Informatics and Control Problems 44 Issue 2 (2024)
journal homepage: www.icp.az

Determining the location of a moving device based on video images

U.M. Goyushova

Institute of Control Systems, Baku, Azerbaijan

ARTICLE INFO

Article history:
Received 13.06.2024
Received in revised form 25.06.2024
Accepted 11.07.2024
Available online 25.12.2024

Keywords:
UAV
Feature detection
Reference
Object recognition
Object detection

ABSTRACT

The study is devoted to the development of an algorithm for the navigation of unmanned aerial vehicles based on video images, which requires less computational resources when comparing images taken at different times. Here, resource refers to the use of both time and memory space. In the proposed method, the images of the objects that the UAV can encounter are obtained from the Google Earth platform. Features are detected in images and a database of found features is created. The same features are detected in the images taken by the UAV during its movement. Current found features are compared with the features previously saved in the database. The location of the device is determined based on the coordinates of the object to which the features belong. The software of the algorithm is simulated in the C++ programming language and is intended to be used in onboard computers.

1. Introduction

Navigation of moving devices, including UAVs, is one of the topical issues. Various navigation methods are available for such devices. One of these methods is image-based navigation. The essence of this method is that the device determines its location based on the areas and objects whose coordinates are known in advance. People can make decisions about where they are and in which direction they will go during their daily activities based on the objects they observe. They become aware of their location by familiar buildings, bridges, statues, roads, etc. By this logic, during image-based navigation, the device receives images of the earth's surface when it is in motion, images are processed, and object detection and recognition algorithms are applied. Information about the coordinates of detected objects is taken as known in advance. UAV navigation is carried out based on the coordinates of the objects found with the application of recognition algorithms [1]. Carrying out such navigation enables solving the problems of image-based object detection and recognition.

There are various approaches to image-based object detection and recognition. One such approach is the use of **neural networks**. Various neural network architectures are available: R-CNN, YOLO, RetinaNet, SSD, etc. Development of neural networks consists of the stages of the training of the system and obtaining results. The basis of the application of neural networks is the training stage. At this stage, image samples and templates of objects that the UAV can find are fed to the neural network. The more samples are fed to the system at this stage, the more accurate the results can be. On the other hand, if there are many samples, the training stage may take longer to complete. After the training of the system is completed, images of objects on the ground surface are taken during the movement of the UAV and transmitted to the neural network. An object is recognized in an image using a neural network. The coordinates of the device are determined based on the coordinates of the

E-mail address: kadaseva.ulviyyee@gmail.com (U.M. Goyushova)

www.icp.az/2024/2-06.pdf https://doi.org/10.54381/icp.2024.2.06

object [2].

Another approach to image-based object recognition is to open both images side by side and identify differences or similarities between them. Although this method does not require additional knowledge and tools, it is intuitive. The possibility of human error can be high and it is not considered a good method for large size images.

The object detection approach by pixel-by-pixel comparison of images consists in comparing each pixel of two images. For each of the pixels that make up the images, there are R, G, B color distributions. If the images or objects in the image are identical, the pixels of the compared images will have the same R, G, B values. In such a comparison, the comparison of images can be performed quite accurately, however, this method is impractical for large images or images with small variations [3].

The object detection approach by calculating the standard deviation between the pixels of the images is based on the measurement of the mean square difference between the corresponding pixels of two images. The value indicating the total difference between the images is calculated. A small value indicates a high similarity between the images. The advantage of the method is that it is easy to calculate just one value. The downside is that it is sensitive to noise and external interference [4].

One approach to image-based object detection and comparison is histogram comparison. The histograms of the color distributions of the images are calculated. Images are compared by identifying differences in the color content of the images. The method is computationally efficient but sensitive to changes such as lighting [5, 393-395].

Feature extraction is a method of detecting and matching the edges, corners, vertices and other parts called points of interest in images. The type of feature detected may vary depending on the objects in the images. Computer vision techniques are used to extract features. Images are compared based on the differences and similarities between these features [6].

This study investigates the problem of storing and comparing features.

2. Problem statement

The essence of the video image-based navigation method is determining the location of the device based on the coordinates of the objects in the captured images. The coordinates of the objects detected in the images must be known in advance. Two types of images are considered here. One of them is images obtained by a certain method in advance. After these images are processed, points of interest are found and templates are organized. This data, which we call a template or a reference, is determined and organized before the execution of the algorithm. The second type of images is images taken by the device during its movement. After these images are also processed, the required points of interest are detected and compared with the template.

One of the issues that arise here is determining on which information the templates are to be built. Considering that images consist of a large number of pixels, saving an entire image as a template or detecting and saving objects in an image may require a lot of memory space. On the other hand, such a complex organization of templates also affects the number of comparison operations. For this reason, building templates with a simple structure is considered suitable for the purpose.

Another issue that arises is deciding in what type of file or program to store the templates. When specifying the storage location, we should take a method that does not require installing an additional application or downloading libraries.

We consider the problem of determining the structure and building the algorithm, which requires little resources in terms of time and memory during the building, storage and comparison of templates. Here, resources mean both memory space and computation time.

3. Solution

Two sub-problems are considered when solving the problem of creating templates. One of them is the information based on which templates are organized, and the other is determining where the information is stored and in what type of files. Let us call the template storage location a database.

3.1. Organization of the template structure.

The structure of the templates is organized based on the images of the objects that the device can encounter during its movement. After pre-taken images are processed, reference objects selected from these images, winding non-intersecting roads, are detected. Reference objects mean objects that must be recognized in the image. Examples of such objects are important infrastructure objects, buildings, roads, power plants, steep rocks or other natural objects, etc. In [7], the detection of non-intersecting roads was considered and a method for identifying a road in the form of a broken straight line was proposed. A path identified as a broken straight line in the image consists of straight line segments. The length of the broken straight line representing the path, the number of straight line segments that make up the broken straight line, the lengths of these segments, etc. can be taken to be stored in the template. The ratio of the lengths of the straight line segments in the image is taken as a feature stored in the templates.

3.1.1 Determining straight lines in an image.

As shown in [7], when solving the problem, the sub-problems of image processing and detecting straight lines in the image are solved first. Uninformative pixels can be removed from the images by first processing the images. The processing steps include conversion from color images to grayscale images, and detecting extraneous lines in grayscale images. Additionally, it is possible to make images cleaner by using one of the image smoothing algorithms.

The Hough Transform [8] algorithm for detecting straight lines in images is based on the number of points that satisfy the equation of the same straight line, and each given straight line in the rectangular coordinate system is replaced by a point in the parameter space (polar coordinate system). A relevant example is shown in Fig. 1.

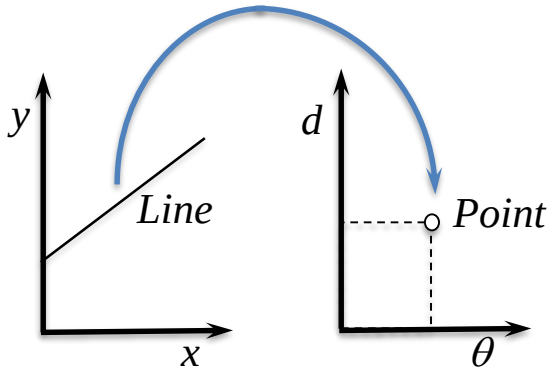

Fig. 1. Transition from the rectangular coordinate system to the polar coordinate system

In Fig. 1, d is the distance from the coordinate origin to the straight line in the rectangular coordinate system, and θ is the angle formed by d in the positive direction of the X axis. The Hough Transform algorithm uses the following formula for the straight line:

$$d = x\cos\theta + y\sin\theta \quad . \tag{1}$$

In formula (1), the angle θ can take values $\overline{0,360}$ degrees, the value of the distance d varies in the interval $\overline{0,d\_max}$. d_max is the size of the diagonal of the image; it is calculated as follows and is an integer:

$$d\_max = \sqrt{widt\square^2 + \square eig\square t^2}. \quad (2)$$

In formula (2) $width$ is the width of the image, and $height$ is the height of the image.

When transferring straight lines to parameter space (polar coordinate system), each straight line is replaced with a point. When looking at the polar coordinate system, the thickenings, i.e., the high density of points, indicate the presence of a straight line corresponding to this density in the rectangular coordinate system. Points that satisfy the same equation belong to a straight line. In this case, even if the number of points satisfying the same equation is small, they are also considered to form a straight line. Therefore, taking into account that it is possible to draw a straight line from any two points, when detecting straight lines in the image, very small straight lines (containing a small number of points) are also seen in the results, and these straight lines are negligible. In order to disregard such small straight lines, a threshold is set, and straight line segments smaller than this threshold value are discarded. Although during the image processing small straight lines are disregarded, the straight lines in the image being of different thickness and length leads to detecting several straight lines with small angle differences that actually are the same straight line. Detecting such a straight line several times with small differences leads to detecting a large number of unnecessary straight lines in the image. Fig. 2 shows the result of detecting straight lines in the image.

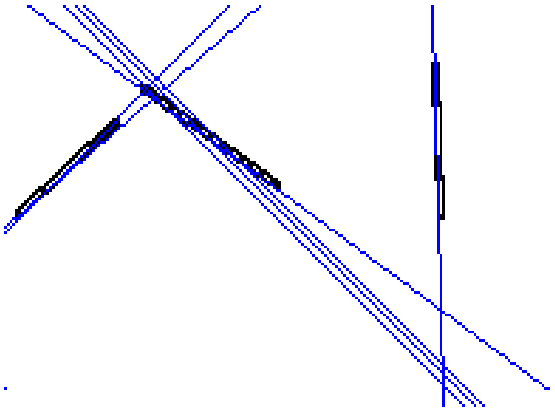

Fig. 2. Result of detecting straight lines in the image

Taking the above into account, a value, a number, is set in advance to refine the results, and only this number of straight lines are taken from the images. As can be seen from Fig. 2, although a straight line is found several times at different angles, according to the results, it is clear that they belong to the same straight line. Other redundant lines can be removed by grouping results belonging to the same line and selecting one line from each group as the result. The straight line segments containing the largest number of points from each group can be determined and selected as the result. The result obtained by applying the algorithm for detecting straight lines in images to roads is given in Fig. 3. Here, the road is shown as a broken straight line consisting of straight line segments.

Fig. 3. Representation of the Agsu Pass as a broken straight line

The path in the form of a broken straight line detected from the image consists of straight line segments and their length ratios are found. Thus, the built reference is composed of the ratio values of the lengths of up to a predetermined number of straight lines.

The steps of image processing and straight line detection are also performed on the images taken by the device during its movement.

3.2. Storing templates.

The second problem that arises is determining the type of files in which these reference values are stored. There are different ways to store these values. A few of them are compared below, with their advantages and disadvantages.

1. Text files with txt extension have a faster processing time and use less memory than other complex text files because they have no formatting or design style. They can be opened on various devices and any operating system [9].
2. Files with csv extension are text files that allow saving data in tabular form. Data are usually separated by commas. In the absence of spreadsheet programs such as MS Excel and the like to open the files, the information appears as comma-separated data [10].
3. Relational databases offer powerful query capabilities and data integrity features. However, creating and maintaining a database takes more time than using flat files. A database can handle large datasets and complex data structures efficiently [11].
4. JSON files store data as key/value pairs. Such storing makes the, more readable. Storing data with such a special structure and comma separation takes more time and memory than text files. [12].

Images of places and objects where the UAV can move are taken from the Google Earth platform. The stages of image processing and detection of non-intersecting winding roads in the form of broken lines are performed. A predetermined number of straight lines are detected in the images.

The ratios of the lengths of these straight lines identifying the road are stored in a txt file. The mentioned stages are also performed on the images received by the device at the current time, the ratios of the lengths of straight lines are calculated, compared with the values stored in the txt text file, and an appropriate template is selected.

3.3. Comparison of features.

Another problem that arises when comparing the values stored as a template in the txt file and the values calculated based on the features of the image obtained by the device during movement is how much these values conform with each other. The Hausdorff metric can be used to determine the conformity [13].

The Hausdorff metric is applied to this problem is as follows. If the features of pre-selected reference objects stored in the database as a template are set *A*, and the features of objects detected in the images obtained by the device during movement are set *B*, then the distance from each element of set A to the elements of set *B* is calculated by the following formula:

$$H(A,B) = \max\left\{ max_{b\in B}\{min_{a\in A}\|a-b\|\}, max_{a\in A}\{min_{b\in B}\|a-b\|\}\right\}. \quad (3)$$

$H(A,B)$ in formula (1) is called the Hausdorff distance and shows how far two sets are from each other. If this value is small, it means that the two given sets are close to each other. Thus, the most suitable one is found between the feature set B of the image obtained by the device and the feature set A stored in the database. The coordinates of the area where the device is moving are determined based on the coordinates of the object to which these features belong, that is, the road [14].

The software of the proposed algorithm for solving the problem is written in the Code Blocks environment, in the C++ programming language.

4. Conclusion

We have developed an algorithm that requires less computational resources when comparing images taken at different times.

Application of the algorithm ensures optimal use of computing systems. For example, enough memory space is used for storage of the values selected as references. The use of established references and appropriate comparison algorithms reduces the computation time. On the other hand, there is no need to install additional software and libraries to implement the proposed algorithm and the selected solution method. This, in turn, eliminates the problems of establishing connections with the programming language.

The result of this work is software that uses memory space efficiently and requires less computation time. This software can be efficient for use in real-time devices.

References

[1] Tae Hyun Kim, Denny Toazza, Navigation control of an Unmanned Aerial Vehicle (UAV), Bachelor Thesis in Computer and Electrical Engineering, November 2009.

[2] L. Hardesty, Explained: Neural networks, MIT News Office, 14 April 2017.

[3] S.Y. Woo, K. Kitaek, G.K. Jung, Y. Yeongsik, Extraction of Color Information and Visualization of Color Differences between Digital Images through Pixel-by-Pixel Color-Difference Mapping, Heritage. 5 No.4 (2022). pp. 3923-3945. doi.org/10.3390/heritage5040202.

[4] J. Gareth, D. Witten, T. Hastie, R. Tibshirani, An Introduction to Statistical Learning: with Applications in R. Springer (2021) pp. 29-33. ISBN 978-1071614174.

[5] K. Reinhard, Concise Computer Vision, An introduction into theory and algorithms, Springer-Verlag London, (2014).

[6] G. Kumar, P.K. Bhatia, A detailed review of feature extraction in image processing systems, 2014 Fourth International Conference on Advanced Computing & Communication Technologies, Rohtak, India, 2014, pp.5-12. doi: 10.1109/ACCT.2014.74.

[7] U.M.Goyushova, Algorithms for finding non-intersecting roads on images, Advanced Information Systems. 7 No.2 (2023) pp.5-8.

[8] Leandro A.F. Fernandes, Manuel M. Oliveira, Real-time line detection through an improved Hough transform voting scheme, Pattern Recognition. 41 No.1 (2008) pp.299-314.

[9] M. Lawnik, A. Pełka, A. Kapczyński, A new way to store simple text files, Algorithms. 13 No.4 (2020) p.101. https://doi.org/10.3390/a13040101.

[10] S.M.H. Mahmud, M.A. Hossin, H. Jahan, S.R.H. Noori, T. Bhuiyan, CSV-ANNOTATE: Generate annotated tables from CSV file, 2018 International Conference on Artificial Intelligence and Big Data (ICAIBD), Chengdu, China, 2018, pp. 71-75, doi: 10.1109/ICAIBD.2018.8396169.

[11] M. Choina, M. Skublewska-Paszkowska, Performance analysis of relational databases MySQL, PostgreSQL and Oracle using Doctrine libraries, Journal of Computer Sciences Institute. 24 (2022) pp. 250-257. doi.org/10.35784/jcsi.3000 .

[12] F. Pezoa, J. L. Reutter, F.Suarez, M. Ugarte, D. Vrgoč. Foundations of JSON Schema. In Proceedings of the 25th International Conference on World Wide Web. Republic and Canton of Geneva, CHE, pp. 263–273. https://doi.org/10.1145/2872427.2883029.

[13] A.A. Taha, A. Hanbury, An efficient algorithm for calculating the exact hausdorff distance. IEEE Transactions on Pattern Analysis and Machine Intelligence. 37 No.11 (2015) pp.2153-2163.
Doi: 10.1109/TPAMI.2015.2408351.

[14] E.N. Sabziev, Determining the location of an unmanned arial vehicle based on video camera images, Advanced Information Systems. 5 No.1 (2021) pp.136-139.