Neural network approach to recognition of visible constellations by sky photo image

V.A. Galkin, A.V. Makarenko
Institute of Control Sciences of Russian Academy of Sciences, Moscow
galckin.vsevolod@gmail.com, avm.science@mail.ru

Abstract. The current paper demonstrates the effective capabilities of deep neural networks in solving the problem of identification of constellations from a photo of the sky in conditions of a priori uncertainty, incomplete observability and stochastic disturbances. The quality of solution 0.927 by metric F1 is obtained. In order to achieve the result, the original ResNet-like architecture of the convolution neural network was synthesized; statistical analysis of the dataset was carried out, the function of losses and strategy of neural network training were developed, and an accurate criterion of constellation observability in the image was formed.

1. Introduction
Celestial navigation is a set of methods for determining the navigational parameters of objects, based on information about stars position in the sky. Celestial navigation is often used for orientation and control of various mobile ground-based, surface, air and space objects. Astronavigation is currently used as a supplement to the inertial navigation system, but it should be noted that celestial navigation, unlike other navigation systems, is autonomous, protected against intentional distortion, does not accumulate errors and has an unlimited in space scope of application. Therefore, the development and improvement of celestial navigation methods is an urgent task. The publication [1] presented a new method of astronavigation of ground objects based on the image of the starry sky and a number of parameters based on a deep neural network, which is an effective solution for a quick “cold start” of the navigation system. The resulting solution has few drawbacks, the main of which is the low resulting accuracy of the algorithm, which is associated with the fundamental mathematical property of the astronavigation problem, which belongs to the class of incorrect inverse problems.

Finding navigational parameters of the object naturally results from the construction of navigation triangles linking the equatorial and horizontal celestial coordinate system. When the navigation triangle is solved, different relations arise between the parameters of the two systems of coordinates, namely the horizontal system parameters: \( h, A \) – altitude and azimuth angle of the star and the parameters of the equatorial system: \( \delta, \alpha \) – declination and right ascension of the star respectively, as well as the parameters of the Earth observer \( \varphi, \lambda, \Omega \) – latitude, longitude and GMT.

Here is the basic ratio of celestial navigation (for one star):

\[
\cos \left( \frac{\pi}{2} - h \right) = \sin(\delta) \sin(\varphi) + \cos(\delta) \cos(\varphi) \cos(\Omega - \lambda - \alpha),
\]

(1)
in which the two target variables to be found are \( \varphi \) and \( \lambda \). The stars parameter \( h \) and time \( \Omega \) are measured with instruments. Obviously, two equations are needed to find two variables, so several stars (at least two) are used to find the observer's position. It is also worth mentioning that you need to know the parameters of the stars \( \delta \) and \( \alpha \), being measured a priori, i.e. you need to know which stars are to be measured. Otherwise, under the condition of a priori uncertainty, the tasks of celestial navigation, as well as those of celestial orientation, become incorrectly defined, because the condition of uniqueness of the solution is not fulfilled. That is, the uncertainty of the observer's position arises.

There are various ways to reduce the task described above to the correct one, for example, by taking additional measurements of a selected star at a fixed point in space, but at a different time. However, a more effective approach to removing this uncertainty is to solve the task of identifying stars.

2. Statement of the Problem and Related Work
The key to the astronavigation task is to solve the task of identifying stars, using star sensors [2] in order to compare stars on the observed image and stars in a catalogue.

There are two separate groups of methods:

- **Lost-In-Space algorithms** when there is no a priori information about the position of the system or the observed area of the starry sky.
- **Recursive algorithms** where a priori information is available.

It should be noted that the algorithms of the second group work much faster, but they require a priori information that is not always available to the observer, so the development of the algorithms of the first group is a more relevant task, but significantly more complicated.

The input data for the Lost-In-Space algorithms is a list of vectors containing the navigation parameters of stars (coordinates) and some of their characteristics (luminosity), which are obtained by processing, filtering and selecting stars from the original image of the starry sky. A star catalogue with which stars are compared is also required.

Approaches to the Lost-In-Space task can be divided into two groups of algorithms by the type of features extraction from data (star vectors):

- **Algorithms based on subgraph isomorphism.** The stars represent the nodes of a graph, and the characteristics (features) between the stars are the edges with some weights, and the task of identifying the stars is to search for an isomorphic subgraph in the general graph of the star catalogue with the graph obtained from the original image. The following algorithms belong to this group: polygon angular matching algorithm, triangle algorithms, group match algorithms, and the pyramid algorithm and other.
- **Algorithms based on pattern recognition.** A pattern is associated with each star in the image and star directory. In this case, the task of identifying the stars is to compare the patterns and find the most "similar" pattern. This group includes such approaches as: grid algorithms, the singular value method algorithm, the Log-Polar transform algorithm, the Hidden Markov Model based algorithm and other.

A comprehensive description of these algorithms, an assessment of their computational complexity and an analysis of their noise is included into the review [3, 4, 5].

All of the above algorithms are based on the same idea – to construct the features “manually” in such a way as to maintain resistance to noise and achieve a high speed of comparison with similar features in a star catalogue. Modern approaches to solving multiple tasks, based on deep learning, make it possible to automatically identify the most relevant features for the task and “put” the learning data and algorithm for solving the task inside neural network. Hence the idea to apply deep neural networks to the task of identifying stars. As a result, there is no need to design features from the data and make comparisons with the star catalogue, and at the prediction stage the algorithm has a constant computational complexity of \( O(1) \).

It should be noted that most star identification works are based on working with the navigation parameters and characteristics of the stars extracted from the image. However, this approach requires the development of additional selection algorithms and does not take into account all available
information in the image of the starry sky. Attempts have been made to identify stars directly from an image of a starry sky [5, 6, 7]. The proposed approaches are based on a comparison of the resulting image of the starry sky and pre-constructed, template images. The main disadvantage of this approach is that it is necessary to store template images, the number of which depends exponentially on the size of the angle of view of the stellar sensor.

The idea that neural networks can be used for the identification of stars purposes dates back to 1989. Early works used simple MLP architectures with only one hidden layer, and the input for neural networks was features that were also “manually” extracted. An article [8] was published in 2019 and it presented a method for identifying stars based on two neural networks, which can be referred to as pattern recognition algorithms. The authors' idea is as follows, using a neural network-generator on the context (neighbouring stars) for the selected guiding star, a unique and optimal pattern is created, which is then recognised by the neural network-classifier. Their method is currently one of the most accurate, 98% of the actual star data set and the fastest, and it is worth noting that there is no comparison with the star directory and there is no need to store it. Moreover, the method presented is extremely noise-resistant. However, the input data for the algorithm is the extracted navigation parameters and luminosity of the stars. Identifying stars from the original, raw picture is therefore still a relevant task.

In addition to the task of identifying stars, a simplified task can be investigated – the identification of constellations, which is to identify the constellations observed in the image of a starry sky. In modern astronomy, the constellations are called the 88 sections of the celestial sphere (sometimes 89, breaking the Serpens constellation on its Serpens Caput and Serpens Cauda), which are used for convenient orientation in the starry sky. The boundaries of the constellations are arcs on a sphere in the equatorial coordinate system, which in turn is not connected with the body's own rotations. It follows that by neglecting the stars' proper motions, celestial bodies do not move between constellations.

The results from the constellation recognition task can be used as a priori information for recursive star identification algorithms [4], using observable constellations, constellation graphs and markings of constellation belonging to specific stars. The task can be interpreted as “bridge” between the Lost-In-Space tasks and the identification task for stars with a priori information. The solution of the constellation identification task based on comparison of templates was presented in report [10]. The declared accuracy on test images is 71.4% and the average recognition time is 85 seconds, which is unacceptable for most practical applications.

This paper presents an approach to the constellation recognition task based on a deep convolution neural network. The input to the neural network is the raw images of the starry sky, without other a priori information (as in the Lost-In-Space task), and the resulting (output) values are the marks of the constellations present in the image.

3. Experiment Results

3.1 Raw Data and Dataset Preparation
To train the neural network, iteratively on the parameters of the Earth observer (location of the observer, viewing angles and GMT), a data set was formed, which includes 1 284 780 images of the starry sky with the size of $240 \times 240$ pixels, the field of view angle of $20 \times 20$ and answers to the image, representing a binary vector with 89 coordinates long, in which 1 – the constellation is observed and 0 – the constellation is absent. To test the generalization of the model, a test data set was generated that does not intersect with the tutorial. The number of copies in the verification dataset is 48 772. The dataset was generated from the data of the star catalog Tycho-2.

3.2 Neural networks architecture
To solve the problem of recognition of constellations, the architecture of a deep convolution neural network was designed, which is a modification of the classic ResNet architecture [12]. Software implementation of the neural network is carried out in the pytorch framework version 1.6.0. The depth of the network is equal to 26 trained layers. The size of the network input $1 \times 240 \times 240$ is a single-
channel image, the output size of the resulting vector is 89. Activation function on the output layer is a sigmoid function, characterizing the probability of class appearance (observation of the constellation). The total number of configurable network parameters is 415,193. To estimate the quality of network approximation $F_1$ score was used [11].

3.3 Classifier Training and Work Flow Tuning

Initially, a binary cross entropy [11] was used to teach the neural network, which was further weighed to neutralize class imbalance. The Adam optimizer was used for learning for 15 epochs with standard learning speed and batch size of 256. The results of testing the resulting solution are shown in Table 1.

| Table 1 | Accuracy of the reference solution on test data set |
|---------|---------------------------------------------------|
| $F_1$, min | 0.234 | $F_1$, median | 0.450 | $F_1$, max | 0.799 |

It can be seen from the results presented in Table 1 that the neural network is actually non-functional. Proceeding from the fact that the deep neural network is a universal approximator [13] and the solution of the problem is possible, the exploratory analysis of the data was carried and in accordance with its results, the training strategy of the neural network was corrected, the loss function was adapted and the criterion of observability of the constellation in the image was refined.

During the data-set analysis, the observed distribution (within the picture) of the following main characteristics was analyzed: the number of boundary constellations, the area of the constellation, the number of stars included in the constellation.

To train the neural network, we developed our own loss function based on the binary cross entropy weighted by relative constellation areas. The basic learning strategy was changed: for the first 7 epochs, the Adam optimizer with standard parameters was used, then, for 15 epochs, the SGD optimizer was used, with an initial learning rate of 0.05, which is then reduced by 2 times every 3 epochs. Moreover, starting from the 15th epoch SGD optimizer was used with the change of the loss function to a logarithmically weighted product of the constellation area and number of stars in it, binary cross entropy. For all the components of the presented training strategy, the mini-batch size is 256 samples.

The results of the training and the test after the above modifications are presented in Table 2, where experiment №1 is performed on “pure data”, and experiment №2 on noisy data (the training and the test). It is necessary to state that introduced noise disturbances physically correspond to influences to which in a reality the star sensor is exposed, namely:

1. Noise with normal distribution law – corresponds to the noise of the image quantization.
2. Impulse noise distributed by binomial law -- corresponds to impulse effects that lead to “false stars”.
3. Closing a random part of the image is equivalent to obstructing the vision of the part of the stars that are actually in view.
4. Random rotation of the image related to the center can be regarded as rotation of the camera around the line of sight.
5. Random mirroring – image orientation errors in the star sensor.

| Table 2 | Accuracy of the modified solution on a test set of data. Experiment: №1 – “clear images”, №2 – images with superimposed noise. |
|---------|-----------------------------------------------------------------------------------------------------------------|
| $F_1$, min | 0.940 | 0.81 |
| $F_1$, median | 0.981 | 0.927 |
| $F_1$, max | 0.996 | 0.971 |
3.4 Outcome Analysis
From the presented results it follows that the artificial neural network successfully approximates the mapping between the image of the starry sky and the space of constellations, and also has a generalizing ability and copes with the solution of the problem in conditions of a priori uncertainty, incomplete observability and stochastic disturbances.

It is necessary to notice that in experiment №2, with noisy images of the star sky, on an input of a neural network any “pure image” during training did not arrive. However, the neural network formed its representation in its latent space and successfully identified on the new images.

4. Additional section

4.1 Network invariance relatively to the local movements of stars
Let's study the effect of local changes in the position of stars on neural network functionality. This type of noise impact occurs naturally when light is refracted from stars or for other reasons.

To begin with, we investigate the effect of this type of transformation on the accuracy of already trained networks from the previous sections. When constructing images of the starry sky, we will add a random perturbation to the coordinates of the true centre of the star \([x_0, y_0]\) distributed uniform in the \(U[0,5]\) range, so that the new, random star centre will be placed in a square \([x_0 - 5, x_0 + 5] \times [y_0 - 5, y_0 + 5]\).

The results of the evaluation of previously trained neural networks in the case of accidental wandering of stars relatively each other are shown in Table 3. It can be seen from the results that although the network trained in the conditions of noise for large constellations shows functionality (but with low accuracy), in general there is a tendency for the accuracy of the networks to drop, which leads to their inoperability.

| Neural network №1 | Neural network №2 |
|-------------------|-------------------|
| \(F_1\), min      | 0.02              | 0.311              |
| \(F_1\), median   | 0.092             | 0.493              |
| \(F_1\), max      | 0.383             | 0.698              |

In order to improve the quality and resistance of networks to this type of noise and to extend the conditions under which the network is able to operate, we will train the network under the conditions of local uncertainty. Due to computational features, it is not possible to effectively create an adequate training set of data, nor is it possible to generate images with a random “on-the-fly” shift of stars. The stars' wandering will be therefore replacing with blur, which simulates the smearing of stars and causes uncertainty in both the position and size of stars. Blur will be introduced using the following strategy: we will use a motion blur with a random grease size \([8, 15]\) pixels, a random grease angle \([-30, 30]\) degrees and a random number of times \([0, 3]\). We will also introduce Gaussian noise and false stars by simulating Bernoulli's distributions.

Let's conduct network training in the configuration described above. The results of the training are shown in Table 4, where we used an assessment on the same test data set but under different noise conditions. Experiment №1 was performed in data transformation to evaluate the same distribution as the training dataset, while experiment №2 was performed under the same conditions as the experiments in Table 3.

We can see the adaptability of the trained network to local shifts in the star centres, although in fact we have replaced this type of noise by introducing local uncertainty through the blur. To sum up, the artificially introduced uncertainty has expanded the area of invariant neural network solution. The next
step is to integrate motion blur noise into the general noise superimposition scheme, which is presented in section 3.3.

| Table 4 | Accuracy of the trained neural network considering blur. Experiment 1 - evaluation on a test data set when all noise is entered. Experiment 2 - evaluation on a dataset with a random stars centres. |
|---------|----------------------------------------------------------------------------------------------------------|
|         | Experiment №1                                      | Experiment №2                                      |
|         | $F_1$, min          | 0.714                  | 0.596                  |
|         | $F_1$, median      | 0.885                  | 0.707                  |
|         | $F_1$, max         | 0.960                  | 0.823                  |

We will use two types of neural networks to conduct this experiment:

1. The network used in the previous sections.
2. Exactly the same neural network as in point 1, but the width of each layer is doubled to increase the capacity of the model. This is done under the assumption that all types of noise increase the complexity and volume of the general distribution, which a standard network would most likely not be able to approximate.

Let's summarize the results in Table 5. The trained neural networks (especially the wide network) show acceptable assessment accuracy on noisy data.

| Table 5 | Accuracy of the trained neural network considering blur. Experiment 1 – evaluation on a test data set when all noise on a conventional network is entered. Experiment 2 – evaluation on the test data set when all noise on a high-capacity network is entered into the test data set. |
|---------|----------------------------------------------------------------------------------------------------------|
|         | Experiment 1                                      | Experiment 2                                      |
|         | $F_1$, min          | 0.577                  | 0.738                  |
|         | $F_1$, median      | 0.849                  | 0.894                  |
|         | $F_1$, max         | 0.971                  | 0.951                  |

4.2 Random distortion of the pattern
To summarise the general distribution on which the neural network operates, random distortion of the constellation pattern is applied, using a random prospective image transformation.

This transformation of the geometry of stars can be implemented in the following way. For each angle of the image $[x_i, y_i]$ we'll set a random number in the square $[x_i - 30, x_i + 30] \times [y_i - 30, y_i + 30]$ and apply a random perspective transformation from four angles. This type of transformation makes the task very difficult because it is non-linear and it randomly transforms the constellation pattern. A strategy identical to the previous sections was used to train the neural network. The evaluations obtained are shown in Table 6. It should be noted that with the exception of small classes, the neural network shows high adaptability to accidental distortion of the constellation pattern. This experiment indicates that the neural network does not simply memorise the spherical, pixel spacing of stars or other features, but forms exactly the patterns of star clusters.

| Table 6 | Neural network accuracy on test data set under conditions of random transformations |
|---------|----------------------------------------------------------------------------------|
|         | $F_1$, min          | $F_1$, median      | $F_1$, max         |
|         | 0.514                  | 0.841                  | 0.970                  |

5. Conclusion
The presented paper has demonstrated that deep neural networks can be effectively applied to identify constellations in a photo of the night sky. The quality of solution 0.927 by $F_1$ score has been obtained. To achieve the result, an original ResNet similar neural network architecture has been synthesized (26 trainable layers, 415 193 configurable parameters), and a statistical analysis...
of dataset structural characteristics has been carried out. On the basis of this analysis, a special neural network training strategy has been developed and an adequate criterion of constellation observability in the image has been formed. In addition, a study of the impact of noise and errors in input data on the quality and stability of the resulting solutions has been conducted.

Further research of this issue involves a comprehensive analysis of the resulting constellation identification solution and its integration into astronomical navigation and astronomical orientation.

References

[1]. Galkin V.A., Makarenko A. V. Study of the possibility of solving the problem of celestial navigation by deep learning methods. 13 Vserossijskoe soveshhanie po problemam upravlenija VSPU-2019 [Proc. 13th Conf. “All-Russian conference on control problems”]. – 2019. p. 2096-2100.

[2]. C. C. Liebe, “Accuracy performance of star trackers—A tutorial,” IEEE Trans. Aerosp. Electron. Syst., vol. 38, no. 2, pp. 587–599, Apr. 2002

[3]. Na M., Jia P. A survey of all-sky autonomous star identification algorithms //2006 1st International Symposium on Systems and Control in Aerospace and Astronautics. – IEEE, 2006. – C. 6 pp.-901.

[4]. Spratling B. B., Mortari D. A survey on star identification algorithms //Algorithms. – 2009. – T. 2. – №. 1. – C. 93-107

[5]. Rijlaarsdam D. et al. A Survey of Lost-in-Space Star Identification Algorithms since 2009 //Sensors. – 2020. – T. 20. – №. 9. – C. 2579.

[6]. Yoon, H.; Lim, Y.; Bang, H. New star-pattern identification using a correlation approach for spacecraft attitude determination. J. Spacecr. Rocket. 2011, 48, 182–186.

[7]. Delabie, T.; Schutter, J.D.; Vandenbussche, B. Highly efficient attitude-estimation algorithm for star trackers using optimal image matching. J. Guid. Control. Dynam 2013, 36, 1672–1680.

[8]. Xu L., Jiang J., Liu L. RPNet: A Representation Learning-Based Star Identification Algorithm //IEEE Access. – 2019. – T. 7. – C. 92193-92202

[9]. M. E. J. Newman. 2006. Modularity and community structure in networks. Proceedings of the National Acad. Sci. 103, 23 (Apr. 2006), 8577–8582.

[10].S. Ji, J. Wang, X. Liu, Constellation detection, in: Final Project for Spring 2014-2015, Stanford press, 2015.

[11].A. Goodfellow, Y. Bengio, and A. Courville, “Deep learning,” 2016, book in preparation for submitted for publication in MIT Press. [Online]. Available: http://www.deeplearningbook.org

[12].He K. et al. Deep residual learning for image recognition //Proceedings of the IEEE conference on computer vision and pattern recognition. – 2016. – C. 770-778.

[13].Cybenko G. Approximation by superpositions of a sigmoidal function //Mathematics of control, signals and systems. – 1989. – T. 2. – No. 4. – C. 303-314.