Aerial SLAM algorithm with global alignment using satellite imagery

Jakub Karbowski[1], Bartosz Bartoszewski[2], Daria Kokot[3], Michał Jan Kwiecień[4], Tymoteusz Turlej[4]

[1] AGH University of Krakow, Faculty of Computer Science, Krakow
[2] AGH University of Krakow, Faculty of Electrical Engineering, Automatics, Computer Science and Biomedical Engineering, Krakow
[3] AGH University of Krakow, Faculty of Physics and Applied Computer Science, Krakow
[4] AGH University of Krakow, Faculty of Mechanical Engineering and Robotics, Krakow

Abstract: Simultaneous localization and mapping (SLAM) algorithms can be used in aerial vehicles with a down-facing camera for navigation and terrain mapping. However, they suffer from an incremental position error. Each neighbouring image pair match introduces an alignment error which accumulates to a global position error proportional to the map size. By using a reference satellite image map this error can be reduced to a constant factor. This work proposes a novel image alignment algorithm called Embed Match which can be applied to any SLAM algorithm. Embed Match uses a neural network to represent a map region as a field of embedding vectors which are a semantic representation of the area. By comparing the embedding fields of a satellite image and the camera image, an alignment function is defined. Maximising this function leads to image alignment. The proposed algorithm shows a radically different approach from keypoint-based alignment methods which usually struggle with matching drastically different images, such as satellite and real camera images.

Keywords: SLAM, aerial navigation, mapping, neural network

ALGORYTM AERIAL SLAM Z GLOBALNYM USTAWIENIEM
PRZY UŻYCIU ZDJĘĆ SATELITARNYCH

Streszczenie: Algorytmy symultanicznej lokalizacji i mapowania (SLAM) mogą być stosowane w pojazdach powietrznych z kamerą skierowaną w dół do nawigacji i mapowania terenu. Jednakże występuje błąd inkrementacyjny pozycji. Każde dopasowanie sąsiedniej pary obrazów wprowadza błąd wyrównania, który wpływa na błąd globalnej pozycji proporcjonalnie do rozmiaru mapy. Korzystając z referencyjnej mapy obrazów satelitarnych, ten błąd można zredukować do stałego współczynnika. W tej pracy proponowany jest nowatorski algorytm wyrównywania obrazów o nazwie Embed Match, który może być stosowany do dowolnego algorytmu SLAM. Embed Match wykorzystuje sieć neuronową do reprezentowania regionu mapy jako pola wektorów osadzania, które są semantyczną reprezentacją obszaru. Porównując pola osadzania obrazu satelitarnego i obrazu kamery, definiuje się funkcję wyrównywania. Maksymalizacja tej funkcji prowadzi do wyrównywania obrazu. Proponowany algorytm stanowi radykalnie inne podejście niż metody wyrównywania oparte na punktach charakterystycznych, które zazwyczaj mają trudności z dopasowywaniem się do drastycznie różnych obrazów, takich jak obrazy satelitarne i rzeczywiste obrazy z kamery.

Słowa kluczowe: SLAM, nawigacja satelitarna, mapowanie, sieć neuronowa

https://doi.org/10.7494/978-83-67427-74-6_5

1. Introduction

SLAM algorithms play a vital role in the field of aerial robotics by providing real-time localization and constructing maps of the environment using internal sensor data (Gupta and Fernando 2022). SLAM algorithms not using external reference information such as pre-loaded maps exhibit an incremental mapping error, which accumulates over time and adversely affects the accuracy of the constructed maps. This accumulated error becomes particularly critical in scenarios involving extensive map coverage, long-distance navigation, and tasks reliant on global spatial relationships.

Most advances in SLAM algorithms are related to the reduction of this error. SLAM algorithms can utilise many different sensor types. Of particular interest are visual SLAM algorithms, which can be divided into visual-only, visual-inertial and visual-depth systems (Macario Barros et al. 2022).

SLAM algorithms incrementally build and refine a map of the environment. When new sensor data such as a camera image is collected, it is used to update the map. First, an estimate of the robot's position is used to roughly align new sensor data with the current map. Next, the new data is locally aligned with the nearest map features. Finally, a global alignment procedure takes into account features of the entire map created up to this point in time to perform long distance alignment and loop closing. Without the global alignment step, each local alignment of new sensor data would introduce an incremental error to the map. Loop closing helps reduce this error, as long as the robot visits the same position many times. However, the global position error of each point in the map is proportional to the size of the map. This kind of error cannot be eliminated without external reference data. A SLAM algorithm with loop closing, assuming robust operation, guarantees a relative positioning error between two map points that is proportional to the distance between these points. An ideal system would guarantee a constant positioning error, independent of the relative position of two points. If this error is measured in relation to a global frame of reference, such as the Earth, the system could be used to replace external global positioning systems.

The primary objective of this study is to introduce a novel approach that reduces the incremental positioning error to a constant factor, particularly focusing on top-down aerial SLAM algorithms (Avola et al. 2019). The proposed algorithm is described as a standalone component which can be used to align the map in any SLAM algorithm.

Traditional methods for image alignment often rely on detecting and matching characteristic keypoints, accompanied by complex feature descriptor computations (Bay et al. 2006, Rublee et al. 2011). However, these methods face challenges when dealing with images from different sources that exhibit visual disparities due to varying lighting conditions and camera characteristics. Moreover, such methods tend to be computationally demanding and may require distinct, visually characteristic points in the image.

The proposed Embed Match algorithm uses image embeddings generated by a neural network. These embeddings encode the semantic content of a given map region and enable alignment across domains, irrespective of lightning conditions and camera characteristics. A similar method is used in the SuperPoint algorithm (Quach et al. 2021) and HFNet-SLAM (Liu et al. 2023). These algorithms focus on detecting high quality keypoints that can be used for matching images from different sources. Embed Match does not detect individual keypoints but instead considers the entire image as a dense field of embedding vectors.

In summary, this paper introduces the Embed Match algorithm as a novel solution to the challenge of global alignment in aerial SLAM algorithms. By using neural network-generated semantic embeddings, Embed Match offers a new approach that holds promise for enhancing the accuracy and applicability of aerial mapping and navigation tasks.

2. Methods

2.1. High level proof of the algorithm

Embed Match works alongside an external SLAM algorithm. The goal of Embed Match is to guarantee a constant relative position error between any two points on the created map. The map is represented by a point cloud where each point is associated with an image taken by the aerial vehicle at that position. To simplify the proof of the algorithm's correctness, we consider a 1-dimensional SLAM problem. Let x^i_{real} and x^i_{est} be the actual and estimated positions of image point i. We want to guarantee that:

$$\exists e_{\text{em}} \forall i : \left| x^i_{\text{real}} - x^i_{\text{est}} \right| < e_{\text{em}} \tag{1}$$

holds true even when new points are added to the map. We call e_{em} the Embed Match error. We define the *EmbedMatch*(i) procedure which, given an image point i, adjusts x^i_{est} so that:

$$\left| x^i_{\text{real}} - x^i_{\text{est}} \right| < e_{\text{em}} \tag{2}$$

The limitation of *EmbedMatch*(i) is that it requires the initial condition:

$$\left| x_{\mathrm{real}}^{i} - x_{\mathrm{est}}^{i} \right| < e_{\max} \tag{3}$$

where $e_{\max}$ is the maximum Embed Match error – the maximum allowed divergence of the image point being aligned. The *EmbedMatch*(i) procedure shall be described later. Next, we assume the existence of a *LocalAlign*(i, j) procedure, which aligns neighbouring image points i and j. Such a procedure is provided by the SLAM algorithm used alongside Embed Match. In the case of ORB-SLAM3 (Campos et al. 2021), it is the matching algorithm using the ORB feature descriptor (Rublee et al. 2011). *LocalAlign*(i, j) assures that:

$$\left| \left(x_{\mathrm{real}}^{i} - x_{\mathrm{real}}^{j} \right) - \left(x_{\mathrm{est}}^{i} - x_{\mathrm{est}}^{j} \right) \right| < e_{\mathrm{loc}} \tag{4}$$

for neighbouring points i and j, where e_{loc} is the local alignment error. Using *LocalAlign*(i, j) for a sequence of neighbouring image points accumulates e_{loc} over the length of the sequence. This accumulation results in a position error that is proportional to the size of the map, and is the main problem solved by Embed Match.

Initially, the SLAM algorithm collects the x_{est}^{0} origin point estimate. If the robot's global starting position x_{real}^{0} can be estimated with an error less than $e_{\max}$, *EmbedMatch*(0) can be used to assure that:

$$\left| x_{\mathrm{real}}^{0} - x_{\mathrm{est}}^{0} \right| < e_{\mathrm{em}} \tag{5}$$

Next, let us consider a generic point i that is initialised to point 0. For this point, Equation (2) is initially satisfied because of Equation (5). The robot now collects the next image point $x_{\mathrm{est}}^{i+1} = x_{\mathrm{est}}^{j}$. Since this point will be neighbouring with point i, *LocalAlign*(i, j) can be used to satisfy Equation (4). From Equations (2) and (4), it follows that (omitting exact steps for brevity):

$$\left| x_{\mathrm{real}}^{j} - x_{\mathrm{est}}^{j} \right| < e_{\mathrm{em}} + e_{\mathrm{loc}} \tag{6}$$

which is simply the accumulation of the Embed Match error and the local alignment error. Under the condition that:

$$e_{\mathrm{em}} + e_{\mathrm{loc}} < e_{\max} \tag{7}$$

EmbedMatch(j) can be used to satisfy:

$$\left| x_{\mathrm{real}}^{j} - x_{\mathrm{est}}^{j} \right| < e_{\mathrm{em}} \tag{8}$$

This method can be used to satisfy Equation (1) by being applied to each image point in the sequence of collected sensor samples.

2.2. Embed Match procedure

The Embed Match procedure is used to align the input image to a global frame of reference. It works by maximising the satellite alignment score of the input image. The satellite alignment score is defined as a function of the estimated latitude and longitude of the input image. Given an input image and a latitude and longitude, the score is the mean patch similarity between the input image and the corresponding satellite image taken around the given latitude and longitude. The real-world size of the area represented by the input image is assumed to be known. It can be computed from the altitude of the camera and its focal length. To compute the mean patch similarity between two images, both images are divided into overlapping patches with a moving window technique. The moving window size and stride are hyper-parameters of Embed Match and depend of the used camera setup. For images with a resolution of 512×512 pixels, a window of size 128 pixels and stride 32 pixels was chosen empirically. After obtaining image patches, the mean embedding alignment (MEA) score is computed between pairs of corresponding patches. The MEA score is used as the satellite alignment score, taking values from –1 to 1.

2.3. Mean embedding alignment score

The MEA score is the core idea behind Embed Match. It is a measure of alignment between two images. For each patch in the input images, an embedding vector is computed with a neural network, in this case the OpenAI CLIP model (Radford et al. 2021). This process is illustrated in Figure 1. The MEA score is the mean of cosine similarity metrics between pairs of corresponding patch embeddings. If two patches show the same area, the cosine similarity of their embeddings will be close to 1. As the patches become more distinct, the cosine similarity approaches –1. The choice of an embedding neural network has significant effects on the quality of MEA. The CLIP model produces embeddings which are spatially-invariant. Spatially-invariant embeddings can only be used with a small patch size because embeddings of larger patches are less sensitive to small translations of the image. Since CLIP is trained with a text-based contrastive learning objective, its embeddings encode a general description of a given patch. With a smaller patch size, two neighbouring patches will have distinct contents which is used to guide image alignment.

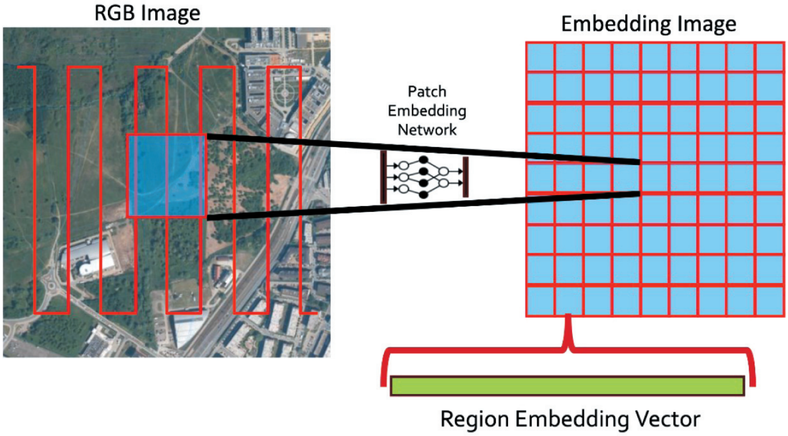

Fig. 1. Representing images with Embed Match

2.4. Maximising mean embedding alignment

Since the MEA is defined as a pure function of two variables, many numerical optimisation algorithms can be used. Embed Match assumes a maximum Embed Match error $e_{\max}$. Therefore, Bayesian optimisation could be an efficient method to optimise MEA as a Gaussian process. With the assumption that the area within $e_{\max}$ of the global maximum of MEA contains only one maximum, a gradient descent algorithm is used instead. The function gradient is approximated with the symmetric difference quotient. The discrete step h is a hyper-parameter of Embed Match defined proportionally to the size of the input image. With a step of 10%, the MEA is sampled at a position translated by 10% of the image size. After numerically computing the gradient with regard to the estimated latitude and longitude, a gradient descent iteration updates the position estimate. When the MEA stops increasing, the learning rate of gradient descent is halved. This iterative process is repeated for a fixed number of iterations. The result of the process is the latitude and longitude with the highest MEA score.

3. Experiments

Measuring the effectiveness of Embed Match when paired with an actual SLAM algorithm is out-of-scope for this work. Such an application is explained by the proof illustrating the intended usage of Embed Match. Instead, we focus exclusively on the image alignment procedure which is the core component of Embed Match. The model used for generating embeddings is the OpenAI CLIP ViT base variant with a patch size of 32 (not to be confused with the Embed Match sliding window patch size). The experiments were carried out based on custom data collected with a DJI Mavic Zoom quadcopter.

3.1. Visualising patch embeddings

The first experiment is meant to provide insights into the quality of the embeddings used for alignment. Two example images are used for the experiment. One is a satellite map image from the mapy.geoportal.gov.pl service and the other is a photo taken by a DJI Mavic Zoom quadcopter of the same area from an altitude of 100 m. The images are split into patches according to the Embed Match algorithm and patch embeddings are computed. The embeddings are reshaped into a 3-dimensional tensor where the third dimension is the embedding vector of one patch. The embeddings are processed with primary component analysis (PCA) to produce an RGB image. Bicubic interpolation is used to visualise the image. The resulting images can be interpreted as a semantically-compressed version of the source images.

3.2. MEA convergence

The MEA score is intended to be the highest when images are perfectly aligned. This assumption is tested by computing the MEA between a source image and target images translated by values taken from an even grid of points. A grid of 5×5 cells with a cell size of 5 m is created around the true geographic origin of the input image. This grid of MEA scores is visualised as a 3-dimensional surface showing whether the MEA has a correct global maximum.

3.3. Image alignment

Finally, the entire Embed Match procedure is used to align a real drone image with a satellite map reference. The path taken by the gradient descent algorithm is analysed, as well as the observed values of MEA. The final image alignment result is visualised. This experiment shows the intended usage of Embed Match alongside a SLAM algorithm.

4. Results

The quadcopter drone image and the corresponding satellite image used for the experiments are shown in Figure 2.

Fig. 2. Comparison of drone (a) and satellite (b) images of the same area

4.1. Visualising patch embeddings

Figure 3 shows side-by-side the patch embeddings generated with images from Figure 2. Both PCA-reduced images contain visually similar areas. The content of the source images differs greatly since one was taken years before the other. Many objects have been moved, some have disappeared, and some have been added. Even though the images come from completely different sources, their embeddings are similar.

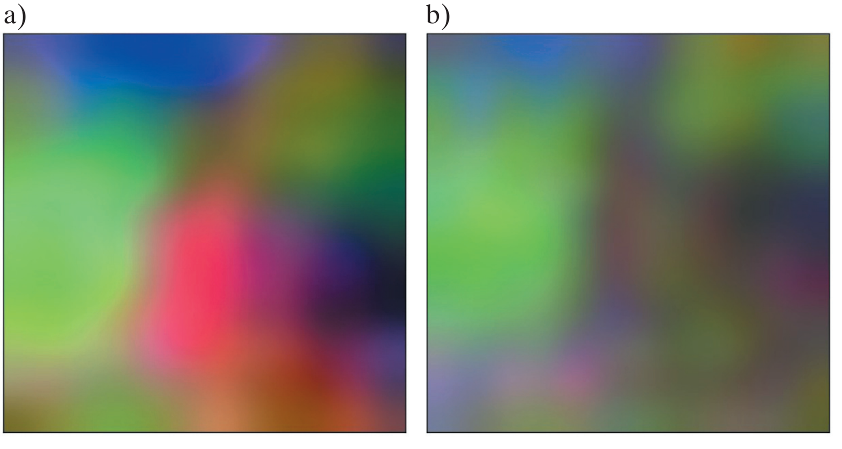

Fig. 3. Comparison of drone (a) and satellite (b) image embeddings

4.2. MEA convergence

Figure 4 visualises the MEA score as a 3-dimensional function of latitude and longitude. The global maximum of MEA is in the middle, corresponding to the actual global position of the image being aligned. No other local maxima can be seen, and the function has a smooth surface. This validates the use of gradient descent and shows that the MEA correctly assigns the highest value to the place of perfect alignment.

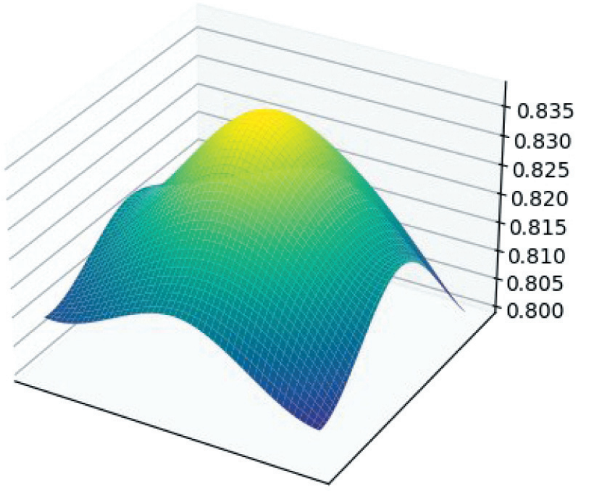

Fig. 4. Visualisation of the MEA score function

4.3. Image alignment

Figure 5 shows the results of Embed Match image alignment. The procedure begins with the actual drone image and an initial guess of the latitude and longitude. After ten iterations of Embed Match, the image is correctly aligned. The iterative progress of Embed Match is depicted in Figure 6, illustrating the path traversed by the algorithm. After a few initial large steps, the algorithm converges on the global maximum of MEA. Figure 7 shows the value of MEA at each iteration of Embed Match.

a) b) c)

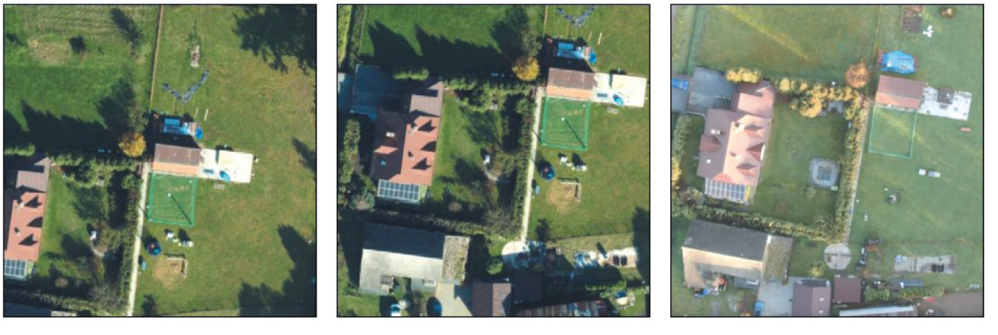

Fig. 5. Aligning images with Embed Match: a) initial guess; b) aligned; c) actual

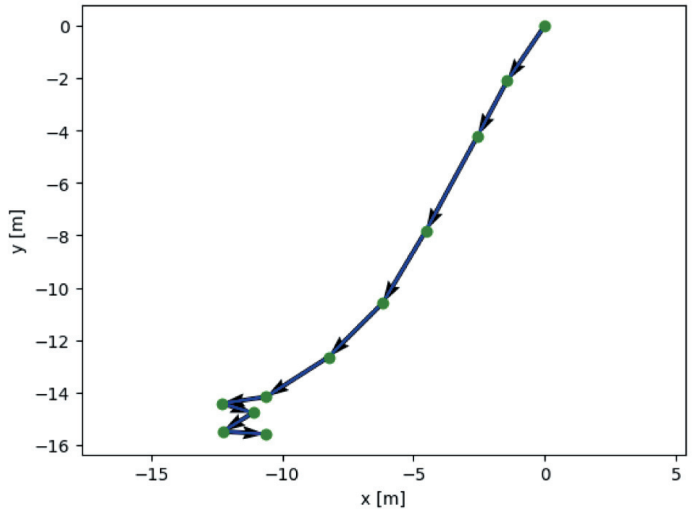

Fig. 6. Path taken by Embed Match

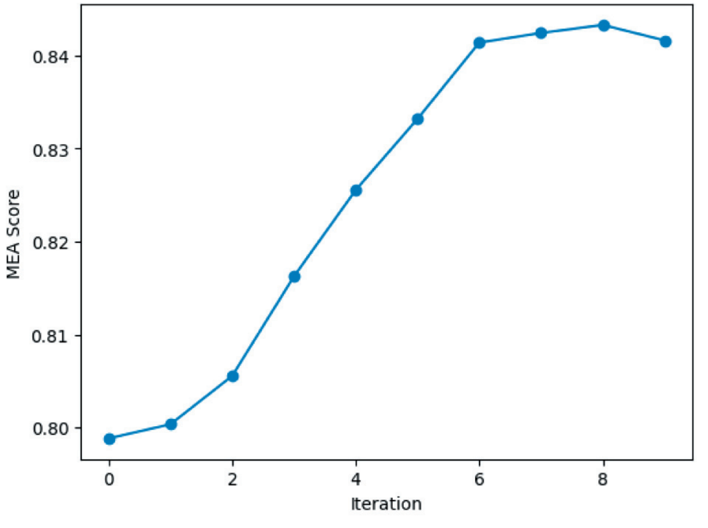

Fig. 7. Optimising MEA with gradient descent

5. Discussion

The results show that Embed Match successfully aligns drone images to a satellite map. The visual similarity of PCA-reduced embeddings can be questioned. Since PCA performs a reduction from 512 embedding features to 3 RGB values, most of the information is lost.

However, analysing the MEA function shows that comparing cosine similarity of patch embeddings is a valid method of aligning images. Only the relative comparison of embeddings is important. Further work could analyse the MEA score of images where one is covered by snow and other anomalies.

The algorithm could be improved by implementing a more advanced MEA optimisation method, such as Adam (Kingma and Ba 2015). Many other optimisation methods can be used, both gradient-based and gradient-free. It is also possible to use MEA to align an image without an initial position guess. In such case an optimisation method that can avoid local maxima has to be used.

The final result obtained with Embed Match demonstrates its effectiveness. The algorithm adjusts the initial position guess by 19 m, which is 30% of the input image width. This shows that Embed Match can be used to align the map created by a SLAM algorithm even with a large initial error.

6. Conclusions

This work explored the application of neural embedding models to the task of cross-domain image alignment. A novel algorithm called Embed Match is introduced. A mathematical proof is used to describe a visual top-down aerial SLAM algorithm using satellite imagery. The proposed solution guarantees a constant upper bound on the global position error of each point on the created map. The Embed Match algorithm introduces a new class of image alignment methods based on whole-image embedding similarity. Experiments are conducted that explore the behaviour of the proposed methods. The results show that the algorithm correctly aligns images. Future research directions for embedding-based image alignment algorithms are discussed.

References

Avola D., Cinque L., Fagioli A., Foresti G.L., Massaroni C., Pannone D., 2019, *Feature-Based SLAM Algorithm for Small Scale UAV with Nadir View*, [in:] Ricci E., Rota Bulň S., Snoek C., Lanz O., Messelodi S., Sebe N. (eds.), *Image Analysis and Processing – ICIAP 2019: 20th International Conference, Trento, Italy, September 9–13, 2019, Proceedings, Part II*, Lecture Notes in Computer Science, vol. 11752, Springer, Cham, pp. 457–467. https://doi.org/10.1007/978-3-030-30645-8_42.

Bay H., Tuytelaars T., Van Gool L., 2006, *SURF: Speeded Up Robust Features*, [in:] Leonardis A., Bischof H., Pinz A. (eds.), *Computer Vision – ECCV 2006: 9th European Conference on Computer Vision, Graz, Austria, May 7–13, 2006, Proceedings, Part I*, Lecture Notes in Computer Science, vol. 3951, Springer, Berlin, Heidelberg, pp. 404–417. https://doi.org/10.1007/11744023_32.

Campos C., Elvira R., Rodríguez J.J.G., Montiel J.M.M., Tardós J.D., 2021, *ORB-SLAM3: An accurate open-source library for visual, visual-inertial, and multimap SLAM*, IEEE Transactions on Robotics, vol. 37(6), pp. 1874–1890. https://doi.org/10.1109/TRO.2021.3075644.

Gupta A., Fernando X., 2022, *Simultaneous localization and mapping (SLAM) and data fusion in unmanned aerial vehicles: Recent advances and challenges*, Drones, vol. 6(4), 85. https://doi.org/10.3390/drones6040085.

Kingma D.P., Ba J.L., 2015, *Adam: A method for stochastic optimization*, [in:] *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7–9, 2015, Conference Track Proceedings*. https://doi.org/10.48550/arXiv.1412.6980.

Liu L., Aitken J.M., 2023, *HFNet-SLAM: An accurate and real-time monocular SLAM system with deep features*, Sensors, vol. 23(4), 2113. https://doi.org/10.3390/s23042113.

Macario Barros A., Michel M., Moline Y., Corre G., Carrel F., 2022, *A comprehensive survey of visual SLAM algorithms*, Robotics, vol. 11(1), 24. https://doi.org/10.3390/robotics11010024.

Quach C.H., Phung M.D., Le H.V., Perry S., 2021, *SupSLAM: A robust visual inertial SLAM system using SuperPoint for unmanned aerial vehicles*, [in:] Bao V.N.Q., Luong M.D. (eds.), *2021 8th NAFOSTED Conference on Information and Computer Science (NICS): NICS 2021: Proceedings: December 21–22, 2021, Hanoi, Vietnam*, IEEE, Piscataway, pp. 507–512. https://doi.org/10.1109/NICS54270.2021.9701527.

Radford A., Kim J.W., Hallacy C., Ramesh A., Goh G., Agarwal S., Sastry G., Askell A., Mishkin P., Clark J., Krueger G., Sutskever I., 2021, *Learning transferable visual models from natural language supervision*, [in:] Meila M., Zhang T. (eds.), *Proceedings of the 38th International Conference on Machine Learning: 18–24 July 2021, Virtual*, Proceedings of Machine Learning Research, vol. 139, pp. 8748–8763. https://doi.org/10.48550/arXiv.2103.00020.

Rublee E., Rabaud V., Konolige K., Bradski G., 2011, *ORB: An efficient alternative to SIFT or SURF*, [in:] Metaxas D.N., Quan L., Sanfeliu A., Van Gool L. (eds.), *IEEE International Conference on Computer Vision, ICCV 2011, Barcelona, Spain, November 6–13, 2011*, IEEE, Piscataway, pp. 2564–2571. https://doi.org/10.1109/ICCV.2011.6126544.