Security of Infocommunication Systems and Internet of Things

2025 Vol 3, No 2

https://doi.org/10.31861/sisiot2025.2.02006

Received 09 April 2025; revised 03 November 2025; accepted 10 November 2025; published 30 December 2025

Overview of Visual Position Recognition

Andriy Kramar*

Department of Computer Systems and Technologies, Private Higher Educational Institution "Bukovinian University", Chernivtsi, Ukraine

*Corresponding author (E-mail: andriykramar@gmail.com)

ABSTRACT One of the promising areas in the development of artificial intelligence is the creation of computer vision technology – a system that enables computerized systems to acquire, analyze, and interpret information from photos, videos, or digital images. This opens up extensive opportunities for process automation across various fields, including robotics, autonomous transportation, industry, and medicine. One of the key challenges in computer vision research is the problem of visual position recognition – assessing a robot's coordinates and orientation based on video or photo data obtained from its cameras. In robotic systems, precise position recognition is critical for navigation, adaptation to environmental changes, and interaction with objects. The article attempts to formulate the problem in terms of approximating the probability density function of the robot's states within the space of input data. In addition to theoretical aspects, the study examines a set of algorithms currently in use – both classical approaches and neural network-based models – their universality, and their integration potential with other computer vision technologies. The interpretation of these algorithms is presented from the perspective of dimensionality reduction in the input data space during localization. Furthermore, a list of relevant datasets for training and testing visual position recognition models is provided, along with key metrics for evaluating their performance. Thus, the study not only summarizes modern approaches to solving this problem but also outlines directions for further technological advancements that can ensure more efficient and accurate robot localization in space.

KEYWORDS computer vision, visual position recognition, object recognition, machine learning, neural networks.

I. INTRODUCTION

Visual Position Recognition (VPR) is an important section in modern computer vision systems that uses visual information to determine the position and orientation of objects in space.

The main purpose of VPR is to determine the exact location of the robot in a given coordinate system. To do this, compare current images (for example, obtained from the robot's cameras) with a set of reference images or maps (or sets of descriptors prepared based on them in a certain way).

This method has applications in various fields such as robotics, autonomous vehicles, and augmented and virtual reality. Examples of VPR applications:

- Robotics – a practical example of the application of VPR methods is modern robot vacuum cleaners: the models are equipped with an IR camera, which allows the robot to determine its position in the room based on the image of the surroundings;
- Autonomous vehicles – there are experimental courier delivery robots that use VPR to refine GPS data;
- Augmented and virtual reality – "Google Lens" AR app in museums: VPR helps the app recognize paintings, sculptures, or other objects and offer contextual information.

Tourist navigation systems (e.g., in the Louvre): The systems are guided by the VPR to show where the main exhibits are, such as the Mona Lisa.

This article outlines the main concepts and approaches used to solve this type of problem, as well as the advantages and disadvantages of each. Familiarizing oneself with existing VPR implementation examples will enable the selection of a strategy and the construction of an optimal technology for solving specific targeted tasks.

II. PROBLEM STATEMENT

Formally, the problem can be reduced to constructing a probability distribution function ρ on the input vector space $\{(V, x, y, z, \alpha, \beta, \gamma)\}$, where V is the set of pixel values of the image obtained from the robot's cameras, x, y, z are the coordinates in some defined reference frame, α, β, γ are the angles that describe the orientation of the robot's eigenaxes relative to the axes of this reference frame. Knowing the distribution function, it is quite easy to obtain the coordinates and orientation of the robot based on the image vector V from the cameras:

$$\begin{pmatrix} \bar{x} \\ \bar{y} \\ \bar{z} \\ \bar{\alpha} \\ \bar{\beta} \\ \bar{\gamma} \end{pmatrix}_V = \frac{\int_S \begin{pmatrix} x \\ y \\ z \\ \alpha \\ \beta \\ \gamma \end{pmatrix} \rho(V,x,y,z,\alpha,\beta,\gamma)dx\,dy\,dz\,d\alpha\,d\beta\,dy}{\int_S \rho(V,x,y,z,\alpha,\beta,\gamma)dx\,dy\,dz\,d\alpha\,d\beta\,dy}. \quad (1)$$

Here, integration is carried out over S – the whole range of possible values of $x, y, z,$ and α, β, γ: $(x, y, z, \alpha, \beta, \gamma) \in S$.

The learning process of a model, in fact, is a process of approximating ρ over a set of known vectors from a learning set.

As wide as the possible applications of VPR systems are, so diverse are the options for their design (construction) and the requirements for them, and therefore for the desired distribution function. Let's consider a few of the most characteristic cases.

A. Ground robot delivery service. Such a robot can be equipped with several cameras oriented in the direction of movement and for viewing the environment. The robot can have a set of auxiliary sensors, such as LIDAR or TOF, to determine the distances to objects around. The algorithm

should take into account the change in perspective when moving, and the parameters of the camera lenses.

B. Flying robot delivery service. The robot can have a camera directed downwards, as well as in the direction of travel, with auxiliary sensors – typically, these include a compass, accelerometer, and altimeter. A change in flight altitude changes the scale of the observed objects, and a change in the inclination of a robot in flight changes its perspective projection. The view of the cameras can be blocked by cloudiness or fog, landscape details can be covered with snow, change color with the seasons, etc.

From the above examples, it can be seen that ρ should give close to true values under the condition of significant distortion in the input data.

Another significant difficulty lies in the enormous dimensionality of the space of the input vectors: having an RGB image of 320 × 240 at the input, we get a vector with 320 × 240 × 3 + 6 = 230406 components. An obvious way to simplify is to reduce the dimensionality, for example, by searching and considering only the position of certain key points or zones in the input image. Indeed, having N key zones and 2 of their coordinates in the image, we get the dimension of the input vector $2 \times N + 6$, which at certain values of N can be significantly less than the indicator given above.

III. FEATURES AND KEY AREAS APPROACHES

Many VPR methods are based on highlighting key areas or features such as corners, edges, or texture elements. Notable algorithms include: SIFT, SURF, ORB, and others.

A. Scale-Invariant feature transform (SIFT). This algorithm is used to identify, describe, and compare key points (features) in the images [1, 2]. Its main advantage is that it is invariant to changes in scale, rotation, and even partial changes in lighting or perspective. The main stages of SIFT's work:

- Key Point Detection: the algorithm detects special points (such as corners, edges, or other unique areas) in an image by searching for extremums in scale space. To create a large-scale space, Gaussian blur is used at different scales;
- Key Point Localization: the algorithm discards faint (low-contrast) points or those near the edges to leave only the most significant features;
- Orientation of key points: for each point, its orientation is determined based on the direction of the gradients in adjacent pixels. This makes the algorithm invariant to rotations;
- Descriptor formation: a descriptor is created around each key point, a vector that describes the local characteristics of the image. Typically, a histogram of gradient orientations is used;
- Key Point Comparison: Key point descriptors from different images are compared to find similar points. This allows you to detect matches between images.

Advantages: Invariance to scale and rotation, resistance to changes in lighting and noise, versatility for various tasks in computer vision.

Disadvantages: High computational complexity, patent restrictions (until 2020, the algorithm was patented).

For tasks where speed is required, alternatives are often used, such as Oriented FAST and Rotated BRIEF (ORB), which is faster but less accurate.

B. Speeded-up robust features (SURF). SURF is a faster and more efficient alternative to SIFT. It is described in detail in [3] and its open-source implementations are compared in [4]. SURF has found applications in many tasks, such as object recognition, image stitching [5], building 3D scenes, tracking objects, etc. Some comparative analysis of SIFT and SURF methods for local feature detection in satellite imagery is presented in [6].

Key features of SURF:

- Scale-invariance: SURF can detect key points, regardless of changes in object scale;
- Rotational invariance: The algorithm remains resistant to object rotations;
- Speed: SURF uses integral images and efficient mathematical methods, making it faster than SIFT;
- Key Point Detection: SURF uses Gaussian filter-based techniques to identify features in an image;
- Description of key points: The algorithm creates a feature vector that describes the detected points, which helps to identify them between images.

The main stages of SURF's work:

- Key points detection: The Hessian matrix is used to search for points with high intensity;
- Orientation of key points: The algorithm determines the orientation of each point to ensure invariance to rotation;
- Descriptor computation: A compact descriptor is created that represents the local structure around a key point;
- Descriptor comparison: Descriptors from different images are compared to identify matching key points.

SURF is actively used in the following tasks:

- Object recognition;
- Create panoramas: Stitch images together by key points;
- Computer graphics and augmented reality;
- Video analysis for object tracking.

SURF is a popular algorithm, although with the advent of more modern deep learning-based techniques (such as ORB or deep learning features), its use is gradually decreasing.

The review discusses various methods of isolating features for visual recognition tasks, among which SURF is one of the most famous.

C. Oriented FAST and rotated BRIEF. This method is highlighted in [7, 8]. It combines two main stages:

- Features from Accelerated Segment Test (FAST) is an algorithm for identifying key points. It focuses on finding points that are stable and unique to the image;
- Binary Robust Independent Elementary Features (BRIEF) is a method for describing these points. BRIEF generates binary features for each key point, which can then be used to compare different images.

How FAST works:

- Pixel analysis around each point: The algorithm looks at each pixel of the image and checks it against the 16 pixels that are located in a circle around it;

- Pixel "curiosity" check: For each pixel point, it is determined whether it is significantly different from neighboring pixels, i.e., whether it is significantly lighter or darker than them. If this is the case, then the point is considered potentially interesting.

Advantages: speed (FAST is optimized for fast processing, making it suitable for real-time), simplicity (the algorithm does not require complex calculations and can work even with a large number of pixels).

Disadvantages: Sensitivity to orientation bias – FAST does not take into account the orientation of the image, so it is not resistant to rotation. Compared to other methods, such as SIFT or SURF, FAST may be less accurate in determining the best points to describe.

BRIEF is an algorithm for creating compact and efficient key point descriptors in images. Its main goal is to describe the identified key points in the form of binary vectors that can be quickly compared with each other to find similar points.

How BRIEF works:

- Key Point Selection: Initially, the algorithm uses a different method (such as FAST) to find the key points in the image;
- Patch around a point: For each key point, BRIEF looks at a small square patch (e.g., 31 × 31 pixels) around it;
- Pixel Pair Selection: Within this patch, BRIEF generates a large number of random pixel pairs;
- Pixel comparison: For each pair of pixels, a comparison of their intensity values is performed - if the intensity of the first pixel is greater than the intensity of the second, the result = 1. If less, the result = 0. This creates a binary vector (e.g., 128 or 256 bits) that is unique to each key point;
- Key Point Description: A binary vector is used as a descriptor for a given key point.

ORB improves BRIEF by adjusting it to handle changes in image orientation. This allows the points to be stable even when the orientation of the image changes. ORB is fast and efficient, so it is widely used for tasks such as object tracking, 3D reconstruction, image detection, and comparison.

IV. USES OF NEURAL NETWORKS

In recent years, deep neural networks (Convolutional Neural Networks (CNNs), Transformers) have been actively used for VPR, which allows detecting complex relationships between pixels and improving localization accuracy [9]. Their applications include:

- Feature Extraction;
- Deep neural networks (e.g., CNNs) are used to automatically isolate features from images. Such features are resistant to changes in lighting, angles, or weather conditions.

Creating Descriptors. Neural networks create compact vector representations (descriptors) of images that facilitate comparison with a reference database [10].

Comparison with the database. With similarity metrics such as cosine distance or Euclidean distance, neural networks help identify the most similar locations in the database [11].

Handling changes in the environment. Recurrent Neural Networks or Transformer architectures can take into account the sequence of images to determine location, even with significant changes in the environment (e.g., seasonal changes) [12].

Self-Supervised Learning. The use of self-learning approaches allows you to model VPRs without the need for large amounts of manual data labeling [13].

Online updates. Neural networks can dynamically update their models in real time, adapting to new data or changes in the environment [14].

A. You Only Look Once (YOLO). A popular algorithm for computer vision problems that is used to detect objects in images or videos. Its key feature is that it performs object detection in a single pass through a neural network, making it extremely fast and efficient in real-time [15]. Key Features of YOLO:

- Simultaneous localization and classification: YOLO splits the image into a grid (e.g., 13×13, 19×19, etc.). For each cell, it provides: the probability of the presence of an object; object class (e.g., person, car, dog, etc.); coordinates of the object's frame (bounding box);
- Speed: With a single pass (simultaneous processing of the entire image), YOLO can process real-time video, which is important for applications such as offline driving, video surveillance, and others;
- Global Context: The algorithm takes into account all image information, not just local areas, which helps reduce false positives;
- Architecture: YOLO is based on CNNs. In different versions of YOLO (v1, v2, v3, v4, v5, v6, etc.), the architecture is constantly being improved to improve accuracy and performance.

How does YOLO work:

- The image is divided into a uniform grid;
- Bounding boxes and object classes are provided for each cell;
- The algorithm uses filters to identify objects and determine which class they belong to;
- NMS (non-maximum suppression) functions are used to eliminate overlapping boxes and leave only the most relevant ones.

Advantages of YOLO:

- High speed (suitable for real-time);
- High accuracy in detecting large objects;
- Ease of use and customization.

Disadvantages of YOLO:

- Less accurate in detecting small objects;
- Can skip objects if they are too close together.

B. Using pre-trained models. Popular models such as ResNet, VGG, or NetVLAD adapt to the VPR task, providing high accuracy in difficult conditions [16].

V. METHODS BASED ON SEMANTIC SEGMENTATION

Semantic segmentation in VPR tasks is a powerful approach to improve the accuracy of place recognition, especially in difficult environments (changing lighting, seasons, partial obstacles, etc.).

Semantic segmentation is a computer vision task that consists of dividing an image into segments that belong to

certain categories (for example, buildings, roads, trees, sky, etc.). In the context of a VPR, this allows you to highlight relevant objects in the scene and ignore unnecessary "noise" (e.g., dynamic objects like people or vehicles).

How Is Semantic Segmentation Used in VPR:

1. Filter out irrelevant objects. The scene image is segmented, and objects that might interfere with the recognition of the place (such as cars or people) are excluded. This reduces the impact of dynamic elements;
2. Selecting stable semantic objects. Elements that are highly resistant to change (e.g., buildings, roads, trees) are used to create place descriptors. This helps to create a more reliable representation of the scene;
3. Semantic-weighted descriptors. Semantic information can be integrated into image descriptors used to compare places. For example, you can increase the weight of pixels related to important objects (buildings or roads) and reduce the weight of less important objects (grass or sky);
4. Adaptation to environmental changes. Semantic segmentation allows you to recognize places even with significant changes in the environment. For example, if a building is covered with snow or shade, its semantic category will remain unchanged.

Methods based on semantic segmentation:

1. SegMatch. This is one of the early methods that uses semantic labels to compare 3D points in the cloud. In the context of VPR, SegMatch isolates objects from point clouds and compares them based on semantic matching [17];
2. Semantic VPR. Methods in this category integrate semantic information into CNN, adding an extra layer of generalization. For example, segmented maps are used instead of the original images [18];
3. Semantically Consistent Feature Matching. This method first performs semantic segmentation of the scene, and then only objects that belong to the same categories (for example, only buildings with buildings) are compared. This reduces the number of false positives [19];
4. Hierarchical Semantic Mapping. Combines semantic segmentation with a hierarchical map structure to recognize locations in large and complex environments [20].

VI. INTERPRETATION OF THE OPERATION OF DESCRIPTOR PRODUCING ALGORITHMS IN TERMS OF PROBABILITY DENSITY DISTRIBUTION

Typically, a descriptor is a (128-bit) number. So, in the process of operation of the SIFT, SURF, and ORB algorithms, the input matrix V of the image is converted into a matrix of descriptors:

$$D(V) = \\ = D\begin{pmatrix} (r\ g\ b)_{00} & \cdots & (r\ g\ b)_{0\,W-1} \\ \vdots & \ddots & \vdots \\ (r\ g\ b)_{H-1\,0} & \cdots & (r\ g\ b)_{H-1\,W-1} \end{pmatrix} = \\ = \begin{pmatrix} 0 & d1_{s1y\,s1x} & 0 \cdots 0 & 0 & 0 \\ & & \dots & & \\ 0 & 0 & 0 \cdots 0 & dN_{sNy\,sNx} & 0 \\ & & \dots & & \\ 0 & 0 & 0 \dots 0 & 0 & 0 \end{pmatrix}. \quad (2)$$

Here, D is the function of finding descriptors, r_{ij}, g_{ij}, b_{ij} is the value of R, G, B, the component of the pixel in the row $i \in [0, H-1]$, column $j \in [0, W-1]$ of the input image matrix with a width of W and a height of H pixels, $d1$ is the value of the first detected descriptor, sX_1, sY_1 are its screen coordinates along the X and Y axis, d_N is the value of the Nth detected descriptor, and sX_N, sY_N are its screen coordinates.

Most of the elements of the resulting matrix are zeros, so it is represented in a compact form:

$$\mathrm{C}(D(V), N) = \mathrm{C}\begin{pmatrix} 0 & d1_{sY1\,sX1} & 0 \cdots 0 & 0 & 0 \\ & & \dots & & \\ 0 & 0 & 0 \cdots 0 & dN_{sYN\,sXN} & 0 \\ & & \dots & & \\ 0 & 0 & 0 \dots 0 & 0 & 0 \end{pmatrix} = \\ = C_V = \begin{pmatrix} (d_1\ sX_1\ sY_1) \\ \dots \\ (d_N\ sX_N\ sY_N) \end{pmatrix}. \quad (3)$$

Here, C is an optimizer function – it selects N "strongest" descriptors, and returns them in the form of a matrix C_V shown in (3).

The desired probability density distribution now needs to be determined on the space of values C_V, x, y, z, α, β, γ:

$$\rho_V(V, x, y, z, \alpha, \beta, \gamma) \to \rho_C(C_V, x, y, z, \alpha, \beta, \gamma). \quad (4)$$

Suppose that the descriptors d_k, d_q are formed by compact regions with global coordinates of the center x_k, y_k, z_k and x_q, y_q, z_q. (see Fig. 1).

Their on-screen coordinates are determined by projection on the camera plane oriented according to the position of the observer (x_o, y_o, z_o, α_o, β_o, γ_o):

$$(sx_k \quad sy_k) = \Pr(x_k, y_k, z_k, x_o, y_o, z_o, \alpha_o, \beta_o, \gamma_o), \quad (5)$$

$$(sx_q \quad sy_q) = \Pr(x_q, y_q, z_q, x_o, y_o, z_o, \alpha_o, \beta_o, \gamma_o). \quad (6)$$

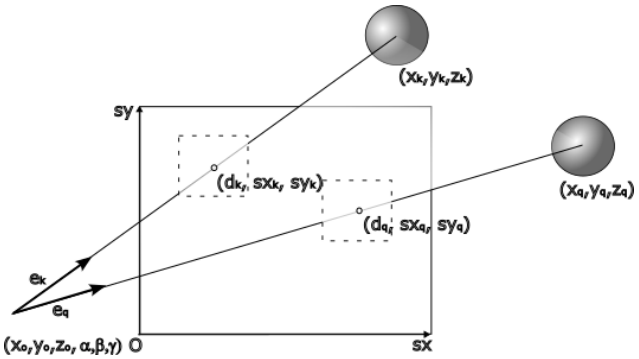

FIG. 1. Directional unit vectors.

Inversely, the on-screen coordinates and orientation of the observer allow us to obtain unit guide vectors **ek**, **eq**, indicating the centers of the regions:

$$(e_{kx} \quad e_{ky} \quad e_{kz}) = Rp(sx_k, sy_k, \alpha, \beta, \gamma), \quad (7)$$

$$(e_{qx} \quad e_{qy} \quad e_{qz}) = Rp(sx_q, sy_q, \alpha, \beta, \gamma). \quad (8)$$

That is, each descriptor corresponds to its unit vector of direction. Let's form a matrix E from these vectors. Then the probability density distribution function can be rewritten as:

$$\rho_C(C, x, y, z, \alpha, \beta, \gamma) = \rho_E(E, x, y, z, \alpha, \beta, \gamma). \quad (9)$$

For geometric reasons (see Fig. 2), it can be represented as:

$$\rho_E(E,x,y,z,\alpha,\beta,\gamma) = \prod_{k=1}^{N} \rho_{ek}(\vec{e}_k,x,y,z,\alpha,\beta,\gamma). \quad (10)$$

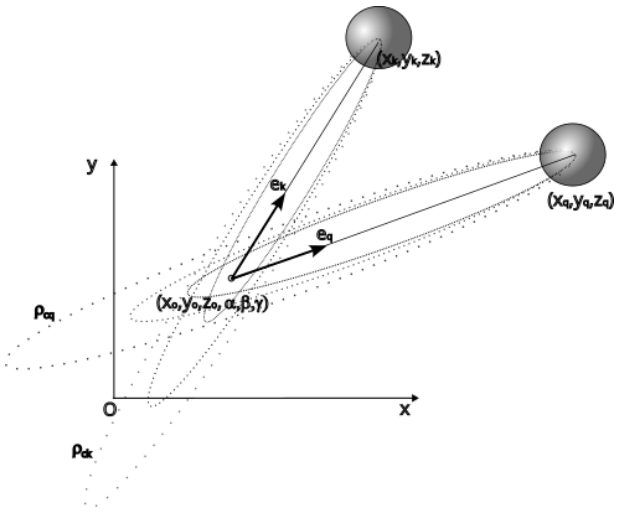

FIG. 2. Probability distributions.

Here, ρ_{ek} is the probability distribution of the observation of descriptor $C.d_k$ in the direction of $\boldsymbol{e_k}$ being at position (x, y, z, α, β, γ).

Returning to the representation in terms of descriptors:

$$\rho_C(C,x,y,z,\alpha,\beta,\gamma) = \prod_{k=1}^{N} \rho_{ck}(C_k,x,y,z,\alpha,\beta,\gamma). \quad (11)$$

Here:

$$\rho_{ck}(C_k,x,y,z,\alpha,\beta,\gamma) = \\ = \rho_{ek}(Rp(sx_k,sy_k,\alpha,\beta,\gamma),x,y,z,\alpha,\beta,\gamma). \quad (12)$$

Let's say we have a training set of images and coordinates $\boldsymbol{L} = \{(Ls, x_s, y_s, z_s, \alpha_s, \beta_s, \gamma_s)\}$, $s \subset [1, S]$, where S is the number of samples.

First, the algorithm tries to match the image Ls from $\boldsymbol{L}$ by the available descriptors set C, and then refines the position, taking into account changes in the screen coordinates of the descriptors.

In view of the above, the approximate probability density distribution can be written as follows:

$$\rho_C(C,x,y,z,\alpha,\beta,\gamma) = \\ \sum_{s=1}^{S} \prod_{k=1}^{N} \delta(C.d_k, Ls.d_k) \rho_{csk}(sx_k, sy_k, x, y, z, \alpha, \beta, \gamma). \quad (13)$$

Here δ is the delta character ($\delta(a, b) = 1$ if $a = b$, $\delta(a, b) = 0$ if $a \neq b$), $C.d_k$, $Ls.d_k$ is the value of the descriptor from the k-th line of the matrix C or Ls, respectively.

The ρ_{csk} functions are unknown, but they can be approximated by comparing images obtained from the same location (with one set of descriptors) at small position offsets.

From consideration of Eq. (13), it is already possible to make certain remarks on algorithms built on the definition of descriptors (features):

- if the input matrix C has a set of descriptors not present in any of the training samples, all $\delta = 0$, and therefore $\rho_c = 0$ – the position cannot be determined. Such a situation may arise, for example, when there is a significant change in the contrast of the observed area of the environment due to changes in lighting or weather conditions.
- The quality of the ρ_{csk}: approximation is a key factor in improving position accuracy.

VII. VPR TRAINING AND TESTING DATASETS

One of the key resources needed to successfully build a VPR system is datasets (datasets) for training for testing. Here are some popular sets:

1. Oxford RobotCar Dataset: A large dataset containing images from different locations in the city of Oxford, including different weather conditions [21];
2. Mapillary Vistas Dataset: A dataset containing millions of images from different locations around the world, with different lighting and weather conditions [22];
3. KITTI Dataset: A dataset used to train and test computer vision systems for automobiles, including images from various cameras [23];
4. VP-Air Dataset: These datasets are important for training and testing VPR algorithms because they provide a variety of images and conditions, allowing you to test their effectiveness and accuracy [24].

Datasets for creating VPR models are often supplemented with synthetic data. Synthetic data can be useful when real-world data is limited or when models need to be improved to handle different conditions or environments.

The main advantages of using synthetic data for VPR:

1. Variety of situations: Synthetic data can generate different scenarios that may not be present in real data, such as different weather conditions, changes in lighting, or variations in camera position;
2. Increase in data volume: The use of synthetic data allows you to create large datasets for training, which increases the overall efficiency of the models;
3. Condition Control: Conditions such as viewing angle, distance, and surface textures can be precisely controlled, which helps in testing and customizing models for specific tasks.

Synthetic data can be generated through graphics engines (such as Unity or Unreal Engine), allowing you to create varied, photorealistic scenes.

VIII. METRICS FOR EVALUATING THE QUALITY OF VPR MODELS

To assess the quality of VPR models, both commonly used metrics (Accuracy, Precision, Recall, F1 Score, Mean Average Precision, Intersection over Union, Root Mean Squared Error) and more specialized ones are used.

A. Localization Error. This metric measures the deviation between the predicted and real position of the object in the image. It is important for assessing the accuracy of determining the location of objects.

B. Orientation Error. This metric assesses how accurately the model determines the orientation of an object relative to its real position. This is critical for applications where object orientation is of great importance, such as in autonomous vehicles.

C. Pose Estimation Error. This metric is used to assess the accuracy of determining the posture (position and orientation) of an object in three-dimensional space. It is important for robotics and other industries where three-dimensional recognition is necessary.

D. Recall @N. This metric measures the proportion of correct predictions among the N highest (ranked) predictions of the model. For example, Recall @5

measures how well the model identifies the correct objects among the top five predictions.

E. Top-K Accuracy. This metric evaluates the accuracy of the model, taking into account the first K predictions. It is useful for evaluating patterns in problems where multiple correct answers may be acceptable.

F. Cumulative Match Characteristic (CMC) Curve. The CMC curve shows the probability that the correct object will be found among the first N predictions of the model. This metric helps to evaluate the effectiveness of the model in problems where it is important to find the right object among a large number of possible options.

These metrics help to assess the quality of VPR models in more detail, taking into account the specific requirements and applications of this technology.

IX. CONCLUSIONS

The analysis of publications on the topic discussed here indicates that as of January 2025, VPR continues to actively develop, focusing on increasing accuracy, universality, and efficiency. At the same time, recent achievements and trends can be noted:

- Versatility and adaptability: Laborers like AnyLoc strive to create versatile VPR solutions that are able to operate in a variety of environments without the need for retraining. This approach allows systems to adapt effectively to new environments, which is essential for real-world applications [25];
- Integration with other technologies: Combined studies like STA-VPR show that combining VPR with other methods, such as dynamic time alignment, can significantly improve the accuracy and robustness of systems [26];
- Expanding applications: Other works, such as PanoVPR, extend the capabilities of VPR, allowing you to work efficiently with panoramic images, which opens up new horizons for use in various fields, including autonomous driving and robotics [27];
- Efficiency improvement: Models like PRAM focus on improving the efficiency of VPRs by reducing memory and processing time requirements, making them more suitable for real-world applications [13].

Focusing efforts on increasing universality, efficiency, and integration with other technologies opens up new opportunities for the application of VPR in the real world. Specifically, it can be expected that in the coming years, VPR will become more integrated with other technologies, such as artificial intelligence and the Internet of Things, allowing for the creation of more complex and adaptive systems for various applications – from autonomous transportation to security systems in “smart cities”.

ACKNOWLEDGMENT

I would like to express my sincere gratitude to Prof. Serhiy Ostapov for their valuable advice and insightful suggestions during the development of this work. Their expertise and guidance greatly contributed to the understanding and refinement of the subject.

AUTHOR CONTRIBUTIONS

A.K. – conceptualization, methodology, investigation; writing (original draft preparation), writing (review and editing).

COMPETING INTERESTS

The author declares no competing interests.

REFERENCES

[1] D. G. Lowe, “Distinctive image features from scale-invariant keypoints,” [Online]. Available: https://www.cs.ubc.ca/~lowe/papers/ijcv04.pdf

[2] T. Lindeberg, “Scale invariant feature transform,” [Online]. Available: https://www.researchgate.net/publication/235355151_Scale_Invariant_Feature_Transform

[3] H. Bay, A. Ess, T. Tuytelaars, and L. V. Gool, “SURF: Speeded up robust features,” 2008. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1077314207001555

[4] D. Gossow, P. Decker, and D. Paulus, “An evaluation of open source SURF implementations,” 2010. [Online]. Available: https://doi.org/10.1007/978-3-642-20217-9_15

[5] E. Abbadi and A. Hassani, “Panoramic image stitching techniques based on SURF and singular value decomposition,” 2022. [Online]. Available: https://doi.org/10.1007/978-3-030-93417-0_5

[6] A. Riabko and Y. Averyanova, “Comparative analysis of SIFT and SURF methods for local feature detection in satellite imagery,” 2024. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1077314214000391

[7] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, “ORB: An efficient alternative to SIFT or SURF,” 2011. [Online]. Available: https://ieeexplore.ieee.org/document/6126544

[8] C. Campos, R. Elvira, J. J. Gómez Rodríguez, J. M. M. Montiel, and J. D. Tardós, “SLAM3: An accurate open-source library for visual, visual-inertial and multi-map SLAM,” [Online]. Available: https://arxiv.org/abs/2007.11898

[9] V. P. Lysechko, B. I. Sadovnykov, O. M. Komar, and O. S. Zhuchenko, “A research of the latest approaches to visual image recognition,” [Online]. Available: https://pdfs.semanticscholar.org/2cc6/befc9db461b20f4cae44a54707ed1257a1d3.pdf

[10] B. Ferrarini, M. Milford, K. D. McDonald-Maier, and S. Ehsan, “Binary neural networks for memory-efficient and effective visual place recognition in changing environments,” [Online]. Available: https://arxiv.org/pdf/2010.00716

[11] S. Dhar, “Visual place recognition. Introduction,” [Online]. Available: https://medium.com/@sd5023/visual-place-recognition-8999307ebb2f

[12] S. Hussaini, M. Milford, and T. Fischer, “Spiking neural networks for visual place recognition via weighted neuronal assignments,” [Online]. Available: https://arxiv.org/pdf/2109.06452

[13] F. Xue, I. Budvytis, and R. Cipolla, “PRAM: Place recognition anywhere model for efficient visual localization,” [Online]. Available: https://arxiv.org/pdf/2404.07785

[14] S. Hussaini, M. Milford, and T. Fischer, “Applications of spiking neural networks in visual place recognition,” [Online]. Available: https://arxiv.org/pdf/2311.13186

[15] C.-Y. Wang, I.-H. Yeh, H.-Y. M. Liao, and C. Yuan, “YOLOv9: Learning what you want to learn using programmable gradient information,” [Online]. Available: https://arxiv.org/pdf/2402.13616

[16] “Visual place recognition – Papers with Code,” [Online]. Available: https://paperswithcode.com/task/visual-place-recognition

[17] R. Dube, D. Dugas, E. Stumm, and J. I. Nieto, “SegMatch: Segment based place recognition in 3D point clouds,” [Online]. Available: https://www.researchgate.net/publication/318693876_SegMatch_Segment_based_place_recognition_in_3D_point_clouds
[18] S. Arshad, “SVS-VPR: A semantic visual and spatial information-based hierarchical visual place recognition for autonomous navigation in challenging environmental conditions,” 2024. [Online]. Available: https://www.mdpi.com/1424-8220/24/3/906
[19] K. Song, S. Zhang, Z. An, Z. Luo, T. Wang, and J. Xie, “Semantics-consistent feature search for self-supervised visual representation learning,” [Online]. Available: https://arxiv.org/pdf/2212.06486
[20] B. Chen, X. Song, H. Shen, and T. Lu, “Hierarchical visual place recognition based on semantic-aggregation,” 2020. [Online]. Available: https://www.mdpi.com/2076-3417/11/20/9540
[21] Oxford Robotics Institute, “Oxford RobotCar Dataset,” [Online]. Available: https://robotcar-dataset.robots.ox.ac.uk/
[22] Meta Platforms Ireland Limited, “Mapillary Vistas Dataset,” [Online]. Available: https://www.mapillary.com/dataset/vistas
[23] A. Geiger, P. Lenz, and R. Urtasun, “Vision meets robotics: The KITTI dataset,” 2012. [Online]. Available: https://www.cvlibs.net/datasets/kitti/
[24] M. Schleiss, F. Rouatbi, and D. Cremers, “VPAIR: Aerial visual place recognition and localization in large-scale outdoor environments,” 2022. [Online]. Available: https://github.com/AerVisLoc/vpair
[25] N. Keetha, A. Mishra, J. Karhade, K. M. Jatavallabhula, S. Scherer, M. Krishna, and S. Garg, “AnyLoc: Towards universal visual place recognition,” [Online]. Available: https://arxiv.org/pdf/2308.00688
[26] F. Xue, B. Chen, X.-D. Zhou, and D. Song, “STA-VPR: Spatio-temporal alignment for visual place recognition,” [Online]. Available: https://arxiv.org/abs/2103.13580
[27] Z. Shi, H. Shi, K. Yang, Z. Yin, Y. Lin, and K. Wang, “PanoVPR: Towards unified perspective-to-equirectangular visual place recognition via sliding windows across the panoramic view,” [Online]. Available: https://arxiv.org/abs/2303.14095

Andriy Kramar

Graduated from Chernivtsi National University in 2001, master of Physics, teacher of physics and computer science at the College of Private Higher Educational Institution "Bukovinian University" since 2024. Scientific interests: machine learning, genetic algorithms, and digital signal processing.

ORCID ID: 0009-0006-7371-4508

Огляд алгоритмів візуального визначення положення в просторі

Андрій Крамар*

Кафедра комп’ютерних систем і технологій/ Факультет інформаційних технологій та економіки/ПВНЗ «Буковинський університет», Чернівці, Україна

*Автор-кореспондент (Електронна адреса: andriykramar@gmail.com)

АНОТАЦІЯ Одним із перспективних напрямів розвитку штучного інтелекту є створення комп’ютерного зору (Computer Vision) – технології, що дозволяє комп’ютеризованим системам отримувати, аналізувати та інтерпретувати інформацію з фото, відео або цифрових зображень. Це відкриває широкі можливості для автоматизації процесів у різних сферах, зокрема в робототехніці, автономному транспорті, промисловості та медицині. Одним із актуальних викликів у дослідженнях комп’ютерного зору є проблема візуального визначення положення (Visual Position Recognition) робота у просторі, що включає оцінку його координат та орієнтації на основі відео- або фотоданих, отриманих з камер робота. В роботизованих системах точне визначення положення має критичне значення для навігації, адаптації до змін у середовищі та взаємодії з об’єктами. Це питання набуває особливої важливості в контексті мобільних роботів, зокрема роботів-прибиральників, автономних дронів і роботів-кур’єрів. У статті зроблено спробу сформулювати проблему з точки зору апроксимації густини розподілу ймовірності станів робота у просторі вхідних даних. Окрім теоретичних аспектів, розглянуто набір алгоритмів, що застосовуються на даний час (як класичних, так і на основі нейронних мереж), їхню універсальність та можливості інтеграції з іншими технологіями комп’ютерного зору. Подано інтерпретацію роботи зазначених алгоритмів з точки зору задачі зменшення розмірності простору вхідних даних при визначенні локалізації. Крім того, наведено список актуальних наборів даних для навчання та тестування моделей візуального визначення положення, а також ключові метрики для оцінювання їхньої ефективності. Таким чином, дослідження покликане не лише узагальнити сучасні підходи до вирішення задачі, а й окреслити напрями подальшого розвитку технологій, що можуть забезпечити більш ефективну та точну локалізацію роботів у просторі.

КЛЮЧОВІ СЛОВА комп'ютерний зір, візуальне розпізнавання положення, розпізнавання об'єктів, машинне навчання, нейронні мережі.