Neural Computing and Applications (2023) 35:16487–16508
https://doi.org/10.1007/s00521-023-08515-y

ORIGINAL ARTICLE

Environment modeling and localization from datasets of omnidirectional scenes using machine learning techniques

Sergio Cebollada[1,3] · Luis Payá[1] · Adrián Peidró[1] · Walterio Mayol[2] · Oscar Reinoso[1,3]

Received: 26 September 2022 / Accepted: 21 March 2023 / Published online: 20 April 2023

Abstract
This work presents a framework to create a visual model of the environment which can be used to estimate the position of a mobile robot by means of artificial intelligence techniques. The proposed framework retrieves the structure of the environment from a dataset composed of omnidirectional images captured along it. These images are described by means of global-appearance approaches. The information is arranged in two layers, with different levels of granularity. The first layer is obtained by means of classifiers and the second layer is composed of a set of data fitting neural networks. Subsequently, the model is used to estimate the position of the robot, in a hierarchical fashion, by comparing the image captured from the unknown position with the information in the model. Throughout this work, five classifiers are evaluated (Naïve Bayes, SVM, random forest, linear discriminant classifier and a classifier based on a shallow neural network) along with three different global-appearance descriptors (HOG, *gist*, and a descriptor calculated from an intermediate layer of a pre-trained CNN). The experiments have been tackled with some publicly available datasets of omnidirectional images captured indoors with the presence of dynamic changes. Several parameters are used to assess the efficiency of the proposal: the ability of the algorithm to estimate coarsely the position (hit ratio), the average error (cm) and the necessary computing time. The results prove the efficiency of the framework to model the environment and localize the robot from the knowledge extracted from a set of omnidirectional images with the proposed artificial intelligence techniques.

Keywords Machine learning · Hierarchical localization · Omnidirectional vision · Global-appearance description

1 Introduction

Over the past few years, omnidirectional imaging has become a widespread technology to solve localization tasks in mobile robotics, thanks mainly to the great quantity of information that omnidirectional systems can capture with only one snapshot [44]. Regarding how to extract relevant information from this type of images, the present work focuses on the use of global-appearance descriptors, which have been commonly used for these purposes.

Concerning the structure of the model, arranging the topological information hierarchically constitutes an efficient approach to carry out, subsequently, the localization process. This method consists in organizing the information in several layers, with different levels of granularity. The high-level layers permit a rough but fast localization and the low-level layers provide more accurate information which is used to refine the estimation. Some authors have proposed creating hierarchical maps, such as Valgren et al.

✉ Sergio Cebollada
sergio.cebollada@umh.es

Luis Payá
lpaya@umh.es

Adrián Peidró
apeidro@umh.es

Walterio Mayol
walterio.mayol-cuevas@bristol.ac.uk

Oscar Reinoso
o.reinoso@umh.es

[1] Department of Systems Engineering and Automation, Miguel Hernández University, Elche, Spain

[2] Department of Computer Science, University of Bristol, Bristol, UK

[3] Valencian Graduate School and Research Network for Artificial Intelligence, Valencian Community, Spain

[53], who tackle an on-line topological mapping through the use of incremental spectral clustering or Shi et al. [50], who propose the use of a differential clustering method to improve the compression of telemetry data. In previous works ([37] and [8]), a clustering approach to compact the model was proposed with the objective of creating a two-layer hierarchical structure. The experiments showed that the proposal is a feasible alternative to build robust compact models.

In recent years, due to the emergence of more efficient hardware devices, artificial intelligence (AI) techniques have contributed to solve a variety of problems in computer vision and mobile robotics, as in [11], where an automatic method based on thermal image processing and CNN is proposed for victim identification in post-disaster environments using a quadruped robot. In many applications, data augmentation constitutes a common solution when training the AI tools, as it can avoid overfitting while increases the training instances.

In the light of the above information, the aim of the present work is twofold: first, to retrieve the structure of the environment from a dataset composed of omnidirectional images captured along it, and second, to use this model to estimate the position of the robot. For this purpose, several machine learning techniques and global-appearance description approaches are used and their performance is studied. The efficiency of these tools will be evaluated through their ability to estimate robustly the position of the robot using the information stored in the model, which presents a trajectory structure.

To tackle the proposed evaluation, the unique source of information used to carry out modeling and localization is a dataset of images obtained by an omnidirectional vision sensor [47] installed on the mobile robot. A variety of publicly available datasets is used in the experiments, obtained from several indoor environments under real-operation conditions. The approach consists in the use of a variety of classifiers, data fitting neural networks and clustering algorithms, used in combination with global-appearance visual descriptors, to solve the localization problem, and their effectiveness is measured with several metrics: the ability to estimate coarsely the position (hit ratio), the average error (cm) of the final measurement and the computing time.

The novelty of the present work is a machine-learning-based hierarchical approach that is used to solve efficiently the localization task. In broad lines, the idea of the present work is (1) to train and use a classifier to estimate in which room or area the robot currently is (rough localization step) and (2) refining this localization in the retrieved room (fine localization step) by using data fitting neural networks. Both steps use holistic descriptors as input information.

Our main contributions in this work can be summarized as follows.

- We train some classifiers with holistic descriptors of omnidirectional images to retrieve the room or area where an input image was obtained.
- We evaluate the use of a clustering method to improve the rough localization step.
- We address the fine localization step by means of a set of data fitting neural networks which are trained to estimate the coordinates of the robot in the ground plane, and compare the results with previous approaches.
- We study the use of the proposed machine learning approaches to solve the hierarchical localization and measure the accuracy of the methods in position estimation.

The remainder of the paper is structured as follows. Section 2 introduces a review of the related literature. Section 3 outlines the machine learning tools used throughout this work. After that, Sect. 4 gives details for the machine learning techniques and the global-appearance descriptors proposed to build the hierarchical models and Sect. 5 presents all the experiments that were carried out to test the validity of the proposed methods to solve the localization. Finally, the conclusions are presented in Sect. 6.

2 Related works

This section describes the related literature in the field of visual description methods (Sect. 2.1) and machine learning techniques in mobile robotics (Sect. 2.2).

2.1 Visual description methods

Omnidirectional cameras have been commonly proposed in mobile robotics during the past few years to capture information from the environment. For example, Liu et al. [27] use omnidirectional images to estimate the position and orientation in outdoor environments and Román et al. [48] use them to create an incremental model of an indoor environment. Reich et al. [43] propose an omnidirectional visual odometry framework to carry out localization and modeling with flying robots. Li et al. [26] propose a method to avoid obstacles for autonomous wheeled robots using HyperOmni Vision and through the proposed method, they won the FIRA avoidance challenge championship 2019.

Regarding how to extract relevant information from omnidirectional images, holistic descriptors have been commonly used for these purposes. For instance, Amorós et al. [1] use this kind of descriptors to develop a loop

closure detection and correction algorithm in a visual odometry framework. Korrapati and Mezouar [24] use global-appearance descriptors to tackle topological modeling using omnidirectional images and also to detect loop closures. Also, Amorós et al. [2] use this kind of descriptors for estimation of position and orientation. Comparing to local-features descriptors (such as SIFT [28], SURF [5] or ORB [49]), global-appearance approaches usually lead to more direct localization algorithms based on a pairwise comparison between descriptors, since each image is represented by a unique descriptor. However, they lack metric information so they have been traditionally used to build topological models of the environment [37].

The present work carries out the modeling task by using global-appearance description. Three methods are evaluated in this paper: the Histogram of Oriented Gradients (HOG), the *gist* of the scenes and a global descriptor obtained from a Convolutional Neural Network (CNN). The selection of these three methods is based on the results obtained in previous works [8].

These methods depart from panoramic images, hence, before calculating the proposed descriptors, a conversion from omnidirectional to panoramic images must be tackled. Once the panoramic image ($im \in \mathbb{R}^{N_x \times N_y}$) is available, the proposed methods are used to calculate the corresponding global-appearance descriptor vector ($\vec{d} \in \mathbb{R}^{l \times 1}$).

Regarding HOG, it was introduced by Dalal and Triggs [13] for pedestrians detection. The version used in the present work consists of dividing the image into k_1 horizontal cells and calculating a histogram from the gradient orientation per cell with b bins per histogram [25]. These histograms, arranged in an unique column vector, compose the final descriptor $\vec{d} \in \mathbb{R}^{b \cdot k_1 \times 1}$.

As for the *gist* descriptor, Oliva et al. [36] firstly proposed this method, which has been used for scenes recognition. The version used in the present work consists of the following steps. Firstly, m_2 images are created from the original panoramic image with different resolution. Then, a set of Gabor filters are applied over the m_2 images with m_1 different orientations, uniformly distributed to cover the whole circumference. Finally, the pixels of each image are grouped into k_2 horizontal blocks and finally, the average value of each block is calculated and these values are arranged to create a vector, which is the resultant descriptor $\vec{d} \in \mathbb{R}^{m_1 \cdot m_2 \cdot k_2 \times 1}$. A more detailed description of the configuration of the HOG and *gist* methods can be found in [45].

Regarding the use of Convolutional Neural Networks (CNNs), in this work, the information contained in intermediate layers is used to obtain global-appearance descriptors. This idea has previously been proposed by some authors such as Mancini et al. [30], who explain the fundamentals of this method and use it to carry out place categorization with the Naïve Bayes classifier. Moreover, this method has already been used in previous works ([37] and [8]), to create hierarchical visual models. The CNN architecture used in the present work is *places* [58], trained with around 2,5 million images to categorize 205 possible kinds of scenes. The descriptors extracted from this network correspond to the ones calculated in the layer *'fc7'*. These descriptors contain 4096 ($\vec{d} \in \mathbb{R}^{4096 \times 1}$) components. It is worth highlighting the fact that this CNN is used only with the purpose of obtaining a holistic descriptor per scene, thanks to the ability that the 'places' training confers it to extract relevant information from the input scene. Therefore, no training or definition of its architecture is carried out in the present work.

The use of global-appearance descriptors in environment modeling has increased during the past few years. For instance, Roman et al. [48] propose a mapping method from holistic description and solve the localization using omnidirectional vision. They also present in [38] a comparative analysis of some global-appearance descriptors for mapping. Rituerto et al. [46] propose the use of the *gist* [35] descriptor to build topological models based on omnidirectional images. More recently, Faessler et al. [16] present a vision-based quadrotor system to model a dense three-dimensional area. Korrapati and Mezouar [24] propose the use of omnidirectional images through global-appearance descriptors to build topological maps and also a loop closure detection method.

Moreover, in the past few years, hierarchical models have been proposed to tackle the localization task. A recent example of this framework was developed by da Silva et al. [12], who propose a localization approach for mobile robots through the use of topological maps and global-appearance descriptors from omnidirectional visual information. Moreover, previous works [8, 9] have also proved the effectiveness and the robustness of clustering methods along with holistic descriptors to create hierarchical models and to subsequently solve the localization problem under challenging situations, such as changes of illumination. Those works rely on arranging the visual information (obtained by global-appearance description methods) in several layers. Afterwards, the localization task is solved by means of an image retrieval problem in two steps: a fast localization in an area of the environment (rough localization) and a local localization step which provides more accuracy within that area (fine localization).

2.2 Machine learning techniques

In recent years, machine learning techniques have contributed to solving a variety of problems in robotics [6]. For

example, Meattini et al. [33] propose a human-robot interface system. This system is based on electromyography sensors. By means of merging pattern recognition and factorization techniques, the robot learns the optimal hand configuration for grasping. Gonzalez et al. [18] use machine learning to detect different levels of slippage for robotic missions in Mars; and Dymczyk et al. [15] propose a boosted classifier to classify landmark observations in a localization framework. The purpose of the present work is to retrieve the structure of the environment from a dataset composed of omnidirectional images captured along it, and use this model to solve the localization problem. Several machine learning techniques and global-appearance description approaches are used and their performance is studied. The efficiency of these tools will be evaluated through their ability to estimate robustly the position of the robot using the information stored in the model, which presents a trajectory structure. As for the use of machine learning tools together with visual information, numerous works can be found in the related literature. For example, Wang et al. [55] use a method based on Support Vector Machine (SVM) with the aim of classifying images of traffic signals extracted from captured videos; Murthy and Jadon [34] use feed-forward neural networks to detect hand gestures; and Fan et al. [17] use a feed-forward neural network with back-propagation to predict the texture characteristics from food surface images. Furthermore, the use of these techniques is widely extended in the mobile robotics field. For instance, Triebel et al. [52] propose the Informative Vector Machine (IVM) classifier for semantic mapping in autonomous mobile robotics; and Duguleana and Mogan [14] propose a path planning algorithm based on the use of Q-learning and artificial neural networks for mobile robots obstacle avoidance. More recently, Wozniak and Kwolek [56] use a salient CNN-based regional representation to calculate local features that are then fed to classifiers with the aim of estimating blur intensity. CNNs can also be re-trained with the aim of solving a specific task and, at the same time, extracting descriptors from the re-trained model [7].

In some applications that require to group visual data information, clustering techniques have proved to be a good solution [8, 37]. In [8], several clustering algorithms were tested and proved to be useful to compact the model with the aim of creating a two-layer hierarchical structure. Basically, the descriptors, calculated from the images captured at the modeling step, compose the visual dataset ($D = \{\vec{d}_1, \vec{d}_2, ..., \vec{d}_{Ndata}\}$). Then, a clustering process is carried out, and the descriptors are grouped according to their mutual similitude (in clusters $C = \{C_1, C_2, ..., C_{n_c}\}$, where n_c is the number of clusters). Consequently, through this method, a high level model can be obtained, composed of the representatives of each cluster. Additionally, the descriptors in each cluster can be considered a set of low-level models. This hierarchical structure permits estimating the position of the robot in an efficient way [9].

Concerning classifiers, they consist in predicting the class of given data instances. Classes are also known as labels or targets and they represent categories. The use of these machine learning techniques is suitable for cases when the data size is large. For instance, Rahimi and Recht [41] have proposed the use of local features and machine learning tools to face regression and classification tasks. Those tasks are based on large scale data and they have output results that are competitive with state-of-the-art algorithms regarding accuracy, training time, and evaluation time. Ballesta et al. [3] use a CNN to address a room classification task and Marinho et al. [31] define some strategic points in the environment and navigation is addressed by means of classification with rejection option. Rebouças et al. [42] use different classifiers based in machine learning to carry out a mobile localization task from sonar data.

Regarding the training process for machine learning tools, having large datasets plays an important role. However, the training dataset is occasionally smaller than required and, hence, the model can not be properly trained. In order to solve this problem, the data augmentation technique has been widely proposed as a method to improve performance. To cite one example, Guo and Gould [19] used data augmentation to improve a CNN training to solve an object detection task. Data augmentation basically consists in creating new pieces of 'data' by applying different effects over the original images. Some authors have already used data augmentation to solve their deep learning tasks. Shorten and Khoshgoftaar [51] introduce a survey concerning methods for data augmentation. While traditional data augmentation techniques apply systematically a variety of effects to the training images, in the present work, we define specifically these effects in such a way that they represent the challenging situations that can occur when the mobile robot moves through an environment under real operation conditions.

Therefore, this paper proposes the use of techniques based on machine learning to carry out the hierarchical localization, since they are expected to provide advantages both in the rough and in the fine localization steps. The improvements reached through the use of these tools with respect to previous works are shown throughout the paper, and they can be outlined as follows:

- Improved robustness against severe changes in lighting conditions.
- Efficient solution when the sizes of the data and the model are large.

- Higher hit ratio in localization in the high-level model.
- More accurate estimation of the position of the robot, since there will not be limitations due to the resolution of the model (distance between consecutive images within the dataset). Hence, more accuracy in the fine localization step.

3 Machine learning tools

Machine learning methods try to automate the construction of analytic models from data analysis. These methods belong to the AI (Artificial Intelligence) branch and they are based on the idea that the systems can learn to identify patterns departing from the data. Common machine learning techniques include decision trees, support vector machines and ensemble methods. The machine learning tools used throughout this work are (a) clustering, (b) classification, and (c) data fitting neural networks. Their fundamentals are outlined in the next paragraphs.

A clustering algorithm groups data vectors according to some given criteria. The most usual criterion is the distance or the similitude between the vectors. Considering previous works [8, 37], spectral clustering [29] is the method selected to tackle the clustering step throughout the present work, as it presented the most accurate results among a variety of approaches. This algorithm will be used to perform a non-supervised labeling of the dataset with the objective that the clusters contain images captured from near positions in the environment. Figure 1 shows an example of the clustering carried out in an indoor environment.

Regarding the classification technique to predict the class of given data instances, the network uses a set of training samples to learn a modeling function from the input variables (x_{train}) to discrete output variables (y_{train}). Thanks to it, the model is expected to achieve a well tuned configuration and it is ready to receive new data (x_{test}) and estimate their categories ($y_{\text{estimated}}$). The present work considers five types of classifications, since they have shown a robust performance in mobile robotics tasks:

- *Naïve Bayes (NB) classification* It is based on Bayes' theorem with independence assumptions between the features. This classification was introduced by Maron [32] as a method for text categorization to solve the problem of judging documents (such as spam or legitimate, sports or politics, etc.) with word frequencies as features. For instance, Posada et al. [39] propose a naïve Bayes classifier to predict the occupancy in a local environment of the robot by fusing the evidence provided by different segmentations and models and hence to carry out visual navigation.
- *Shallow neural network pattern recognition classification* The shallow neural network is a framework to solve different tasks through processing different data inputs [54]. Such systems "learn" to perform tasks by considering examples (training data), generally without

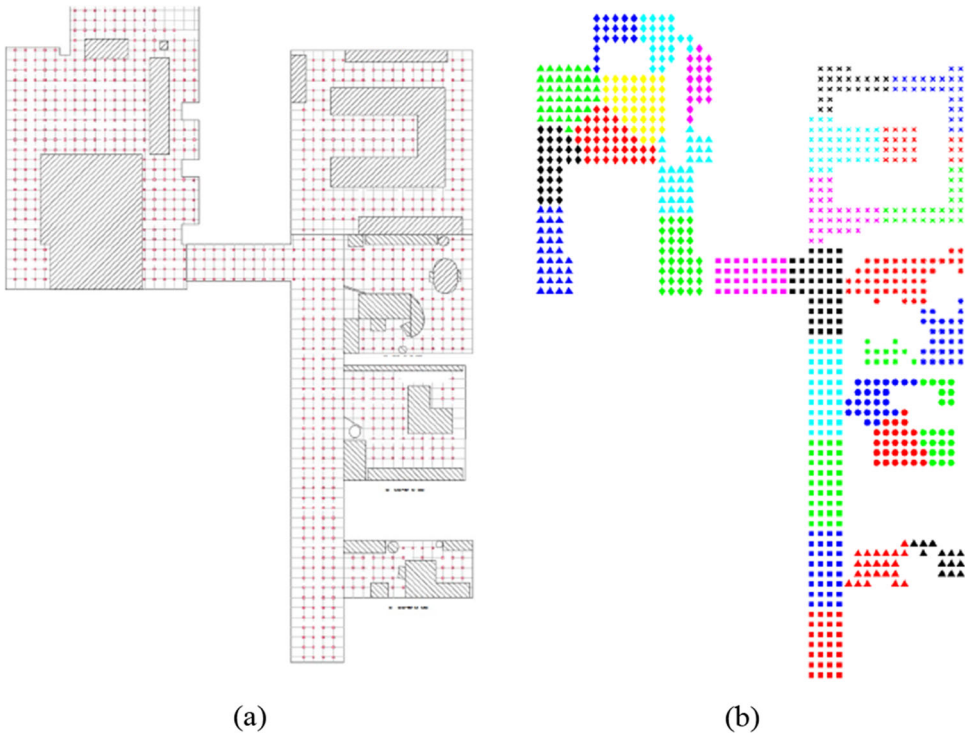


Fig. 1 Example of a non-supervised labeling through clustering. (**a**) The red dots show the positions where the images were captured. (**b**) Result of the clustering process. Each image is grouped according to its similitude with the rest of them. Only visual information will be used to carry out this clustering process

being programmed with any task-specific rules. The network automatically generates identifying characteristics from the learning material that it processes. For example, Barshan et al. [4] used this type of classifier in mobile robotics to differentiate with higher accuracy targets obtained by SONAR.

- *Support vector machine (SVM) classification* Introduced by Cortes and Vapnik [10], SVM can be used either for classification or regression purposes. The algorithm considers each data item as a point in an n-dimensional space (where n is the number of features) with the value of each feature being the value of a particular coordinate. Then, a classification is tackled by finding the hyper-plane or hyper-planes that best differentiate the categories. As an example of use, Iagnemma and Ward [22] propose a signal-recognition based approach for detecting autonomous mobile robot immobilization on outdoor terrain by using SVM classifiers, which define class boundaries in a feature space composed of statistics related to inertial and wheel speed measurements.
- *Random forest classification* Random decision forest is an ensemble learning method which consists in constructing a multitude of decision trees at training time and it was initially introduced by Ho [20]. Random Forest can be used either for classification or regression purposes like SVM. Basically, it consists of a large number of individual decision trees that operate as an ensemble. To address classification tasks, each individual tree in the random forest carries out a class prediction and the class with most votes becomes in the prediction of the model. An example of use in robotics can be found in [57], where a Random Forest model is trained to predict the best grasp for novel objects.
- *Linear discriminant analysis (LDA) classifier* LDA is a generalization of Fisher's linear discriminant method that is widely used in statistics and other fields to find a linear combination of features that characterize or separate classes. The resulting combination can be used as a linear classifier, or even for dimensionality reduction before further classification. LDA assumes that independent variables follow a multivariate normal distribution and the model has the same covariance matrix for each class. Under this modeling assumption, the classifier infers the mean and covariance parameters of each class. An example of using LDA for classification is given by Kang et al. [23], who carry out mode classification strategies for exoskeletons. Their approach uses the LDA classifier for hip exoskeleton applications using wearable sensors.

Finally, data fitting neural networks or function approximation networks are used to fit practical functions. The neural network is not trained to predict the correct category of the input data but to estimate a value among a specific range ($x = [x_{\min}, x_{\max}]$, where $x_{\min}, x_{\max} \in \mathbb{R}$). The training of the network is carried out through a supervised learning, i.e., the learning process assumes the availability of a labelled set of training data made up of N_{train} input–output examples. This work proposes the use of this type of neural networks to estimate the position (x, y) of the robot within a specific area or room.

Figure 2 shows the diagrams of the three machine learning tools used throughout this work to solve the modeling task. In Fig. 2, *im* is the panoramic image; (x, y) are the coordinates where a specific image was captured (obtained from the ground truth). $\vec{d}$ is the global-appearance descriptor calculated for a specific image and k is the number of categories. C_i is the *i-th* category and N_{train} is the total number of images used for training. Moreover, regrading the data to test the proposed methods, im_{test}, $(x_{\text{test}}, y_{\text{test}})$ and $\vec{d}_{\text{test}}$ are respectively the image, coordinates and descriptor calculated for a test image. The likelihood that the input $\vec{d}_{\text{test}}$ belongs to the category c_i is given by $p(c_i)$. Additionally, (x_i,y_i) are the coordinates of the position within the environment which correspond to the *i-th* image; and $(x_{\text{estimated}}, y_{\text{estimated}})$ are the coordinates which have been estimated for a test image. These tools carry out the training step departing from the data provided by the dataset. This dataset consists of N_{train} global-appearance descriptors which were calculated from N_{train} images captured along the environment. In clustering (Fig. 2a), the set of descriptors is introduced and the algorithm groups the information in k categories by considering the similitude between descriptors. Regarding classification (Fig. 2b), a training phase is previously performed. A set of training data items (visual descriptors and their correct categories) are introduced and the configuration parameters are tuned according to those values. After that, the classification can be performed when a new input descriptor is presented. As for the data fitting network (Fig. 2c), it is similar to classification, but in this case, the target data are not categories but they are real values within a range.

Furthermore, regarding the use of data augmentation, it consists in applying a variety of effects over the original training images. Through this technique, the machine learning tools are supplied by N_{effects} times the total number of training images ($N_{train\_augmented} = N_{train} \times N_{effects}$), where N_{effects} is the number of effects applied over the original images dataset.

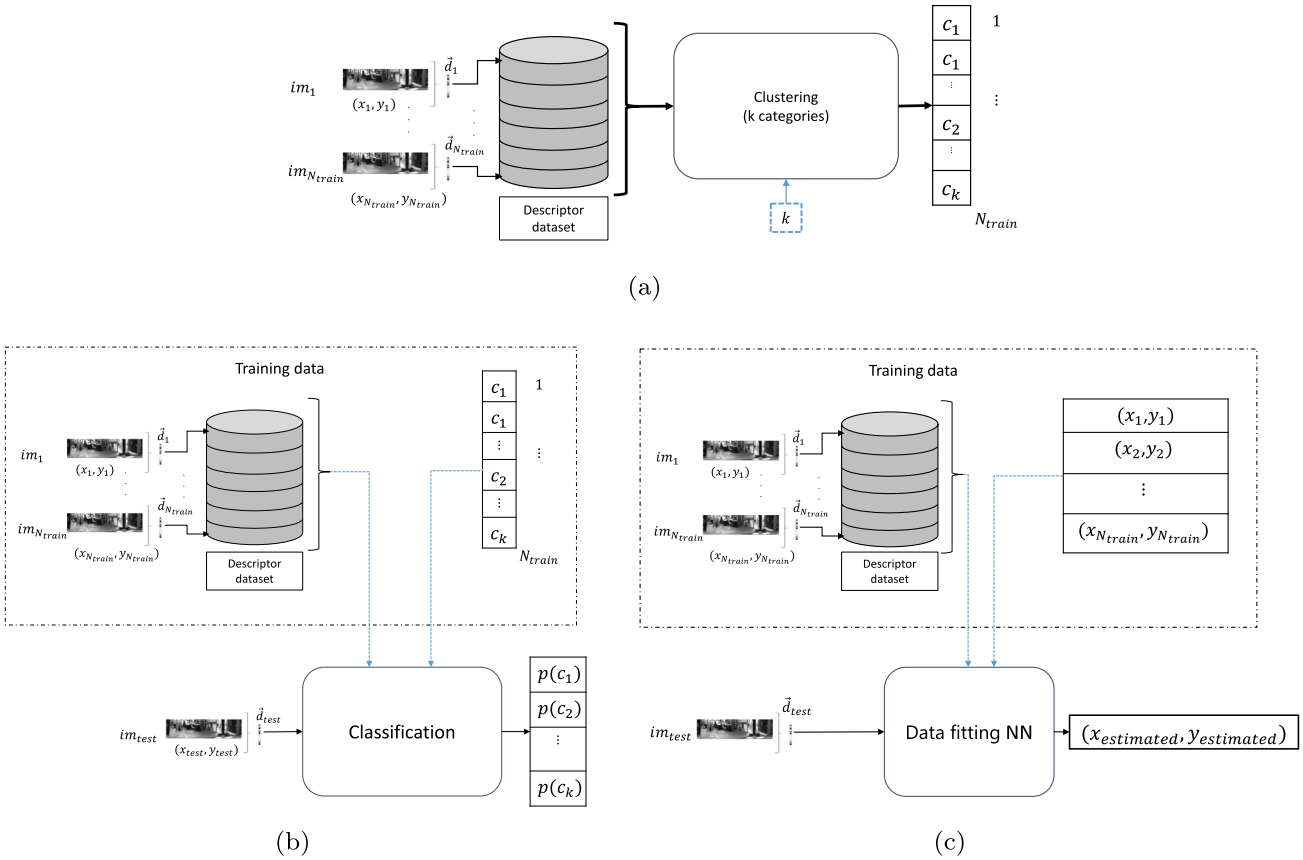


Fig. 2 Diagrams of (**a**) the clustering tool, (**b**) the classification tool and (**c**) the data fitting neural network tool used throughout this work to solve the modeling task. The clustering tool groups the visual description data in k clusters regarding their similitude. Every descriptor is then associated to a specific cluster ($C_1, C_2, ...,$ or C_k). The classification tool learns to classify global-appearance descriptors departing from the training data (descriptor dataset) and their labeling information. The data fitting neural network works similarly to the classifiers, but in this case, coordinates are introduced to the network instead of labels to train the network

4 modeling and localization through visual descriptors

4.1 Environment modeling and hierarchical localization

In the present work, the movement of the robot is contained in the floor plane and it captures images using a catadioptric vision system mounted on it. Every image is described using a global-appearance method in such a way that the initial dataset is composed of a set of N_{train} descriptors, $D = \{\vec{d}_1, \vec{d}_2, ..., \vec{d}_{Ntrain}\}$ where each descriptor is $\vec{d}_i \in \mathbb{R}^{l\times 1}$ and corresponds to the image im_i.

In the present work, the model is structured into two layers. First, the low-level layer is composed of a set of descriptors, each one corresponding to an image in the original training dataset. Second, the set of descriptors $D = \{\vec{d}_1, \vec{d}_2, ..., \vec{d}_{Ntrain}\}$ is divided into n_c groups by using a labeling method, where each group of descriptors is expected to contain information from zones which are visually distinctive. Once the labeling process has been tackled, each zone of the high level model is represented by a representative descriptor. Therefore, a set of representatives is obtained $R = \{\vec{r}_1, \vec{r}_2, ..., \vec{r}_{n_c}\}$, where $\vec{r}_i$ is the representative obtained for the *i-th* group c_i, and this set is the high-level model.

Once the model is built, the localization process can be solved hierarchically, as shown in Fig. 3 [8]. Basically, it consists of the following steps.

1. The robot captures a test image (im_{test}). From an unknown position
2. It calculates the global-appearance descriptor of this image ($\vec{d}_{\text{test}}$).
3. In the rough localization step, the distances between $\vec{d}_{\text{test}}$ and the representative descriptors (high level model) are calculated and stored in a vector of distances $\vec{l}_i$; and a zone is selected as the most likely one (c_i).

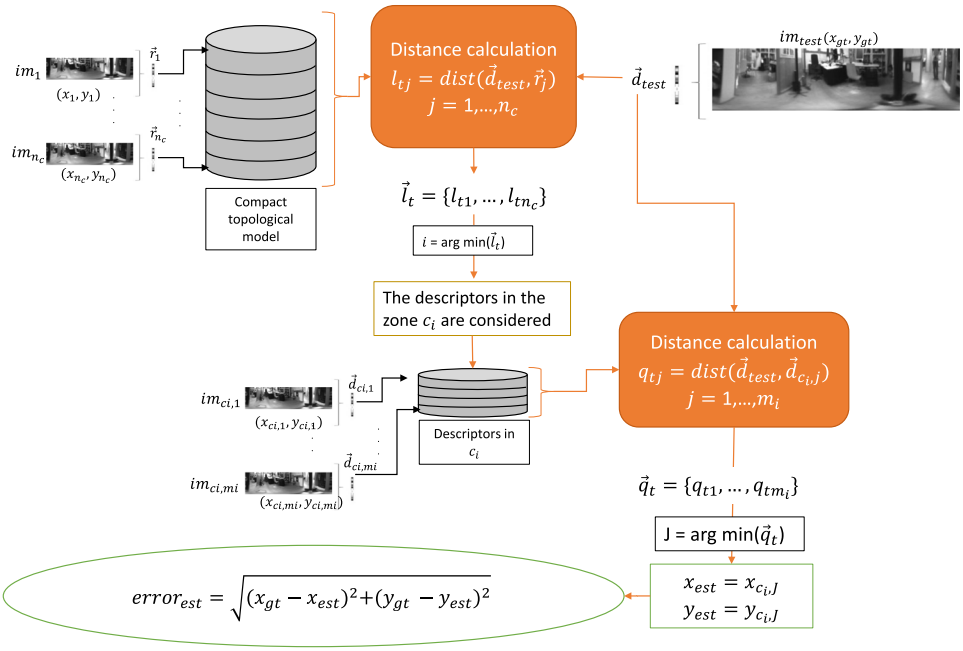


Fig. 3 Hierarchical localization diagram. The high-level model contains representative descriptors for each zone ($\vec{r}_1, \ldots, \vec{r}_{n_c}$). The process starts when a new image is captured (im_{test}) and its holistic descriptor is calculated ($\vec{d}_{\text{test}}$)

4. The descriptor ($\vec{d}_{\text{test}}$) is compared again but in this case with the descriptors of the low-level model which are contained in the selected zone c_i.
5. The most similar descriptor is selected ($\vec{d}_{c_i,k}$) and the position of im_{test} is estimated as the position of the robot from which the image whose descriptor is $\vec{d}_{c_i,k}$ was captured.

4.2 Classifiers for localization in the high-level model

This paper proposes an alternative hierarchical localization method based on the use of classifiers to estimate the position in the high level layer (the rough localization). It will be solved by introducing the descriptor into a classifier that will output the corresponding group or zone. The use of this tool is expected to provide some advantages, compared to the method presented in Fig. 3. The localization algorithm is not only expected to be more robust against changes of illumination, but also to provide an improved success selecting the correct area within the high-level model despite visual aliasing. The use of the classifier is included in the step (3) of the hierarchical localization process described in Sect. 4.1. In this step, the descriptor $\vec{d}_{\text{test}}$ is fed into a classifier, which predicts the most likely area. Basically, the steps to label the information are the following.

1. The global-appearance descriptors of the training images are calculated.
2. These descriptors are introduced into the spectral clustering algorithm and a number of clusters n_c is manually selected.
3. This algorithm outputs a vector of labels which specifies the cluster to which each descriptor belongs (more information regarding the spectral clustering process can be found in [8]).
4. Once the vector of labels is available, the classifier is trained (Fig. 2b).

Therefore, the high-level modeling process is the following. (1) Global-appearance descriptors are obtained from the training images. (2) Then, this visual information is firstly used to obtain automatically the labels, using spectral clustering (n_c is the number of clusters). (3) Finally, the global-appearance descriptors together with the labels are used to train the classifier.

In this case, once the model is ready, the localization process consists of the following steps:

1. The robot captures an image from an unknown position im_{test}.
2. The global-appearance descriptor of this image is calculated $\vec{d}_{\text{test}}$.
3. The rough localization is solved by means of the classifier, which predicts the most likely area c_i from $\vec{d}_{\text{test}}$.

4. After the prediction, the fine localization is solved by comparing the descriptor with the descriptors of the low-level model included in the selected area. The most similar descriptor $\vec{d}_{c_i,J}$ is retained.
5. Finally, the position of the robot is estimated as the position which corresponds to the most similar descriptor from the selected area (see Fig. 4).

Nevertheless, as it was explained in Sect. 3, the classifier must be trained before using it to predict the cluster. A complete study of the performance of the different classifiers is included in Sect. 5.1.

4.3 Solving the fine localization problem using function approximation through a data fitting neural network

The coarse and fine localization steps outlined in Sect. 4.1 use only visual information. Previous works have shown the main limitations of it [9], which are, mainly, the visual aliasing and the resolution in localization due to the distance between consecutive acquisition points in the training dataset. To overcome these limitations, the proposed neural network is trained to approximate the coordinates (x, y) of the test image. The network is trained with a set of visual descriptors from a training dataset, along with the

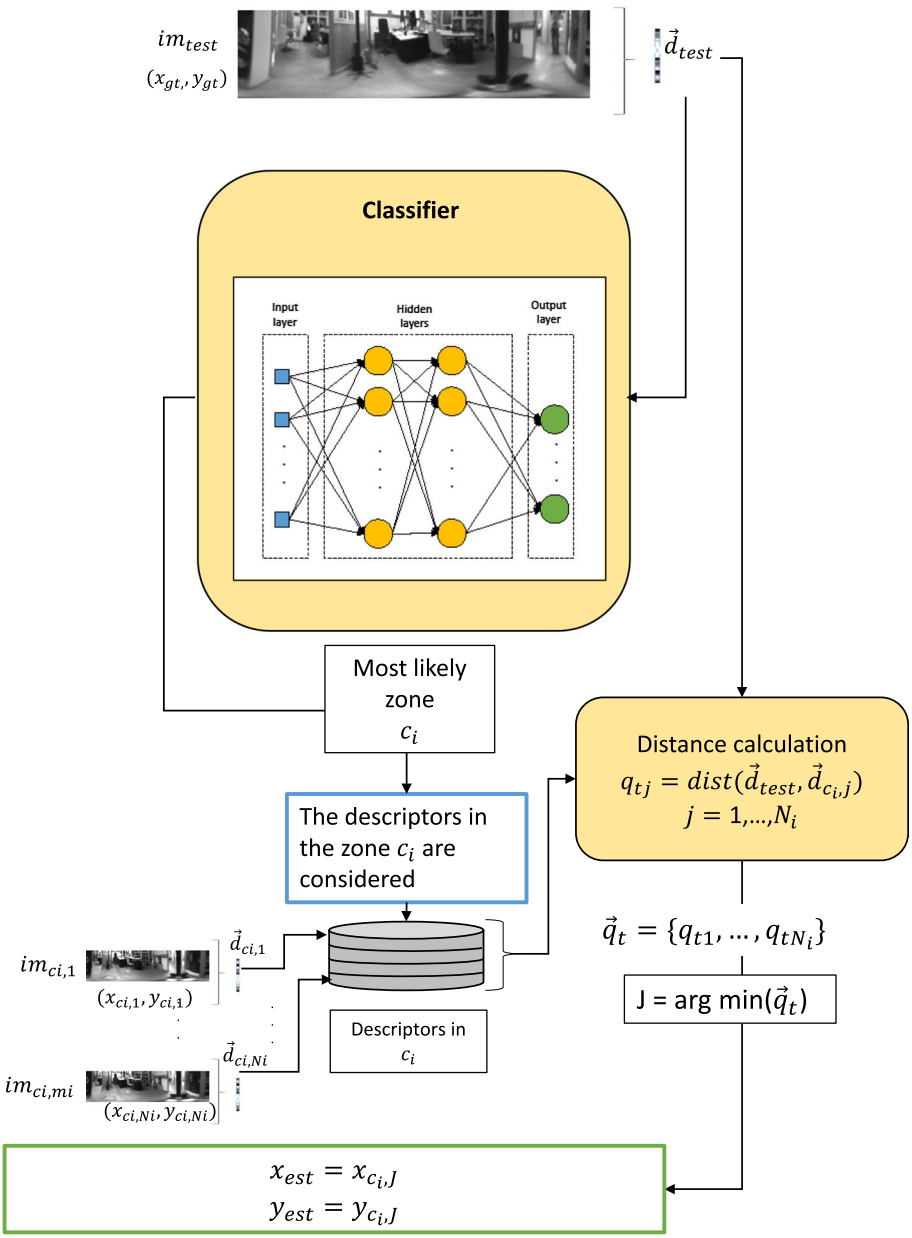


Fig. 4 Hierarchical localization diagram. A classifier has been previously trained. To start the localization process, a new image im_{test} is captured and its holistic descriptor $\vec{d}_{\text{test}}$ is obtained

coordinates (x, y) in the floor plane from which each training image was captured.

Therefore, a data fitting Neural Network can be used to solve the fine localization as a function approximation problem. Before the localization task, the model construction is carried out. This process consists of training a network for each area by using the descriptors included in that area and the (x, y) coordinates from which each image was captured. After training the n_c networks, the complete localization process is addressed through the following steps:

1. The robot captures a test image from an unknown position im_{test}.
2. The global-appearance descriptor of this image is calculated $\vec{d}_{\text{test}}$.
3. This descriptor is fed into a classifier, which predicts the most likely zone c_i.
4. After the prediction, the descriptor $\vec{d}_{\text{test}}$ is fed into the previously trained neural network corresponding to the zone retrieved in the previous step.
5. The outputs of this network are the predicted coordinates of the test image, which are an estimation of the position of the robot (see Fig. 5).

The diagram related to this process can be seen in Fig. 5. Concerning the data augmentation, the present work proposes an approach that has been designed specifically to obtain a robust Data fitting NN for localization. Hence, we consider a variety of visual effects to each training image with the aim of obtaining new samples that reflect the visual changes that usually appear when the robot operates in real conditions. Therefore, through this data augmentation, the CNN is expected to be more robust against the challenging conditions that can occur in the scenario where the robot moves. Considering it, the effects that we consider to perform the data augmentation are:

- *Rotation* A random rotation between 10 and 350 degrees is applied over each omnidirectional image. This effect emulates the different orientations that the robot may have at a specific point in the ground plane when acquiring a new image.
- *Occlusion* This effect simulates the cases when some parts of the picture are hidden either by some event (such as piece of furniture or a person in front of an object) or by some parts of the sensor setup. This effect is applied by introducing geometrical objects over random parts of the image.
- *Blur effect* Some degrees of blur are applied to each training image to emulate the case in which the image is captured while the robot is moving.
- *Gaussian noise* It emulates the possible noise that the visual sensor can introduce in the image. White Gaussian noise is added to the image.
- *Brightness* The low intensity values are re-adjusted (increased) in order to create a new image brighter than the original one.
- *Darkness* The high intensity values are re-adjusted (decreased) in order to create a new image darker than the original one. The brightness and darkness effects try to imitate the changes that the lighting conditions of the environment may experience during the day. No darkness and brightness are applied at the same time on the same image.

The data augmentation and training processes are offline steps which must be completed before starting the localization task. Once the data fitting neural networks are available, the fine localization can be solved: the descriptor of the test image ($\vec{d}_{\text{test}}$) is introduced into the neural networks of the area/room retrieved in the rough localization step, and the coordinates (x and y respectively) are estimated.

5 Experiments

The database used to perform the experiments is the COLD (COsy Localization Database) [40]. This open access dataset is composed of several types of data (monocular images and videos, laser-based ground truth, etc.) captured in indoor environments. Among the different data provided, the omnidirectional images are selected to carry out the experiments and the ground truth provided by the database, which is used throughout the experiments, was obtained by means of a laser sensor. Furthermore, the database contains images that were collected under three different lighting conditions (cloudy days, sunny days and at nights) and they also contain dynamic changes and blur effects. Among the different datasets, the Freiburg dataset captured during a cloudy day is used as training data, because this environment is the largest one provided by COLD and the presence of wide windows makes the localization task more challenging. Moreover, the cloudy conditions introduce the minimum effect produced by illumination. Subsequently, cloudy, sunny and night images will be used to test the robustness of the methods. It will allow us to analyse the domain shift problem (changes of lighting conditions).

The Freiburg dataset is composed of 9 different rooms: a printer area, a kitchen, four offices, a bathroom, a stair area and a long corridor which connects the rooms. This dataset includes several challenging phenomena which make the experiments suitable to analyze the performance of the

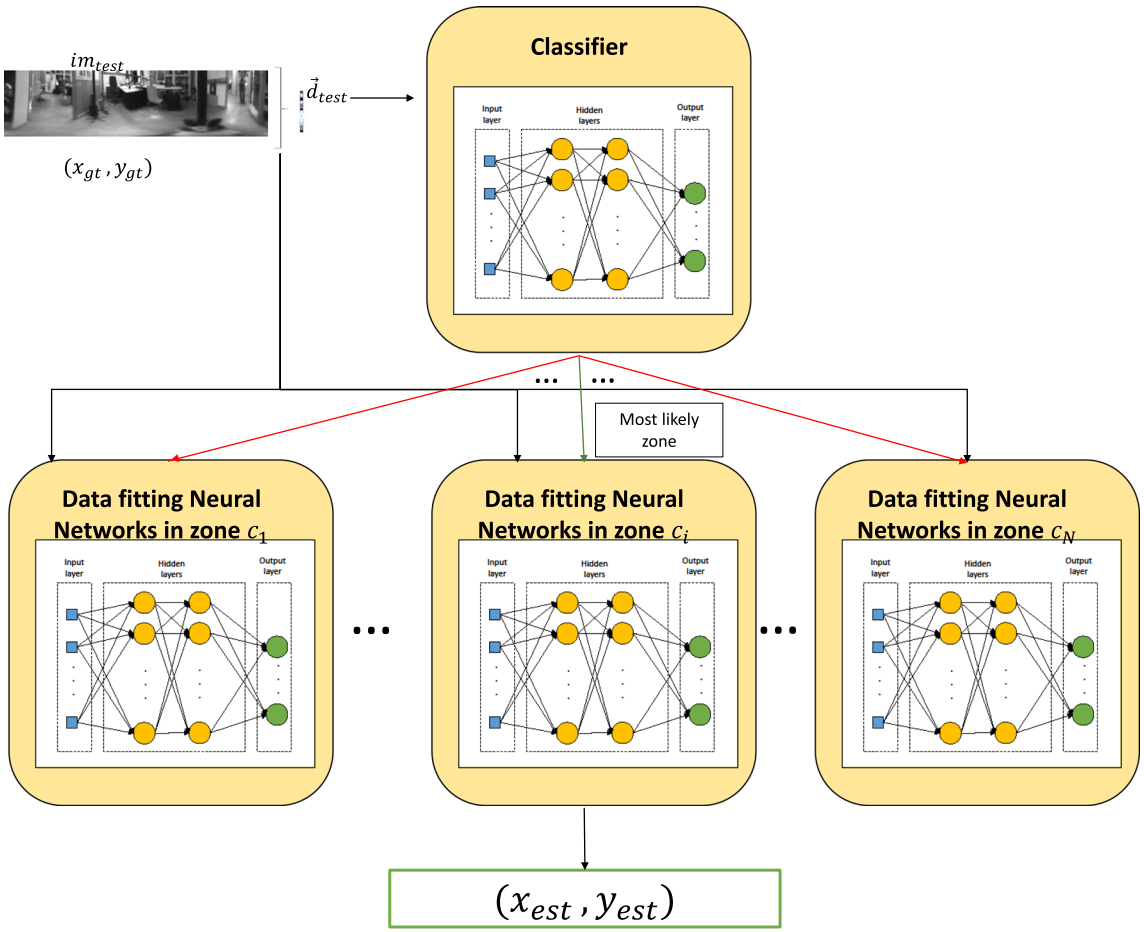


Fig. 5 Hierarchical localization diagram. A classifier and a set of c_N data fitting neural networks have been previously trained to solve, respectively, the coarse and fine localization steps

algorithm under real-operation conditions, apart from those described in the previous paragraph. There are dynamic changes in the environment such as in the position of furniture and objects or people walking, there is blur effect over some images due to the movement of the camera and also, a high degree of visual aliasing can be perceived in the images. The original cloudy dataset is downsampled to keep visual information every 30 cm on average. It allows us to obtain results which are comparable to those ones obtained in previous works ([8] and [9]). Hence, after downsampling, a training dataset composed of 519 images is considered as starting point. From this initial dataset, a data augmentation is performed to obtain an augmented dataset with 49824 images, which are used to train the data fitting

Furthermore, departing separately from the cloudy, sunny and night datasets, three test datasets are created with 2596, 2231 and 2876 images respectively. These images were selected randomly across the whole model. The cloudy test dataset (same domain) contains images which are not included in the training dataset. The sunny and night test datasets allow us to analyse the problem of domain shift. Figure 6 shows some examples of omnidirectional images under the three different illumination conditions. The experiments have been carried out in a computer with a CPU Intel Core i7-7700® at 3,6 GHz and through Matlab® programming.

In this section, the results of the experiments are presented. First, Sect. 5.1 shows the performance of the classifiers to solve the rough localization step. Second, in Sect. 5.2, the fine localization is addressed by means of data fitting neural networks and the results are analysed. Finally, the complete hierarchical localization process is tested in Sect. 5.3.

5.1 Experiment 1: rough localization

This subsection focuses on the rough localization. The first experiment studies the performance of the five classifiers to carry out the selection of the corresponding area within the environment. These classifiers are (a) Naïve Bayes, (b) a classifier based on a shallow neural network (c) SVM

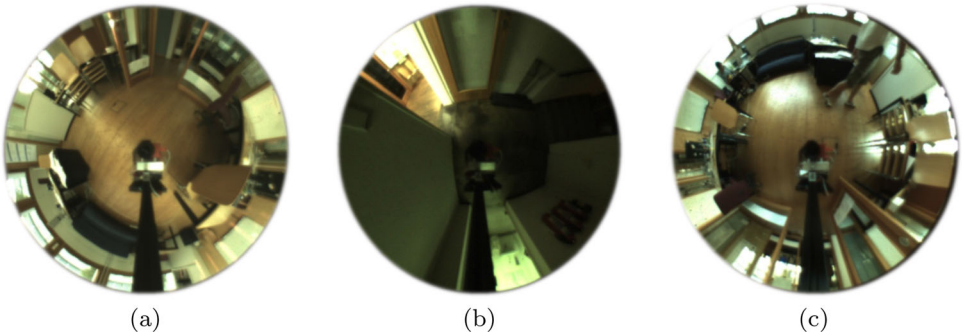


Fig. 6 Sample omnidirectional images from the Freiburg environment under (**a**) cloudy, (**b**) night and (**c**) sunny illumination conditions

(d) Random Forest and (e) LDA classifier. These classifiers are all trained with the global-appearance descriptors of the training images, along with the labels, which indicate the corresponding category for every descriptor. In the first part of the experiment, a manual labeling of the data is carried out, in which the labels correspond to the number of room in which each image was captured, so these labels are integers in the range [1, 9] as the Freiburg dataset contains 9 rooms.

The three kinds of global-appearance descriptors used to train each classifier are *gist*, HOG and a descriptor obtained from the layer 'fc7' of the CNN *places* (this descriptor is named CNN-fc7 throughout the paper). Therefore, to sum up, every classifier is first trained with the set of training descriptors (extracted from the cloudy dataset) and the labels, and after that, the classifier is ready to receive new descriptors (those of the test images) and output the correct label. Moreover, in order to evaluate the ability of these tools to cope with changes of the illumination conditions, the experiment is also carried out by using as test images those contained in the night and sunny Freiburg datasets.

To evaluate the relative performance of each method, the hit ratio of every configuration (classifier + description method) is collected, i.e. the percentage of success of the classifier, using as input the test data (new data which has not been used to train the classifier). In order to carry out a comparison between methods, this experiment also shows the hit ratios obtained when using the nearest neighbour method to solve the rough localization, instead of a classifier, since this was the method proposed in previous works ([8]). In this case, the high-level model is composed of one representative descriptor per room (calculated as the average descriptor of the training images contained in this room). The rough localization calculates the distance between each test descriptor and the representatives, and retrieves the room whose representative is the nearest neighbour.

Table 1 shows the results obtained, showing separately the hit ratio of the test data for the three illumination conditions and also, the necessary time to complete the rough localization process. According to this table, the shallow neural network and the SVM classifier work successfully with the cloudy test images, since both provide hit ratios around 98% when *gist* or CNN-fc7 are used to describe the visual information. The best results are obtained with LDA and SVM. On the one hand, LDA presents the best results in cloudy conditions (98.96%). On the other hand, SVM outputs the best hit ratios in night and sunny conditions (94.09 and 85.03% respectively). Both classifiers present outstanding results in general terms when used along with the CNN-fc7 descriptor. The confusion matrix obtained with the cloudy test images is shown in Fig. 7. In this matrix, correct predictions are higher than 96% in all the areas of the environment. The worst case is produced in the 2-person office 2. In this case,

Table 1 Results of experiment 1. Accuracy of the classifiers. Hit ratio and average computing time of the rough localization process under three illumination conditions (domain shift). The best result for each illumination condition is highlighted in bold

Classifier	Descriptor	Time (s)	Hit ratio test (%)		
			Cloudy	Night	Sunny
Naïve Bayes	*gist*	0.17	86.74	75.66	70.10
Naïve Bayes	CNN-fc7	0.51	86.13	77.78	62.39
Naïve Bayes	HOG	0.05	4.24	4.10	9.86
Neural net	*gist*	0.05	98.57	83.55	82.61
Neural net	CNN-fc7	0.02	97.42	93.25	76.20
Neural net	HOG	0.02	4.97	4.35	6.77
SVM	*gist*	0.09	98.61	84.63	**85.03**
SVM	CNN-fc7	0.15	98.50	**94.09**	82.03
SVM	HOG	0.05	7.24	5.29	5.92
Random forest	*gist*	0.33	79.15	76.43	45.06
Random forest	CNN-fc7	1.37	90.67	79.68	63.99
Random forest	HOG	0.13	5.20	6.21	5.17
LDA	*gist*	0.28	90.79	65.61	46.53
LDA	CNN-fc7	2.54	**98.96**	93.75	84.15
LDA	HOG	0.05	5.97	6.32	7.20
Nearest Neigh	*gist*	0.06	80.89	70.58	70.10
Nearest Neigh	CNN-fc7	0.01	78.23	76.95	53.65
Nearest Neigh	HOG	0.02	14.07	11.27	19.77

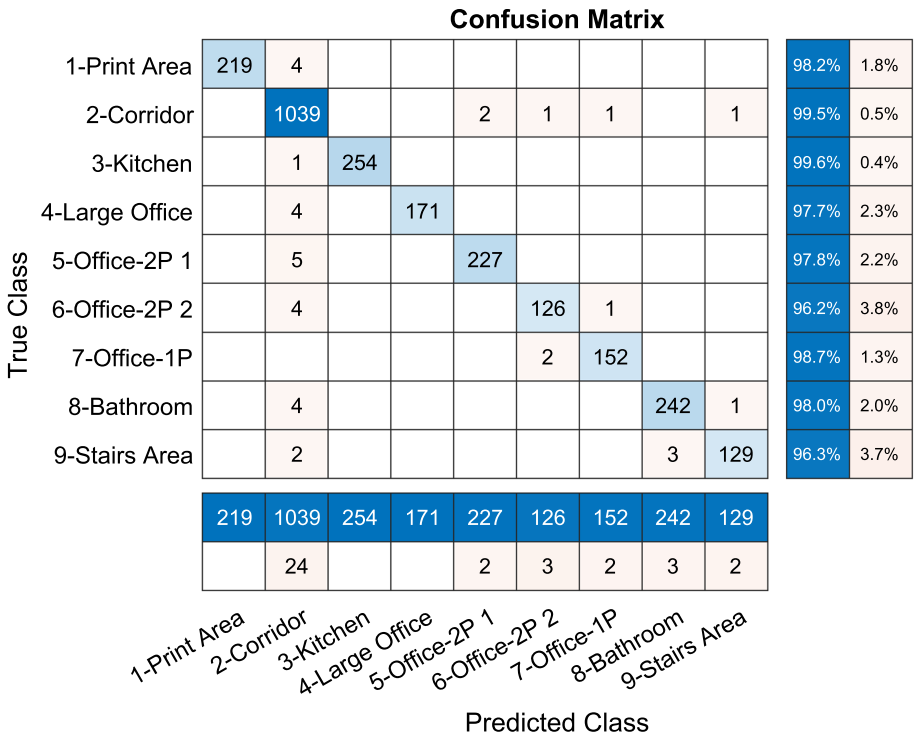


Fig. 7 Confusion matrix of the classifier SVM along with the *gist* descriptor to estimate the area of the test images in the cloudy test dataset

there are 5 false positives, 4 with the corridor and 1 with the 1-person office, which are not considered critical, since these false positives are produced with adjacent rooms. Nevertheless, these mistakes should be taken into consideration with the aim of avoiding incorrect fine localization subsequently. Additionally, the Naïve Bayes and Random Forests classifiers return the worst solutions and the HOG description proves not to be valid in combination with these machine learning tools.

Regarding different illumination conditions (sunny and night), the hit ratio decreases for most of the areas. This effect was expected and was observed in previous works ([9]). Notwithstanding that, the results do not differ substantially from the ones obtained with the cloudy test images. Thus, the conclusion achieved is that the classifiers present robustness to carry out the rough localization task even when substantial changes of lighting conditions occur. The benchmark method, based on the Nearest Neighbour search, presents, in general, a lower calculation time. However, its hit ratio, even with the best configuration (using *gist*) is substantially lower than the hit ratios provided by the shallow neural network and SVM. Therefore, these classifiers prove to be a robust and computationally efficient option to address the rough localization. Considering the results shown in Table 1, for future experiments, only the combinations shallow neural network, LDA or SVM along with *gist* or CNN-fc7 will be considered.

Once proved the utility and robustness of the classifiers to carry out the rough localization task, the second part of this experiment evaluates the labeling of the training data. So far, the labeling provided by the dataset has been used. Therefore, these experiments have assumed a manual labeling of the dataset. This way, the training of the classifier has consisted basically in introducing the visual description of the images and its related label. Nevertheless, using AI to automatically label the images can be a more practical solution. Hence, we can consider a tool which groups the images regarding its visual similitude (automatic labeling) instead of using the labeling provided by the dataset (manual labeling). To tackle the automatic labeling, spectral clustering has been selected, because it provided good solutions in previous works [8] compared to other clustering frameworks.

For this experiment, the descriptor *gist* is used to obtain the visual information, $n_c = 2, ..., 13$ clusters are considered and either the classifiers SVM, LDA or shallow neural network are used. Moreover, the nearest neighbour method is also used as a reference method to compare the performance of the classifiers with the results obtained in previous works. The results of this experiment are shown in Fig. 8. This figure shows the average error under cloudy (blue), night (orange) and sunny (green) illumination conditions. Classifiers (SVM, LDA and shallow neural network) work better than the nearest neighbour method independently of the illumination conditions. The use of classifiers with cloudy test images provides hit ratios which

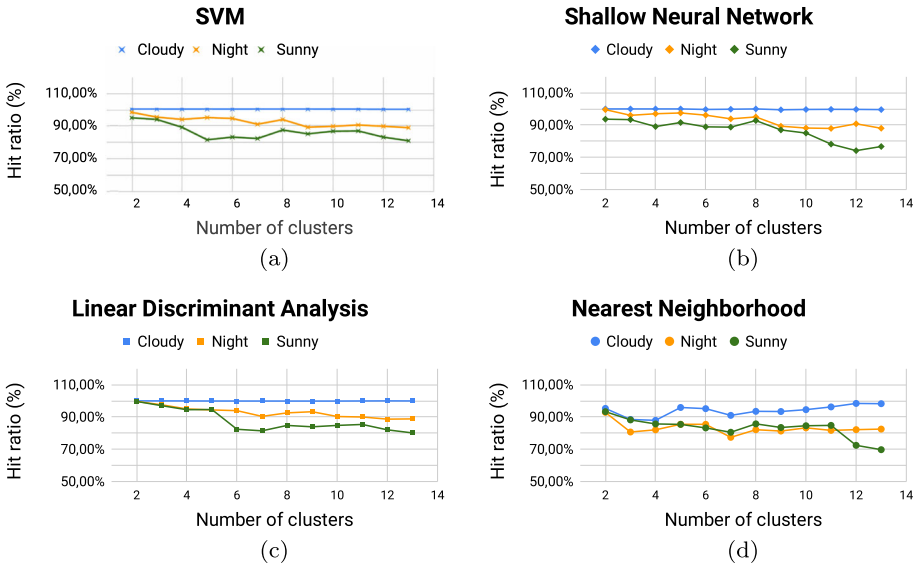


Fig. 8 Experiment 1. Results of rough localization using automatic labeling. Hit ratio for the test images versus number of clusters. The localization step is carried out by means of (**a**) the SVM classifier, (**b**) a shallow neural network classifier, (**c**) the LDA classifier and (**d**) the nearest neighbour method (benchmark method)

are always around 100%. Among the four methods studied to evaluate the use of automatic labeling LDA and SVM present competitive results in the cases of domain shift (night and sunny conditions). The hit ratio is improved by using automatic labeling in comparison to the results obtained by manual labeling (see Table 1). The hit ratios reached with neural network classifier and manual labeling (98.57, 83.55 and 82.61% for cloudy, night and sunny test datasets respectively) are lower than those obtained with the automatic labeling considering 9 clusters (99.65, 89.29 and 86.96% for cloudy, night and sunny test datasets respectively). Also, the same comparison using the SVM classifier outputs better results by using automatic labeling, for example, the hit ratio improves from 98.61, 84.63 and 85.03% to 100, 89.19 and 85.21% respectively with the three illumination conditions. Only when the number of clusters is higher than 10 under sunny illumination, neural network classifier produces hit results lower than 80%. Therefore, the conclusion reached through this experiment is that the combination of spectral clustering and a classifier improves considerably the rough localization step with respect to the nearest neighbour search and performs more robustly against changes of illumination.

5.2 Experiment 2: fine localization

Section 5.1 has shown the utility of some classifiers to solve the rough localization step considering either manual or automatic data labeling. The present section goes one step ahead and tries to solve the fine localization. Considered jointly, both steps will form a complete hierarchical localization and have proved to be an efficient method to solve this task [9]. Hence, after selecting the area, the fine localization consists in estimating the position within the selected area. In these experiments, changes of the lighting conditions are again considered and the labels to train the classifiers correspond either to the room number (manual labeling) or those obtained through spectral clustering (automatic labeling), as in Sect. 5.1. In the present section, we assume that the proper room or area is always correctly selected in the previous rough localization step. Therefore, we focus on the performance of the methods proposed for fine localization. Later, in Sect. 5.3, the joint performance of both processes will be analyzed.

As described in section 4, the fine localization is solved either with a data fitting neural network (one network per area has been created and trained) by means of a nearest neighbour search with the images included in the selected area. To evaluate the performance of each method, the average localization error has been obtained for each configuration (localization method + description method), i.e. the average error, using test images which have not been used to train the data fitting neural networks and images captured with different illumination conditions. Also, the average computing time is collected. Table 2 shows the results obtained. According to this table, the computing time for the data fitting neural network option is lower, with time values of 57.34 and 23.55 ms whereas, in the case of the nearest neighbour method, the average time is 86.64 and 30.30 ms (when using *gist* or the CNN-fc7 descriptor, respectively). Regarding the localization error, the table shows that the fine localization method based on image retrieval outputs slightly better results than the alternative using data fitting neural networks with the exception of the results with sunny test images. Nevertheless, a more detailed analysis of the results provided by these methods leads to deeper conclusions. Table 3 shows the data collected for the fine localization by using nearest

Table 2 Experiment 2. Accuracy of the methods studied to solve the fine localization. Average localization error and average computing time considering test images captured under three illumination conditions (domain shift)

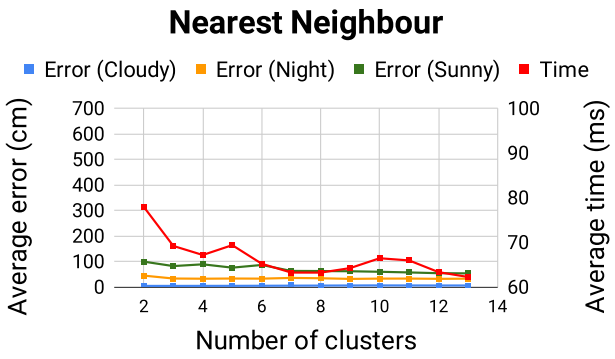
Method	Descriptor	Avg. time (ms)	Avg. Error (m)		
			Cloudy	Night	Sunny
Nearest Neigh	*gist*	89.64	1.82	1.83	2.18
Nearest Neigh	CNN-fc7	30.30	1.82	1.82	2.31
Neural net	*gist*	57.34	1.86	1.91	1.98
Neural net	CNN-fc7	23.55	1.90	1.95	1.98

Table 3 Results for fine localization using nearest neighbour and data fitting neural network with gist descriptor. The average errors are collected separately for each area

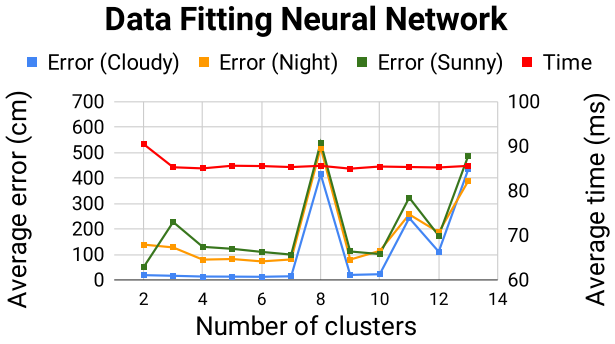
Area	Average error (nearest neighbour) (cm)	Average error (neural network) (cm)
1	5.22	4.96
2	417.17	430.08
3	4.19	2.16
4	167.54	168.40
5	4.11	3.61
6	5.23	3.72
7	5.11	3.15
8	4.09	3.18
9	6.20	4.67

neighbour and data fitting neural networks, but showing in detail the information per area. The table shows that the average error with the neural network is quite low in most of the areas: 1, 3, 5, 6, 7 and 8 present an error between 2.16 and 4.96 cm, but areas 2, 4 and 9 present worse results. The same problems are presented when the nearest neighbour method is used. These results are due to the fact that the areas 2 (corridor) and 4 (kitchen) are the largest rooms, thus, their training can be more challenging. Despite these issues, this analysis permits realizing that the use of data fitting neural networks can be an interesting tool to estimate the position of the robot within a non-large and well delimited environment or when abrupt changes of the lighting conditions are expected. Additionally, the estimation with data fitting neural network does not present the limitation of resolution given by the distance between consecutive images in the training set.

Furthermore, a comparison between manual and automatic labeling is carried out. Figure 9 shows the average error and computing time of the fine localization through (a) nearest neighbour and (b) data fitting neural network respectively. Both methods depart from automatic labeling through spectral clustering and they also use the *gist* descriptor. The graphs show that the nearest neighbour method performs better than the data fitting neural network. Regarding the computing time, both options provide similar values and they decrease as the number of clusters does, as expected. Finally, to compare the labeling options, the spectral clustering is configured with a number of clusters $n_c = 9$, since the labeling provided by the ground truth (manual labeling) defines 9 categories regarding the 9 rooms which compose the dataset. Through this comparison, automatic labeling proves to behave better than the manual option. For instance, comparing the fine localization through manual labeling and nearest neighbour method (see Table 2; with *gist* descriptor) with the automatic labeling ($n_c = 9$ clusters) and nearest neighbour method (see Fig. 9a), the average error results obtained for the second option (6.6; 32.5 and 62.5 cm respectively for the three illumination conditions) improves substantially the results obtained for the manual labeling option (182; 183 and 218 cm respectively). In addition, Fig. 10 shows the average silhouettes vs. the number of clusters. The silhouette provides information about how compact the clusters are. For this case, we have used the silhouette of descriptors, which measures the compactness of the clusters from a visual point of view, since the mapping process is purely visual. The silhouette takes values in the range $]-1, 1[$ and evaluates the degree of similarity between the instances within the same cluster and at the same time the dissimilarity with the instances which belong to others clusters. Low values of silhouette denote that the clusters are not visually consistent and high values indicate that they are visually compact and consistent. Therefore, this



(a)



(b)

Fig. 9 Experiment 2. Results for fine localization using automatic labeling. Average error (cm) (blue, orange and green lines) and average computing time (ms) (red line) versus number of clusters. The localization is carried out by means of (**a**) the nearest neighbour method and (**b**) data fitting neural networks

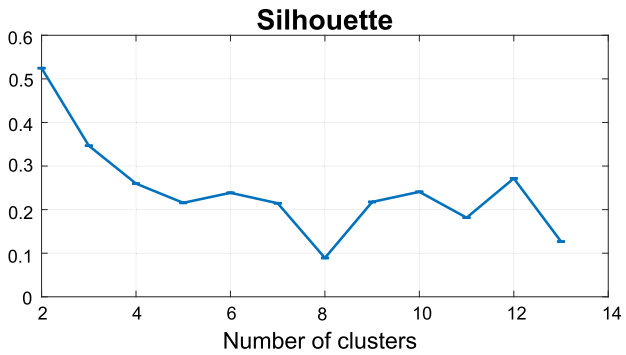


Fig. 10 Results of silhouette versus number of clusters

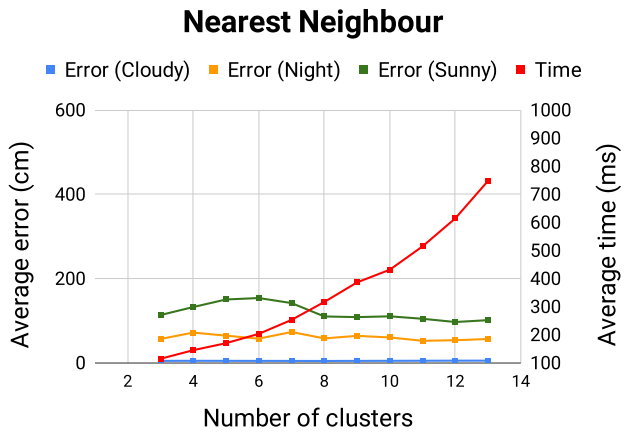
parameter allows us to quantify the goodness of the clustering process, hence, it is used to decide the number of clusters to take into consideration. For instance, it can be seen in Fig. 10 that for eight clusters, the silhouette value is low and hence, the produced clusters are not visually consistent. Therefore, this environment should not be clustered into 8 clusters to create the automatic labelling.

5.3 Experiment 3: complete hierarchical localization

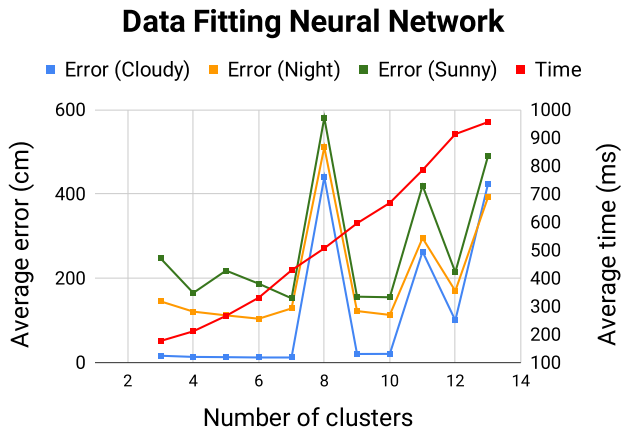
Previous experiments have shown the utility of some classifiers to solve the rough localization step (Sect. 5.1) and two methods to carry out the fine localization step (Sect. 5.2). The present subsection joins both steps and tries to solve the complete hierarchical localization. Therefore, after selecting the area through classifiers, the position within the selected area is estimated. In these experiments, changes of the illumination conditions are also considered and the labels to train the classifiers correspond to those obtained through spectral clustering (automatic labeling). Two alternatives are considered to carry out the fine localization step: either (a) through an image retrieval approach, by calculating the nearest neighbour or (b) through the use of a data fitting neural network.

To evaluate the performance of each method, the average localization error has been collected using cloudy test images and also images captured during different illumination conditions (sunny and night). Moreover, the average computing time is collected. The proposed hierarchical localization process has been evaluated by using the SVM classifier in the rough step and the *gist* descriptors, because the previous experiments have proved that this configuration works efficiently.

Figure 11 shows the results obtained in this experiment. Blue, orange and green lines show the average localization error for the three illumination conditions and the red line represents the average computing time measured for the cloudy test dataset. These figures show that the computing time increases as the number of cluster does, since the rough step (use of classifiers) requires more computing time than the fine step. Moreover, the nearest neighbour option is slightly faster. The average time values are between 115 and 748 ms whereas, in the case of the neural network, the average time goes from 177 to 956 ms. Nevertheless, these results would be good enough to tackle an online localization task. Regarding the localization error, the figures show that the hierarchical localization method based on image retrieval for fine localization outputs better results than the alternative using data fitting neural network. However, the results obtained through data fitting neural network provide accurate values which are also good enough to carry out the localization task. For instance, in the case of cloudy test data, the localization error takes values around 13 cm, which is quite low considering that the distance between the training images is around 30 cm. The results for $n_c = 8$ clusters provide high error values. This is due to the fact that the clustering algorithm was not capable of finding a compact representation of the environment, as shown in the previous

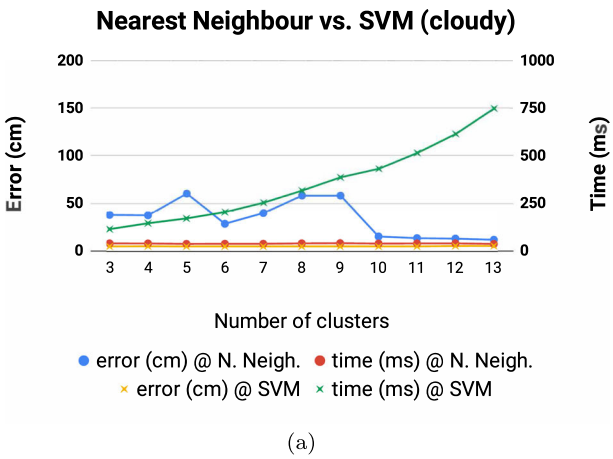


(a)

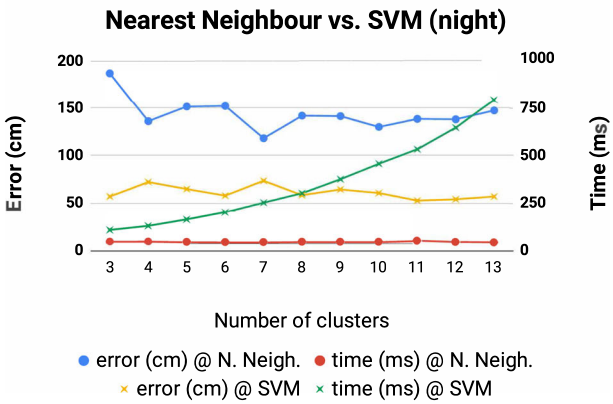


(b)

Fig. 11 Experiment 3. Results for complete hierarchical localization using automatic labeling. Average error (cm) (blue, orange and green lines) and average computing time (ms) (red line) versus number of clusters. The localization step is carried out by means of (**a**) a nearest neighbour search and (**b**) data fitting neural networks



(a)



(b)

Fig. 12 Comparison of hierarchical localization methods under (**a**) cloudy and (**b**) night illumination conditions. Solving the rough localization by calculating the nearest neighbour (N. Neigh.) to the representatives obtained by spectral clustering and through the SVM classifier. Average localization error and average computing time

subsection. Regarding the changes of illumination, for both localization methods, the error increases, as expected. Nevertheless, the increment is smoother with the data fitting neural network option.

Finally, two comparisons are carried out. First, a comparison between hierarchical localization methods is addressed. On the one hand, the methods proposed in this work (using classifiers in the rough step and image retrieval in the fine step) and, on the other hand, the method proposed in previous works ([8] and [9]) by using a representative per area obtained through spectral clustering in the rough step and image retrieval in the fine step. Figure 12 shows the localization results (average localization error and computing time) versus the number of areas (clusters). The results show that the classifier introduces a more efficient alternative regarding the localization error at the expense of a higher computing time. Furthermore, Fig. 12b shows the results obtained when the test dataset is composed of images captured at night. Both methods are affected by this illumination conditions. Nevertheless, the method based on classifiers presents an improved robustness against this effect.

To conclude, a final comparative evaluation with a recognized visual localization approach is addressed. On the one hand, the method proposed in this work (using classifiers in the rough step and image retrieval in the fine step) and, on the other hand, a localization method based on visual place recognition which proved an excellent performance in a work by Horst and Möller [21]. This method consists basically in tackling an image retrieval, comparing the test image with a set of training images, by using an edge filter and distance known as e-NSAD (Normalized Sum of Absolute Differences on edge-filtered images). The results obtained for each method are shown in Table 4. The Visual Compass presents a similar

Table 4 Comparison with a benchmarking visual localization approach. Results of the proposed method (hierarchical localization using machine learning) and results of the Visual Compass [21]. The localization methods are evaluated under three illumination conditions (domain shift)

	Method proposed			Visual Compass [21]		
	Cloudy	Night	Sunny	Cloudy	Night	Sunny
Avg. loc. error (cm)	5.09	52.74	77.32	6.82	101.21	74.02
Avg. comp. time (s)	0.71			1150.33		

localization error comparing to the proposed approach, when no changes of illumination or sunny illumination conditions are considered (by using the cloudy or sunny test datasets). Nevertheless, the localization error with the night dataset is better with the proposed approach. Furthermore, concerning the average computing time, the proposed method presents results significantly faster than the provided by visual compass. Therefore, visual compass is a suitable solution when no dark illumination conditions are considered, and computing time does not play an important role. The proposed method presents more robustness against changes of illumination and it is also a feasible solution when the localization process should be solved in real time.

5.4 Experiment 4: evaluation in a different environment

This subsection studies the proposed methods to carry out the hierarchical localization in a different building. The aim of this subsection is to evaluate the suitability of the framework in an indoor environment which is different from the environment used in the previous experiments (Freiburg dataset). The experiments in this subsection are developed by using the dataset obtained in the Saarbücken building, which is also available in the COLD database [40]. The Saarbrucken dataset is composed of 5 rooms: corridor, kitchen, 1-person office, printer area and toilet. Like the Freiburg dataset, it includes challenging effects, as described in Sect. 5. Moreover, experiments are only presented under cloudy illumination conditions, since the suitability of the methods with changes of the lighting conditions has been already demonstrated in the previous subsections.

First, the performance of classifiers to carry out the selection of the corresponding area within the environment is evaluated. Table 5 shows the hit ratio of the test data and the time to complete the rough localization process. According to this table, the classifiers shallow neural network, SVM and LDA present hit ratios over 90% when the CNN-fc7 descriptor is used, in line with the results obtained previously in the Freiburg environment. On the contrary, the Naïve Bayes and the Random Forest classifiers present the worst hit ratios.

Table 5 Results of experiment 1 in the Saarbrücken environment. Accuracy of the classifiers. Hit ratio and average computing time of the rough localization process under cloudy conditions

Classifier	Descriptor	Time (s)	Hit ratio test (%)
Naïve Bayes	*gist*	0.35	76.00
Naïve Bayes	CNN-fc7	1.98	82.96
Naïve Bayes	HOG	0.17	27.33
Neural net	*gist*	3.60	79.63
Neural net	CNN-fc7	0.59	91.67
Neural net	HOG	0.34	21.45
SVM	*gist*	0.48	77.18
SVM	CNN-fc7	0.16	90.30
SVM	HOG	0.09	21.45
Random forest	*gist*	0.8×10^{-3}	85.21
Random forest	CNN-fc7	17.1×10^{-3}	85.80
Random forest	HOG	0.1×10^{-3}	21.45
LDA	*gist*	0.2×10^{-3}	85.99
LDA	CNN-fc7	4.3×10^{-3}	95.59
LDA	HOG	0.01×10^{-3}	20.76
Nearest Neigh	*gist*	0.06	80.89
Nearest Neigh	CNN-fc7	0.01	78.23
Nearest Neigh	HOG	0.02	14.07

After proving the robustness of the classifiers to carry out the rough localization task, an automatic labeling is tackled by means of spectral clustering. The results are shown in Fig. 13. Again, SVM, LDA and shallow neural network present better results than using nearest neighbour. The best results are obtained with LDA.

Concerning the fine localization step, Table 6 shows the results obtained with data fitting neural networks or with a nearest neighbour search (average localization error and average computing time). Regarding the localization error, the table shows that the best result in fine localization is obtained with the nearest neighbour search along with the CNN-fc7 descriptor.

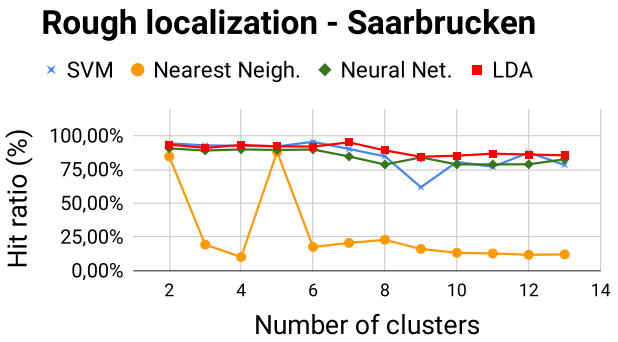


Fig. 13 Results of the rough localization using automatic labeling in the Saarbrücken environment. Hit ratio for the test images versus number of clusters. The localization step is carried out by means of the SVM classifier (blue line), a shallow neural network classifier (green), the LDA classifier (red) and the nearest neighbour method (orange)

Table 6 Experiment 2 in the Saarbrücken environment. Accuracy of the methods studied to solve the fine localization. Average localization error and average computing time considering test images captured under cloudy illumination conditions

Method	Descriptor	Avg. time (ms)	Avg. Error (m)
Nearest Neigh	*gist*	47.41	1.40
Nearest Neigh	CNN-fc7	24.02	0.61
Neural net	*gist*	62.51	1.12
Neural net	CNN-fc7	40.29	1.23

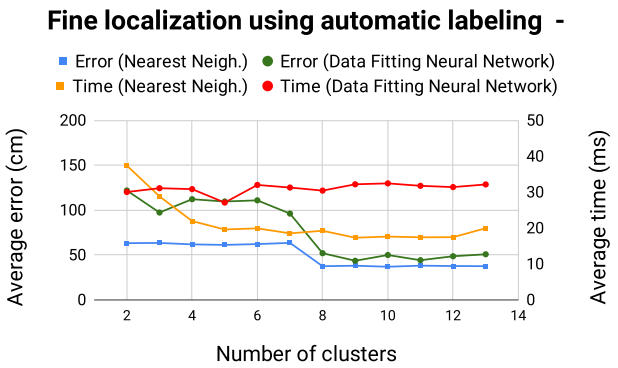


Fig. 14 Experiment 2 in the Saarbrücken environment. Results of the fine localization using automatic labeling (considering that the rough step always selects the correct room). Fine localization step carried out either by means of nearest neighbour search or by means of data fitting neural networks

Furthermore, Fig. 14 shows the average error and computing time of the fine localization step by means of nearest neighbour search and data fitting neural networks departing from automatic labeling (performed by spectral clustering). This figure presents the average error (cm) and average computing time (ms) with blue and orange lines respectively for the nearest neighbour method. The results of the data fitting neural networks are depicted with a green line (average error, cm) and a red line (average computing time, ms). These results are obtained by considering that the rough step selects always the correct room as in Sect. 5.2, with the objective of focusing on the performance of the fine localization. When the number of clusters is low, the nearest neighbour method provides a considerably lower error than the data fitting neural network. Nevertheless, if the number of clusters is higher than 7, the differences between methods are reduced. As for the computing time, in general terms, the nearest neighbour search is faster than the method based in data fitting neural networks.

To conclude this section, the complete hierarchical localization process is performed in the Saarbrücken environment and the results are presented in Fig. 15. The rough localization step is solved by means of the SVM classifier and the fine localization step is solved either by means of the nearest neighbour search or with data fitting neural networks. An automatic labeling (by spectral clustering) is also performed. This figure presents the average error (cm) and average computing time (ms) with blue and orange lines respectively for the nearest neighbour method, and with green and red lines for the data fitting neural networks. The figure shows that, in general, the hierarchical localization based on nearest neighbour search for fine localization outputs better results than using data fitting neural networks, in line with the results obtained in the Freiburg environment. Concerning the computing time, it increases as the number of clusters does for both methods. Among them, the nearest neighbour method is slightly faster for $n_c < 10$ and slower for $n_c > 10$.

Finally, we compare the performance of the methods proposed in this work for hierarchical localization (using classifiers in the rough step and image retrieval in the fine

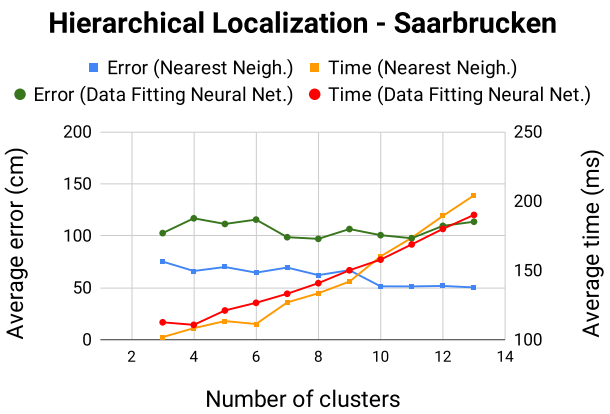


Fig. 15 Experiment 3 in the Saarbrücken environment. Results of the complete hierarchical localization using automatic labeling. Rough localization step with the SVM classifier. Fine localization step carried out either by means of nearest neighbour search or using data fitting neural networks

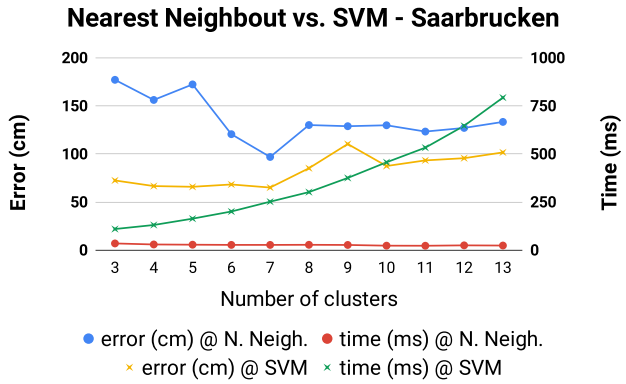


Fig. 16 Comparison of hierarchical localization methods in the Saarbrücken environment. Solving the rough localization by calculating the nearest neighbour (N. Neigh.) to the representatives obtained by spectral clustering and through the SVM classifier. Average localization error and average computing time

step) and the method proposed in previous works ([8] and [9]) by using a representative per area obtained through spectral clustering in the rough step and image retrieval in the fine step. Figure 16 shows the average localization error and the average computing time versus the number of clusters. The results confirm that the use of a classifier in the rough step (SVM in this experiment) provides a lower localization error at the expense of a higher computing time, as previously observed in the Freiburg environment.

6 Conclusion

Throughout the present work, we have evaluated the use of machine learning tools to carry out hierarchical localization with mobile robots using the knowledge extracted from a dataset composed of omnidirectional scenes. The experiments were carried out with an indoor dataset that presents dynamic changes and blur effects. The dataset also provides images captured under different illumination conditions (during cloudy days, during sunny days and at night). The work shows that most of the machine learning techniques proposed provide good localization results departing from a compact model. In this paper, several studies have been tackled. First, the classifiers have been validated as an efficient tool to perform the rough localization. SVM, LDA and shallow neural network classifiers together with global-appearance descriptors (*gist* and CNN-fc7) provide high hit ratios to retrieve the corresponding room or area. Second, a data fitting neural network was proposed for the fine localization step. Although it does not improve the results obtained by the image retrieval option, it actually works relatively well and robustly for most of the rooms or areas. The key to obtain better results would consist in either optimizing the training step in the most challenging areas or finding a global-appearance descriptor which suits better the training of the network. Moreover, these techniques (classifiers and data fitting neural network) present robustness against changes of illumination. Third, through the use of automatic labeling (spectral clustering), the localization results are improved in comparison to the ones with manual labels (provided by the ground truth). Furthermore, a comparison between hierarchical localization methods is tackled. Whereas the hierarchical localization based on classifiers provides more accurate localization results, the hierarchical localization based on representatives obtained from spectral clustering works faster. Additionally, the proposed methods improve the performance of a localization approach based on a Visual Compass under substantial changes of the lighting conditions, and thanks to the hierarchical approach, the proposal works faster than this benchmarking method. Finally, to provide a more complete analysis, an additional experiment is carried out to prove the validity of the framework in a different environment. Future works will focus on optimizing the results regarding the data fitting neural network to estimate the position within an area. Also, a study of deep learning tools to develop a complete SLAM framework with the use of omnidirectional images will be developed.

Acknowledgements This work is part of the project PID2020-116418RB-I00 funded by MCIN/AEI/10.13039/501100011033, and of the project PROMETEO/2021/075 funded by Generalitat Valenciana. This work has also been supported by the ValgrAI (Valencian Graduate School and Research Network for Artificial Intelligence).

Funding Open Access funding provided thanks to the CRUE-CSIC agreement with Springer Nature.

Data Availability The database COLD (COsy Localization Database) [40] that supports the findings of this study are available in https://www.cas.kth.se/COLD/.

Declarations

Conflict of interest All authors declare that they have no conflicts of interest.

References

1. Amorós F, Payá L, Marín JM, Reinoso O (2018) Trajectory estimation and optimization through loop closure detection, using omnidirectional imaging and global-appearance descriptors. Expert Syst Appl 102:273–290. https://doi.org/10.1016/j.eswa.2018.02.042
2. Amorós F, Payá L, Mayol-Cuevas W, Jiménez LM, Reinoso O (2020) Holistic descriptors of omnidirectional color images and their performance in estimation of position and orientation. IEEE Access 8:81822–81848. https://doi.org/10.1109/access.2020.2990996
3. Ballesta M, Payá L, Cebollada S, Reinoso O, Murcia F (2021) A CNN regression approach to mobile robot localization using omnidirectional images. Appl Sci. https://doi.org/10.3390/app11167521
4. Barshan B, Ayrulu B, Utete SW (2000) Neural network-based target differentiation using sonar for robotics applications. IEEE Trans Robot Autom 16(4):435–442. https://doi.org/10.1109/70.864239
5. Bay H, Tuytelaars T, Gool LV (2006) Surf: speeded up robust features. In: European conference on computer vision, Springer, pp 404–417. https://doi.org/10.1007/11744023_32
6. Cebollada S, Payá L, Flores M, Peidró A, Reinoso O (2021) A state-of-the-art review on mobile robotics tasks using artificial intelligence and visual data. Expert Syst Appl 167:114195. https://doi.org/10.1016/j.eswa.2020.114195
7. Cebollada S, Payá L, Flores M, Román V, Peidró A, Reinoso O (2020) A deep learning tool to solve localization in mobile autonomous robotics. In: ICINCO 2020, 17th international conference on informatics in control, automation and robotics, INSTICC, pp 232–241 (Online streaming, 7–9 July 2020)
8. Cebollada S, Payá L, Mayol W, Reinoso O (2019) Evaluation of clustering methods in compression of topological models and visual place recognition using global appearance descriptors. Appl Sci 9(3):377. https://doi.org/10.3390/app9030377
9. Cebollada S, Payá L, Román V, Reinoso O (2019) Hierarchical localization in topological models under varying illumination using holistic visual descriptors. IEEE Access 7:49580–49595. https://doi.org/10.1109/ACCESS.2019.2910581
10. Cortes C, Vapnik V (1995) Support-vector networks. Mach Learn 20(3):273–297. https://doi.org/10.1007/BF00994018
11. Cruz Ulloa C, Prieto Sánchez G, Barrientos A, Del Cerro J (2021) Autonomous thermal vision robotic system for victims recognition in search and rescue missions. Sensors 21(21):7346. https://doi.org/10.3390/s21217346
12. da Silva SPP, da Nbrega RVM, Medeiros AG, Marinho LB, Almeida JS, Filho PPR (2018) Localization of mobile robots with topological maps and classification with reject option using convolutional neural networks in omnidirectional images. In: 2018 international joint conference on neural networks (IJCNN), pp 1–8. https://doi.org/10.1109/IJCNN.2018.8489328
13. Dalal N, Triggs B (2005) Histograms of oriented gradients for human detection. In: In Proceedings of the IEEE conference on computer vision and pattern recognition, San Diego, vol II, pp 886–893. https://doi.org/10.1109/CVPR.2005.177
14. Duguleana M, Mogan G (2016) Neural networks based reinforcement learning for mobile robots obstacle avoidance. Expert Syst Appl 62:104–115. https://doi.org/10.1016/j.eswa.2016.06.021
15. Dymczyk M, Gilitschenski I, Nieto J, Lynen S, Zeisl B, Siegwart R (2018) Landmarkboost: efficient visualcontext classifiers for robust localization. In: 2018 IEEE/RSJ international conference on intelligent robots and systems (IROS), pp 677–684. https://doi.org/10.1109/IROS.2018.8594100
16. Faessler M, Fontana F, Forster C, Mueggler E, Pizzoli M, Scaramuzza D (2016) Autonomous, vision-based flight and live dense 3D mapping with a quadrotor micro aerial vehicle. J Field Robot 33(4):431–450. https://doi.org/10.1002/rob.21581
17. Fan F, Ma Q, Ge J, Peng Q, Riley WW, Tang S (2013) Prediction of texture characteristics from extrusion food surface images using a computer vision system and artificial neural networks. J Food Eng 118(4):426–433. https://doi.org/10.1016/j.jfoodeng.2013.04.015
18. Gonzalez R, Apostolopoulos D, Iagnemma K (2018) Slippage and immobilization detection for planetary exploration rovers via machine learning and proprioceptive sensing. J Field Robot 35(2):231–247. https://doi.org/10.1002/rob.21736
19. Guo J, Gould S (2015) Deep CNN ensemble with data augmentation for object detection. https://doi.org/10.48550/ARXIV.1506.07224. arXiv:1506.07224
20. Ho TK (1995) Random decision forests. In: Proceedings of 3rd international conference on document analysis and recognition, IEEE, vol 1, pp 278–282. https://doi.org/10.1109/ICDAR.1995.598994
21. Horst M, Möller R (2017) Visual place recognition for autonomous mobile robots. Robotics 6(2):9. https://doi.org/10.3390/robotics6020009
22. Iagnemma K, Ward CC (2009) Classification-based wheel slip detection and detector fusion for mobile robots on outdoor terrain. Auton Robot 26(1):33–46. https://doi.org/10.1007/s10514-008-9105-8
23. Kang I, Molinaro DD, Choi G, Camargo J, Young AJ (2022) Subject-independent continuous locomotion mode classification for robotic hip exoskeleton applications. IEEE Trans Biomed Eng 69(10):3234–3242. https://doi.org/10.1109/TBME.2022.3165547
24. Korrapati H, Mezouar Y (2017) Multi-resolution map building and loop closure with omnidirectional images. Auton Robot 41(4):967–987. https://doi.org/10.1007/s10514-016-9560-6
25. Leonardis A, Bischof H (2000) Robust recognition using eigenimages. Comput Vis Image Underst 78(1):99–118. https://doi.org/10.1006/cviu.1999.0830
26. Li S, Chou L, Chang T, Yang C, Chang Y (2019) Obstacle avoidance of mobile robot based on hyperomni vision. Sens Mater 31(3):1021–1036. https://doi.org/10.18494/SAM.2019.2226
27. Liu R, Zhang J, Yin K, Pan Z, Lin R, Chen S (2018) Absolute orientation and localization estimation from an omnidirectional image. In: pacific rim international conference on artificial intelligence, Springer, pp 309–316. https://doi.org/10.1007/978-3-319-97310-4_35
28. Lowe DG (1999) Object recognition from local scale-invariant features. In: The proceedings of the seventh IEEE international conference on Computer vision, 1999, vol 2, pp 1150–1157. https://doi.org/10.1109/ICCV.1999.790410
29. Luxburg U (2007) A tutorial on spectral clustering. Stat Comput 17:395–416. https://doi.org/10.1007/s11222-007-9033-z
30. Mancini M, Bulò SR, Ricci E, Caputo B (2017) Learning deep NBNN representations for robust place categorization. IEEE Robot Autom Lett 2(3):1794–1801. https://doi.org/10.1109/LRA.2017.2705282
31. Marinho LB, Rebouças Filho PP, Almeida JS, Souza JWM, Souza Junior AH, de Albuquerque VHC (2018) A novel mobile robot localization approach based on classification with rejection option using computer vision. Comput Electr Eng 68:26–43. https://doi.org/10.1016/j.compeleceng.2018.03.047
32. Maron ME (1961) Automatic indexing: an experimental inquiry. J ACM 8(3):404–417. https://doi.org/10.1145/321075.321084
33. Meattini R, Benatti S, Scarcia U, De Gregorio D, Benini L, Melchiorri C (2018) An sEMG-based human-robot interface for robotic hands using machine learning and synergies. IEEE Trans

Compon Packag Manuf Technol 8(7):1149–1158. https://doi.org/10.1109/TCPMT.2018.2799987
34. Murthy GRS, Jadon RS (2010) Hand gesture recognition using neural networks. In: 2010 IEEE 2nd international advance computing conference (IACC), pp 134–138. https://doi.org/10.1109/IADCC.2010.5423024
35. Oliva A, Torralba A (2001) Modeling the shape of the scene: a holistic representation of the spatial envelope. Int J Comput Vis 42(3):145–175. https://doi.org/10.1023/A:1011139631724
36. Oliva A, Torralba A (2006) Building the gist of a scene: the role of global image features in recognition. In: Visual perception, progress in brain research, vol 155, Elsevier, pp 23–36. https://doi.org/10.1016/S0079-6123(06)55002-2. https://www.sciencedirect.com/science/article/pii/S0079612306550022
37. Payá L, Peidró A, Amorós F, Valiente D, Reinoso O (2018) Modeling environments hierarchically with omnidirectional imaging and global-appearance descriptors. Remote Sens 10(4):522
38. Payá L, Reinoso O, Berenguer Y, Úbeda D (2016) Using omnidirectional vision to create a model of the environment: a comparative evaluation of global-appearance descriptors. J Sens 2016:1209507. https://doi.org/10.1155/2016/1209507
39. Posada LF, Narayanan KK, Hoffmann F, Bertram T (2010) Floor segmentation of omnidirectional images for mobile robot visual navigation. In: 2010 IEEE/RSJ international conference on intelligent robots and systems, IEEE, pp 804–809. IROS.2010.5652869
40. Pronobis A, Caputo B (2009) COLD: COsy localization database. Int J Robot Res (IJRR) 28(5):588–594. https://doi.org/10.1177/0278364909103912
41. Rahimi A, Recht B (2008) Random features for large-scale kernel machines. In: Advances in neural information processing systems, pp 1177–1184
42. Rebouças Filho PP, da Silva SPP, Ohata EF, Almeida JS, de Sousa PHF, Nascimento NMM, dos Santos Silva FH (2019) A new strategy for mobile robots localization based on omnidirectional sonar images and machine learning. In: Anais Estendidos da XXXII conference on graphics, patterns and images, SBC, pp 168–171. https://doi.org/10.5753/sibgrapi.est.2019.8321
43. Reich S, Seer M, Berscheid L, Wörgötter F, Braun JM (2018) Omnidirectional visual odometry for flying robots using low-power hardware. In: VISIGRAPP (5: VISAPP), pp 499–507. https://doi.org/10.5220/0006509704990507
44. Reinoso O, Payá L (2020) Special issue on visual sensors. Sensors. https://doi.org/10.3390/s20030910
45. Reinoso O, Payá L (2020) Special issue on mobile robots navigation. Appl Sci. https://doi.org/10.3390/app10041317
46. Rituerto A, Murillo AC, Guerrero J (2014) Semantic labeling for indoor topological mapping using a wearable catadioptric system. Robot Auton Syst 62(5):685–695. https://doi.org/10.1016/j.robot.2012.10.002
47. Rituerto A, Puig L, Guerrero J (2010) Visual slam with an omnidirectional camera. In: 2010 international conference on pattern recognition, IEEE, pp 348–351. https://doi.org/10.1109/ICPR.2010.94
48. Román V, Payá L, Peidró A, Ballesta M, Reinoso O (2021) The role of global appearance of omnidirectional images in relative distance and orientation retrieval. Sensors. https://doi.org/10.3390/s21103327
49. Rublee E, Rabaud V, Konolige K, Bradski G (2011) Orb: An efficient alternative to sift or surf. In: International conference on computer vision, IEEE, pp 2564–2571
50. Shi X, Shen Y, Wang Y, Bai L (2018) Differential-clustering compression algorithm for real-time aerospace telemetry data. IEEE Access 6:57425–57433. https://doi.org/10.1109/ACCESS.2018.2872778
51. Shorten C, Khoshgoftaar TM (2019) A survey on image data augmentation for deep learning. J Big Data 6(1):60. https://doi.org/10.1109/ACCESS.2018.2872778
52. Triebel R, Grimmett H, Paul R, Posner I (2016) Driven learning for driving: How introspection improves semantic mapping. In: Robotics research, Springer, pp 449–465
53. Valgren C, Lilienthal A (2010) Sift, surf & seasons: appearance-based long-term localization in outdoor environments. Robot Auton Syst 58:149–156. https://doi.org/10.1016/j.robot.2009.09.010
54. van Gerven M, Bohte S (2017) Editorial: artificial neural networks as models of neural information processing. Front Comput Neurosci 11:114. https://doi.org/10.3389/fncom.2017.00114
55. Wang LL, Ngan HYT, Yung NHC (2018) Automatic incident classification for large-scale traffic data by adaptive boosting SVM. Inf Sci 467:59–73. https://doi.org/10.1016/j.ins.2018.07.044
56. Wozniak P, Kwolek B (2021) Deep embeddings-based place recognition robust to motion blur. In: Proceedings of the IEEE/CVF international conference on computer vision, pp 1771–1779
57. Zhang J, Li M, Feng Y, Yang C (2020) Robotic grasp detection based on image processing and random forest. Multimed Tools Appl 79(3):2427–2446. https://doi.org/10.1007/s11042-019-08302-9
58. Zhou B, Lapedriza A, Xiao J, Torralba A, Oliva A (2014) Learning deep features for scene recognition using places database. In: Advances in neural information processing systems, pp 487–495