Localization of mobile robot in prior 3D LiDAR maps using stereo image sequence

I.V. Belkin [1,2], A.A. Abramenko [2], V.D. Bezuglyi [1], D.A. Yudin [1,3]
[1] Moscow Institute of Physics and Technology, 141700, Russia, Moscow Region, Dolgoprudny, Institutskiy per. 9;
[2] LLC Integrant, 127204, Russia, Moscow, Dolgoprudnenskoe highway. 3;
[3] Artificial Intelligence Research Institute (AIRI), 121170, Russia, Moscow, Kutuzovsky ave. 32 c1

Abstract

The paper studies the real-time stereo image-based localization of a vehicle in a prior 3D LiDAR map. A novel localization approach for mobile ground robot, which successfully combines conventional computer vision techniques, neural network based image analysis and numerical optimization, is proposed. It includes matching a noisy depth image and visible point cloud based on the modified Nelder-Mead optimization method. Deep neural network for image semantic segmentation is used to eliminate dynamic obstacles. The visible point cloud is extracted using a 3D mesh map representation. The proposed approach is evaluated on the KITTI dataset and a custom dataset collected from a ClearPath Husky mobile robot. It shows a stable absolute translation error of about $0.11 - 0.13$ m. and a rotation error of $0.42 - 0.62$ deg. The standard deviation of the obtained absolute metrics for our method is the smallest among other state-of-the-art approaches. Thus, our approach provides more stability in the estimated pose. It is achieved primarily through the use of multiple data frames during the optimization step and dynamic obstacles elimination on depth image. The method's performance is demonstrated on different hardware platforms, including energy-efficient Nvidia Jetson Xavier AGX. With parallel code implementation, we achieve an input stereo image processing speed of 14 frames per second on Xavier AGX.

Keywords: vehicle localization, optimization, deep learning, stereo camera, semantic segmentation, embedded systems.

Citation: Belkin IV, Abrameko AA, Bezuglyi VD, Yudin DA. Localization of mobile robot in prior 3D LiDAR maps using stereo image sequence. Computer Optics 2024; 48(3): 406-417. DOI: 10.18287/2412-6179-CO-1369.

Introduction

Autonomous robots' safe movement is impossible without high-precision localization. Ones often rely on the Inertial Navigation System (INS) to solve this problem; however, when the Global navigation satellite system and the Real-time kinematic (GNSS+RTK) data are neither available nor provide reliable information, an error begins to accumulate, which does not guarantee the robot traffic safety.

In addition to the INS/GNSS+RTK, LiDAR and the camera are popular sensors used for localization under challenging conditions [1, 2]. Significant progress has been made in the domain of visual and LiDAR simultaneous localization and mapping (SLAM) methods [3, 4, 5, 6]. LiDAR is a high precision sensor. This makes the LiDAR SLAM more accurate than the visual SLAM. However, LiDAR is a rather expensive sensor. So, it is very costly to create groups of robots using LiDAR for localization. From this point of view, it is much more attractive to use visual SLAM. However, both SLAM approaches accumulate errors over time. In the process of the robot movement, the LiDAR point clouds are distorted, which leads to minor errors that accumulate over time. In visual SLAM, the errors accumulate due to inaccuracy in determining feature points and depth

images. And if these errors are not compensated, it will lead to a wrong localization over time. Loop closure and INS using helps to compensate ones partially. Despite this, it is impossible to get rid of the errors accumulation completely. Therefore, these approaches are not suitable for the long-term localization of robots over a large environment. The problem of long-term localization can be solved using a prior map [7, 8, 9].

The essence of the map localization approach is to match sensor observations with a known map. Due to the development of map tools, it is possible to build a 3D map of the indoors or outdoors environment quickly and easily. Quite often, the map is presented and stored as a 3D point cloud. Point clouds capture the structural and geometrical features of the environment that are less likely to change over time and, therefore, do not require re-mapping unless major changes have occurred (e.g., construction or alteration of a road) [10]. This reduces the cost of compiling and maintaining the map up to date. In this regard, the use of such a map for localization becomes promising.

Localization on a prior map involves matching the map and data from the robot's sensors. For example, the combined usage of a 3D LiDAR point cloud map, on-board stereo camera images, and inertial measurement unit (IMU) data significantly improves visual localization

accuracy [11]. The camera is a lightweight, widespread, and lowcost sensor. Thanks to this, scaling to a large number of robots and a significant cost reduction of robots without the loss of high-precision localization is possible.

The successful results of deep learning in improving the quality of image processing and extracting additional information from them are impressive [12, 13, 14]. In this regard, it is relevant to use modern methods based on neural networks to improve the quality of localization [15, 16]. Thus, the well-known problem of degrading the quality of SLAM methods in environments with a large number of moving objects can be solved by segmentation and tracking of such objects. Obtaining dense depth images using deep learning seems to be a promising solution to the problem of a lot of noise on them.

In this paper, we pay special attention to localization using images from the lowcost stereo camera in a prior 3D LiDAR map. The main contributions of this work are as follows:

1. A novel real-time stereo camera based localization approach against prior 3D LiDAR map for mobile ground robot is proposed. It successfully combines conventional computer vision techniques, neural network based image analysis and numerical optimization. It achieves decimeter-level accuracy and high robustness against dynamic objects.

2. A novel algorithm for matching a noisy depth image and prior LiDAR point cloud map is proposed. It includes robustified loss function between depth image and point cloud, procedure of actually visible point cloud extraction from the whole map and an optimization procedure on $se(3)$ Lie Group based on Nelder-Mead optimization method.

3. The proposed approach is evaluated on the KITTI dataset and a custom dataset collected from a ClearPath Husky mobile robot. It shows a stable absolute translation error of about $0.11 - 0.13$ m and a rotation error of $0.42 - 0.62$ deg, which is comparable to methods which use the data from expensive LiDARs. Its performance is demonstrated on different hardware platforms, including energy-efficient Nvidia Jetson Xavier AGX, where it achieves up to 14 frames per second (FPS).

1. Related work

A dense depth map obtained with a stereo camera can be used directly for localization in the point cloud [8]. By minimizing the differences between the depth map and the projection of the prior 3D map points onto the image plane, the authors estimate the 6DoF position of the camera. For noise compensation on the depth map error, the weighting is applied based on its gradients. In [11], the dense point cloud, reconstructed using a stereo camera from several frames, is compared using the normal distribution transform (NDT) with an a priori map. The key element in NDT [17] is a representation for the map point cloud. Instead of matching the data point cloud to the points in the map directly, the probability of finding a point at a certain position is modeled by a linear combination of normal distributions. Because the points in the target scan are not used directly for matching, there is no need for computationally expensive nearest-neighbor search, as in Iterative closest point (ICP). The authors in [18] also uses stereo, but sparse. The position of the key points of the visual stereo SLAM is determined, taking into account the planes pre-selected from the map. The disadvantages of this group of methods include the fact that the map is built on the basis of a stereo point cloud, the accuracy of which is much lower than LiDAR data.

The presence of a prior map with the surrounding space geometry makes it possible to better solve the 7DoF pose estimation problem (6 degrees of freedom for position and orientation, one for scale), which is necessary when determining the position of a monocular camera. Thus, in the papers [7, 19], the local map of the key points of the visual monocular SLAM is matched using the ICP scheme with a prior map in the point cloud form. The disadvantage of such approaches is the increased computational complexity due to ICP.

Matching with a prior map can be based on a comparison of line segments extracted from the 3D terrain map in advance and from the image in the process of work [20, 21]. These methods, however, require a terrain map built with an expensive stationary laser scanner that allows obtaining a dense point cloud suitable for the line extraction.

Localization using the camera can be carried out along the road marking lines extracted from the image and compared with the map. Extraction from the image can be carried out using the edge detection methods [22] or using the semantic segmentation neural network [16, 23]. It is necessary to take into account the limited scope of this approach. Obviously, it will not work off-road and on roads without marking lines.

Free-form contours extracted from the image also allow localization relative to a pre-built map [24, 25]. Contour extraction from the map occurs after rendering an image based on it. In this case, in [24], prior contours are re-projected back to 3D using synthesized depth maps. Note that rendering, extracting and matching of contours require powerful computers for the operation of this group of methods in real time.

Information about the color of the map surfaces can also be taken into account for localizing with the camera. So in [26], a color map is built with the LiDAR and the camera. Then it is converted into a mesh. The position is found by minimizing mutual information between the camera image and the projection of such a map. A similar approach with minimizing the Normalized Information Distance is used in [27, 28]. The methods require a sufficiently detailed and dense map, which is not always possible to build.

The generation and evaluation of the hypotheses about the position based on the matching of the observed and synthesized map data were proposed in the papers [29, 30]. In [29], the LiDAR map contained an intensity field. Localization was carried out by minimizing the Normalized Mutual Information between the image and the projection of the intensity field of the terrain map. In [30], the matching was done between the image and the synthetic depth map.

The prior map can be used as a constraint when constructing a new map using visual SLAM methods. These restrictions are taken into account when solving the bundle adjustment task. In [31], the restrictions are introduced on the coincidence of planes in the current visual and prior maps. Point-to-point and point-to-plane constraints between SLAM key points and prior map points have been introduced in [10]. In [32] in addition to point-to-point constraints between visual landmarks and map points, a tightly-coupled fusion with IMU is used. The map representation in the form of surfels makes it possible to introduce more flexible restrictions on individual key points of the visual map [33]. The map can be presented as a mixture of Gaussians [9] to introduce similar constraints on points. In [34], the authors used Signed Distance Field, penalizing points as they move away from surfaces in the map. To perform registration between sparse point cloud of visual odometry (VO) and prior LiDAR map a probabilistic weighted normal distributions transformation (ProW-NDT) is proposed [35]. Prior map is used both for visual SLAM map enhancement and pose estimation.

It has recently been shown that a neural network can solve the task of localization with a prior map in an end-to-end way. In [15], the deep neural network evaluates the pose correction based on the input image and the depth map synthesized from the terrain map. The authors of the paper [36] used a neural network to estimate the vector field of pixel displacements for aligning the image and projection of the map onto its plane. The final assessment of the pose is done using PnP + RANSAC approaches. They demonstrated improvement the portability of the trained model between datasets and invariance of the proposed method to the internal parameters of the camera. In [37] dense scene matching by convolutional neural network (CNN) with subsequent PnP solver is proposed for camera localization. Compressed point cloud map representation HyperMap is proposed in [38]. Pose estimation is performed online by CNN, which takes as inputs camera image and projected from map virtual feature image. Feature extraction from map performed offline by neural network. The whole pipeline is trained end-to-end. The disadvantage of this group of methods is the low resistance to changes in lighting conditions and terrain.

Our research shows that there are no stereo image-based localization approaches on a prior map that provide real-time performance on energy-efficient platforms (e.g., Xavier AGX). Real-time localization means the ability to localize with a frequency of more than the frame rate of the input image stream, providing an accurate pose for every frame. In the proposed approach, the real-time operation is achieved by parallel run of two threads: fast visual SLAM and accurate error compensation based on the prior 3D LiDAR map.

2. Methodology
2.1. Scheme of the proposed localization approach

An overall scheme of the proposed localization approach is shown in the Fig. 1. The method takes as input a left and right images of a stereo camera. The images are then utilized for depth estimation and semantic segmentation of the visible scene. Next, the depth image and the left image are fed into the Odometry estimator, which gives an approximate displacement between two consecutive frames. Semantic segmentation is used for depth image postprocessing in Dynamic obstacle elimination module. All this data is then utilized in the Localization module which estimates the robot pose w.r.t. the prior 3D LiDAR map.
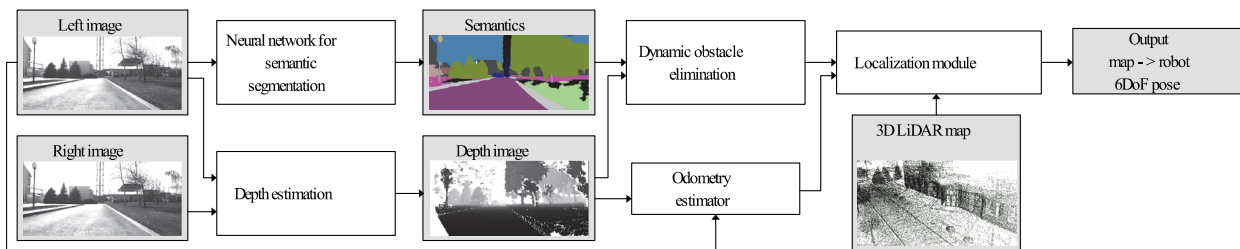

Fig. 1. Structure of localization approach proposed

2.2. Depth estimation

For the depth estimation we compare two approaches: rSGM [39] and AnyNet-M [14]. The first one is an analytical approach based on disparity estimation of the images from calibrated stereo camera. It is an accelerated version of Semi-global matching (SGM) [40, 41, 42, 43] for efficient execution on a CPU. The AnyNet-M is a modification of a real-time deep neural network AnyNet [13] with less downsampling of feature maps in the backbone. It is trained on the KITTI dataset [44].

Both depth estimation approaches show comparable metrics on the KITTI-2012 [45] dataset in terms of Out-

All (3px) metric – 5.8 % for rSGM and 4.9 % for AnyNet-M. This metric shows a ratio of pixels, at which a disparity error exceeds 3 pixels over all image pixels. Details on the choice and development of the AnyNet-M method can be found in our paper [14].

The impact of depth estimation accuracy on localization quality is explored in the next sections.

2.3. Odometry estimator

To estimate a displacement (i.e., odometry) between consecutive frames, each of which consists of an image and a depth map, we use ORB-SLAM2 [3] - fast and robust visual SLAM method with open source implementation.

The method first extracts 2D key points from the image and their ORB descriptors. Then associates them to the 3D key points of an environment map and estimates pose of current frame w.r.t. map. This map is built online by solving a bundle adjustment problem, which includes simultaneous optimization of keyframes poses and keypoints 3D coordinates. This map differs from prior 3D LiDAR map. It is more sparse, less accurate and accumulates drift over the time. But such approach to visual odometry estimation outperforms methods which rely only on two last frames.

ORB-SLAM2 also supports loop closure, which reduces drift accumulation of the map. But performing it online lead to pose jumps. Moreover, as will be shown next, the proposed localization approach successfully eliminate such drift.

2.4. Semantic segmentation for dynamic obstacle elimination

In order to increase localization robustness in environments with dynamic objects such as pedestrians and cars, we propose to employ semantic segmentation of input images.

Let $\mathbb{L} = \{1, 1, \ldots, n-1\}$ is a set of semantic labels. Let $m \in \mathbb{L}^{H \times W}$ – segmentation mask, H and W – the height and the width of the input image, $l = m(u, v) \in \mathbb{L}$ – segmentation label of pixel (u, v). Let $\mathbb{D} \subset \mathbb{L}$ – subset of dynamic classes. In our experiments, we consider pedestrians and cars as dynamic.

Segmentation is used for depth image masking as shown in the Fig. 2. We apply next transformation for depth image: $D'(u, v) = D(u, v) \cdot [m(u, v) \notin \mathbb{D}]$, where $[m(u, v) \notin \mathbb{D}]$ – binary mask that contains zeros at pixels, which belong to dynamic objects. In new depth image $D'(u, v)$ zeros pixels are ignored in further operations.

In our experiments we use FCN-ResNet-M-OC [46] provided by the authors as a neural network for semantic segmentation. The model was trained on the Mapillary Vistas [12] dataset and show a segmentation quality at the level of 37.1 % IoU on the Mapillary Vistas (see [46]). The model shows promising results on real-world data in the problem of dynamic obstacle elimination for the occupancy grid generation. FCN-ResNet-M-OC is also optimized for Nvidia Jetson Xavier, that is a target hardware platform in our research. Its inference time is 33 ms on 345×1242 resolution.
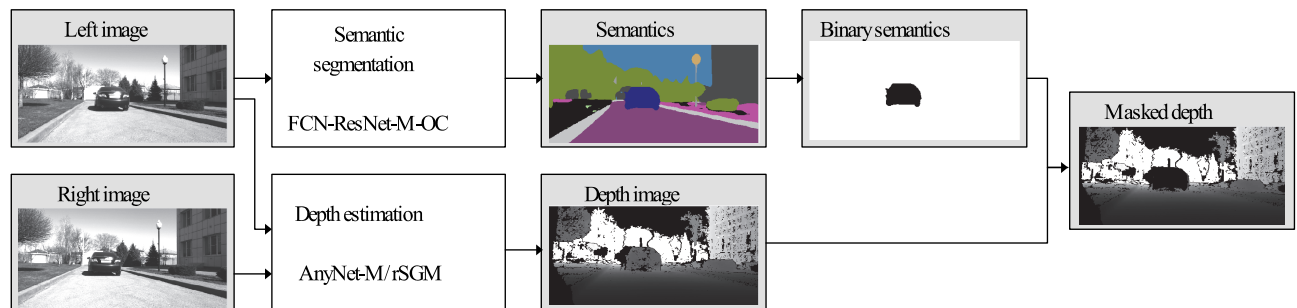

Fig. 2. Depth masking based on semantic segmentation

2.5. Proposed localization module

Detailed scheme of the proposed Localization module is shown in the Figure 3. It represents the main scientific and practical novelty in the proposed approach. Next, we consider in detail its components related to prior map preparation, visible point cloud extraction, the original matching algorithm of a noisy depth map and a visible 3D LiDAR point cloud, features for achieving real-time performance.

2.5.1. Prior map preparation

We follow the approach suggested in [47] when preparing a prior 3D LiDAR map. LiDAR point clouds with estimations of corresponding poses are used as initial guess. They compose the pose graph, which is then optimized with offline graph SLAM methods implemented in [48].

Additionally, in the Prior map preparation module (see Fig. 3), a mesh representation is built from the resulting overall point cloud. This is done in offline mode. We use the Greedy Projection Triangulation method [49] which is suitable for noisy point clouds and implemented in the Point Cloud Library (PCL) [50]. The estimation of the normals needed to build the mesh also done using PCL tools.

2.5.2 Visible point cloud extraction

The 3D points do not have a physical size, so when they are projected onto the image plane, we can see points of actually hidden objects standing in front of the camera. This is especially important during movement

near an obstacle because it can lead to a divergence in the localization process.

Representing the map as a mesh allows us to effectively deal with this problem. The surfaces included in it are used to filter out invisible points. At the same time, there are no strict requirements for the quality of the mesh. The main thing is to preserve the geometry of the space. It should not contain surfaces where there are none, and vice versa. The opaque areas should be covered with a mesh.
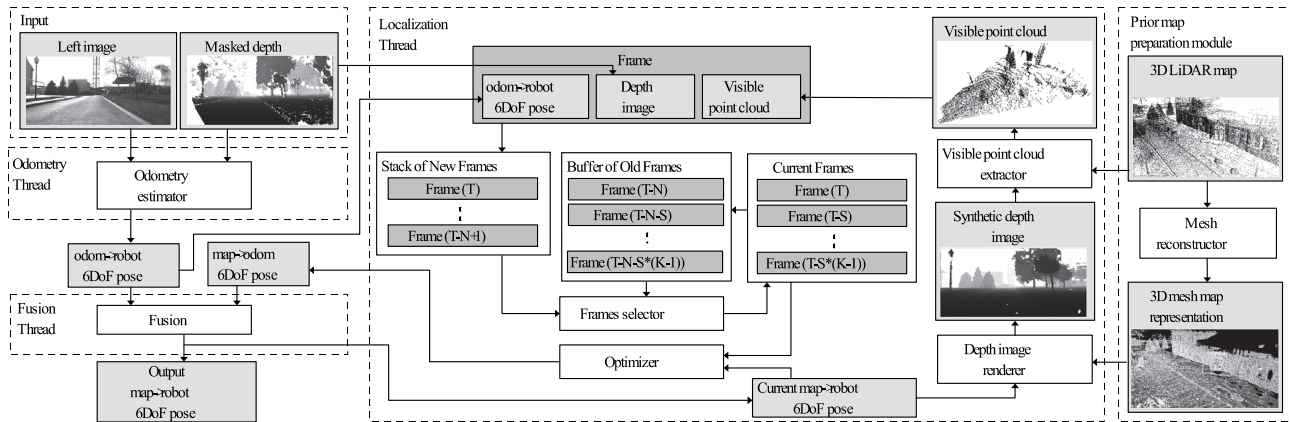

Fig. 3. Structure of the proposed Localization module

The choice of the 3D points visible by the camera from the source cloud is carried out as follows. The position of the robot on the map at the current moment (denote as $map \rightarrow robot$) is obtained as a result of combining the current 6DoF pose estimation ($odom \rightarrow robot$) in the Fusion Thread and evaluating the correction ($map \rightarrow odom$). Then the Synthetic depth image is rendered with a fused pose in the Depth image renderer block. This is done using the OpenGL API, which allows using the hardware video accelerator available on the target Nvidia Jetson Xavier AGX platform. In the next step, the points located in the camera frustum (Fig. 4a) are projected onto the image plane and filtered according to the values of the z coordinate known from the depth buffer. This is how the Visible point cloud V^i (Fig. 4b) at iteration i is formed.

The projection of the Visible point cloud V^i_{proj} aligned with the depth image is shown in the Fig. 5. This projection is done according to the formula:

$$V^i_{proj} = P^i_{bo} P^i_{om} V^i,$$

where P^i_{bo} is known pose of the camera in odom coordinate system, P^i_{om} is the resulting point cloud pose in the same coordinate system. The proposed original algorithm for finding P^i_{om} will be described in the next section.

This projection is added to the Frame along with the depth image and the odometry estimation. This filtering further reduces the number of points used in the optimization process and increases the overall performance of the method.

a)   *b)*

Fig. 4. Visible point cloud extraction: (a) Points in camera frustum; (b) Visible point cloud

2.5.3. Matching of a noisy depth map and a visible 3D LiDAR point cloud

Matching of a depth map and a visible point cloud is formulated as an optimization problem. Because depth maps are noisy, the optimization process may fall at a local minimum. Moreover, the depth map is a raster image meaning objective is not differentiable. Under such conditions gradient-free optimization methods can be applied. Particle filter [51] is widely used in localization approaches for solving such problems. It requires handling and updating a set of particles, which would be time-consuming in our case. Instead we adopt the Nelder-Mead [52] optimization algorithm to our problem. This is

zero-gradient method, which is known to show good results on noisy objective functions. It still may fall in the local minimum, but given a good initial guess from visual odometry converges fast to the solution, which compensate odometry drift. Mathematical models described next are implemented using an Eigen library.
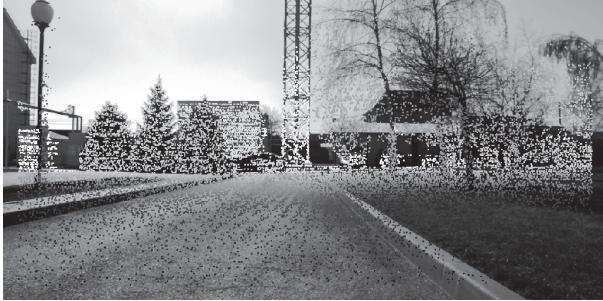

Fig. 5. Visible point cloud projected onto the image

Let $T \in \mathbb{N}$ be the ordinal of the current frame, where $\mathbb{N}$ – is the set of all natural numbers. We propose Algorithm 1 of noisy depth maps and a visible point cloud matching which accepts as input:

- $P_{om}^{T-1} \in SE(3)$ – a previous odometry correction estimation $map \rightarrow odom$,
- a set of K frames with step S between them, containing $P_{ro}^{T-1} \in SE(3)$ – the estimation of $odom \rightarrow robot$ pose by the Odometry estimator,
- the depth map $D^i \in \mathbb{R}^{H \times W}$, H and W – are the height and the width of the input image, $\mathbb{R}$ – is the set of all real numbers,
- the visible point cloud $V^i = \{x_i\}_{j=1}^{n_i}$, $x_j \in \mathbb{R}^3$, where n_i – is the size of point cloud V^i, $i = T, T - S, ..., T - S(K-1)$.

As $\pi : \mathbb{R}^3 \rightarrow \mathbb{R}^2$, we denote the projection operator of the three-dimensional point to the image plane, $D^i(u)$, $u \in \mathbb{R}^2$ is the depth value at point u in the image. It is worth noting that if the point does not lie within the image after projection, it is not considered while computing the objective function value.

As a kernel of the objective function, we use the modified Huber loss function. If an error exceeds ϵ_2, it is clipped to that value. We use $\epsilon_1 = 0.5$, $\epsilon_2 = 1.5$ in all the experiments.

Algoritm 1 Matching of a noisy depth map and a visible 3D LiDAR point cloud.

1. Input: P_{om}^{T-1} – previously estimated map to odom pose, P_{ro}^i – odom to robot pose, $(D')^i$ – masked depth image, V^i – visible point cloud, $i = T, T-S, ..., T-S(K-1)$.
2. Set $h(x)$ – modified Huber loss function with clipping, $x \in \mathbb{R}$

$$h(x) = \begin{cases} x^2, if \ |x| < \epsilon_1, \\ 2\epsilon_1 |x| - \epsilon_1^2, if \ |x| < \epsilon_2, \\ 2\epsilon_1\epsilon_2 - \epsilon_1^2, otherwise. \end{cases}$$

3. $P_{bo} = P_{ro}^{T-SK/2}$ – pose of the middle frame w.r.t. odom.
4. $P_{bm}^{T-1} = P_{bo}P_{om}^{T-1}$ – pose of the middle frame w.r.t. map.
5. $P_{rb}^i = P_{ro}^i P_{bo}^{-1}, i = T, T - S, ..., T - S(K-1)$ – poses of frames w.r.t. middle frame.
6. Set $e^i(P, x)$ – error function, $P \in SE(3)$, $x \in \mathbb{R}^3$

$$e^i(P, x) = [P_{rb}^i Px]_z - (D')^i(\pi P_{rb}^i Px).$$

7. Set $H(P)$ – total loss, $P \in SE(3)$, M – number of points, which lie within image after projection

$$H(P) = \frac{1}{M} \sum_{j=0}^{K-1} \sum_{x \in V^{T-Sj}} h(e^j(P, x)).$$

8. Set $L(\xi)$ – objective function, $\xi \in se(3)$

$$L(\xi) = H(e^\xi).$$

9. $\xi^{T-1} = log(P_{bm}^{T-1})$.
10. $\xi^T = $ optimize $L(\xi)$ with Nelder-Mead and initial guess ξ^{T-1}.
11. $\delta = \xi^T - \xi^{T-1}$.
12. if $\|\delta\| > \beta$ then $\eta = \xi^{T-1} + \alpha\beta(\delta / \|\delta\|)$ else $\eta = \xi^T$.
13. $P_{bm}^T = e^\eta$; $P_{om}^T = P_{bo}^{-1} P_{bm}^T$.
14. Output: P_{om}^T – current estimated map to odom pose.

Optimization is performed on the $se(3)$ manifold. The manifold elements are denoted as ξ and η. The mappings between $SE(3)$ and $se(3)$ are referred to as $log: SE(3) \rightarrow se(3)$ and $e^\xi: se(3) \rightarrow SE(3)$. Since $se(3)$ space does not have a norm, it is impossible to define a simplex needed for the Nelder-Mead algorithm. Instead we define $\Delta = \{\xi + \delta_1 e_i, 1 \le i \le 3\} \cup \{\xi + \delta_2 e_i, 4 \le i \le 6\}$, where $\delta^1 = 0.2$, $\delta^2 = 0.4$, which replaces a simplex in Nelder-Mead optimization process.

To improve the method stability in poor conditions, we apply optimization step clipping. It is regulated by parameters α and β. β indicates a maximum optimization step, which is considered as valid. $0 < \alpha \le 1$ regulates the actual optimization step.

Another option for matching a depth map and a point cloud, is to convert depth map to another point cloud and apply approaches like Normal Distribution Transform (NDT) or Iterative Closest Point (ICP) similar to [7, 19]. However, formulating an objective function in the 2D space is more native since account for the nature of observations – 2D images. And, as will be shown next, leads to better accuracy.

2.5.4. Localization with real-time performance

Visual odometry P_{ro}^T is quite reliable on small distances if tracking does not fail. In such conditions, there is no need to relocalize after each frame. Instead, the Localization Thread (see Figure 3) estimates the odometry correction w.r.t. map $map \rightarrow odom$ P_{om}^T with a lower frequency. This correction is fused with odometry

in the Fusion Thread using equation, which gives robot pose at last iteration T:

$$P_{rm}^T = P_{ro}^T P_{om}^T.$$

This approach produces a robot pose with frequency and delay equal to odometry frequency and delay. Simultaneously, there is no error accumulation, which is characteristic of the odometry and SLAM approaches.

To tune the frequency and the quality of the $map \rightarrow odom$ correction estimation, we introduce parameters S and K. The K value controls the number of frames that will be used for localization. Increasing this parameter reduces the frequency of producing the correction $map \rightarrow odom$, but at the same time, increases its quality and stability. The S value controls the step between the frames. In dynamic environments and fast movements, it should be decreased. Otherwise, it can be increased.

3. Experimental results

The proposed approach was evaluated on two datasets: our own dataset collected from a mobile ground robot Clearpath Husky (Husky dataset) and the sequences 00-10 from the KITTI Odometry dataset (KITTI), which is widely used as a benchmark for visual localization of a car in urban environment. Demo video is available at the link: https://youtu.be/M3FqPPb9njQ .

3.1. Datasets

The KITTI Dataset [53] consists of sequences containing stereo images of resolution 1240×376 captured at 10 Hz, point clouds, obtained from LiDAR Velodyne HDL-64 with the same frequency, and ground truth poses. The sequences are recorded in an urban environment and were used for evaluation. We reveal that the LiDAR maps for sequences built by aggregating point clouds based on ground truth poses contain local artifacts. To reduce them, each map was optimized by the Offline Graph SLAM [48] approach. Then, maps were subsampled to resolution 0.2 m. The final point clouds were used for a mesh reconstruction.

The proposed approach was also evaluated on our dataset, collected from the ground robot ClearPath Husky equipped with a stereo camera with a baseline ~40 cm. and a resolution of 1200×600, LiDAR Velodyne HDL-32, a high-quality GNSS/INS+RTK system, used as ground truth in our experiments. The dataset contains 14 tracks, two of which were used for the map reconstruction, others were used for the evaluation. The data were collected on different days and contain dynamic objects: pedestrians and cars.

3.2. Localization results

Tab. 1 contains the average metrics of the localization quality on all 12 tracks of the Husky dataset. It includes ablation study for the proposed approach, results for ORB-SLAM2 [3], two open-source localization approaches [9, 54], which uses a prior map and a stereo camera, and results for LiDAR-only localization method LOL [55] based on LeGO-LOAM [6]. The table shows that the proposed approach outperforms its peers.

Tab. 1. Average metrics of the localization quality on the Husky dataset (00-12 tracks)

Method	Trans. error [m]	Rot. error [deg]
Localization module (our)	0.166	0.553
+odometry	0.132	0.539
+odometry+multifrаme	0.125	0.542
+odometry+multifrаme+mask	**0.115**	**0.419**
ORB-SLAM2 [3]	0.292	0.636
GMMLoc [9]	0.289	0.715
Iris [54, 56] (stereo)	0.546	2.520
Iris [54, 56] (RGBD)	0.387	2.775
LOL [55]	0.138	0.644

The ORB-SLAM2 [3] is used in our work as an Odometry estimator. The number of keypoints is set to 100. We found such setup for ORB-SLAM2 providing a good trade-off between performance and quality. This method is not a localization method and its results are included only for reference.

The GMMLoc [9] method requires the map to be represented as a Gaussian Mixture Model. We built such a map from our LiDAR map by applying normal distribution transform. According to the original paper, on high-quality map build by expensive stationary laser scanner, the method can provide centimeter-level accuracy. However, because the map was obtained by a LiDAR on a moving mobile platform and not a stationary laser scanner, the method shows lower quality than expected.

The MapIV/iris [54, 56] method uses a point cloud as a map and was tested on the same map as our approach without any additional processing. The original implementation uses stereo images for localization. We adapted it to work with our depth images obtained by rSGM [39]. The results of the modified version are also presented in the table.

In addition, the results for LOL [55] are also included in the table. This method is LiDAR-only localization method. It means, that both the prior map of the environment and the observations in run-time are obtained by the same sensor – LiDAR. Such approach is widely used, can provide high quality, but more expensive than localization in LiDAR map using a stereo camera in operation time.

The first line of the Table 1 contains the results only of the Localization module with disabled Odometry estimator. In this setup only the latest image and depth are used. Next, we enable Odometry estimator, which improve the metrics up to LiDAR-only method. This effect show, that good initial guess for the Localization module is important. Incorporating multiple frames

further improve the results. We set $S=2$, $K=5$ in this experiment. In the last experiment we research the effect of dynamic object elimination by masking the depth. In a full setup the proposed method provide better localization quality than LiDAR-only method in terms of both translation and rotation errors.

Fig. 6 shows the ground truth trajectory, ORB-SLAM2 trajectory, and the proposed localization method trajectory on the 06 track of the Husky dataset. Our approach eliminates the accumulation of odometry error providing unbiased estimation of the robot pose on the map.
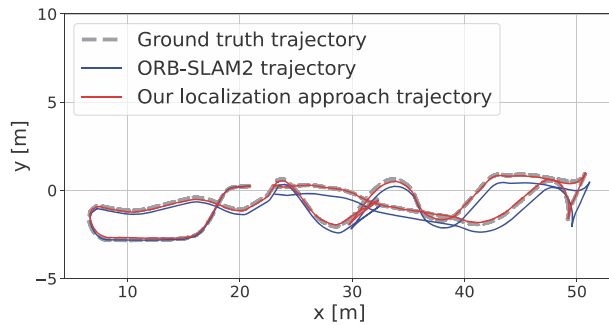

Fig. 6. Result trajectories on Husky dataset (06 track)

Table 2 contains localization metrics on the 06 track of the Husky dataset. The first row corresponds to the results obtained with depth images estimated by rSGM. The second line – to the results obtained with depth images from neural network AnyNet-M. It can be seen, that localization quality is much poorer for AnyNet-M. We argue that without training on the target dataset the deep learning approach will not provide reliable depth information. We leave this problem for further research and do not perform evaluation of localization with depth estimated by AnyNet-M on the other tracks.

Tab. 2. Metrics of the localization quality on the 06 track of the Husky dataset

Method	Trans. error [m]	Rot. [deg]
Localization module with rSGM +odometry+multiframe	**0.102**	**0.387**
Localization module with AnyNet-M +odometry+multiframe	0.826	2.369

Table 3 shows the mean translation and rotation errors with their standard deviations on the 00 KITTI sequence. The data used for the map reconstruction and for the evaluation are taken from the same sequence. It is a limitation of this dataset because it contains only one sequence for each area. Because of this, dynamic objects are the same in the map and in observations and its elimination does not lead to any improvement. Moreover, such scenario is not practical. For this reason, we do not include the results with masked depth in the table. The first line contains the results of the proposed localization method in multiframe mode ($S=1$, $K=3$) and enabled

odometry estimator. The second one – the results of ORB-SLAM2 [3] with the same parameters as on Husky dataset. Other rows correspond to existing localization methods based on the image analysis. Methods [7, 8, 31, 36] do not have public code and were not evaluated on Husky dataset. For comparison with these approaches we use the original results provided by the authors except the MapIV/iris [54, 56]. Metrics for the MapIV/iris are obtained by ourselves. GMMLoc were not evaluated on the KITTI by authors and we were unable to run it on the KITTI dataset for comparison since its computational complexity is quadratic of the map size. The Tab. 3 shows that the proposed method demonstrates an error at the level of the current state-of-the-art solutions, but has the least variance in both translation and rotation.

Tab. 3. Metrics of the localization quality on 00 KITTI sequence (avg ±std)

Method	Trans. error [m]	Rot. error [deg]
Localization module (our) +odometry+multiframe	**0.13** ±**0.08**	0.62 ±**0.27**
ORB-SLAM2 [3]	9.11 ±4.49	2.75 ±1.25
Kim [8]	**0.13** ±0.11	**0.32** ±0.39
Caselitz [7]	0.30 ±0.11	1.65 ±0.91
CMRNet++[36]	0.21 ±0.30	0.43 ±0.42
Lu [31]	15.92 ±8.04	-
Iris [54, 56] (stereo)	0.74 ±0.48	3.12 ±1.20
Iris [54, 56] (RGBD)	0.43 ±0.31	2.86 ±0.97
Zuo [35]	0.47	0.87
HyperMap [38]	0.48	1.42

We also tested the proposed method on several other sequences from the KITTI dataset. We chose sequences $04-07$, which are close to our target scenes where the autonomous robot should function. Table 4 shows the results of these experiments. As one can see, the proposed method shows comparable results, but it is not very stable and the quality of localization changes from sequence to sequence. This is due to some noise in the data, which affects the quality of visual odometry and sometimes creates flashes in pose estimation that our method cannot handle. This especially affects rotation estimation. The Iris [54] and Lu [31] methods did not provide results for some sequences, as shown by dashes in Table 4. Note that methods Kim[8] and Zuo [35] were tested by the authors on powerful desktop computers and, unlike our method, are not capable of working in real time on a less powerful but energy efficient platform.

Fig. 7 shows the ground truth trajectory of 00 KITTI sequence, ORB-SLAM2 trajectory, and the proposed localization method trajectory. Figure 8 shows the plots of the translation error and the rotation error versus frame number. Low variance means higher stability of the estimated trajectory when compared to the other approaches. Localization stability is of crucial importance for the algorithms running on the robot, which use the localization results

for motion planning and trajectory following. The largest error of 0.6 meters is achieved at the end of the track when the car moves in the area free from close objects. This behavior is a methodological limitation of depth-image based approaches in structure-less environments.

It should be noted, that localization metrics depends not only on the localization approach, but on the map itself. The KITTI dataset does not contain a prebuilt map. We, as well as the authors of the other approaches, reconstruct the environment map from LiDAR sweeps. These maps are slightly different and the comparison is not truly objective. To resolve this problem for the future research we make our 3D LiDAR map publicly available at the link: https://github.com/mr-abramenko/stereo-localization-in-lidar-map.
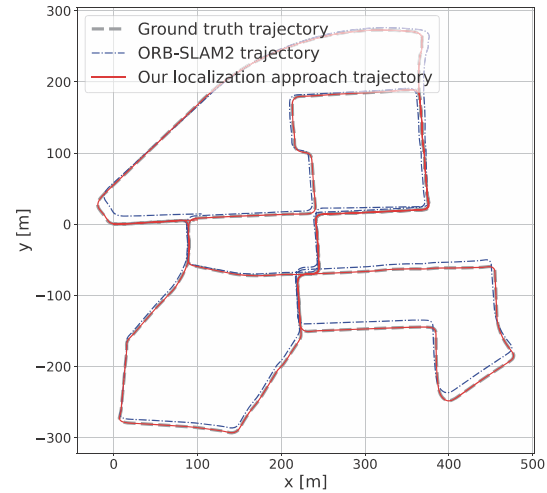

Fig. 7. Result trajectories on KITTI (00 sequence)

Tab. 4. Metrics of the localization quality on KITTI dataset (translation/rotation)

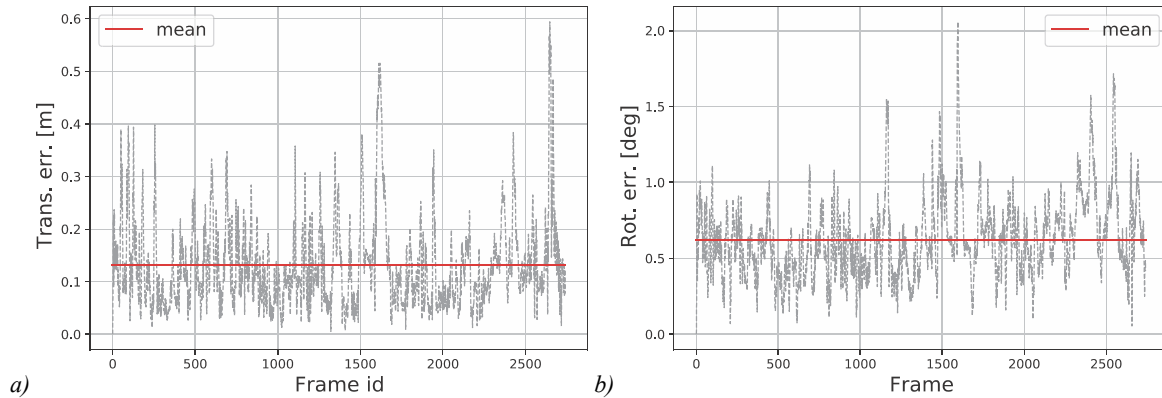| Sequence	Our	Kim [8]	Zuo [35]	Iris [54]	Lu [31]
00	0.13/0.62	0.13/0.32	0.47/0.87	2.6/-	15.92/-
04	0.22/1.47	0.45/0.88	0.23/0.72	-/-	-/-
05	0.64/2.05	0.15/0.34	0.26/0.45	-/-	-/-
06	0.16/1.25	0.38/0.85	0.33/0.50	-/-	-/-
07	0.57/1.39	0.13/0.49	0.16/0.46	-/-	4.55/-

Fig. 8. Localization errors on KITTI-00: (a) Translation error [m]; (b) Rotation error [deg]

3.3. Performance

The proposed approach was tested on two hardware setups: 1) Desktop: AMD Ryzen Threadripper 1900X, Nvidia GeForce RTX 2070; 2) Xavier: Nvidia Jetson Xavier AGX. The power consumption of Nvidia Jetson Xavier AGX is 30W, which makes it attractive for use in mobile robots and unmanned vehicles. It has an 8-core processor and a video accelerator with 512 CUDA cores with OpenGL support. The latter allows one to quickly render a depth map based on a mesh.

The performance test results are shown in the Tab. 5. The measurements were taken on the Husky dataset. The latency of the method is the same as the latency of the Odometry estimator, meaning that localization results are provided at the rate of 14 FPS for our setup on Xavier. But, as apposed to a single Odometry Estimator, does not accumulate an error. The average processing time of one frame is taken as the operating time of the Depth image renderer, Visible point cloud extractor, and Optimizer blocks. Thus, in total 164 ms on the Nvidia Jetson Xavier AGX. This speed allows the $map \rightarrow odom$ odometry correction to be updated using one frame at six FPS, using two frames at three FPS, etc. If the error accumulation of the Odometry Estimator is small (which is true for the visual odometry), it is enough to correct its drift once a second or at lower rate.

Tab. 5. Localization performance (in milliseconds per frame), measured on the Husky dataset

Platform	Odometry estimator	Depth image renderer	Visible point cloud extractor	Optimizer
Xavier	69.59	15.36	17.02	131.60
Desktop	49.61	6.05	11.72	64.85

Conclusion and future work

The proposed approach for mobile robot localization shows a stable absolute translation error of about $0.11 - 0.13$ meters and rotation error of $0.42 - 0.62$ degrees both on KITTI and the custom dataset from ClearPath Husky. It was revealed that the standard deviation of the obtained absolute metrics is the smallest among other state-of-the-art approaches. This was achieved through the use of a sequence of multiple data frames during the optimization step and dynamic obstacles elimination on depth image. The use of a neural network of semantic segmentation of an RGB image made it possible to successfully mask dynamic objects (cars, pedestrians) and, as a result, to reduce the localization error caused by them. In addition, due to the use of the 3D LiDAR map, the method allows us to almost completely eliminate the drift of inaccurate visual odometry. In contrast to most of other approaches, the proposed approach software implementation demonstrates good performance on different hardware platforms, included energy efficient Jetson, which is appropriate for real-time applications. It is achieved mainly for parallelization on CPU and GPU. Thus, the proposed approach opens up great opportunities for using light and lowcost cameras for high-precision localization without the LiDAR or GNSS usage for a long time.

Acknowledgment

References

[1] Myasnikov VV, Dmitriev EA. The accuracy dependency investigation of simultaneous localization and mapping on the errors from mobile device sensors. Computer Optics 2019; 43(3): 492-503. DOI: 10.18287/2412-6179-2019-43-3-492-503.

[2] Goshin YV, Kotov AP. Method for camera motion parameter estimation from a small number of corresponding points using quaternions. Computer Optics 2020; 44(3): 446-453. DOI: 10.18287/2412-6179-CO-683.

[3] Mur-Artal R, Tardós JD. ORB-SLAM2: An open-source SLAM system for monocular, stereo, and RGB-D cameras. IEEE Trans Robot 2017; 33(5): 1255-1262.

[4] Labbé M, Michaud F. RTAB-Map as an open-source lidar and visual simultaneous localization and mapping library for large-scale and longterm online operation. J Field Robot 2019; 36(2): 416-446.

[5] Zhang J, Singh S. Laser–visual–inertial odometry and mapping with high robustness and low drift. J Field Robot 2018; 35(8): 1242-1264.

[6] Shan T, Englot B. Lego-loam: Lightweight and ground-optimized lidar odometry and mapping on variable terrain. in: 2018 IEEE/RSJ Int Conf on Intelligent Robots and Systems (IROS) 2018: 4758-4765.

[7] Caselitz T, Steder B, Ruhnke M, Burgard W. Monocular camera localization in 3D LiDAR maps. 2016 IEEE/RSJ Int Conf on Intelligent Robots and Systems (IROS) 2016: 1926-1931.

[8] Kim Y, Jeong J, Kim A. stereo camera localization in 3D LiDAR maps. 2018 IEEE/RSJ Int Conf on Intelligent Robots and Systems (IROS) 2018: 1-9.

[9] Huang H, Ye H, Sun Y, Liu M. GMMLoc: Structure consistent visual localization with gaussian mixture models. IEEE Robot Autom Lett 2020; 5: 5043-5050.

[10] Ding X, Wang Y, Li D, Tang L, Yin H, Xiong R. Laser map aided visual inertial localization in changing environment. 2018 IEEE/RSJ Int Conf on Intelligent Robots and Systems (IROS) 2018: 4794-4801.

[11] Zuo X, Geneva P, Yang Y, Ye W, Liu Y, Huang G. Visual-inertial localization with prior LiDAR map constraints. IEEE Robot Autom Lett 2019; 4: 3394-3401.

[12] Neuhold G, Ollmann T, Rota Bulo S, Kontschieder P. The mapillary vistas dataset for semantic understanding of street scenes. Proc IEEE Int Conf on Computer Vision 2017: 4990-4999.

[13] Wang Y, Lai Z, Huang G, Wang BH, Van Der Maaten L, Campbell M, Weinberger KQ. Anytime stereo image depth estimation on mobile devices. 2019 Int Conf on Robotics and Automation (ICRA) 2019: 5893-5900.

[14] Kasatkin N, Yudin D. Real-time approach to neural network-based disparity map generation from stereo images. In Book: Kryzhanovsky B, Dunin-Barkowski W, Redko V, Tiumentsev Y, Klimov VV, eds. Advances in neural computation, machine learning, and cognitive research V. Cham: Springer Nature Switzerland AG; 2022.

[15] Chen Y, Wang G. EnforceNet: Monocular camera localization in large scale indoor sparse LiDAR point cloud. arXiv Preprint. 2019. Source: <https://arxiv.org/abs/1907.07160>.

[16] Pauls JH, Petek K, Poggenhans F, Stiller C. Monocular localization in HD maps by combining semantic segmentation and distance transform. 2020 IEEE/RSJ Int Conf on Intelligent Robots and Systems (IROS) 2020: 4595-4601.

[17] Magnusson M, Lilienthal A, Duckett T. Scan registration for autonomous mining vehicles using 3D-NDT. J Field Robot 2007; 24(10): 803-827.

[18] Han D, Zou Z, Wang L, Xu C. A robust stereo camera localization method with prior LiDAR map constrains. 2019 IEEE Int Conf on Robotics and Biomimetics (ROBIO) 2019: 2001-2006.

[19] Sun M, Yang S, Liu H. Scale-aware camera localization in 3D LiDAR maps with a monocular visual odometry. Comput Animat Virtual Worlds 2019; 30(3-4): e1879.

[20] Yu H, Zhen W, Yang W, Scherer S. Line-based camera pose estimation in point cloud of structured environments. arXiv Preprint. 2019. Source: <https://arxiv.org/abs/1912.05013>.

[21] Yu H, Zhen W, Yang W, Zhang J, Scherer S. Monocular camera localization in prior LiDAR maps with 2D-3D line correspondences. 2020 IEEE/RSJ Int Conf on Intelligent Robots and Systems (IROS) 2020: 4588-4594.

[22] Lu Y, Huang J, Chen Y, Heisele B. Monocular localization in urban environments using road markings. 2017 IEEE Intelligent Vehicles Symposium (IV) 2017: 468-474.

[23] Jeong J, Cho Y, Kim A. HDMI-Loc: Exploiting high definition map image for precise localization via bitwise particle filter. IEEE Robot Autom Lett 2020; 5: 6310-6317.

[24] Qiu K, Liu T, Shen S. Model-based global localization for aerial robots using edge alignment. IEEE Robot Autom Lett 2017; 2: 1256-1263.

[25] Wong D, Kawanishi Y, Deguchi D, Ide I, Murase H. Monocular localization within sparse voxel maps. 2017 IEEE Intelligent Vehicles Symposium (IV) 2017: 499-504.

[26] Pascoe G, Maddern WP, Stewart AD, Newman P. FARLAP: Fast robust localisation using appearance priors. 2015 IEEE Int Conf on Robotics and Automation (ICRA) 2015: 6366-6373.

[27] Pascoe G, Maddern WP, Newman P. Robust direct visual localisation using normalised information distance. British Machine Vision Conf 2015: 1-13.

[28] Oishi S, Kawamata Y, Yokozuka M, Koide K, Banno A, Miura J. C*: Cross-modal simultaneous tracking and rendering for 6-DoF monocular camera localization beyond modalities. IEEE Robot Autom Lett 2020; 5: 5229-5236.

[29] Wolcott RW, Eustice R. Visual localization within LIDAR maps for automated urban driving. 2014 IEEE/RSJ Int Conf on Intelligent Robots and Systems 2014: 176-183.

[30] Neubert P, Schubert S, Protzel P. Sampling-based methods for visual navigation in 3D maps by synthesizing depth images. 2017 IEEE/RSJ Int Conf on Intelligent Robots and Systems (IROS) 2017: 2492-2498.

[31] Lu Y, Lee J, Yeh SH, Cheng HM, Chen B, Song D. Sharing heterogeneous spatial knowledge: Map fusion between asynchronous monocular vision and lidar or other prior inputs. In Book: Amato NM, Hager G, Thomas S, Torres-Torriti M, eds. Robotics research. Cham: Springer Nature Switzerland AG; 2020: 727-741.

[32] Bao H, Xie W, Qian Q, Chen D, Zhai S, Wang N, Zhang G. Robust tightly-coupled visual-inertial odometry with pre-built maps in high latency situations. IEEE Trans Vis Comput Graph 2022; 28(05): 2212-2222.

[33] Ye H, Huang H, Liu M. Monocular direct sparse localization in a prior 3D surfel map. 2020 IEEE Int Conf on Robotics and Automation (ICRA) 2020: 8892-8898.

[34] Huang H, Sun Y, Ye H, Liu M. Metric monocular localization using signed distance fields. 2019 IEEE/RSJ Int Conf on Intelligent Robots and Systems (IROS) 2019: 1195-1201.

[35] Zuo X, Ye W, Yang Y, Zheng R, Vidal-Calleja T, Huang G, Liu Y. Multimodal localization: Stereo over LiDAR map. J Field Robot 2020; 37(6): 1003-1026.

[36] Cattaneo D, Sorrenti DG, Valada A. CMRNet++: Map and camera agnostic monocular visual localization in LiDAR maps. arXiv Preprint. 2020. Source: <https://arxiv.org/abs/2004.13795>.

[37] Tang S, Tang C, Huang R, Zhu S, Tan P. Learning camera localization via dense scene matching. Proc IEEE/CVF Conf on Computer Vision and Pattern Recognition 2021: 1831-1841.

[38] Chang MF, Mangelson J, Kaess M, Lucey S. HyperMap: Compressed 3D map for monocular camera registration. 2021 IEEE Int Conf on Robotics and Automation (ICRA) 2021: 11739-11745.

[39] Spangenberg R, Langner T, Adfeldt S, Rojas R. Large scale semi-global matching on the cpu. 2014 IEEE Intelligent Vehicles Symposium Proc 2014: 195-201.

[40] Hirschmüller H. Accurate and efficient stereo processing by semi-global matching and mutual information. 2005 IEEE Computer Society Conf on Computer Vision and Pattern Recognition (CVPR'05) 2005; 2: 807-814.

[41] Hirschmüller H. Stereo processing by semiglobal matching and mutual information. IEEE Trans Pattern Anal Mach Intell 2007; 30(2): 328-341.

[42] Hirschmüller H. Semi-global matching-motivation, developments and applications. Photogrammetric Week 2011; 11: 173-184.

[43] Hirschmüller H, Buder M, Ernst I. Memory efficient semi-global matching. ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences 2012; 3: 371-376.

[44] Menze M, Geiger A. Object scene flow for autonomous vehicles. Conf on Computer Vision and Pattern Recognition (CVPR) 2015: 1-10.

[45] Geiger A, Lenz P, Urtasun R. Are we ready for autonomous driving? The KITTI vision benchmark suite. Conf on Computer Vision and Pattern Recognition (CVPR) 2012: 3354-3361.

[46] Shepel I, Adeshkin V, Belkin I, Yudin DA. Occupancy grid generation with dynamic obstacle segmentation in stereo images. IEEE Trans Intell Transp Syst 2021; 23(9): 14779-14789.

[47] Belkin I, Abramenko A, Yudin D. Real-time lidar-based localization of mobile ground robot. Procedia Computer Sci 2021; 186: 440-448.

[48] Koide K. Interactive slam: open source 3D LIDAR-based mapping framework. 2019. Source: <https://github.com/SMRT-AIST/interactive_slam>.

[49] Marton ZC, Rusu RB, Beetz M. On fast surface reconstruction methods for large and noisy datasets. Proc IEEE Int Conf on Robotics and Automation (ICRA) 2009: 3218-3223.

[50] Rusu RB, Cousins S. 3D is here: Point cloud library (pcl). 2011 IEEE Int Conf on Robotics and Automation 2011: 1-4.

[51] Del Moral P. Nonlinear filtering: Interacting particle resolution. Comptes Rendus de l'Académie des Sciences – Series I – Mathematics 1997; 325(6): 653-658.

[52] Nelder J, Mead R. A simplex method for function minimization. Comput J 1965; 7: 308-313.

[53] Geiger A, Lenz P, Stiller C, Urtasun R. Vision meets robotics: The KITTI dataset. Int J Rob Res 2013; 32: 1231-1237.

[54] Yabuuchi K, Wong DR, Ishita T, Kitsukawa Y, Kato S. Visual localization for autonomous driving using pre-built point cloud maps. 2021 IEEE Intelligent Vehicles Symposium (IV) 2021: 913-919.

[55] Laser odometry and localization. 2019. Source: <https://github.com/jyakaranda/LOL>.

[56] MapIV. Iris: Visual localization in pre-build pointcloud maps. 2020. Source: <https://github.com/MapIV/iris>.

Authors' information

Ilya V. Belkin received M.S. degree in Computer Science from Moscow Institute of Physics and Technology (MIPT), Moscow, Russia in 2021. Currently, he is doing a Ph.D. degree in "Mathematical modeling, numerical methods and software packages" at the MIPT, Moscow, Russia. From June to September 2018, he was a Data Science Intern in Huawei Russian Research Center. From October 2018 to July 2019, he was a Computer Vision Intern in ABBYY R&D Russia. From June 2019 to the present, he is a researcher and developer in Intelligent Transport Laboratory at MIPT,

Moscow, Russia. From October 2020 he is a developer in LLC Integrant, Moscow, Russia. His research interests include simultaneous localization and mapping, computer vision, deep learning, robotics.

Alexander A. Abramenko was born in 1992. He received specialist degree in Applied Mathematics and Computer Science in 2015 and the Ph.D. degree in Computer Science from the Southern Federal University, Taganrog, Russia in 2020. From 2017 to the present, he is a researcher in Scientific Design Bureau of Computing Systems, Taganrog, Russia. Since 2021 he is a research engineer in LLC Integrant, Moscow, Russia. His research interests include simultaneous localization and mapping, data analysis, computer vision, robotics. He is the author of 20 scientific papers. E-mail: *mr.abramenko@gmail.com*

Vitaly D. Bezuglyi was born in Kyiv, Ukraine in 2000. He received the bachelor degree in Applied Mathematics and Computer Science in 2022 from the National Research Nuclear University 'MEPHi', Moscow, Russia. Since 2021 he is a research engineer in Intelligent Transport Laboratory, Moscow Institute of Physics and Technology, Moscow Region, Dolgoprudny, 141700, Russia. His research interests include 3D mapping, semantic mapping, HD maps, computer vision, and robotics. He is the author of 1 scientific paper. E-mail: *bezuglyi.vd@mipt.ru*

Dmitry A. Yudin was born in Belgorod, Russia in 1988. He received the engineer diploma in Automation of Technological Processes and Production in 2010 and the Ph.D. degree in Computer Science from the Belgorod State Technological University (BSTU) n.a. V.G. Shukhov, Belgorod, Russia in 2014. From 2009 to 2019, he was a researcher and assistant professor with Technical Cybernetics Department in BSTU named after V.G. Shukhov. From 2019 to the present, he is a head of the Intelligent Transport Laboratory in Moscow Institute of Physics and Technology, Moscow, Russia. Since 2021, he has been a senior researcher in Artificial Intelligence Research Institute (AIRI), Moscow, Russia. He is the author of three books, more than 100 articles. His research interests include computer vision, deep learning and robotics. E-mail: *yudin.da@mipt.ru*