
\Omega(y', x'_{out}) = \lambda \sum_i x'_i
\]

For the 3x3 input image module, we need to statistically obtain the square error criterion for the Gaussian damage and reconstruction of the true image and train our self-encoder according to this criterion.

\[
\sum_i \| g(f(x'_i)) - x'_i \|
\]

Of course, some people here will certainly ask, so many values in real-time feedback back, in fact, the feedback parameters of the gap is not actually a small impact on the characteristics, in order to ensure the real-time nature of the original algorithm, we set a switch device here.

\[
k = \begin{cases} 
1 & \| y'_t - x'_{out} \| < \xi \\
\| y'_t - x'_{out} \| & \| y'_t - x'_{out} \| \geq \xi
\end{cases}
\]

Finally, we will organize the above algorithm into a flow chart.

\[\text{Figure 5. Time series self-adaptive matching network}\]
From the algorithm flow chart, this article needs to train two modules, the time series network module is used for parameter prediction, the time image sequence self coding module is used for image denoising, and the feature extraction is performed with the parameter prediction improved by the time series network of the Forbidden City, and finally combined with Coded network for feature matching screening.

4. Experimental results and analysis
In this paper, a two-dimensional color image and a depth time series image are obtained through a three-dimensional camera of a plastering robot, as shown in FIG. 6. The data in the table 1 and table 2 are the average data of a single image, the average loss error.

![Figure 6. Part of the atlas](image)

| Table 1. Feature Extraction and Matching by Default. |
| --- | --- | --- |
| The default situation | SIFT | SURF | ORB |
| Feature Match Number | Match rate | Time (ms) | Feature Match Number | Match rate | Time (ms) | Feature Match Number | Match rate | Time (ms) |
| BF | 350 | 99.2% | 240 | 343 | 98% | 78 | 380 | 76% | 54.725 |
| FLANN | 276 | 78.9% | 140 | 250 | 71.5% | 65 | 355 | 71% | 30.474 |
| RANSAC | 324 | 92.6% | 136 | 311 | 88.7% | 62 | 375 | 75% | 28.131 |

| Table 2. Feature Extraction and Matching of Time Series Self-coding |
| --- | --- | --- |
| Time series self-encoding network | SIFT | SURF | ORB |
| Feature Match Number | Match rate | Time (ms) | Feature Match Number | Match rate | Time (ms) | Feature Match Number | Match rate | Time (ms) |
| BF | 360 | 99.5% | 243 | 355 | 99.3% | 80 | 455 | 91% | 56.725 |
| FLANN | 324 | 92.8% | 142 | 275 | 92.6% | 67 | 425 | 85% | 32.474 |
| RANSAC | 350 | 99.2% | 138 | 322 | 92.4% | 63 | 431 | 86% | 30.131 |

From the comparison between Table 1 and Table 2, we can see that the time series self-encoding neural network effectively provides parameter prediction and denoising feedback. Comparing each feature extraction and feature matching algorithm, the matching rate has been improved. Of course, the threshold judgment of the time image sequence network leads to an average of about 2ms more in the time consumption of the algorithm in this paper. However, considering that this algorithm needs to face a highly complex construction environment, it is obviously at a sacrifice of a certain time to obtain higher accuracy. acceptable.

5. Conclusion
This paper completes the design of self-encoding neural network for time image sequence, aiming at solving the complex scene changes, predicting reasonable feature extraction and matching parameter values through scene changes, and using the self-encoded network output feedback to improve image
quality. Get a better match. The experiment verifies the rationality and enforceability of the algorithm. According to the experimental data, the algorithm can improve the matching rate of the visual odometer by about 5%, which means that the fault tolerance rate is increased by 5%. This algorithm enhances the fault-tolerance of the robot visual odometer in the practical application scene construction plastering robot. In complex and variable light, or dark corners, it can get better feature extraction and matching effect than traditional algorithms.

Acknowledgments
This work was financially supported by Shanghai Holford Group.

References
[1] Pritsker, A. A. B. (1986). Introduction to stimulation and Slam II.
[2] Ng, P. C., & Henikoff, S. (2003). SIFT: Predicting amino acid changes that affect protein function. Nucleic acids research, 31(13), 3812-3814.
[3] Bay, H., Tuytelaars, T., & Van Gool, L. (2006). Surf: Speeded up robust features. Computer vision–ECCV 2006, 404-417.
[4] Rublee, E., Rabaud, V., Konolige, K., & Bradski, G. (2011, November). ORB: An efficient alternative to SIFT or SURF. In Computer Vision (ICCV), 2011 IEEE international conference on (pp. 2564-2571). IEEE.
[5] Kullmann, O. (2017). The Science of Brute Force. Communications of the ACM.
[6] Muja, Marius, and David G. Lowe. "Fast approximate nearest neighbors with automatic algorithm configuration." VISAPP (1) 2.331-340 (2009): 2.
[7] Derpanis, K. G. (2010). Overview of the RANSAC Algorithm. Image Rochester NY, 4(1), 2-3
[8] Friedman J H. Greedy Function Approximation: A Gradient Boosting Machine[J]. Annals of Statistics, 2001, 29(5):1189-1232.
[9] Gold S, Rangarajan A. A Graduated Assignment Algorithm for Graph Matching[J]. IEEE Transactions on Pattern Analysis & Machine Intelligence, 1996, 18(4):377-388.
[10] Schölkopf B, Platt J, Hofmann T. Greedy Layer-Wise Training of Deep Networks[C]// International Conference on Neural Information Processing Systems. MIT Press, 2006:153-160.