APPLICATION OF SUBDEFINITE MODELS IN THE GLOBAL LOCALIZATION OF A MOBILE ROBOT

A. D. Moscowsky

National Research Center "Kurchatov Institute," Moscow, Russia

✉ moscowskyad@yandex.ru

Abstract. This paper considers the application of subdefinite (SD) models, a variation of constraint programming, to the localization problem of a mobile robot. A complex technology with semantic maps and point cloud maps is proposed. The technology is intended to accelerate and increase the accuracy of global localization in large, symmetric, and periodic environments. The conventional localization approach is based on data from rangefinders generating point clouds; the idea proposed instead is, first, to match the objects observed by the robot to those on the semantic map (recognize the scene), and then apply SD computations to perform localization via visual landmarks. SD computations are used to determine interval constraints on the robot's positions, represented by several sets for each hypothesis obtained during the scene recognition. Within the interval constraints, the robot is localized using rangefinder data based on a particle filter initialized within these constraints. According to the experiments conducted on the open KITTI-360 dataset, localization based on SD computations can reduce the search space to 0.2% of the original map size. The complex technology shows a significant advantage compared to approaches involving point clouds or visual landmarks only, especially in scenarios with multiple hypotheses about the matches of observed objects and those on the semantic map.

Keywords: global localization, subdefinite models, constraint programming, scene recognition, semantic maps, mobile robot.

INTRODUCTION

Localization is crucial in modern mobile robotics since the vast majority of robot control approaches are based on the current position knowledge. When it is difficult to use satellite navigation (e.g., in urban or indoor environments), one applies localization methods building a 3D map in the form of a point cloud. The popularity of this approach is determined, among other things, by its versatility: there is no need to place additional labels (devices) in the robot's environment to aid localization. Such a detailed map can also be used in navigation, as it contains information about obstacles. It can be built by means of common sensors such as scanning laser rangefinders and depth cameras. The modern development of SLAM (*Simultaneous Localization and Mapping*) methods [1] allows building quite accurate maps of large areas, but the following global localization problem arises when using such maps further: it is necessary to determine the robot's starting position without knowledge of any initial conditions. This problem is solved during robot initialization (the start of operation) and also when the robot gets lost (its position differs from the expected one) due to errors during localization or, e.g., was moved by third parties. All these situations prevent the use of available data on the robot's previous position and displacement for localization. Finding the robot's position in the entire space (usually limited by the map size) becomes more difficult as the map size increases, as noted in the review [2] of the corresponding approaches. Besides the high volume of necessary computations, the additional problem is that the above urban and indoor environments often contain domains with quite similar geometry, especially in typical buildings. Such "periodicity" and "symmetry" of the environment lead to the so-called local minima on the map, in which the robot can erroneously globally localize itself. And the number of such local minima grows with the map size.

Researchers tackle this problem from different standpoints. One R&D direction involves only point clouds, and the task is to construct qualitative descriptors associated with point clouds in order to identify and compare geometric and other features of their elements, including semantic ones. Generally speaking, this approach includes three main stages: descriptor extraction, descriptor matching, and position refinement. Among descriptor extraction methods, we mention FPFH (*Fast Point Feature Histograms*) [3], NDT (*Normal Distribution Transform*) [4], NeRF (*Neural Radiance Fields*) [5], Minkloc3d [6], and others [7]. Different approaches are used for descriptor matching: RANSAC (*RANdom SAmple Consensus*) [8], graph-based approaches [9], optimization algorithms [10], and learning approaches [11] are widespread. ICP (*Iterative Closest Point*) [12] and its extensions (for example, see [13]) are used to further refine the robot's position. Also note probabilistic approaches, such as the histogram filter or particle filter [14], applied to point clouds [15]. However, these descriptors are primarily based on the geometric features of the mutual arrangement of the points, and the problem of local minima due to the periodicity and symmetry of the spaces cannot be eliminated accordingly. Despite that the descriptors narrows the search space, they remain at the point cloud level and, to some extent, are associated with the above problems, so other approaches pass to a "higher" map representation level, which will be discussed later.

Similar to descriptor construction, Place Recognition [16] encodes the entire environment of a robot as a vector of numbers. Usually, neural network models are applied for this purpose; they are trained so that the same environment, taken from different angles, at different times of day, etc., produces vectors close to each other. Such models often have multimodal input and consider point clouds, "raw" images, and the semantics of the scene. Thus, the map is provided with a set of "key frames" with encoding vectors, and localization is reduced to finding the frame best matching the robot's current observation from a set of sensors. This approach is good in the sense of utilizing all available information about the environment, but it requires dense coverage of the space with key frames. A strong change in camera angle can also deteriorate the frame search for different compositions of the robot's sensors. Such methods demonstrate the best performance for sensor compositions with full view, but this is not possible or reasonable for all robots.

This paper follows another R&D direction with the so-called semantic maps. On such maps, objects are assigned semantic labels (classes). Methods for obtaining semantic maps in an automatic set are being actively developed [17, 18]. A set of same-class objects is extracted in the environment directly observed by a robot. Then it is necessary to recognize the scene, i.e., match the objects of the map to the objects in the environment (scene). Matching at the object level allows narrowing the search space compared to searching in 3D maps. Semantics gives additional "uniqueness" to scenes with monotonous geometry. Once a match is found, localization methods based on visual landmarks can be used. However, one problem of this direction is that such scene recognition methods generate a number of hypotheses about object matching. Therefore, it is required to perform localization for each hypothesis and evaluate the resulting quality of object matching. The direction with semantic maps has several advantages: the methods are less sensitive to changes in camera angle and allow for the manual editing of semantic maps. The latter is an obvious benefit of the method because the environment may change over time and performing a complete mapping procedure often seems unreasonable.

In view of the aforesaid, the aim of this research is to accelerate and increase the accuracy of global localization of a mobile robot on point cloud maps. We choose the R&D direction involving point clouds jointly with semantic maps, which requires scene recognition. This paper focuses on the global localization problem of a mobile robot based on scene recognition results considering such features as high object positioning errors, object matching errors, and selection of the best result among a set of hypotheses.

1. LOCALIZATION METHODS FOR SCENE RECOGNITION

In modern works on localization using semantic maps addressing scene recognition, a prevalent method is an optimization-based approach for matching two sets of 3D points, known as SVD (*Singular Value Decomposition*) [19]. Localization by this method based on scene recognition results was performed in [20–22]. The approach minimizes the root-mean-square (RMS) error of point positions between two sets, but requires exact element-by-element correspondence of one set to another and neglects the position errors in computations. In [23], the ICP approach (see above) was applied not to point clouds but to two sets of objects: on the map and on the scene. In several publications (for example, [24, 25]), robot localization was performed by the 2D-3D *reprojection* technology: the positions of flat objects in the image were matched

to their 3D positions. This approach eliminates the need for the 3D localization of the scene objects in space and, therefore, partially offsets the errors of determining this distance, which can be significant.

A seemingly promising direction is to combine localization by point clouds with data obtained by scene recognition methods. An interesting approach was proposed in [26]: the separate 3D registration of point clouds of selected objects, followed by a refinement using ICP.

At the same time, to the best of the author's knowledge, search probabilistic approaches based on a histogram filter or a particle filter have not been applied in the literature to perform localization via visual landmarks in the setting under consideration. An explanation is that these methods require restricting the search space, causing difficulties for the majority of the localization approaches discussed here: they output a point in space rather than a domain. Imposing interval constraints on the robot's position would settle this problem. The common approach of *constraint programming* [27] yields such data. Its generalization in the form of subdefinite (SD) computations [28] was applied to perform localization via visual landmarks and use a histogram filter within the resulting constraints [29]. Below, this approach will be applied to robot localization based on a set of hypotheses using a particle filter (PF), which determines the robot's position from point clouds within the resulting constraints.

2. THE COMPLEX GLOBAL LOCALIZATION TECHNOLOGY

The complex global localization technology proposed in this paper is within the direction using semantic maps. This technology requires both a map layer (in the form of a point cloud) and a semantic layer (with marked positions of objects and some semantic information about them). In addition to the two map layers, the input data are images and point clouds from the robot's sensors. The complex technology allows solving the localization problem in the following stages:

1. recognizing and localizing all objects in the robot's environment (forming the scene);

2. recognizing the scene (matching the objects recognized to those on the semantic map and obtaining several hypotheses);

3. applying the SD localization approach to these hypotheses (discarding contradictory hypotheses and determining interval constraints on the robot's position);

4. determining the robot's position by means of a PF in the resulting constraints for each hypothesis and calculating quality values;

5. selecting the best solution with the highest quality value.

The first stage is implemented either by the classical method for recognizing objects in images and further localizing them using depth maps or point clouds [30], or by 3D recognition approaches [31].

The second stage, including scene recognition, was described in detail in [32]. Graph theory algorithms were used therein to extract geometric features in the mutual arrangement of groups of objects. CLIP (*Contrastive Language-Image Pre-training*) [33], a foundational visual-language model, was applied to consider the visual similarity of objects in addition to the semantic label, as previously done by most researchers in the field of robot localization with semantic maps. According to the conclusions, the visual similarity criterion of objects used jointly with the geometric feature criterion significantly improves the accuracy of scene recognition compared to methods involving only one of the criteria. However, several problems of working with objects, including high localization errors, the presence of visually similar objects, errors of recognition systems, and the multiplicity of solutions, lead to many hypotheses at the output of such systems, and the correct solution does not always have the highest quality value.

In the third stage, the SD localization approach—the focus of this paper—is applied. The approach serves to perform robot localization via visual landmarks: it imposes interval constraints on the robot's positions and allows identifying possible input data inconsistencies due to the above features of scene recognition.

In the fourth stage, the robot's position is determined using a PF and a map layer (a point cloud). The rationale behind this approach is that a PF can be naturally initialized in interval constraints derived from SD localization. In addition, a PF can handle a wide set of input data, and a quality value of any particle can be obtained for these data; in turn, the value represents a hypothesis about the robot's position in the localization problem.

The fifth stage is to select the best solution for all hypotheses of the second stage (the one with the highest quality value).

The flowchart of the complex global localization technology is shown in Fig. 1.

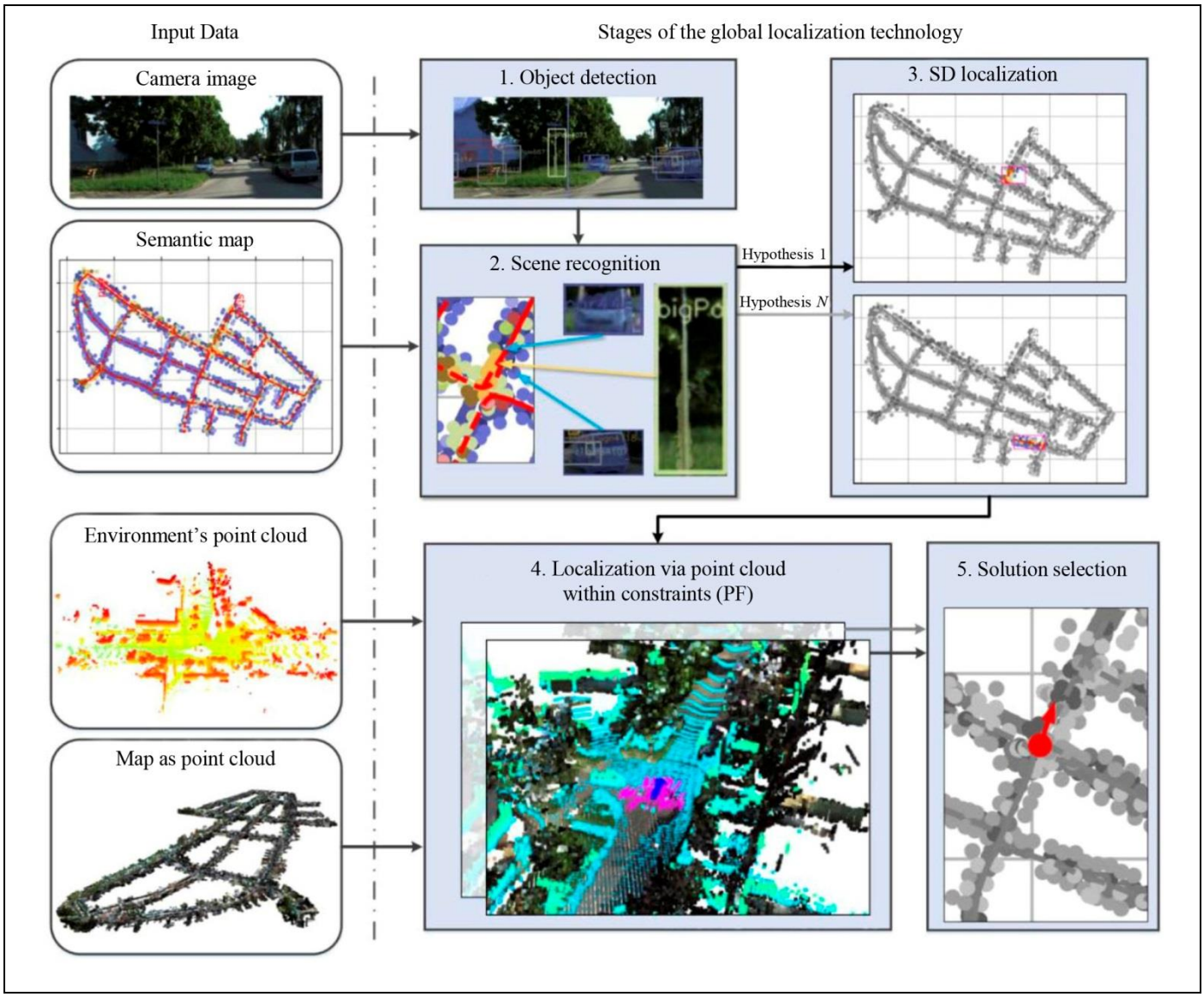

Fig. 1. The flowchart of the complex global localization technology.

2.1. Localization via Landmarks Based on SD Models

To apply SD computations, one has to describe the problem under consideration in the form of an SD model. Such a model represents the desired variables, their definitional domain, as well as interpretation and assignment functions to update them [28]. The desired variables—the robot's position—are specified in the form of multi-intervals, i.e., a non-intersecting set of intervals in ascending order:

$$^{*}a = \left[\left[a^1_{\text{low}}, a^1_{\text{high}}\right],\ldots,\left[a^N_{\text{low}}, a^N_{\text{high}}\right],\right.$$

$$\left. a^i_{\text{low\high}} \in R,\ a^i_{\text{low}} \le a^i_{\text{high}},\ a^i_{\text{high}} < a^{i+1}_{\text{low}}\right],$$

where $^{*}a$ is some SD variable defined by a multi-interval; a^i_{low} and a^i_{high}, $i = \overline{1,\ N-1}$, are lower and upper limits, respectively, of intervals forming a multi-interval. Multi-intervals are a special case of SD variables, denoted by the left-hand superscript "*" throughout this paper.

Originally, these multi-intervals are initialized with all available values. Then, the so-called interpretation functions are defined to reduce the uncertainty (i.e., narrow the limits of the multi-intervals): the new value calculated at each iteration is the intersection of the old one with that obtained by applying the interpretation function. The absence of intersection means that the input data contain inconsistencies and should be excluded from consideration. Interpretation functions are defined based on the following relationship between the robot's position (*X, *Y, *θ), the position of the observed landmarks (X_l, Y_l), and their measurements, which include the distance r_l and angle α_l to the object:

$$^{*}X = \pm\sqrt{\left(r + \Delta r_l\right)^2 - \left(^{*}Y - Y_l\right)^2} + X_l, \quad (1)$$

$$^{*}Y = \pm\sqrt{\left(r + \Delta r_l\right)^2 - \left(^{*}X - X_l\right)^2} + Y_l, \quad (2)$$

$$^{*}X = \frac{^{*}Y - Y_l}{\tan\left(^{*}\theta + ^{*}\Delta a_l + a_l\right)} + X_l, \quad (3)$$

$$^{*}Y = (^{*}X - X_l)\tan\left(^{*}\theta + ^{*}\Delta a_l + a_l\right) + Y_l, \quad (4)$$

$$^{*}\theta = \arctan\left(\frac{Y_l - ^{*}Y}{X_l - ^{*}X}\right) - \left(a_l + ^{*}\Delta a_l\right), \quad (5)$$

where $^{*}\Delta r_l$ and $^{*}\Delta\alpha_l$ are the measurement errors of r_l and α_l, respectively, expressed in intervals (they can be obtained from the RMS errors by the N sigma rule: $^{*}\Delta r_l = [-N\sigma r_l, N\sigma r_l]$. In formulas (1)–(5), some variables are represented as multi-intervals and are calculated using interval arithmetic [34]. For each observed landmark obtained during scene recognition, a different set of the interpretation functions (1)–(5) is formed. The procedure of SD computations is reduced to the iterative application of functions to a set of SD variables. Once a function has been applied, it is removed from the list of active functions; however, if an input variable for some assignment function outside the active list has been updated (its value has changed), this function will be returned to the list of active ones. The computational procedure continues until the list of active functions is empty. In practice, one may also limit the maximum number of iterations or set a desired accuracy for the resulting values of the variables.

At the same time, several situations cannot be resolved by scene recognition methods. These include symmetry situations: algorithms on graphs are unable to distinguish the order of appearance of objects when increasing the angle to them, since the graph contains information only about the distance between two objects. This situation can be resolved at the level of SD models by introducing additional correctness checking functions. Based on the current constraints ($^{*}X$, $^{*}Y$, $^{*}\theta$), it is necessary to calculate the angles to the leftmost and rightmost objects in the scene and check the nonnegativity of their difference:

$$\neg\left(\arctan\left(\frac{^{*}Y - Y_{\text{left}}}{^{*}X - X_{\text{left}}}\right) - \arctan\left(\frac{^{*}Y - Y_{\text{right}}}{^{*}X - X_{\text{right}}}\right) < 0\right), \quad (6)$$

where the subscripts “left” and “right” indicate the leftmost and rightmost, respectively, landmarks by the angle. One should apply additional correctness checking functions not to the full set ($^{*}X$, $^{*}Y$, $^{*}\theta$) but to the Cartesian product where each element is a tuple of three intervals for each variable. The tuples not satisfying condition (6) are excluded from the total set.

The output is the values of the SD variables represented by the set of tuples of intervals. Such tuples describe the union of rectangular domains (with sides parallel to the x- and y-axes, sometimes referred to as bounding rectangles [34]) specifying an angle constraint. However, the answer in this form may be somewhat redundant and contain unnecessary values (Fig. 2a).

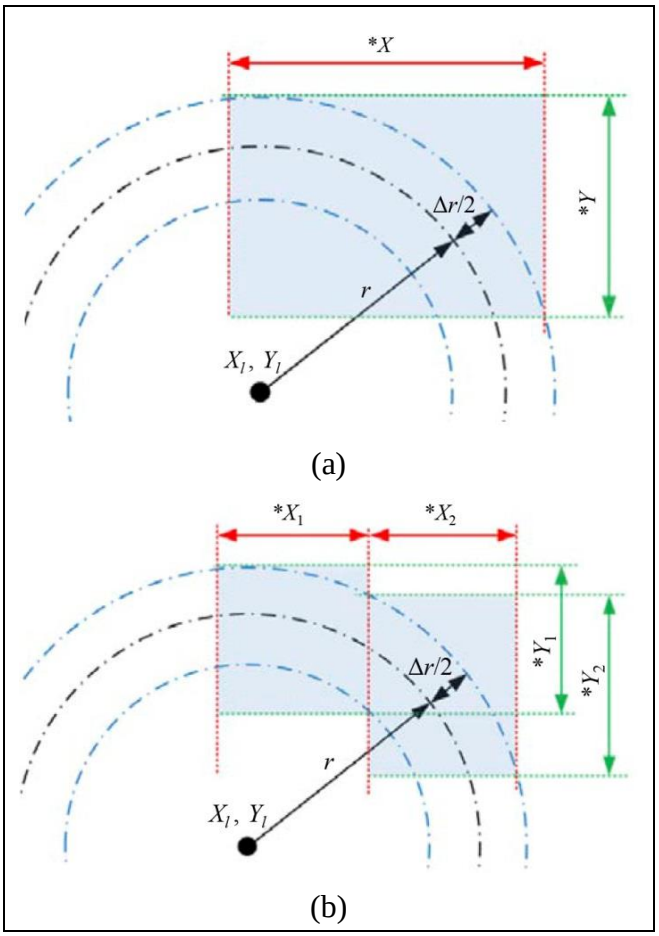

Fig. 2. Exhaustive estimation: (a) the redundant estimation of the robot’s location bounded by blue dash-and-dot lines, where $^{*}X$ is the input SD variable and $^{*}Y$ is the calculated one; (b) the exhaustive estimation of its location by splitting the input variable.

For such situations, exhaustive estimation algorithms are used in interval analysis [34]: the intervals exceeding a given constraint are iteratively split, and the computational procedure is reapplied to the result. Figure 2b shows an example of such splitting after the first iteration of the algorithm on the variable $^{*}X$ and the computation of the new values of $^{*}Y_1$ and $^{*}Y_2$. Thus, the general global localization algorithm based on SD computations (SD localization algorithm) can be represented as the following pseudocode.

Global SD localization algorithm via visual landmarks

```
1.   Algorithm sdm_global_localization({<X1, Y1, r1, α1, σr, σα>}, Nσ,
     xy_lim):
2.       -- X1, Y1 – the positions of objects on the map
3.       -- r1, α1 – the measurements of objects in the scene
4.       -- σr, σα – the measurement errors of objects
5.       -- Nσ – the factor to convert the errors into intervals
6.       -- xy_lim – the limit size of intervals for x and y
7.       V = ∅
8.       *X = *Y = [[-∞, ∞]]
9.       *θ = [[-π, π]]
10.      R = init_R({<X1, Y1, r1, α1, σr, σα>}, Nσ) -- (1)-(5)
11.      Status, *X, *Y, *θ = sdm_process(*X, *Y, *θ, R)
12.      if !Status
13.         return V
14.      C = init_C({<X1, Y1, r1, α1, σr, σα>}, Nσ) -- (6)
15.      function process_leaf(*x, *y, *γ)
16.         Status,*x, *y, *γ = sdm_process(*x, *y, *γ, R)
17.         if Status
18.            if |*x| > xy_lim:
19.               *x1, *x2 = divide(*x)
20.               process_leaf(*x1, *y, *γ)
21.               process_leaf(*x2, *y, *γ)
22.            elif |*y| > xy_lim:
23.               *y1, *y2 = divide(*y)
24.               process_leaf(*x, *y1, *γ)
25.               process_leaf(*x, *y2, *γ)
26.            else
27.               add (*x, *y, *γ) to V
28.      for (*x, *y, *γ) in *X×*Y×*θ do
29.         if C(*x, *y, *γ) then
30.            process_leaf(*x, *y, *γ)
31.      return V
```

The **sdm_global_localization** algorithm takes as input a set of landmark positions and their measurements, the factor $N\sigma$ to convert the RMS errors into intervals, and the limit *xy_lim* for the intervals. The **init_R** auxiliary function initializes the parameters of the assignment function (1)–(5) based on landmark data. The **sdm_process** function computes the SD variables as described above (also, see [28]) and returns the status (success or failure) and the updated values of the SD variables. The **init_C** function initializes the parameters of the additional correctness checking functions (6). The **divide** function splits an interval into two parts at its center, and the algorithm returns a set of interval tuples that do not exceed the limit *xy_lim* in the *x*- and *y*-axis and have passed all additional checks.

2.2 Localization Using a Particle Filter within the Constraints

The constraints yielded by the **sdm_global_localization** algorithm describe a domain where the robot can be located. With these constraints available, one can carry out a search using a PF. Each particle in this context is a hypothesis about the robot's position. It is required to initialize the initial distribution within the constraints uniformly. For the operation of the PF, it is necessary to evaluate the quality of each particle. The idea is to use dense map data in the form of a point cloud and a point cloud obtained from the robot. To compare the two point clouds, one should calculate the shortest distances between the points in the robot's cloud and those in the map cloud.

The arithmetic mean of these distances will show the overlapping quality of the clouds. However, it is recommended to use a more "sensitive" estimate to accelerate the convergence of the PF:

$$p(d, \sigma) = \frac{e^{-(d/\sigma)^2}}{\sqrt{2\pi}\sigma}, \quad (7)$$

where p is the desired particle estimate; d is the mean of the shortest distances; finally, σ is the desired sensitivity threshold. At each iteration of the PF operation, the weights of all particles are calculated by transferring the point cloud from the robot to the particle coordinates and calculating the estimate (7). Next, a *resampling* procedure is applied to discard low-weight particles and multiply high-weight ones; the probability with which a particle will fall into the new sample is proportional to its weight. Because some particles are represented in multiple instances in the resampling process, it makes sense to add some random noise to each particle at the beginning of each iteration in order to separate the same hypotheses and cover the search space more densely. The procedure consisting of the above steps can be performed either for a fixed number of iterations or until reaching target values of some criteria, e.g., the mean weight of the particles, the values of the covariance matrix constructed from the particle distribution, etc. There are different practices to select the desired position: the weighted average of all particles, the particle with the maximum weight, and initial clustering of particles with the subsequent application of the above methods to local clusters. Once the desired robot's position is computed, its quality can be evaluated using the same metric (7). The global localization algorithm using a PF is as follows.

Global localization algorithm via point clouds using a particle filter within constraints

```
1.  Algorithm pf_global_localization(V, σ, N, iter, Δ, PCD_map, PCD_robot):
2.     -- V – constraints on the robot's position
3.     -- σ – sensitivity
4.     -- N – the number of particles
5.     -- iter – the number of iterations
6.     -- Δ – noise parameters
7.     -- PCD_map – map point cloud
8.     -- PCD_robot – robot's point cloud
9.     P = ∅
10.    for n in N do
11.       v = sample_element(V)
12.       p = sample_particle(v)
13.       add p to P
14.    for i in iter do
15.       W = ∅
16.       P' = ∅
17.       for p in P do
18.          p = random_shift(p, Δ)
19.          add p to P'
20.          PCD = transform_cloud(PCD_robot, p)
21.          d = get_mean_p2p(PCD, PCD_map)
22.          w = get_w(d, σ) -- (7)
23.          add w to W
24.       P = resampling(P', W)
25.    pose_final = get_pose(P, W)
26.    PCD_final = transform_cloud(PCD_robot, pose_final)
27.    d_final = get_mean_p2p(PCD_final, PCD_map)
28.    w_final = get_w(d_final, σ) -- (7)
29.    return pose_final, w_final
```

The **pf_global_localization** algorithm takes as input a set of constraints yielded by the **sdm_global_localization** algorithm, a series of parameter values, and point clouds from the robot and map. The **sample_element** function randomly selects one of the interval tuples in proportion to the total domain size (the product of the sizes of all intervals). The **sample_particle** function equiprobably generates a particle including the position on the *x*- and *y*-axis and the angle so that all values will belong to the selected intervals. The **random_shift** function adds random white noise with the specified parameters to the particles. The **transform_cloud** function performs a geometric transfer of the point cloud to the particle's coordinates. The **get_mean_p2p** function computes the shortest distances between the points of the first and second clouds. The **get_w** function computes the particle weight by formula (7). The **resampling** function resamples the particles, and the **get_pose** function computes the robot's position in one of the above ways (e.g., by taking the weighted average, by multiplying the particle matrix by a weight vector). The algorithm returns the robot's position and the weight of this position as a quality criterion.

Thus, the **sdm_global_localization** and **pf_global_localization** algorithms are sequentially applied to each hypothesis selected for matching scene objects to map objects; in the resulting set of solutions with calculated quality criteria, the best one is chosen in terms of the introduced metric.

3. AN EXPERIMENTAL STUDY

The complex technology was experimentally studied on KITTI-360 [35], an adapted open dataset, for the scene recognition task proposed in [32]. The prepared data[1] contain descriptions of semantic maps and scenes in the form of object sets with an indication of matches between these sets. As input data for the **sdm_global_localization** algorithm, we chose the results of the scene recognition method based on the search of isomorphic subgraph *sig_lite_clip* [32], which showed the best results in terms of the metric of finding a completely correct answer. The answer for the scene recognition task is a match between observed objects and objects on the map; this match is used to form a localization task based on landmark data. Nevertheless, a significant part of the answers did not contain the correct answer (when all objects are matched correctly) in the first place in the list of hypotheses ordered by the certainty factor (a quality value of scene recognition results). To test the effectiveness of the technology in dealing with such input data, the ten answers with the maximum certainty factors were selected for each input scene.

The selected scenes were divided into several groups: *super_true*, *has_true*, and *wrong*. The *super_true* group corresponds to those scene recognition results in which the correct answer of matching scene objects to map objects is present and has the highest certainty factor among the selected ten answers. The *has_true* group contains the answers in which the correct answer is present, but the certainty factor is not the highest among the other hypotheses. The *wrong* group contains results without completely correct answers. The size of the groups was correlated as 68%, 27%, and 4% to the total number of scenes, and the remaining number was not processed by recognition methods. All experiments were carried out on a computer with an AMD Ryzen7 2700X Eight-Core 3.70GHz processor and 32GB RAM, without involving GPUs.

For the ten hypotheses obtained, the SD localization procedure (SDM, *Subdefinite Models*) with $N\sigma = 4$, in the variants with ($xy\_\mathrm{lim} = 10$ m) and without ($xy\_\mathrm{lim} = \infty$ m) additional interval splitting was performed for each scene (see Fig. 2). The resulting interval constraints on the robot's position for all hypotheses of the same scene were evaluated for interval accuracy and interval coverage (Table 1). The interval accuracy was calculated by determining whether the true robot's position, known from the dataset, belongs to the calculated interval constraints (the larger the value is, the better the result will be). The interval coverage reflects the reduction of the search space. It was evaluated by distributing a million particles (robot's positions) uniformly over the entire map area, determining the number of particles belonging to any of the resulting constraint sets, and taking the ratio of the resulting number to the entire set (the smaller its value is, the better the result will be).

According to Table 1, the *wrong* group (no completely correct answers) is far behind the other groups in interval accuracy. Also, there is an order-of-magnitude difference in interval coverage for the variant of the algorithm with splitting, albeit with a small loss of interval accuracy, compared to the variant without splitting. However, an order-of-magnitude decrease in interval coverage causes an order-of-magnitude increase in the running time of the algorithm.

[1] URL: https://github.com/MoscowskyAnton/scene_recognition_kitti_360

Table 1

Testing results for the SD localization approach

Method	Interval accuracy, %, ↑			Interval coverage, %, ↓			|*t*|, s
	Group			Group			All groups
	super_true	*has_true*	*wrong*	*super_true*	*has_true*	*wrong*	
SDM with splitting	97.1	98	17.6	**0.23**	**0.41**	0.65	0.69
SDM without splitting	**99.8**	**100**	**37**	3.22	4.44	**0.37**	**0.03**

The proposed technique was compared with other global localization methods to evaluate the accuracy of determining the robot's position. All approaches under study were implemented by the author or taken from open source libraries. The methods working with the scene recognition result received all ten variants for each scene as input and chose the best one by the quality value; for the best solution, the position and angle errors and the running time were calculated accordingly. The errors were described by the mean (|.|) and the median (*M*). The running time of the scene recognition method was not considered; for the method and sequence selected, it was 41.55 ± 46.5 s [32] on the same computational hardware. Note that in the KITTI-360 set, the point clouds for the map are presented not in a monolithic version but as several overlapping domains (sub-maps); where possible, one of the sub-maps best fitting the search space was selected. The following approaches were considered in the study:

– **RANSAC+ICP**, a classical localization approach based on point clouds only, with the calculation of FPFH descriptors [3], searching for their matches using the RANSAC method [8], and further refinement using ICP [12]. This approach was chosen for consideration because of the available open implementations. RANSAC matching was performed for each sub-map, and ICP refinement was performed for the position with the highest *fitness* value.

– **SVD**, a localization method based on scene recognition data [19]. For different scene recognition results, the quality of the solution was determined by the average position error of each object between the scene and the map.

– **SVD+ICP**, localization from scene recognition data using the SVD method and further position refinement using ICP via point clouds. The *fitness* parameter of the ICP method was taken as a quality value.

– **SDM+PF**, the SD localization technology via landmarks (proposed in this paper) with the **sdm_global_localization** algorithm ($N\sigma$ = 4, xy_lim = 10 m) and a PF used within the constraints by the **pf_global_localization** algorithm (N = 150; σ = 0.1; *iter* = 5; Δ = (0.5; 0.1)). The particle with the highest weight was selected as the solution; the weight also served as a criterion for selecting solutions for different scene recognition results.

– **SDM+RANSAC+ICP**, the **RANSAC+ICP** approach applied to a point cloud derived from the constraints calculated by the **sdm_global_localization** algorithm. The computed constraints were extended by the range of the rangefinder. The *fitness* value of the ICP method was also taken as a measure of solution evaluation. The variant of the SDM method without splitting (xy_lim = ∞) was used, and the other parameters corresponded to those in **SDM+PF**.

Additionally, the *recall@k* (*R@k*) metric, often encountered in global localization tasks, was calculated for the results. It shows the percentage of the answers falling within some certainty region for *k* answers. In the literature on global localization [36, 37], the certainty region is usually chosen to be 20 m: from a practical viewpoint, it is no longer important how much the error exceeds this value.

Table 2 presents the numerical results of the methods obtained for the *super_true* group.

According to the results in Table 2, the widespread SVD approach without reference to point clouds significantly outperforms the other methods in almost all parameters. The matter is that the certainty factor used for normalizing answers in scene recognition methods ideologically coincides with the quality metric of the SVD method; therefore, for the *super_true* group, this method makes almost no mistake, selecting the hypothesis with the highest certainty factor (actually, the correct one). This fact can also be observed when comparing SVD with its extension SVD+ICP, where the mean error increases significantly due to the appearance of scenes with a different hypothesis preferred, but the median error is reduced compared to SVD as ICP improves the robot's position on the point cloud data in over 50% of the cases. Other methods using scene recognition results show a comparable median but a strongly higher mean due to incorrectly chosen hypotheses. (In some cases, they are at other ends of the map, thereby significantly affecting the mean.) The impact of errors "distant" from the correct answer is also indirectly confirmed by the superior *R@k* values for SDM+PF, since the *R@k* metric describes a threshold estimate.

The results for the *has_true* group are of greater interest within the study because they reflect the ability of different methods to find the correct answer in a set of similar hypotheses. Table 3 summarizes the numerical results for this group.

In Table 3, SVD (the best method of Table 2) has significant accuracy losses and shows poor performance. The reason is that it neglects additional point cloud data, in contrast to the other methods. This fact confirms the importance of the complex approach proposed above. However, additional correction using ICP significantly improves the accuracy and provides almost better results in all parameters and good time performance of the algorithm. Generally speaking, the values of the accuracy measures of the methods based on the results of scene recognition and position refinement via point clouds insignificantly differ from those for the *super_true* group (see Table 2). Hence, it is possible to find a solution among many similar hypotheses. The strong decrease in the *R*@1 value is due to the incorrect best answer in the *has_true* group; the decrease in the *R*@5 value is because the correct answer may fall outside the top five solutions.

The group of *wrong* solutions (without a completely correct answer) is also of interest in localization via visual landmarks: the problem statement contains knowingly false data. The numerical results are given in Table 4.

According to the experimental results in Table 4, the lack of a correct answer from the scene recognition system has a negative impact on the results (all accuracy measures decrease significantly compared to Tables 2 and 3). Therefore, it is important to develop robust scene recognition methods. The SD localization approach based on correctness checking functions allows discarding some hypotheses: on average, 17.5% of all processed hypotheses are discarded, and this value varies insignificantly within the groups under study (16.6, 20.1, and 18.5%, respectively).

Figure 3 provides the values of *recall*@1 and *recall*@5 without division into groups.

The proposed methods based on search space reduction (SDM+PF and SDM+RANSAC+ICP) show comparable results, in terms of accuracy, with the SVD-based approach, even excelling them in some cases (see Fig. 3). Hence, this R&D direction is

Table 2

Experimental results for the *super_true* group of solutions

Method	$\|r\|$, m	$M(r)$, m	$\|\alpha\|$, rad	$M(\alpha)$, rad	$\|t\|$, s	$M(t)$, s	R@1, %	R@5, %
RANSAC+ICP	233.3	190.1	0.71	0.12	4418.1	4171.5	37	-
SVD	**4.7**	2.3	**0.26**	**0.06**	**0.005**	**0.004**	94.4	95.6
SVD+ICP	39.5	1.6	0.29	**0.06**	18.6	17.7	94.9	96.1
SDM+PF	83.7	2.1	0.39	0.06	95.1	97.1	**98.4**	**99.3**
SDM+RANSAC+ICP	100.6	**1.4**	0.28	0.05	1009	976	69.4	83.7

Table 3

Experimental results for the *has_true* group of solutions

Method	$\|r\|$, m	$M(r)$, m	$\|\alpha\|$, rad	$M(\alpha)$, rad	$\|t\|$, s	$M(t)$, s	R@1, %	R@5, %
RANSAC+ICP	207.5	158.9	1	0.46	4038.3	4002	27.7	–
SVD	191.1	28.9	1.15	0.83	**0.005**	**0.004**	44.3	85.8
SVD+ICP	**38.4**	2	0.35	0.04	19.6	19.6	44.5	**85.9**
SDM+PF	80.4	**1.5**	0.29	0.05	94.8	95.6	**62.2**	63.9
SDM+RANSAC+ ICP	71.5	6.7	**0.22**	**0.03**	911.2	878.3	61.5	84.6

Table 4

Experimental results for the *wrong* group of solutions

Method	$\|r\|$, m	$M(r)$, m	$\|\alpha\|$, rad	$M(\alpha)$, rad	$\|t\|$, s	$M(t)$, s	R@1, %	R@5, %
RANSAC+ICP	289	262.3	0.87	**0.06**	4501	4255	**33.3**	–
SVD	351	298.3	1.22	0.92	**0.004**	**0.004**	24.3	**30.7**
SVD+ICP	206.7	**95.2**	1.13	0.86	17.8	17.2	24.5	**30.7**
SDM+PF	**203.4**	150.8	1.34	1	79.1	83.4	25.7	25.7
SDM+RANSAC+ICP	210.5	285.5	**0.59**	0.07	1076.6	1030.9	13.3	20

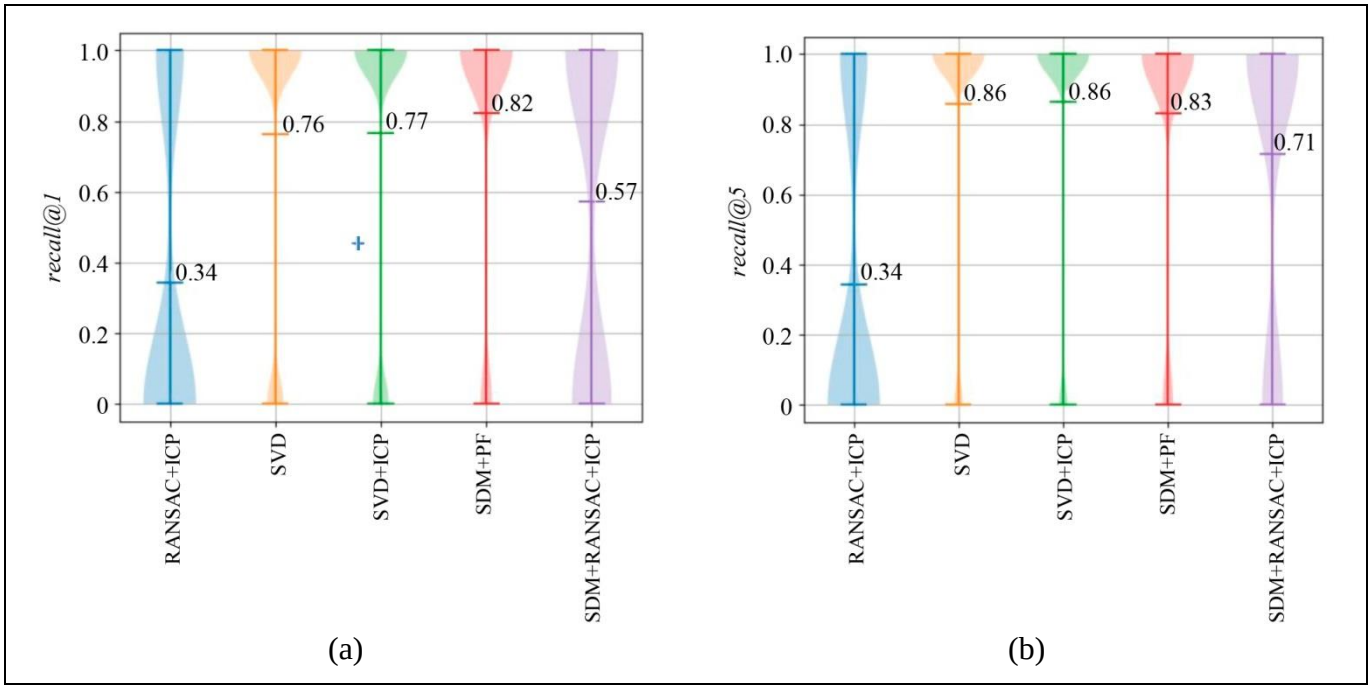

Fig. 3. The distributions and means of *recall@k* for different methods: (a) *recall*@1 and (b) *recall*@5.

promising for solving global localization problems. The values of the operating parameters of the algorithms used in this study are rather for orientation, since no software optimization issues arose during the implementation of the methods. The PF-based approach using the point-to-point distance for cloud comparison has a limitation: it performs well only near the correct answer and with small deviations in angle. Investigating and applying more sensitive metrics for comparing point clouds, including their color characteristics, would potentially improve the result as well. Note separately that SDM+RANSAC+ICP has demonstrated both a significant increase in accuracy and a reduction in running time compared to the classical RANSAC+ICP approach. This finding suggests that the search space reduction technology based on SD localization can be used for other global localization methods with point clouds.

CONCLUSIONS

This paper has described a subdefinite (SD) localization approach as part of a complex global localization technology for a robot on 3D maps equipped with a semantic layer. By recognizing and localizing the objects in the robot's field of view, one can solve the scene recognition problem by matching them with the map objects. The SD localization approach is applied to the resulting data to determine constraints on the robot's position, which in turn can be used to reduce the search area for a localization method based on rangefinder data.

According to the experiments, the point cloud localization methods applied within the constraints obtained by the complex technology significantly improve the accuracy and time performance of the algorithm compared to the basic global localization approach without semantics. The reduction of the search space to 0.2% of the map size confirms the applicability of this technology for any localization methods based on rangefinder data. The experiments have also demonstrated that combining localization via landmarks with localization via point clouds is effective when the correct scene recognition answer "hides" among several similar hypotheses. Thus, based on the experimental results, the aim of this research—improving the quality of global localization—has been achieved. The quality of the algorithm has been improved owing to SD computations, which are used within the integrated localization technology with semantic maps.

Further research will analyze and apply more sensitive point cloud registration methods to increase the accuracy of robot positioning and hypothesis estimation. In addition, it is reasonable to consider the proposed approach jointly with Place Recognition to reduce the search space among the available key frames: they are often provided with a coordinate label, and the latter can be used to determine compliance with the constraints yielded by SD localization. Note that a

promising task is the multicriteria optimization of the parameters of the methods used, which requires significant computational resources. However, this task goes beyond the scope of the paper, as the parameters have been chosen empirically.

Acknowledgments. *This work was carried out under the state assignment of the National Research Center "Kurchatov Institute."*

REFERENCES

1. Chen, P., Zhao, X., Zeng, L., et al., A Review of Research on SLAM Technology Based on the Fusion of LiDAR and Vision, *Sensors*, 2025, vol. 25, no. 5, art. no. 1447.
2. Yin, H., Xu, X., Lu, S., et al., A Survey on Global LiDAR Localization: Challenges, Advances and Open Problems, *Int. J. Comput. Vis*., 2024, vol. 132, no. 8, pp. 3139–3171.
3. Rusu, R.B., Blodow, N., and Beetz, M., Fast Point Feature Histograms (FPFH) for 3D Registration, *Proceedings of 2009 IEEE International Conference on Robotics and Automation*, Kobe, Japan, 2009, pp. 3212–3217.
4. Saarinen, J., Andreasson, H., Stoyanov, T., et al., Normal Distributions Transform Monte-Carlo Localization (NDT-MCL), *Proceedings of 2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Tokyo, Japan, 2013, pp. 382–389.
5. Mildenhall, B., Srinivasan, P.P., Tancik, M., et al., NeRF: Representing Scenes As Neural Radiance Fields for View Synthesis, *Commun. ACM*, 2022, vol. 65, no. 1, pp. 99–106.
6. Komorowski, J., MinkLoc3D: Point Cloud Based Large-Scale Place Recognition, *Proceedings of 2021 IEEE Winter Conference on Applications of Computer Vision (WACV)*, Waikoloa, HI, USA, 2021, pp. 1789–1798.
7. Rosen, D.M., Doherty, K.J., Terán Espinoza, A., et al., Advances in Inference and Representation for Simultaneous Localization and Mapping, *Annu. Rev. Control. Robot. Auton. Syst*., 2021, vol. 4, no. 1, pp. 215–242.
8. Fishler, M. and Bolles, R., Random Sample Consensus: A Paradigm for Model Fitting Applications to Image Analysis and Automated Cartography, *Communication of the ACM*, 1981, vol. 24, pp. 381–395.
9. Yang, H., Shi, J., and Carlone, L., TEASER: Fast and Certifiable Point Cloud Registration, *IEEE Trans. Robot*., 2021, vol. 37, no. 2, pp. 314–333.
10. Lusk, P.C. and How, J.P., CLIPPER: Robust Data Association without an Initial Guess, *IEEE Robot. Autom. Lett.*, 2024, vol. 9, no. 4, pp. 3092–3099.
11. Choy, C., Dong, W., and Koltun, V., Deep Global Registration, *Proceedings of 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Seattle, WA, USA, 2020, pp. 2511–2520.
12. Rusinkiewicz, S. and Levoy, M., Efficient Variants of the ICP Algorithm, *Proceedings of the 3rd International Conference on 3-D Digital Imaging and Modeling*, Quebec City, Canada, 2001, pp. 145–152.
13. Vizzo, I., Guadagnino, T., Mersch, B., et al., KISS-ICP: In Defense of Point-to-Point ICP – Simple, Accurate, and Robust Registration If Done the Right Way, *IEEE Robot. Autom. Lett.*, 2023, vol. 8, no. 2, pp. 1029–1036.
14. Barfoot, T.D., *State Estimation for Robotics*, Cambridge: Cambridge University Press, 2017.
15. Chen, R., Yin, H., Jiao, Y., et al., Deep Samplable Observation Model for Global Localization and Kidnapping, *IEEE Robot. Autom. Lett*., 2021, vol. 6, no. 2, pp. 2296–2303.
16. Zhang, X., Wang, L., and Su, Y., Visual Place Recognition: A Survey from Deep Learning Perspective, *Pattern Recognit.*, 2021, vol. 113, art. no. 107760.
17. Xia, L., Cui, J., Shen, R., et al., A Survey of Image Semantics-Based Visual Simultaneous Localization and Mapping: Application-Oriented Solutions to Autonomous Navigation of Mobile Robots, *Int. J. Adv. Robot. Syst.*, 2020, vol. 17, no. 3, art. no. 172988142091918.
18. Jin, X., Li, X., Zhu, Y., et al., A Survey of Visual Semantic Mapping, *Proceedings of 2022 3rd Asia-Pacific Conference on Image Processing, Electronics and Computers*, New York, NY, USA, 2022, pp. 295–301.
19. Arun, K.S., Huang, T.S., and Blostein, S.D., Least-Squares Fitting of Two 3-D Point Sets, *IEEE Trans. Pattern Anal. Mach. Intell*., 1987, vol. PAMI-9, no. 5, pp. 698–700.
20. Pramatarov, G., De Martini, D., Gadd, M., et al., BoxGraph: Semantic Place Recognition and Pose Estimation from 3D LiDAR, *arXiv:2206.15154v1*, 2022. DOI: 10.48550/arXiv.2206.15154
21. Wu, Y., Zhang, Y., Zhu, D., et al., An Object SLAM Framework for Association, Mapping, and High-Level Tasks, *arXiv:2305.07299*, 2023. DOI: https://doi.org/10.48550/arXiv.2305.07299
22. Ankenbauer, J., Lusk, P.C., Thomas, A., et al., Global Localization in Unstructured Environments Using Semantic Object Maps Built from Various Viewpoints, *Proceedings of 2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Detroit, MI, USA, 2023, pp. 1358–1365.
23. Guo, X., Hu, J., Chen, J., et al., Semantic Histogram Based Graph Matching for Real-Time Multi-Robot Global Localization in Large Scale Environment, *IEEE Robot. Autom. Lett.*, 2021, vol. 6, no. 4, pp. 8349–8356.
24. Wang, Y., Jiang, C., and Chen, X., GOReloc: Graph-Based Object-Level Relocalization for Visual SLAM, *IEEE Robot. Autom. Lett*., 2024, vol. 9, no. 10, pp. 8234–8241.
25. Matsuzaki, S., Sugino, T., Tanaka, K., et al., CLIP-Loc: Multimodal Landmark Association for Global Localization in Object-Based Maps, *Proceedings of 2024 IEEE International Conference on Robotics and Automation (ICRA)*, Yokohama, Japan, 2024, pp. 13673–13679.
26. Liu, Y., Petillot, Y., Lane, D., et al., Global Localization with Object-Level Semantics and Topology, *Proceedings of 2019 International Conference on Robotics and Automation (ICRA)*, Montreal, Canada, 2019, pp. 4909–4915.
27. *Handbook of Constraint Programming*, Rossi, F., van Beek, P., and Walsh, T., Eds., Amsterdam: Elsevier, 2006.
28. Narin'yani, A.S., Introduction to Subdefiniteness, *Information Technologies*, 2007, no. 4, pp. 1–32. (In Russian.)
29. Moscowsky, A., Subdefinite Computations for Reducing the Search Space in Mobile Robot Localization Task, in *Artificial Intelligence. RCAI 2021*, Kovalev, S.M., Kuznetsov, S.O., and Panov, A.I., Eds., Lecture Notes in Computer Science, Cham: Springer, 2021, vol. 12948, pp. 180–196.
30. Moscowsky, A.D., Extended Object Detection: Flexible Object Description System for Detection in Robotic Tasks, in *Smart Electromechanical Systems*, Gorodetskiy, A.E. and Tarasova, I.L., Eds., Studies in Systems, Decision and Control, Cham: Springer, 2022, vol. 419, pp. 27–43.
31. Contreras, M., Jain, A., Bhatt, N.P., et al., A Survey on 3D Object Detection in Real Time for Autonomous Driving, *Front. Robot. AI*, 2024, vol. 6, no. 11, art. no. 1212070

32. Moscowsky, A.D., Scene Recognition for the Mobile Robot Global Localization Problem Based on Image Vectorization and Graphs Approaches, *Large-Scale Systems Control*, 2025, no. 114, pp. 307–344. (In Russian.)
33. Radford, A., Kim, J.W., Hallacy, C., et al., Learning Transferable Visual Models from Natural Language Supervision, *Proceedings of the International Conference on Machine Learning*, Vienna, Austria, 2021, pp. 8748–8763.
34. Bazhenov, A.N., Zhilin, S.I., Kumkov, S.I., et. al., *Obrabotka i analiz interval'nykh dannykh* (Processing and Analysis of Interval Data), Moscow–Izhevsk: Institute of Computer Studies, 2024. (In Russian.)
35. Liao, Y., Xie, J., and Geiger, A., KITTI-360: A Novel Dataset and Benchmarks for Urban Scene Understanding in 2D and 3D, *IEEE Trans. Pattern Anal. Mach. Intell.*, 2023, vol. 45, no. 3, pp. 3292–3310.
36. Keetha, N., Mishra, A., Karhade, J., et al., Anyloc: Towards Universal Visual Place Recognition, *arXiv:2308.00688*, 2023, DOI: https://doi.org/10.48550/arXiv.2308.00688
37. Puligilla, S., Omama, M., Zaidi, H., et al., LIP-Loc: LiDAR Image Pretraining for Cross-Modal Localization, *Proceedings of 2024 IEEE/CVF Winter Conference on Applications of Computer Vision Workshops (WACVW)*, Waikoloa, HI, USA, 2024, pp. 939–948.

This paper was recommended for publication by R. V. Meshcheryakov, a member of the Editorial Board.

Received May 5, 2025, and revised May 12, 2025. Accepted July 16, 2025

Author information

Moscowsky, Anton Dmitrievich. Research group head, National Research Center "Kurchatov Institute," Moscow, Russia
✉ moscowskyad@yandex.ru
ORCID iD: https://orcid.org/0000-0002-6546-8697

Cite this paper

Moscowsky, A.D., Application of Subdefinite Models in the Global Localization of a Mobile Robot, *Control Sciences* **4,** 55–67 (2025).

Original Russian Text © Moscowsky, A.D., 2025, published in *Problemy Upravleniya*, 2025, no. 4, pp. 64–78.

Translated into English by *Alexander Yu. Mazurov*,
Cand. Sci. (Phys.–Math.),
Trapeznikov Institute of Control Sciences, Russian Academy of Sciences, Moscow, Russia
✉ alexander.mazurov08@gmail.com