Frontiers of Information Technology & Electronic Engineering
www.jzus.zju.edu.cn; engineering.cae.cn; www.springerlink.com
ISSN 2095-9184 (print); ISSN 2095-9230 (online)
E-mail: jzus@zju.edu.cn

An artificial intelligence enhanced star identification algorithm*

Hao WANG†‡[1], Zhi-yuan WANG[1], Ben-dong WANG[1], Zhuo-qun YU[1], Zhong-he JIN[1], John L. CRASSIDIS[2]

[1]*School of Aeronautics and Astronautics, Zhejiang University, Hangzhou 310027, China*

[2]*Department of Mechanical and Aerospace Engineering, University at Buffalo, State University of New York, Amherst, NY 14260-4400, USA*

†E-mail: roger@zju.edu.cn

Received Oct. 30, 2019; Revision accepted Apr. 14, 2020; Crosschecked July 20, 2020; Published online Aug. 26, 2020

Abstract: An artificial intelligence enhanced star identification algorithm is proposed for star trackers in lost-in-space mode. A convolutional neural network model based on Vgg16 is used in the artificial intelligence algorithm to classify star images. The training dataset is constructed to achieve the networks' optimal performance. Simulation results show that the proposed algorithm is highly robust to many kinds of noise, including position noise, magnitude noise, false stars, and the tracker's angular velocity. With a deep convolutional neural network, the identification accuracy is maintained at 96% despite noise and interruptions, which is a significant improvement to traditional pyramid and grid algorithms.

Key words: Star tracker; Lost-in-space; Star identification; Convolutional neural network
https://doi.org/10.1631/FITEE.1900590 **CLC number:** V447

1 Introduction

As a kind of high-precision attitude determination instrument, star trackers are widely used in both orbiting and interplanetary spacecraft. Star trackers perform attitude determination by identifying stars in the field of view (FOV). Typically, a star tracker has two working modes, i.e., lost-in-space (LIS) mode and tracking mode. When prior knowledge of attitude information is unavailable, the star tracker operates in the LIS mode. In this case, a full-sky star identification algorithm (Spratling and Mortari, 2009) is required. Once the initial attitude has been determined, the star tracker switches to the tracking mode. In this case, the previously obtained information will be used to predict the current attitude so that the identification process is much easier than that in the LIS mode. Thus, a reliable LIS star identification algorithm becomes the major problem to be solved to obtain attitude information from a star tracker.

Many star identification algorithms have been developed in the last 40 years to solve the full-sky star identification problem. The existing full-sky autonomous star identification algorithms can be divided roughly into two categories (Padgett and Kreutz-Delgado, 1997). The first kind tends to approach star identification as an instance of subgraph isomorphism. In this case, features used for star identification would be the angular separations. The most representative algorithm is the triangle algorithm (Liebe, 1992). The original triangle algorithm is sensitive to focal plane noise and false stars. To improve robustness and reduce

‡ Corresponding author

* Project supported by the National Natural Science Foundation of China (No. 6152403)

ORCID: Hao WANG, https://orcid.org/0000-0002-0383-7258

identification time, the pyramid algorithm has been introduced (Mortari et al., 2004). This uses at least four stars for pattern creation and the k-vector search technique. However, focal plane noise still affects the performance of the pyramid algorithm. Another algorithm identifies stars by planar triangles along with their area and polar moment to match a catalog of triangles (Cole and Crassidis, 2006). This approach has been expanded using polygons formed by neighboring stars (Hernández et al., 2017). This algorithm is highly robust to focal plane noise, but is sensitive to false stars. Schiattarella et al. (2017) designed a multi-pole algorithm to deal with false objects. The results showed that the identification accuracy can reach 100% considering a large number of false objects. The multi-pole algorithm is robust to moderate angular velocities. For most algorithms, the validation step takes a lot of time because of the need for many lookups, and may fail when the number of star points is not large enough. Focusing on this problem, Wang et al. (2019) proposed a two-step validation algorithm.

The second kind tends to treat star tracking as a pattern identification problem. In this case, each star is associated with a unique pattern determined by its neighboring stars. The initial algorithm is the grid algorithm (Padgett and Kreutz-Delgado, 1997). Compared with triangle algorithms, the grid algorithm is less sensitive to focal plane noise and is less time consuming. Some modified grid algorithms (Na et al., 2009; Aghaei and Moghaddam, 2016) have been developed to better mitigate star position deviation and magnitude noise. However, because of the need to find the reference star's closest neighboring star to generate a star pattern, magnitude noise and false stars may cause grid algorithms to fail to identify the closest neighboring star. A radial and cyclic algorithm (Zhang GJ et al., 2008) has been developed to deal with false stars, but this algorithm is sensitive to magnitude noise. A modified radial and cyclic algorithm (Wei et al., 2019) has been proposed to solve this problem. Besides the morphological feature, the singular value of star vectors can be used as the unique feature. Juang et al. (2003) proposed an original singular value decomposition algorithm. However, this algorithm is also sensitive to star magnitude. Sun et al. (2017) proposed a modified algorithm based on the singular value to solve this problem. To improve the performance of the singular value decomposition algorithm in dynamic conditions, Kim and Bang (2020) proposed a new singular value algorithm. For daytime star tracking, Roshanian et al. (2016) proposed a robust star identification algorithm using a Euclidean distance transform of images. Some algorithms based on different star pattern identification techniques have been proposed (Mehta et al., 2018; Samirbhai et al., 2019).

Researchers have proposed some other methods, such as the multi-purpose panoramic camera based algorithm (Opromolla et al., 2017), adaptive ant colony algorithm (Quan and Fang, 2010), and artificial intelligence algorithm (Hong and Dickerson, 2000; Roberts and Walker, 2005; Jing and Liang, 2012), to deal with star tracking issues. A neural network and fuzzy logic have been proposed to identify stars in Hong and Dickerson (2000). They used a reference star and its two brightest neighbors to form a triplet, ordered the triplet based on star brightness, and fed it into a neural network. Simulation results were obtained with the star identification accuracy higher than 95%, even when the magnitude error was as high as 0.5 arc-seconds and the angular separation error as high as 121.4 arc-seconds. However, this method does not take false stars into account, and this can cause the triplet to be changed. A method based on the counter propagation neural network (Roberts and Walker, 2005) has been proposed. Test results showed that the identification accuracy achieved was over 97% when the star position error in the ascension and declination was up to 0.025°. However, this method does not consider the influence of the magnitude error and false stars. A neural network has been used to improve the identification accuracy in Jing and Liang (2012). It had an identification accuracy over 95% when the position deviation was one pixel. However, false stars are not taken into consideration either.

Generally, in these traditional neural network methods, the star tracking problem is treated as a pattern identification problem. The distance vectors or angular separation between reference stars and their neighbor stars are used to identify reference stars. This means that patterns must be made beforehand. In general, the more the patterns, the higher the accuracy. Unfortunately, too many patterns can lead to a large computational load in network training. Thus, these methods may not be

applicable to a practical mission. Furthermore, it is difficult for these approaches to achieve good identification when there are false stars, because false stars will change the distance vectors or angular separation with the reference stars. Recent advances in convolution neural networks (CNNs) provide an alternative approach (Cheng et al.,, 2018; Zhang QS and Zhu, 2018). It has been shown that CNN performs much better than other neural networks in machine vision circumstances. CNNs have brought a revolution in computer vision, and become the dominant approach for almost all recognition and detection tasks (LeCun et al., 2015). Considering that star tracking identification can be treated as a machine vision problem, CNN is used to solve the star identification problem. With the help of deep learning, the neuron number and computational cost in network training are reduced dramatically. This means that this method could be used for practical space missions. The key to star identification is to extract informative features or patterns from the noisy background.

In this study, a new LIS star identification algorithm based on deep convolutional neural networks (DCNNs) is proposed. The main idea is to use a DCNN to classify the chosen reference stars in the star image. To make the network robust to magnitude uncertainty, false stars, and position deviation, a training dataset is constructed through several ways of data augmentation. Simulation results are presented to show the advantages of this approach.

2 Star identification algorithm based on deep convolution neural networks

The star identification algorithm is composed of a training dataset and a DCNN. The training dataset must be constructed beforehand to make any identification possible.

2.1 Generation of the training dataset

2.1.1 Guide star catalog

The Smithsonian Astrophysical Observatory (SAO) star catalog is chosen as the basic star catalog. FOV of the camera is set to $12^\circ \times 12^\circ$, and the highest visual magnitude detected by the star tracker is set to 6.0 Mv. Guide stars are selected according to the following rules:

1. The magnitude of the guide star should be less than 6.0 Mv.

2. The guide star should have at least five neighboring stars to ensure high precision for star tracking.

3. At least one of the five closest neighboring stars should be brighter than 5.7 Mv.

Based on these rules, a guide star catalog containing 4897 stars is obtained. The star distribution of the catalog is shown in Fig. 1. Then, a training dataset is constructed.

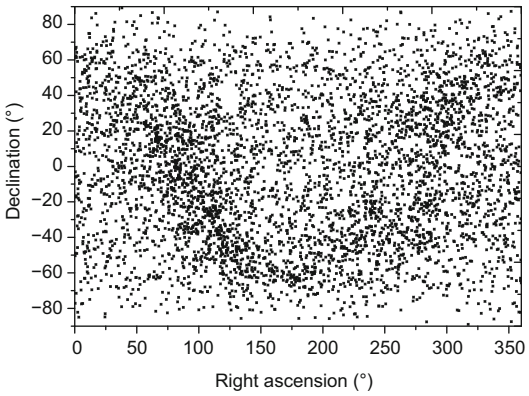

Fig. 1 Distribution of guide stars

2.1.2 Training dataset

The key to constructing a training dataset is to make sure that the distributions of the training dataset and real dataset are as similar as possible. The training dataset contains image samples and the corresponding guide star's ID. The grid algorithm (Padgett and Kreutz-Delgado, 1997) is used in the construction of the training dataset.

Step 1: Select a guide star as the reference star S. Then, a small region is defined by circles with radii pr and br around the reference star. This region is denoted by sky(br, pr).

Step 2: Five closest stars from S inside sky(br, pr) are considered as adjacent stars A_i ($i = 1, 2, \ldots, 5$). For most stars, the magnitude deviation is always lower than 0.3 Mv (Kruzhilov, 2012). Thus, there will be at least one adjacent star for every guide star chosen according to the third guide star selection rule.

Step 3: Each adjacent star is chosen as the orientation star, and the image is rotated until the orientation star lies on the horizontal axis (Fig. 2). Curves

with arrow in Fig. 2 denote the directions of rotation. Then, a basic dataset that contains five images for each guide star is obtained. This basic dataset is denoted by T1. Fig. 2a shows a non-rotating image, and Figs. 2b, 2c, and 2d show the rotated images in T1 when A_1, A_2, and A_3 are chosen as orientation stars, respectively.

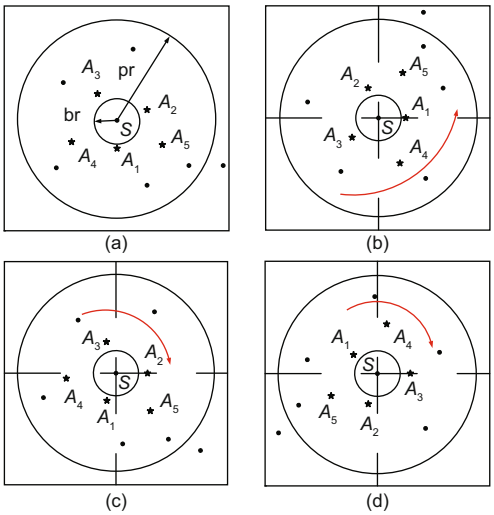

Fig. 2 Construction of the basic dataset: (a) original image; rotated images when A_1 (b), A_2 (c), and A_3 (d) are orientation stars

Step 4: To reduce overfitting and enhance the generalization power of the network, the basic dataset T1 is enlarged using several ways of data augmentation as described below:

(1) Stars at pr pixels or farther away from the reference star are discarded to improve robustness.

(2) A random magnitude deviation varying from −0.3 to 0.3 Mv is added to each star.

(3) Add 1–5 false stars with random positions and magnitudes.

(4) Add white noise to each image. The mean and variance of noise are determined by the noise level of the complementary metal-oxide-semiconductor (CMOS) sensor used.

Step 5: Resize the image to 224 × 224. A function named "imresize" in Matlab is used to resize images. The parameter is set to be "bilinear," which means that the output pixel value is a weighted average of pixels in the nearest 2 × 2 neighborhood.

After these steps, a training dataset T2 containing 734 550 images is constructed. If a false star is chosen accidentally as the reference star or orientation star, fatal errors would occur. Thus, an extra class, i.e., the false star class, is constructed to cope with this situation. A negative sample set is built accordingly. There are one million images generated with a Monte-Carlo method to compose the negative sample set. A false star is chosen as the reference star in half of these images, while the orientation star is a false star in the remaining images. During training process, the number of false star class samples used is three times that of each positive class to balance the data. There are 4897 positive samples and three negative samples in each training course, which are chosen randomly from the positive sample set and negative sample set (Liu et al., 2003), respectively.

2.2 Star identification algorithm

The star identification algorithm based on DCNN is shown in Fig. 3.

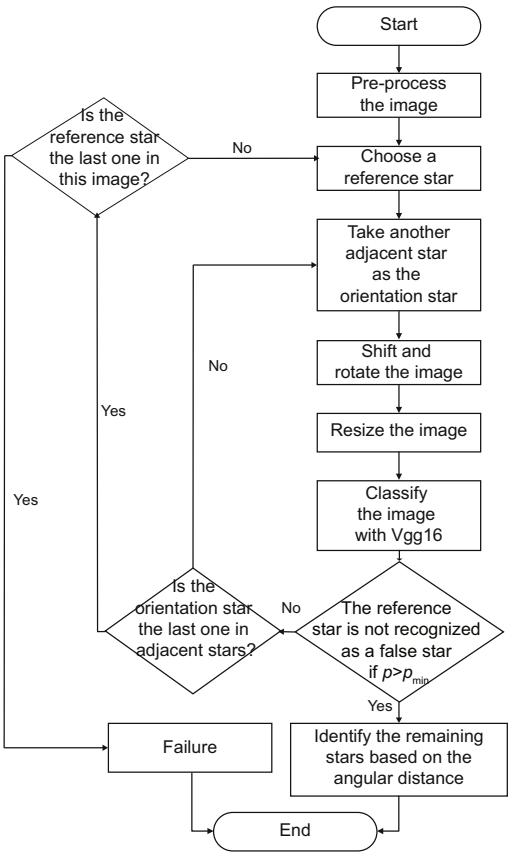

Fig. 3 Flowchart of the star identification algorithm

There are six steps in performing identification:

Step 1: image preprocessing. Im denotes the image. All stars, whose number is n, are extracted from their backgrounds. Their coordinates set is C_n.

Step 2: reference star and orientation star determination. The nearest star to the center of the image is taken as the reference star S. Its coordinates are defined as C_i. The five closest stars near S inside a region, defined by circles of radii pr and br, are taken as adjacent stars A_i $(i = 1, 2, \ldots, 5)$. The nearest one of A_i $(i = 1, 2, \ldots, 5)$ will be chosen as the orientation star. Its coordinates are C_{ja}.

Step 3: image standardization. Move the image to relocate the reference star to the center of the image, rotate the image to make the orientation star on the positive horizontal axis, and then resize the image to 224×224. This step is fulfilled by three functions, called shift, rotate, and resize in the pseudo code.

Step 4: image classification. The pre-processed image is sent to the trained Vgg16 network. According to the output of Vgg16, the ID of the reference star and the corresponding probability p are obtained. This procedure is denoted as function Vgg16_Classify in the pseudo code.

Step 5: reference star validation. To prevent spurious classifications, the identification is considered successful if $p \geq p_{\min}$ and the reference star is not recognized as a false star. Otherwise, another orientation star is chosen from the rest of A_i; this means going back to step 3. If all adjacent stars are used and the identification still fails, then switch to another reference star and repeat step 2.

Step 6: star identification. Identify the remaining stars in the image by their angular distances from the reference star. This is the function called AngularMatch in the pseudo code.

The pseudo code is listed as follows:

```
begin
[n, Cn] = Preprocess(Im)
for (i = 1; i <= n; i++){
  for (j = 1; j <= 5; j++){
    Im = Shift(Im, Ci)
    Im = Rotate(Im, Cja)
    Im = Resize(Im)
    [id, p] = Vgg16_Classify(Im)
    if (id! = false_star_class && p > pmin){
      ID = AngularMatch(id, Cn)
      return ID
    }
  }
}
end
```

There are several well-known CNNs, such as GoogLeNet (Szegedy et al., 2015), Vgg16 (Simonyan and Zisserman, 2015), and AlexNet (Krizhevsky et al., 2017). GoogLeNet and Vgg16 have achieved great success in the 2014 ImageNet competition for classification and detection challenges. In this framework, Vgg16 has been adopted as the base network structure. The architecture of the network is described in detail in the following:

1. CONV layer. This layer is used to filter the input signal in the space domain and produce a two-dimensional feature map. There are 16 CONV layers with 3×3 convolution kernels in the network. The CONV layer is represented as follows:

$$y_{m,n}^k = f\left(\sum_{j=0}^{2}\sum_{i=0}^{2} x_{m+i,n+j} w_{i,j}^k + b^k\right), \quad (1)$$

where $x_{m,n}$ is the input value at position (m, n) $(0 \leq m \leq M,\ 0 \leq n \leq N$, and $M \times N$ is the size of the feature map), w^k and b^k the weight and bias of the k^{th} kernel, respectively, and $f(\cdot)$ the activation function.

2. Activation function. A leaky rectified linear unit (ReLU) function is used in the network:

$$f(x) = \begin{cases} x, & x > 0, \\ \alpha x, & x \leq 0, \end{cases} \quad (2)$$

where α is set to 0.05.

3. Pooling layer. Spatial pooling is carried out by five max-pooling layers, which follow some of the CONV layers. Max-pooling is performed over 2×2 windows with stride 2.

4. Fully connected (FC) layer. After several CONV and pooling layers, a set of feature maps has been obtained. Then, three FC layers are used to combine these features for classifying the input images. The first two FC layers are composed of 500 neurons. The last FC layer serves as the output layer with 4898 neurons, which represent the 4897 classes of the problem and one false star class. In this layer, the softmax function is used to calculate the probability of each class. The FC layer can be represented as follows:

$$\boldsymbol{y} = f(\boldsymbol{W}^{\mathrm{T}} x + \boldsymbol{b}), \quad (3)$$

where x is the set of feature maps, $\boldsymbol{y}$ the vector of output, $\boldsymbol{W}$ the weight matrix, and $\boldsymbol{b}$ the bias vector.

The softmax function is defined as

$$p_n = \mathrm{e}^{z_n} \Big/ \sum_{k=1}^{N} \mathrm{e}^{z_k},\ n \in [1, N], \tag{4}$$

where p_n is the probability of the n^{th} class and z_n the output score of the n^{th} class from the FC layer. The original network is modified in three aspects: the size of the input layer is changed to $224 \times 224 \times 1$, the activation function is changed to leaky ReLU, and the number of neurons of the first two FC layers is changed to 500. The architecture of Vgg16 is demonstrated in Fig. 4.

3 Simulation results and analysis

Simulations are presented to show the performance of the proposed algorithm. Fig. 5a shows the random star image to be identified. Star 127 934 and a false star are chosen as the reference star and orientation star, respectively. Then, the image is shifted and rotated (Fig. 5b), and the directions of rotation are denoted by arrow-headed curves. After the image classification step, the output ID of the reference star is No. 11 983 with an associated probability of 12%. Thus, another orientation star is chosen. In Fig. 5c, Star 127 894 is chosen as the orientation star, and the original image is pre-processed again. Fig. 5c is classified, and the output ID of the reference star is No. 127 934 with an associated probability of 93%.

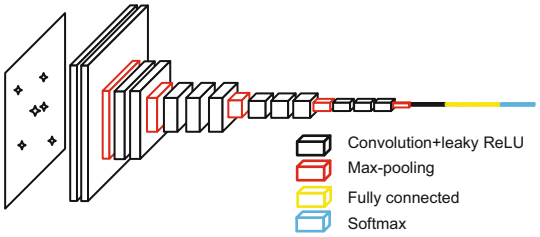

Fig. 4 Architecture of Vgg16

3.1 Network training

Network is trained with the Caffe framework (Jia et al., 2014). A stochastic gradient descent method with a batch size of 128 examples is employed. Momentum and weight_decay, two important parameters of Caffe framework, are set to 0.9 and 0.0005, respectively. Plots of loss/accuracy vs. the number of iterations are shown in Fig. 6.

3.2 Comparison and analysis

Several simulations are conducted to evaluate the performance of the new algorithm. Parameters

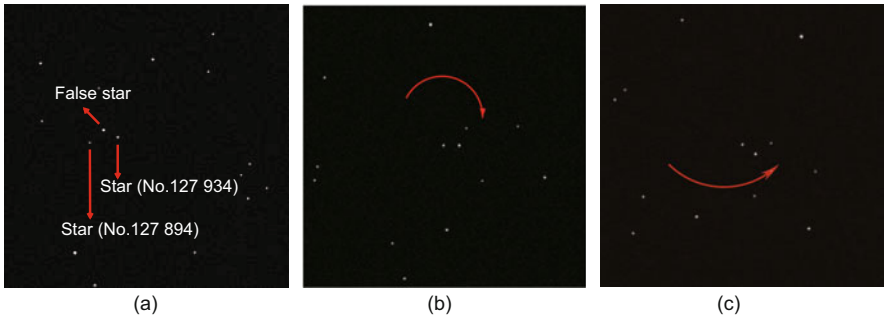

Fig. 5 Examples of the identification: (a) random star image; (b) image after the first classification; (c) image after the second classification

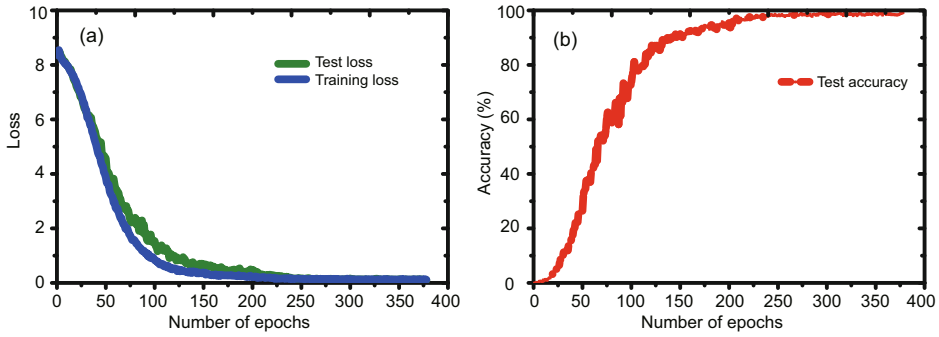

Fig. 6 Loss (a) and accuracy (b) vs. the number of epochs

of the star tracker for the simulations are shown in Table 1. The full well charge is denoted by a gray value of 1024 because the output is a 10-bit analog-to-digital converter (ADC), and the exposure time t_{int} is set to 0.1 s. According to Zhang GJ (2017) and Hancock et al. (2020), the variance of noise is calculated as

$$\delta_{\mathrm{sensor}} = (\delta^2_{\mathrm{Dark\_noise}} + \delta^2_{\mathrm{Temporal\_noise}} + \delta^2_{\mathrm{FPN}} + \delta^2_{\mathrm{Dark\_signal}})^{1/2}, \quad (5)$$

where $\delta_{\mathrm{Temporal\_noise}}$ is the temporal noise, $\delta_{\mathrm{Dark\_noise}}$ the dark noise, δ_{FPN} the fixed-pattern noise, and $\delta_{\mathrm{Dark\_signal}}$ the dark signal, while the mean noise is determined by the actual astronomical background light. For photo electrons corresponding to a 10-Mv star, the value of background noise is $73e^-$, and the mean noise and variance of noise are calculated as 1.196 and 24.812, respectively.

Table 1 Parameters for simulations

Parameter	Value
Resolution of CMOS	1024×1024
Size of one pixel	$6.7\ \mu m \times 6.7\ \mu m$
Focal length	35 mm
Highest visual magnitude	6.0 Mv
Full well charge	$62\ 500e^-$
Temporal noise	$\delta_{\mathrm{Temporal\_noise}} = 2.5$ LSB
Dark noise	$\delta_{\mathrm{Dark\_noise}} = 21e^-$
Fixed-pattern noise	$\delta_{\mathrm{FPN}} = 4.5$ LSB
Dark signal	$\delta_{\mathrm{Dark\_signal}} = 4.5$ (LSB/s) $\cdot\ t_{\mathrm{int}}$
Exposure time t_{int}	0.1 s
Astronomical background noise	10 Mv

LSB: least significant bit

The performance of the proposed algorithm is compared with those of the grid, pyramid, and modified radial and cyclic algorithms. All these algorithms except the modified radial and cyclic algorithm are performed with 4000 images, which are generated by Matlab based on the SAO star catalog. According to Kumar et al. (2010), the pyramid algorithm is more suitable for identifying a measured star pattern with four or more stars. Thus, images containing at least four stars are tested. These algorithms are evaluated under three noise conditions, i.e., position deviation, false stars, and magnitude uncertainty.

3.2.1 Robustness to star position noise

Each star has position noise, which is random Gaussian noise with a zero mean and a certain variance. The standard deviation is set from zero to one pixel with a increment of 0.2 pixels during the simulations. In accordance with the state of star extraction, the largest deviation of a star is set to one pixel in the simulations. In Fig. 7, with increasing position deviation, the identification accuracy of the pyramid algorithm drops from 97% to 92.5%. The pyramid algorithm uses the angular distance as a key feature, so the position noise will possibly cause a matching failure during identification. The other three algorithms are insensitive to position noise. They maintain a level of identification accuracy over 97%.

3.2.2 Robustness to false stars

According to Schiattarella et al. (2017), various phenomena may lead to false stars, such as single event upset, sensor aging, and thermal drift. Fig. 8 shows the false stars' influence on the identification accuracy for different algorithms. The number of false stars increases from zero to five, and the position and magnitude of the false stars are randomly added in the simulations. It can be seen that the DCNN algorithm is robust and maintains its accuracy over 97%. While the accuracy of the modified radial and cyclic algorithm decreases slightly from 98.8% to 95.3% when the number of false stars increases from zero to four, the grid algorithm's identification accuracy drops from 99.3% to 76.5%. As mentioned in Section 1, the key to generating a grid pattern is to find the correct closest neighbor star. Thus, false stars may be selected as the closest neighbor stars, which causes a failure.

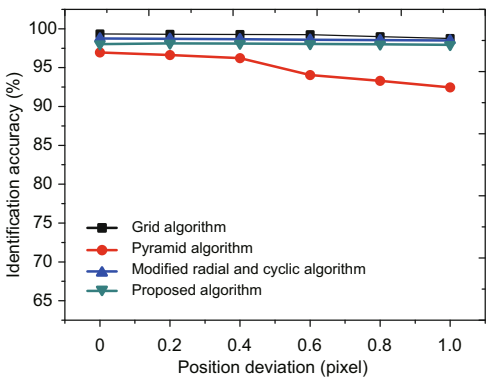

Fig. 7 Identification accuracy vs. position deviation

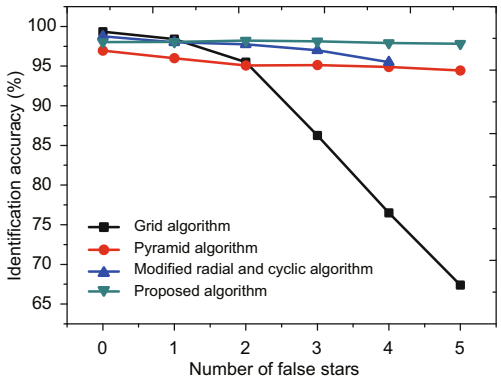

Fig. 8 Identification accuracy vs. the number of false stars

3.2.3 Robustness to magnitude noise

For each star, the magnitude noise is set from 0.05 to 0.30 pixels with an increment of 0.05 pixels during simulations. The maximum magnitude deviation is assigned 0.3 Mv for each star. Performances of these algorithms are shown in Fig. 9. The identification accuracies of the proposed algorithm and the modified radial and cyclic algorithm both remain over 97% when the magnitude noise increases, while the identification accuracy of the grid algorithm drops from 99.3% to 83.5%. The identificiation accuracy of the pyramid algorithm decreases slightly from 97% to 92%.

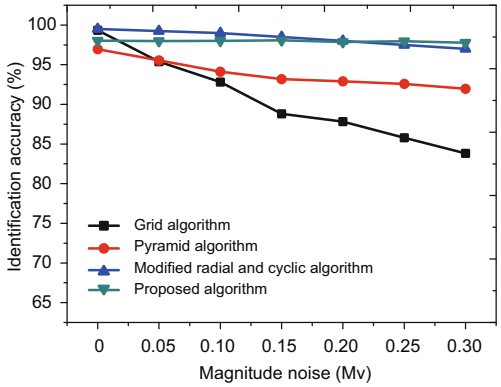

Fig. 9 Identification accuracy vs. magnitude noise

3.2.4 Robustness to angular velocity

Under dynamic conditions, the star-spots on the image sensor move across multiple pixels during the exposure time and form streaks. The coordinates $(x(t), y(t))$ of the star-spot center at time t can be approximately calculated as

$$\begin{cases} x(t) \approx x_0 + \dfrac{f\omega t \cos\theta}{\mu}, \\ y(t) \approx y_0 + \dfrac{f\omega t \sin\theta}{\mu}, \end{cases}$$

where (x_0, y_0) are the star-spot center coordinates at $t = 0$, f the focal length, ω the angular velocity, θ the angle between the streak and the x axis, and μ the pixel size. During the simulations, ω is set from $1°/s$ to $3°/s$ and θ is set randomly from $-\pi$ to π. Table 2 shows the performance of the proposed algorithm. As can be seen, the identification accuracy decreases from 96.95% to 96.23%.

Table 2 Identification accuracy under different angular velocities

Angular velocity (°/s)	Identification accuracy (%)
1	96.95
2	96.65
3	96.23

3.2.5 Analysis of star identification errors

There are 114 identification errors in the 4000 identifications during simulations. Among them, the reference star is identified to a star nearby in 96 images. Star 28 737, the reference star in Fig. 10a, is identified as star 28 738 because they are too close to identify. Fig. 10b is one training image of star 28 738. The remaining 18 images are caused by false stars and the magnitude error. For example, star 81 727 is identified as star 14 908. Star 81 727 is chosen as the reference star in Fig. 10c, while star 14 908 is the true reference star in Fig. 10d. These two images are similar after false star inclusion. The false stars are circled in Fig. 10c. This error is inevitable because false stars are added randomly. The algorithm would return failure only if all the stars have been selected as reference stars and none of them are identified successfully. In our 4000 simulations, no failure case occurs. There may be two reasons for this: First, the number of guide stars is sufficient for identification because every disposed image contains at least three guide stars, which is the requirement for attitude determination; Secondly, the parameter $p_{\min}$ is set as 50%. This threshold is relatively low, so it is easy for the selected reference star to find a corresponding star in the catalog.

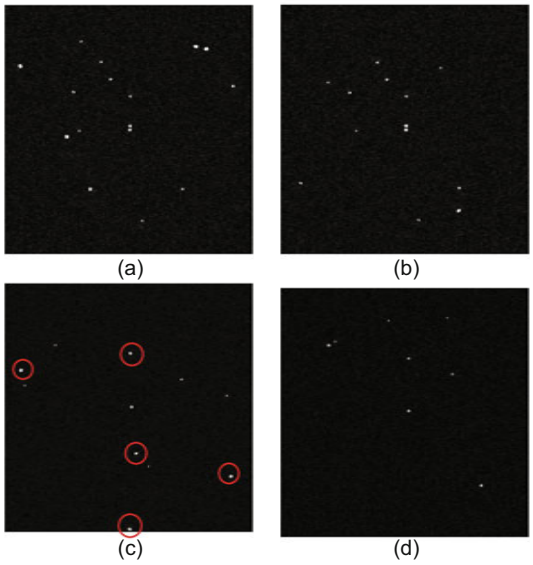

Fig. 10 Examples of identification errors: (a) testing image; (b) training image of star 28 738; (c) testing image with labeled false stars; (d) training image of star 14 908

3.2.6 Time and memory performance

The network is trained and tested on NVIDIA TITAN X. The average identification time is 4 ms. According to Qiu et al. (2016), the implementation of Vgg16-SVD on Xilinx Zynq achieves a frame rate of 4.45 frames/s. The pyramid and grid algorithms are implemented on TMS320C6747 DSP. The average identification time of the grid algorithm is 268 ms, and the identification time of the pyramid algorithm increases from 349 to 1592 ms as the false star count increases from zero to five. The number of parameters of the network is about 14.7 million and the memory requirement is 14 MB when eight quantization values are used.

4 Conclusions

A convolution neural network (CNN) model based on Vgg16 has been applied to classify star images, and a training dataset has been constructed accordingly. A star identification algorithm based on a deep convolution neural network for the lost-in-space mode achieved good performance for realistic star tracker scenarios. Simulation results demonstrated that this algorithm is highly robust to various kinds of noise, including position noise and magnitude noise, false stars, and angular velocity of the trackers. This work showed that CNN is highly effective for star identification. Future work will focus on reducing memory consumption and validating the proposed algorithm in a real star tracker.

Contributors

Hao WANG designed the research and drafted the manuscript. Zhi-yuan WANG and Ben-dong WANG trained the network and processed the data. Zhuo-qun YU helped acquire the data. Zhong-he JIN and John L. CRASSIDIS offered advice. Hao WANG and Zhi-yuan WANG revised and finalized the paper.

Compliance with ethics guidelines

Hao WANG, Zhi-yuan WANG, Ben-dong WANG, Zhuo-qun YU, Zhong-he JIN, and John L. CRASSIDIS declare that they have no conflict of interest.

References

Aghaei M, Moghaddam HA, 2016. Grid star identification improvement using optimization approaches. *IEEE Trans Aerosp Electron Syst*, 52(5):2080-2090. https://doi.org/10.1109/TAES.2016.150053

Cole CL, Crassidis JL, 2006. Fast star-pattern recognition using planar triangles. *J Guid Contr Dynam*, 29(1):64-71. https://doi.org/10.2514/1.13314

Cheng J, Wang PS, Li G, 2018. Recent advances in efficient computation of deep convolutional neural networks. *Front Inform Technol Electron Eng*, 19(1):64-77. https://doi.org/10.1631/FITEE.1700789

Hancock BR, Stirbl RC, Cunningham TJ, et al., 2020. CMOS active pixel sensor specific performance effects on star tracker/imager position accuracy. Symp on Integrated Optics, p.43-53. https://doi.org/10.1117/12.426872

Hernández EA, Alonso MA, Chávez E, et al. 2017. Robust polygon recognition method with similarity invariants applied to star identification. *Adv Space Res*, 59(4): 1095-1111. https://doi.org/10.1016/j.asr.2016.11.016

Hong J, Dickerson JA, 2000. Neural-network-based autonomous star identification algorithm. *J Guid Contr Dynam*, 23(4):728-735. https://doi.org/10.2514/2.4589

Jia YQ, Shelhamer E, Donahue J, et al., 2014. Caffe: convolutional architecture for fast feature embedding. Proc 22nd ACM Int Conf on Multimedia, p.675-678. https://doi.org/10.1145/2647868.2654889

Jing Y, Liang W, 2012. An improved star identification method based on neural network. Proc IEEE 10th Int Conf on Industrial Informatics, p.118-123. https://doi.org/10.1109/INDIN.2012.6301126

Juang JN, Kim HY, Junkins JL, 2003. An efficient and robust singular value method for star pattern recognition and attitude determination. NASA/TM-2003-212142, NASA Langley Research Center, Hampton, USA.

Kim K, Bang H, 2020. Algorithm with patterned singular value approach for highly reliable autonomous star identification. *Sensors*, 20(2):374. https://doi.org/10.3390/s20020374

Krizhevsky A, Sutskever I, Hinton GE, 2017. ImageNet classification with deep convolutional neural networks. *Commun ACM*, 60(6):94-90. https://doi.org/10.1145/3065386

Kruzhilov IS, 2012. Evaluation of instrument stellar magnitudes without recourse to data as to star spectral classes. *J Appl Remote Sens*, 6(1):063537. https://doi.org/10.1117/1.JRS.6.063537

Kumar M, Mortari D, Junkins JL, 2010. An analytical approach to star identification reliability. *Acta Astronaut*, 66(3-4):508-515. https://doi.org/10.1016/j.actaastro.2009.07.005

LeCun Y, Bengio Y, Hinton G, 2015. Deep learning. *Nature*, 521(7553):436-444. https://doi.org/10.1038/nature14539

Liebe CC, 1992. Pattern recognition of star constellations for spacecraft applications. *IEEE Aerosp Electron Syst Mag*, 7(6):34-41. https://doi.org/10.1109/62.145117

Liu B, Dai Y, Li X, et al., 2003. Building text classifiers using positive and unlabeled examples. Proc 3rd IEEE Int Conf on Data Mining, p.179-188. https://doi.org/10.1109/ICDM.2003.1250918

Mehta DS, Chen SS, Low KS, 2018. A robust star identification algorithm with star shortlisting. *Adv Space Res*, 61(10):2647-2660. https://doi.org/10.1016/j.asr.2018.02.029

Mortari D, Samaan MA, Bruccoleri C, 2004. The pyramid star identification technique. *Navigation*, 51(3):171-183. https://doi.org/10.1002/j.2161-4296.2004.tb00349.x

Na M, Zheng DN, Jia PF, 2009. Modified grid algorithm for noisy all-sky autonomous star identification. *IEEE Trans Aerosp Electron Syst*, 45(2):516-522. https://doi.org/10.1109/TAES.2009.5089538

Opromolla R, Fasano G, Rufino G, et al., 2017. A new star tracker concept for satellite attitude determination based on a multi-purpose panoramic camera. *Acta Astronaut*, 140:166-175. https://doi.org/10.1016/j.actaastro.2017.08.020

Padgett C, Kreutz-Delgado K, 1997. A grid algorithm for autonomous star identification. *IEEE Trans Aerosp Electron Syst*, 33(1):202-213. https://doi.org/10.1109/7.570743

Qiu JT, Wang J, Yao S, et al., 2016. Going deeper with embedded FPGA platform for convolutional neural network. Proc ACM/SIGDA Int Symp on Field-Programmable Gate Arrays, p.26-35. https://doi.org/10.1145/2847263.2847265

Quan W, Fang JC, 2010. A star recognition method based on the adaptive ant colony algorithm for star sensors. *Sensors*, 10(3):1955-1966. https://doi.org/10.3390/s100301955

Roberts P, Walker R, 2005. Application of a counter propagation neural network for star identification. AIAA Guidance, Navigation, and Control Conf and Exhibit, p.15-18. https://doi.org/10.2514/6.2005-6469

Roshanian J, Yazdani S, Ebrahimi M, 2016. Star identification based on Euclidean distance transform, Voronoi tessellation, and k-nearest neighbor classification. *IEEE Trans Aerosp Electron Syst*, 52(6):2940-2949. https://doi.org/10.1109/TAES.2016.150642

Samirbhai MD, Chen S, Low KS, 2019. A hamming distance and spearman correlation based star identification algorithm. *IEEE Trans Aerosp Electron Syst*, 55(1):17-30. https://doi.org/10.1109/TAES.2018.2845198

Schiattarella V, Spiller D, Curti F, 2017. A novel star identification technique robust to high presence of false objects: the multi-poles algorithm. *Adv Space Res*, 59(8): 2133-2147. https://doi.org/10.1016/j.asr.2017.01.034

Simonyan K, Zisserman A, 2015. Very deep convolutional networks for large-scale image recognition. https://arxiv.org/abs/1409.1556

Spratling BB IV, Mortari D, 2009. A survey on star identification algorithms. *Algorithms*, 2(1):93-107. https://doi.org/10.3390/a2010093

Sun ZP, Qi XD, Jin G, et al., 2017. Ellipticity pivot star method for autonomous star identification. *Optik*, 137:1-5. https://doi.org/10.1016/j.ijleo.2017.02.067

Szegedy C, Liu W, Jia YQ, et al., 2015. Going deeper with convolutions. Proc IEEE Conf on Computer Vision and Pattern Recognition, p.1-9. https://doi.org/10.1109/CVPR.2015.7298594

Wang XC, Sun CH, Sun T, 2019. A novel 2-step validation algorithm for lost-in-space star identification. *IEEE Trans Aerosp Electron Syst*, 56(3):2272-2279. https://doi.org/10.1109/TAES.2019.2945104

Wei X, Wen DS, Song ZX, et al., 2019. A star identification algorithm based on radial and dynamic cyclic features of star pattern. *Adv Space Res*, 63(7):2245-2259. https://doi.org/10.1016/j.asr.2018.12.027

Zhang GJ, 2017. Star identification: methods, techniques, and algorithms. Springer, Berlin, Germany.

Zhang GJ, Wei XG, Jiang J, 2008. Full-sky autonomous star identification based on radial and cyclic features of star pattern. *Image Vis Comput*, 26(7):891-897. https://doi.org/10.1016/j.imavis.2007.10.006

Zhang QS, Zhu SC, 2018. Visual interpretability for deep learning: a survey. *Front Inform Technol Electron Eng*, 19(1):27-39. https://doi.org/10.1631/FITEE.1700808