Journal of Intelligent & Robotic Systems (2021) 103: 28
https://doi.org/10.1007/s10846-021-01476-1

SHORT PAPER

3D Dense Mapping with the Graph of Keyframe-Based and View-Dependent Local Maps

Krzysztof Zieliński[1] · Rafał Staszak[1] · Mikołaj Nowaczyk[1] · Dominik Belter[1]

Received: 21 October 2020 / Accepted: 4 August 2021 / Published online: 11 September 2021

Abstract

This article concerns the problem of a dense mapping system for a robot exploring a new environment. In this scenario, a robot equipped with an RGB-D camera uses RGB and range data to build a consistent model of the environment. Firstly, dense mapping requires the selection of the data representation. Secondly, the dense mapping system has to deal with localization drift which can be corrected when loop closure is detected. In this article, we deal with both of these problems, and we make several technical contributions. We define local maps which use the Normal Distribution Transform (NDT) stored in the 2D structures to represent the local scene with varying 3D resolution. This method directly utilizes the uncertainty model of the range sensor and provides information about the accuracy of the data in the map. We also propose an architecture that utilizes pose and covisibility graphs to correct a global model of the environment after loop closure detection. We show how to integrate the dense local mapping with the pose graph and keyframes management system in the ORB-SLAM2 localization. Finally, we show the advantages of the view-dependent model over the methods that uniformly divide the space to represent objects in the environment.

Keywords Mapping · Reconstruction · Graph-based dense mapping

1 Introduction

Mobile robots are becoming increasingly popular in factories, warehouses, houses, and even hospitals. They support activities, such as autonomous transport, floor cleaning, or facility surveillance. When the robot is equipped with a robotic arm, the number of possible applications significantly increases. The robot can support the production process, object pick-and-place, load and unload machines with parts or materials, palletize, or assemble objects. In domestic applications, mobile manipulating robots are used to bring objects on demand. In all those applications robots should localize themselves and build a dense model of the environment to avoid collisions with obstacles and navigate safely in the previously unknown environment.

This work is motivated by the imperfections of the existing dense mapping techniques. Simultaneous Localization and Mapping Systems match the current view to the global model (map). When the robot explores *a priori* unknown environment and localizes itself, the localization error accumulates and the estimated pose differs significantly from the real pose of the robot (Fig. 1a). The dense mapping sensors like RGB-D cameras or LiDAR provide depth measurements (point cloud) which are later used to update the dense model of the environment (Fig. 1c). Because storing the raw measurements (point clouds) is not memory efficient and computationally expensive during motion planning, the measurements are used to update the environment representation (voxels, cells, mesh, or surfels). Incorrect pose estimate causes incorrect update of the global map. However, modern localization systems enable robot re-localization when part of the environment is re-observed. Then, the localization drift is reduced and the whole trajectory of the robot is corrected (Fig. 1b). Unfortunately, the global dense model cannot be corrected because all measurements are already integrated in the map. In the proposed approach the global map is closely coupled with the graph-based localization system thus it allows for the correction of the global map whenever the loop closure is detected (Fig. 1d) and the drift of the trajectory is reduced.

The second problem that we tackle in this research is the representation of the map. The dense model of

✉ Dominik Belter
dominik.belter@put.poznan.pl

[1] Institute of Robotics and Machine Intelligence, Poznan University of Technology, Poznan, Poland

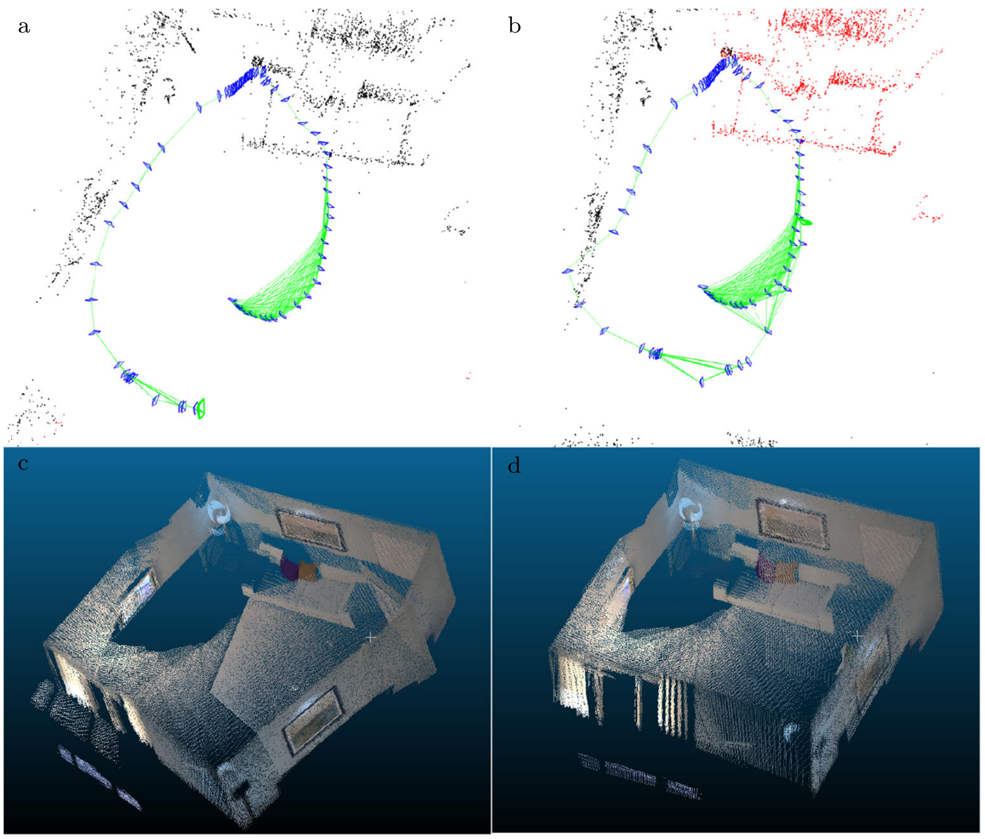


Fig. 1 Typical scenario and the application of the proposed method: a robot explores a new environment, localizes itself (**a**), and builds a dense model using depth measurements. Whenever the robot re-observes the previously seen part of the environment, the localization system closes the loop and corrects the trajectory (**b**). In contrast to global maps like OctoMap or NDT-OM (**c**), the proposed model of the environment is corrected after loop closure detection (**d**) and can be later used for collision avoidance and task planning

the environment might be represented by the elevation map [24], OctoMap [19], Normal Distribution Transform Occupancy Map (NDT-OM) [36], mesh model [37] or sufels [39, 46]. In contrast to sparse maps used for robot localization, which contain features useful for matching the current view to the map [30], the dense map should contain precise information about the shape of the objects in the environment and allow collision avoidance and finding the collision-free motion of the robot. Most of the mapping methods uniformly divide 3D space. As a result, all objects are represented in the map with the same accuracy. Nevertheless, the perception system of the robot has defined properties and in most cases, the objects which are closer to the sensor are measured with higher accuracy than distant objects. Thus, when we integrate measurements in a global map based on voxels, we lose information about the shape of close objects. The model of the environment should also preserve information about the accuracy of the measurements, which is often solved by storing an uncertainty map that corresponds to the geometrical map [11] and utilizes the most accurate measurements.

To deal with this problem, we propose a graph-based structure of the local maps. Local maps in the graph are connected by the $SE3$ transformation which is directly connected to the graph-based structure of the keyframes in the localization system (e.g. ORB-SLAM2). Thus, each local map (node in the graph) is related to the corresponding keyframe created by the localization system. The poses of local maps are modified whenever the localization system improves the estimation of the camera trajectory. This approach allows us to integrate measurements from neighboring frames into a local dense map and later, align local maps to improve the quality of the global map. Moreover, we build local maps in the keyframe pose and integrate measurements in the image space. As a result, we obtain the 3D model which directly utilizes the uncertainty model of the sensor used on the robot (RGB-D camera in this research) and represents objects with accuracy depending on the distance from the sensor.

2 Related Work

2.1 3D Mapping

The basic geometrical representation of the environment used in robotics to model the shape of the obstacles is a 2.5D elevation map [16, 24]. The first implementation proposed by Kweon and Kanade builds a map from multiple sensors. The elevation map is also used to localize the robot by matching the current view to the global map. The locus method which uses the range sensor model is used to interpolate the terrain surface [16]. Elevation maps are widely used when the robot is high above the terrain e.g. on wheeled, tracked, flying, and walking robots [11]. The elevation map can be also extended by the information about the terrain type [4]. The terrain classification is performed with the utilization of voxel-based 3D structures. Plageman et al. solve a regression problem to build a model of the terrain. To this end, the Gaussian Process (GP) is applied. As a result the proposed model naturally deals with the measurement noise and fills in small gaps.

The elevation map cannot properly represent the environment when the obstacles are higher than the robot. The space below higher objects like tunnels or bridges is filled in because the cell of the map stores the highest measured value. To deal with these situations the Multi-Level Surface Maps (MLS) have been proposed [34]. The MLS builds a stack of elevation maps that properly represent gaps below high objects.

A representation of a full 3D model of the environment is a multi-volume occupancy grid [9]. The cells are grouped into vertical volumes to save memory. Another idea is to store information about 3D voxels in the octree. This approach is used in the OctoMap [19]. The size of the voxel is a design parameter and it influences memory usage, computation time, and more importantly the accuracy of the map. The 3D space is discretized and we lose information about details inside each voxel. The space inside each cell is better represented in the Normal Distributions Transform Occupancy Map (NDT-OM) proposed by Saarinen et al. [36]. In this map, each voxel stores information about the occupancy and also the mean value and covariance of measurements inside the voxel, thus it better represents the occupied space inside each cell. The method was successfully applied in a large-scale [35] and for highly dynamic environments [36].

Other 3D example representations of the environment are triangle meshes and surfels. FastFusion represents the surface of the object by a triangle mesh with variable resolution [37]. It also uses an octree structure to efficiently use the memory. In BundleFusion, the dense 3D reconstruction is performed online simultaneously with tracking the motion of the camera [7]. The localization is based on sparse features and dense geometric and photometric matching. The scene model utilizes truncated signed distance (TSDF) stored in the volumetric grid of voxels. The method returns globally optimized poses which are used to re-estimate the 3D mesh model of the environment. Also Canelhas et al. utilize TSDF in the 3D model of the environment [6]. The TSDF is used to represent the alignment error and estimate the motion of the camera. In [13] the TSDF model is extended by the information about object categories. An example of surfel representation can be seen in [39]. In this case, surfels are stored in multi-resolution maps (octrees). View-dependent maps are used because they can be processed efficiently on CPU and matched to localize a robot [22]. Similarly to our approach, the key views are used to estimate the camera motion but in our research, we are focused on dense mapping and reconstruction and we do not contribute in the field of localization. The environment can be also represented by geometric structures like planes [42] or objects with semantic meanings [25].

2.2 Localization

SLAM methods can be divided into few groups according to the core idea. In the first group, the depth measurements are used directly. Because 3D points matching is computationally expensive these methods use GPU units to process data. The example method is Kintinuous [45]. Other methods utilize sparse visual features and optimize their poses in the 3D space to estimate the motion of the camera. In ORB-SLAM2 [30] the depth measurements are utilized to solve the scale ambiguity problem and obtain proper metric information. The matching of visual (RGB) features only is advantageous, because the direction to the feature is measured with higher accuracy than measuring the distance to the feature by the depth sensors. Relying on the depth measurements introduces significant noise. On the other hand, the optimization of the camera pose using 2D features only is more demanding. Another method to recover scale in the monocular system is used in Visual-Inertial Simultaneous Localization and Mapping and Inertial-Visual Odometry [29] which predicts the next pose of the camera based on the IMU measurements. The motion of the camera can be also estimated using intensity changes on the RGB images like in LSD-SLAM [10]. In this paper, we show the advantages of using local maps stored in the pose-based graph. We propose the use of view-dependent NDT-OM maps coupled with ORB-SLAM2 which uses keyframes (RGB images) organized in a graph to represent the transformation between camera poses [30].

The localization error strongly influences the quality of the obtained dense map. An incorrect position of a sensor causes erroneous updates of the global map. This error is later difficult to correct and as a result, the map contains phantom objects. Wrong measurements can be removed when the localization error decreases and the environment is revisited. Then, the new measurement corrects the previous changes. Incorrect objects can be also removed from the map by checking visibility [19]. In this case, phantom objects from the map are removed if they occlude the currently observed objects.

Most of the mapping methods assume that the trajectory of the robot is known, but there are some examples where the localization is integrated with dense mapping. An interesting approach to an integrated dense mapping and localization was presented by David Droeschel et al. [8]. The robot creates local 3D maps that are later aligned to compute the position of the robot and provide a consistent model of the environment. In BundleFusion, the motion of the camera and model of the environment are estimated simultaneously [7]. Also, Stuckler et al. localize the robot using a dense model but in this case, it is based on surfels [39]. The extended version of ElasticFusion uses semantic segmentation of the RGB images to extract objects from the scene and improve the reconstruction accuracy. Each surfel in the map is described by the recognized instance of the object class which helps with data association from the current view and data in the map. Similarly, a globally consistent map of fused surfels is created in [26]. In [25] the produced semantic map, which contains objects only, is based on the Truncated Signed Distance Function (TSDF). Objects and camera poses are stored in the pose-based graph. In [11, 12] the localization and dense mapping are independent, but the accuracy of the localization system is taken into account. The model is based on the elevation mapping, but the map is local and always moves with the robot (the robot is in the center of the map). More importantly, the uncertainty of the map increases for the regions which are far from the current pose of the robot due to the localization drift.

2.3 Mapping with Local Maps

In this article, we propose using local maps which are later aligned to improve a global model. The mapping with submaps is standardized for 2D maps [1]. Strom and Olson introduced graph-based representation to store local maps [38]. Google Cartographer SLAM uses 2D local submaps consisting of probability grids which are built from 2D scans [17]. The submaps are organized in the graph structure and are used for loop closing. In this article, we show that the graph of local maps can be used to create 3D models.

The idea of local maps connected with pose graph and the possibility to correct global map was presented in our previous work [3]. A similar approach is utilized in [27], but we use view-dependent representation as a local map which is memory efficient and better represents the uncertainty of the perception system. Also, Ho et al. correct the global map after loop closure detection [18], but they use Virtual Occupancy Grid Maps. In contrast, ElasticFusion uses a dense surfel-based model obtained by dense frame-to-frame camera tracking without graph optimization [44, 46]. Choi et al. fuse the range images to obtain a surface mesh from a defined number of frames [41]. In our case, the number of frames used to update a local model is dynamic and depends on the number of re-observed visual features from the keyframe. The mapping system presented in [21] creates a new submap when the number of re-observed voxels drops below a threshold. Each scene is represented by the TSDF-based and subsampled depth images that are used for relocalization and loop closure detection. An approach that uses a set of local maps which are connected with covisibility graph is presented in [28]. The strategy is similar to our approach but we use NDTOM instead of TSDF to represent the local maps. In [43] the ORB-SLAM2 [30] is combined with InfiniTAM [20]. The ORB-SLAM2 is used for precise localization, while InfiniTAM is applied for dense volumetric fusion. The keyframes are used only to update the dense model. In our approach, we create dense local models because our goal is to integrate multiple depth frames into a local model and estimate the distribution of the measurements (covariance matrix).

2.4 Approach and Contribution

In this article, we propose a mapping system that is coupled with the localization system. Data from the RGB-D sensor are integrated in local maps which are connected to unique views (keyframes). This approach allows reducing the influence of the sensor noise on the map quality. When the camera moves and observes a new part of the environment, the new local map is created. The local maps are connected by the $SE3$ transformations and are stored in the pose graph. The relative pose between maps can be freely modified. The pose of the local maps is provided by the localization system. To this end, we applied ORB-SLAM2 [30] which is based on keyframes and creates a pose graph in the back-end.

To represent local maps, we choose a view-dependent representation. Instead of dividing a 3D space into a regular grid, like in OctoMap or NDT-OM, we represent objects on the image plane. We propose a new representation which is based on Normal Distribution Transform Occupancy Maps (NDT-OM) [36]. The NDT-OM is computed on the regular 2D grid on the image plane. As a result, the objects which are close to the camera are represented naturally with higher accuracy than distant objects.

The global map of the environment, which can be later used for collision avoidance and motion planning, is represented by the set of ellipsoids defined in the 3D space. The pose of the local maps is used to compute the pose of the ellipsoids in the 3D space. We use the model of the camera to project ellipsoids on the keyframes and to choose the most accurate model of the observed region of the environment.

This article presents a unified method for localization and mapping of a new environment. The proposed structure of the map allows for correction of the global map after loop closure which is presented in our previous work [3]. The local maps were based on standard view-independent NDT-OM which accuracy does not depend on the distance from the camera. We were also using our own method for covisibility-graph management and loop closure detection. In this article, we present a mapping architecture that is integrated directly with the *state of the art* pose-based localization system ORB-SLAM2. In this research, we show that the proposed view-dependent model of the environment is more efficient than the 3D model (NDT-OM or OctoMap) because it adapts the size of the cells of the map to the distance between the sensor (RGB-D camera) and the objects in the environment. The initial work on the view-dependent local maps was presented in our conference paper [47].

The rest of this paper is organized as follows. Section 3 describes the proposed graph-based mapping system. Section 4 provides the results of the evaluation of the proposed method, and Section 5 concludes the article.

3 Graph-based Mapping

3.1 General Architecture

We designed the mapping system taking into account the properties of the real localization systems. Until the loop closure is detected, the global map contains a set of local maps. However, the larger the distance between local maps, the larger the localization drift is. In this case, the global map accumulates the error caused by the localization drift. The robot can use the information about neighboring local maps only which are less affected by the localization drift. The example of this strategy is presented in [11], where the local map stores only recent measurements. After loop closure detection, the localization drift can be reduced. We utilize this property of the localization system in the dense mapping module by storing local maps in the pose graph. When the pose graph is corrected, the alignment of local maps and consistency of a global map is improved.

The structure of the proposed map is presented in Fig. 2. The global map consists of a set of local maps related to keyframes k_i. The local maps are stored in a pose-based graph. The pose of each local map $\mathbf{K}_i$ is defined w.r.t. the first keyframe and is represented by the $SE3$ transformation. We also store edges in the graph that represent covisibility between keyframes $c_{\mathrm{i,j}}$. The covisibility factor defines the percentage of the scene re-observed between keyframes. The covisibility factor allows reducing the time needed to find submaps that describe the same region of the environment. In contrast to our previous work, where the covisibility factor is directly computed from the number of detected features on the keyframes [3], we take the covisibility values directly from the ORB-SLAM2 [30].

3.2 View-dependent Local 3D Map

Our mapping strategy is directly based on the NDT-OM mapping method proposed by Saarinen et al. [36]. Each cell of the map stores information about occupancy and the covariance matrix which represents the distribution of measurements inside cells, and mean position ($[x, y, z]^T$). Ellipsoid obtained from the covariance matrix describes the shape of the object inside the voxel. With the same cell size, the NDT-OM better represents the shape of objects in comparison to OctoMap [19]. We utilize this property of the NDT-OM in the proposed method. Additionally, we propose to store NDT in the structures which change the resolution with respect to the distance from the camera to better represent the uncertainty of the depth measurements from range sensors [33].

The main disadvantage of using a 3D map, which divides the space into regular voxels, is the constant resolution of the model. As a result, we lose information about objects which are close to the camera and are measured with high precision. To deal with this problem, we propose a view-dependent model, which naturally utilizes the uncertainty model of the camera. The general idea of the local view-dependent map is presented in Fig. 3. The container which stores ellipsoids is updated according to the current RGB-D frame. The size of the container is set to cover 1280×960 px image and should be larger than the resolution of the keyframe because we update the container from neighboring camera poses and the points from other frames are projected outside the borders of the keyframe. The size of the cell in the container defines the number of pixels in the image covered by this cell. In the experiments presented in the article, the size of each cell is set to 5×5 px which is a good compromise between the accuracy of the map and the number of cells in the map. The points from the current point cloud are projected on the keyframe and used to update cells. Even though we use the 2D container, the map stores information about 3D shapes (ellipsoids).

The local map is updated using the current RGB and depth images. The color point cloud is obtained from

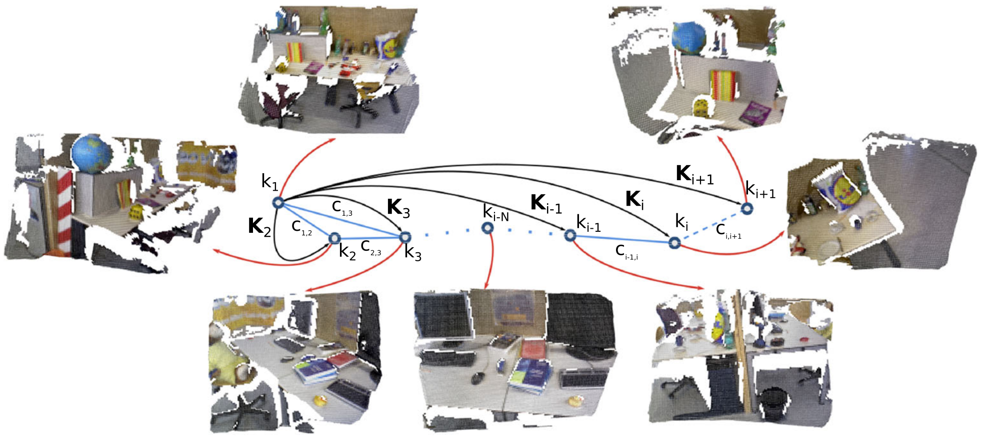


Fig. 2 General structure of the graph-based map. Each node of the graph is related to a local 3D map. Nodes are connected by the $SE3$ transformation $\mathbf{K}_i$ and the covisibility factor c

the RGB-D images and transformed to the pose of the considered keyframe. Before the loop closure, the current keyframe is the last added keyframe in the map. After loop closure, the update order changes according to the detected re-visited keyframe number. Then, we use NDT update methodology [36] to obtain the position, color, and covariance matrix describing an ellipsoid inside each cell of the 2D container associated with the keyframe. First, we compute a transformation matrix $\mathbf{T}$ between the keyframe and current camera pose:

$$\mathbf{T} = \mathbf{K}_0^{-1} \cdot \mathbf{K}_i, \tag{1}$$

where $\mathbf{K}_0$ is the global pose of the keyframe, and $\mathbf{K}_i$ is the global pose of the i-th RGB-D frame from the camera. We use the obtained transformation $\mathbf{T}$ to compute position of points $P = \{\mathbf{p}_1, ..., \mathbf{p}_N\}$ in the current keyframe coordinate system $\mathbf{K}_0$:

$$\mathbf{p}^0 = \mathbf{T} \cdot \mathbf{p}_n, \tag{2}$$

where $\mathbf{p}^0$ is the position of the n-th point in the keyframe coordinate system K_0 and $\mathbf{p}_n$ is the position of the point in the current camera frame.

Later, an inverse model of the camera is used to compute the position of the considered point on the 2D container related to the keyframe. Then, points are grouped according to the u and v coordinates. Each group is used to update the ellipsoid related to cell $c_{u,v}$ in the 2D keyframe container. We use NDT update method [36] to update position of the ellipsoid $\mathbf{x}_{u,v}$ and the covariance matrix $\Sigma_{u,v}$. The N new points added to the cell and current mean value $\mathbf{x}_t$ computed

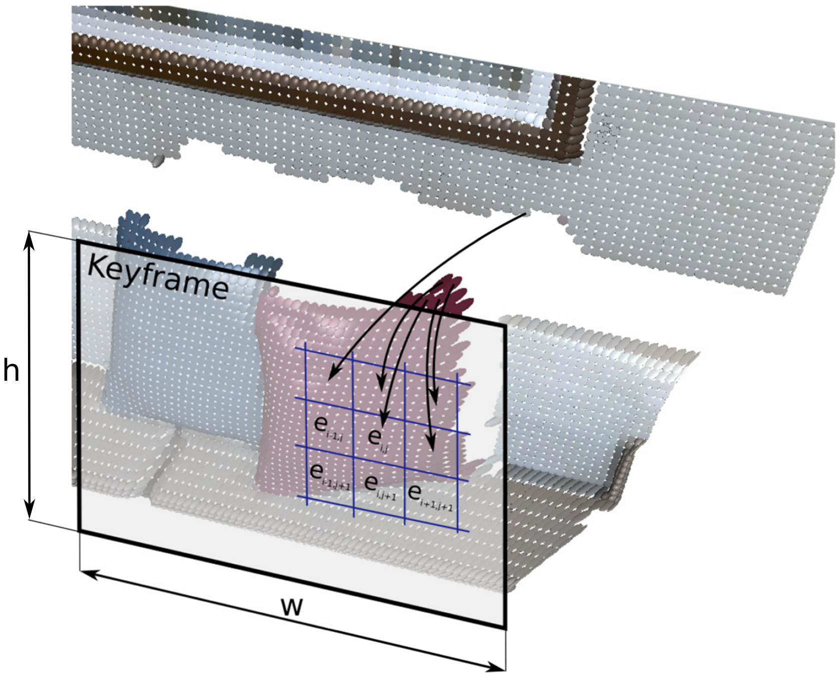


Fig. 3 Local map of the environment: 3D normal distribution transforms (ellipsoids) stored in a 2D container

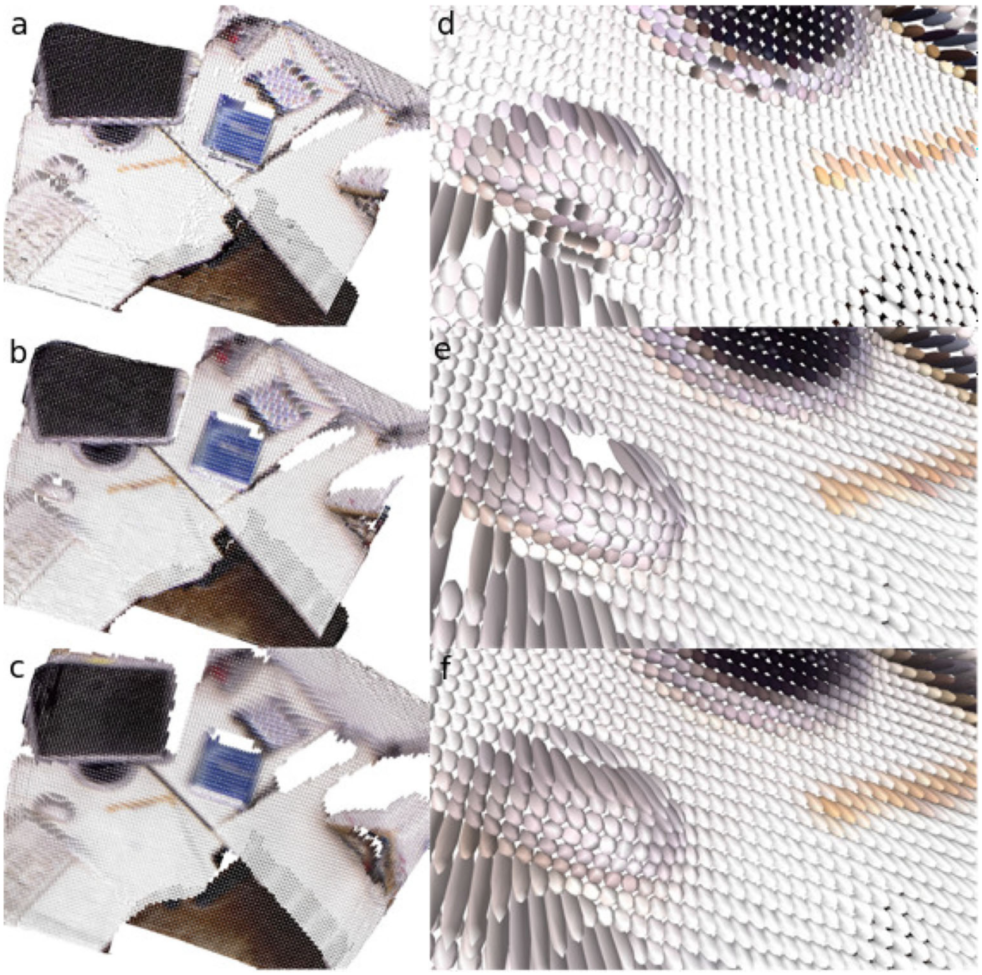


Fig. 4 Example local map generated using, consecutively, 1, 2 and 10 frames, obtained for the `freiburg1_desk` sequence from the TUM dataset [40]: 2D container with ellipsoids (a,b,c) and the enlarged region of the local map (d,e,f)

from total M points are used to update the mean position of the ellipsoid $\mathbf{x}_{t+1}$:

$$\mathbf{x}_{t+1} = \frac{\mathbf{x}_t M + \sum_{n=1}^{N} \mathbf{p}_n}{M+N}. \tag{3}$$

The update procedure is repeated whenever a new point cloud from the sensor is provided. The iterative procedure requires the computation of the sum $\boldsymbol{s}_M = \sum_{m=1}^{M} \boldsymbol{p}_m$ and the total number of points M used to update the considered cell (ellipsoid). We update the covariance matrix using:

$$\Sigma_{t+1} = \Sigma_t + \sum_{n=1}^{N}\left(\mathbf{p}_n - \frac{\mathbf{s}_N}{N}\right)\cdot\left(\mathbf{p}_n - \frac{\mathbf{s}_N}{N}\right)' + \frac{M}{N(M+N)}\left(\frac{N}{M}\mathbf{s}_M - \mathbf{s}_N\right)\left(\frac{N}{M}\mathbf{s}_M - \mathbf{s}_N\right)', \tag{4}$$

where $\mathbf{s}_N = \sum_{n=1}^{N} \mathbf{p}_n$ and N is the size of point cloud used to update the voxel. To update the color of the voxel, we use:

$$\mathbf{c}_{t+1} = \frac{M \cdot \mathbf{c}_t + \mathbf{s}_{c_N}}{M+N}, \tag{5}$$

where $\mathbf{s}_{c_N} = \sum_{n=1}^{N} \mathbf{c}_n$ is the sum of colors for points P used to update voxel, $\boldsymbol{c}_t$ is the current mean color of the voxel.

After each cell update, the total number of points used to update a single ellipsoid is increased $M = M + N$. In Fig. 4 we show the set of ellipsoids stored in the 2D container after 1, 2, and 10 update iterations. In the first iteration, the ellipsoids are updated from a single pair of RGB-D images. Their position and covariance depend on the shape of the object, but the error of the single measurements also plays an important role. The larger size of the ellipsoids naturally represents the larger uncertainty of the measurements. In the following iterations the shape described by the ellipsoids becomes more smooth (compare enlarged region in Fig. 4 after 1 and 10 iterations).

3.3 Integration with the Localization System

We integrated our graph-based mapping system with the ORB-SLAM2 [30]. In Fig. 5 we show the architecture of the final system that consists of two main blocks: localization

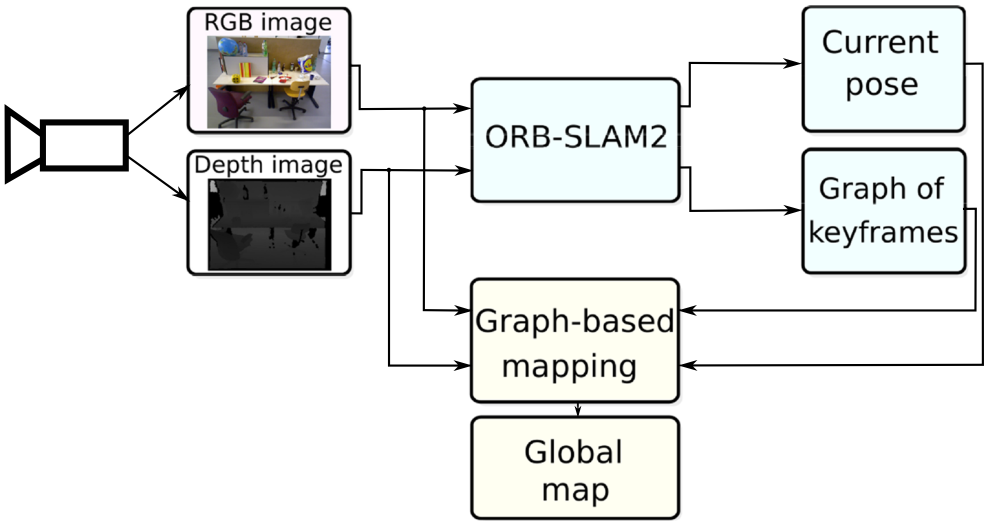


Fig. 5 Architecture of the mapping method integrated with the graph-based localization system

and dense mapping subsystems. We utilize the ORB-SLAM2 with the proposed graph-based mapping system. However, we do not modify the localization system. We use out-of-the-box implementation of the ORB-SLAM2 and modify the sources that allow us to access the pose graph for the keyframes, covisibility graph, and the identifier of the current keyframe. Both sub-systems (mapping and localization) use RGB and depth images. ORB-SLAM2 returns the current pose of the camera. We modified the ORB-SLAM2 software to get information from the back-end of the SLAM. The modified version of the ORB-SLAM2 replaces the methods which we used to define and manage keyframes in our previous work [3].

The obtained pose graph, active keyframe number, and the current pose are used to update the local map. When the camera is close to the current keyframe the pose drift is small. Thus, the localization error has a small influence on the quality of the local map. When the camera moves further from the previous keyframe, the ORB-SLAM2 system creates a new keyframe and the next measurements update another local map. When the robot moves in a new environment, the localization drift between keyframes increases. Generally, the larger distance between keyframes, the larger the localization error we can expect. This error is also reduced by the ORB-SLAM2 when the loop closure is detected.

3.4 Data Filtering

When the point cloud obtained from the RGB-D image is projected on the keyframe, the surfaces of two objects may be measured in a single cell. If the distance between two surfaces is large, the resulting ellipsoid does not properly represent the surface of two objects. This situation happens on the edges of the objects. We propose the procedure which allows removing incorrect ellipsoids from the local map.

We use the uncertainty model of the RGB-D sensor [14] to remove incorrect ellipsoids that appear on the edges of the objects. The covariance matrix computed using NDT updating procedure is closely related to the uncertainty model of the RGB-D sensor [2]. In the filtering procedure, we compare the model of the sensor with the obtained ellipsoids. From the uncertainty model of the sensor, we know the size of the ellipsoids that we can expect at the given distance. The filtering procedure removes ellipsoids that are too long in 3D space according to the uncertainty characteristic of the given RGB-D sensor. Ellipsoids with center points close to the camera origin should have a much smaller magnitude than those further away according to the sensor model. For results obtained from various datasets, we use the same generic uncertainty model of the Kinect sensor [14]. The model of the camera used for filtering data is presented in Fig. 6.

Moreover, to reduce the influence of sparse and erroneous data, we carefully treat the numerical stability when calculating the sample covariance like in [39]. We require a minimum number of samples (>10) to create the covariance matrix.

3.5 Merging Local Maps

The local maps are efficient structures that decrease the influence of the localization error. However, a global map is required to plan the motion of the robot. In our previous work [3], we compute the position of each ellipsoid in the global frame. Then, for the obtained set of ellipsoids,

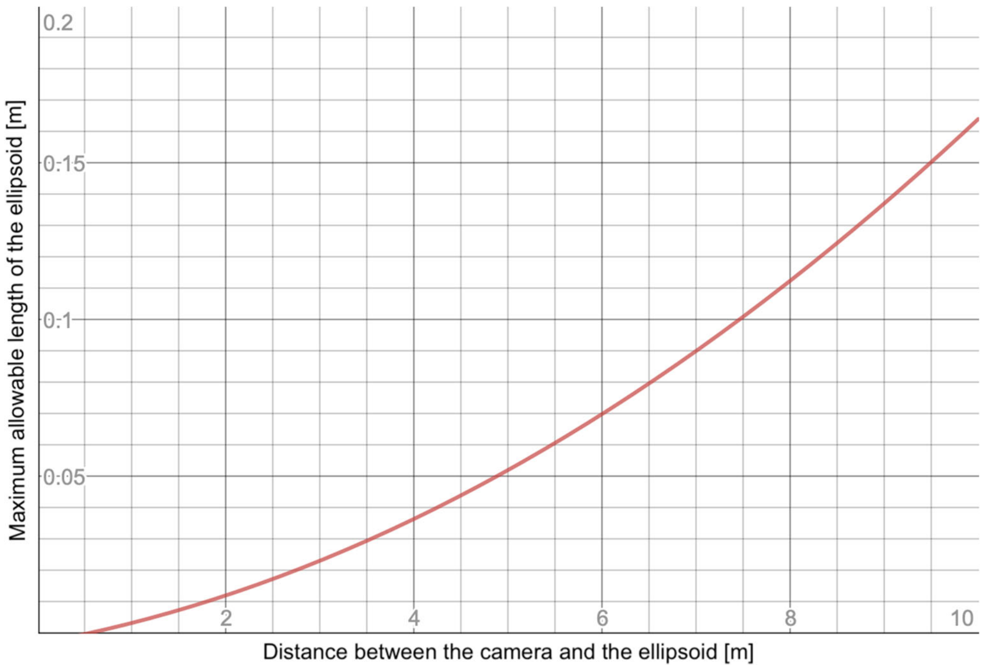


Fig. 6 Example model of the depth camera used to remove incorrect ellipsoids from the local map

we sample 3D points and as a result, we obtain a 3D point cloud. This global representation can be used directly for motion planning or used to update the global model of the environment like the OctoMap. However, in this research, we propose a local map that is view-dependent, and each ellipsoid is obtained from the different distances between the camera and the object. The measurement and obtained ellipsoids that are closer to the object are more accurate than ellipsoids obtained for the same object observed from a distant position. The procedure which merges the local maps should take into account these properties of the mapping system and preserve details of the objects.

We merge local maps to obtain the global model (set of ellipsoids) in two stages. In the first stage, we merge ellipsoids from local maps connected by the edges in the covisibility graph. The local maps connected by the covisibility edges cover a similar region in the 3D space and some parts of the map overlap. In Fig. 7a and in Fig. 7b we show two local maps that overlap. In Fig. 7c we show the local maps in the same coordinate frame. The borders between maps are well visible because the map from Fig. 7b has darker colors, but most of the ellipsoids overlap each other. Thus, we can merge ellipsoids that are close to each other in the 3D space. Because searching for neighboring ellipsoids in the 3D space is computationally demanding we merge ellipsoids on the image (keyframe) plane. We create groups of local maps connected by the covisibility edges and define the map which is in the center of the group. Then, we project ellipsoids from the neighboring maps to the center keyframe using the inverse pinhole camera model: [2]:

$$\begin{bmatrix} u \\ v \\ d \end{bmatrix} = \begin{bmatrix} \frac{x \cdot f_x}{z} + x_c \\ \frac{y \cdot f_y}{z} + y_c \\ z \end{bmatrix}, \tag{6}$$

where $[u, v, d]^T$ is the position of the ellipsoid on the keyframe, $[x, y, z]^T$ is the 3D position of the ellipsoid in the camera frame, x_c, y_c define the position of the optical axis on the image plane, f_x and f_y are focal lengths of the camera. The ellipsoids that are projected on the same 2D cell $c_{u,v}$ in the keyframe are grouped using mean shift clustering. We do not cluster ellipsoids if the distance between them is larger than the threshold (0.25 m) which means that the ellipsoids represent two different surfaces. It might also mean that the localization error is large and the local maps do not align with each other. This error might be corrected when the loop closure is detected. Then, the considered ellipsoids will be merged correctly. For ellipsoids that are located in the same cluster, we compute the mean position, covariance matrix, and color. The results of the clustering are presented in Fig. 7d. The proposed procedure reduces significantly the number of ellipsoids in the global map and smooths out their positions.

In the second stage, we merge information about objects from the local maps which are not connected in the covisibility graph. This situation happens when the part of the scene is observed from a significantly different

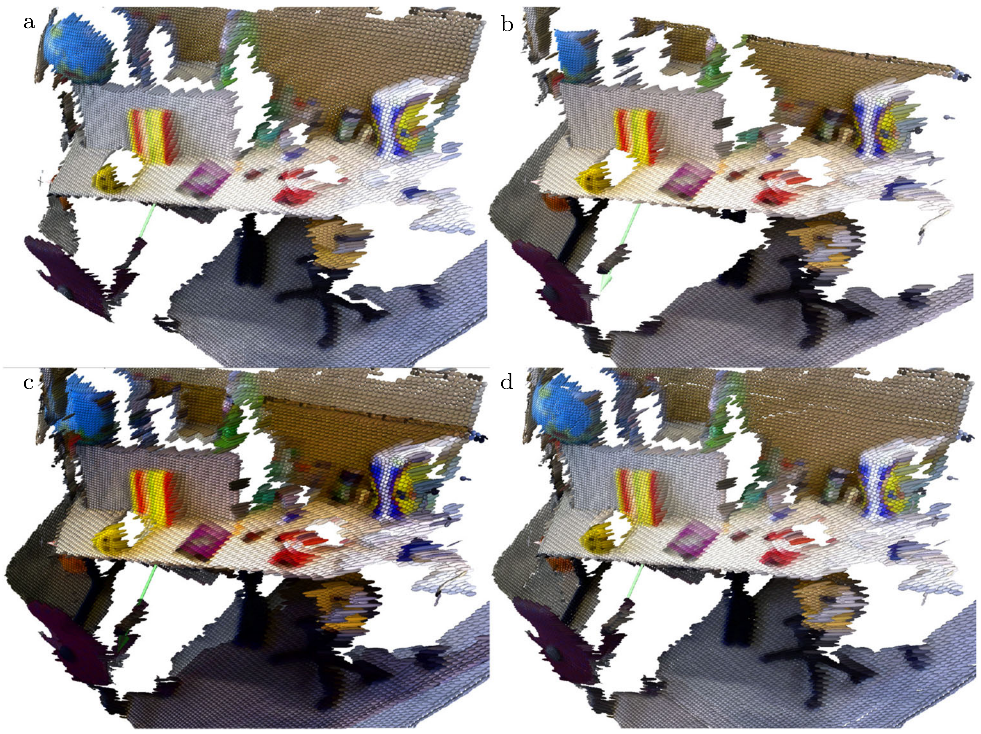


Fig. 7 Merging two local maps (a,b): maps in the common coordinate frame (c) and the obtained set of ellipsoids (d)

perspective and the distance between the camera and/or the orientation of the camera differs significantly. As a result, the ellipsoids which represent the same model also differ significantly. The goal of this procedure is to remove from the map ellipsoids, which occlude other ellipsoids and have larger uncertainty. We use a procedure that is similar to the one from the previous step. We create groups of local maps that are neighbors in the covisibility graphs, but we also add local maps that are not connected in the covisibility graph but are close to each other in the 3D space (Euclidean distance and the angle between camera axes are below the threshold). Then, we check whether ellipsoids from the local maps are occluded by ellipsoids from the other viewpoint (local map). Again, we use the forward and inverse model of the pinhole camera. To speed up computations we store coordinates of each ellipsoid on the keyframe. For the map with 100 local maps, the filtering procedure takes less than 5 s.

4 Results

We performed experiments on various datasets to show the properties of the proposed method[1]. In Fig. 8 we show the example keyframe (local map) of the `freiburg3_desk` sequence from the TUM dataset [40]. In this experiment, we do not estimate the motion of the camera using ORB-SLAM2. We use the ground truth trajectory of the camera registered for the sequence because we test the properties of the dense mapping system only. In Fig. 8 we show the set of ellipsoids resulting from the covariance matrices stored in the 2D container. The map contains information about 3D objects even though the measurements are stored in the 2D structure. The information about the part of the scene is collected for the close camera poses of the keyframe. The properties of the map are well visible in the enlarged regions in Fig. 8. The ellipsoids which are closer to the camera are much smaller than the distant ones. On the other hand, the ellipsoids, which are located on the edges of the objects (a cup, a coke, and a mouse in Fig. 8), are elongated due to measurements (depth values) along the z axis of the camera.

In the second experiment, we show the filtering results of the local view-dependent map. The results of the proposed filtering method are presented in Fig. 9. In Fig. 9a, we show the ellipsoids computed for all cells of the keyframe-based 2D structure. Some ellipsoids are computed for depth values located on the edges of the objects. The measured values inside a cell belong to various surfaces that are distant to each other. It does not represent the surface of the object, but data belonging to two surfaces, and the obtained uncertainty is larger than the uncertainty resulting from the camera model [14]. In Fig. 9b we show the local map without ellipsoids removed according to the uncertainty model of the camera (Fig. 6).

[1]short video from experiments is available at https://https://www.dropbox.com/s/4uasiuoqkf8gh03/map.mp4?dl=0

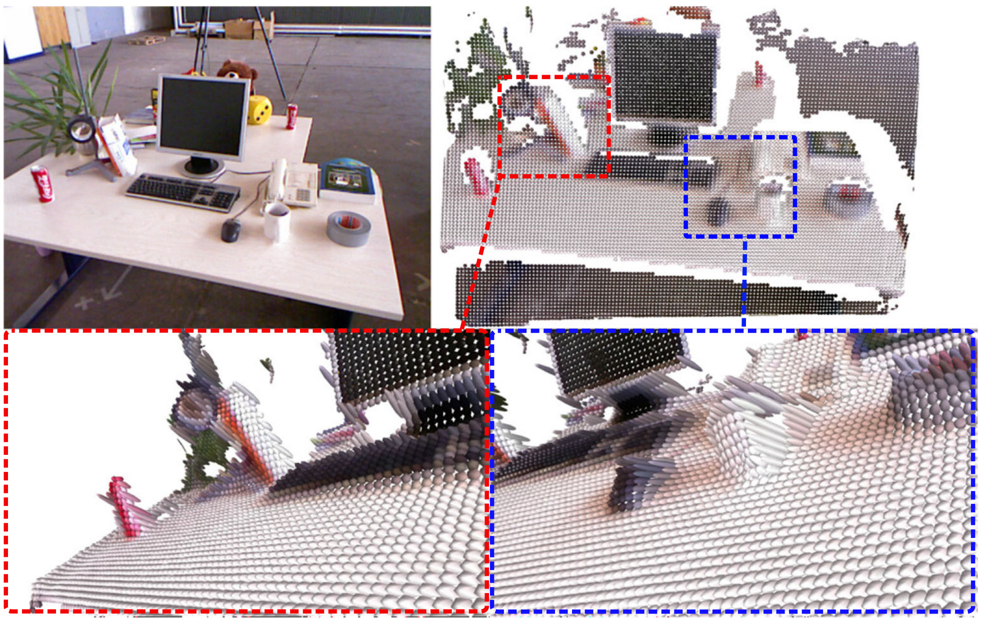


Fig. 8 Local map obtained for the `freiburg3_desk` sequence from the TUM dataset [40]. The enlarged regions of the map are in the red and blue frames

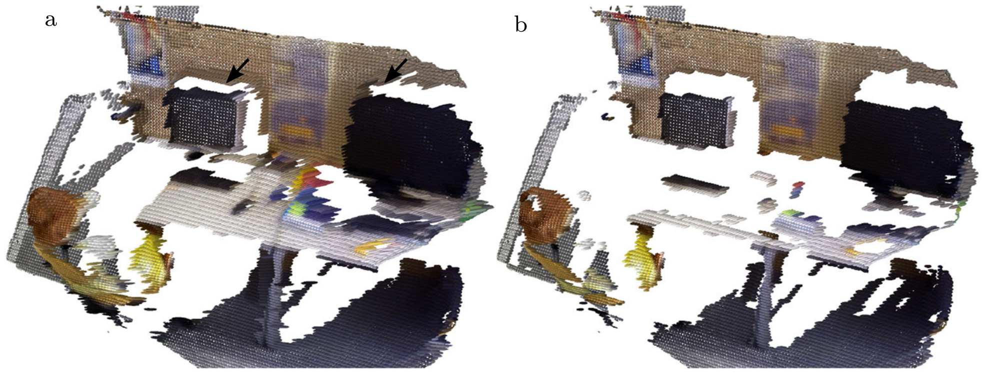


Fig. 9 Filtering of a depth image: 3D ellipsoids obtained from a single depth image (**a**) and 3D ellipsoids related to the planar surfaces (**b**) after removing ellipsoids resulting from potentially erroneous measurements. Arrows indicate ellipsoids on the edges

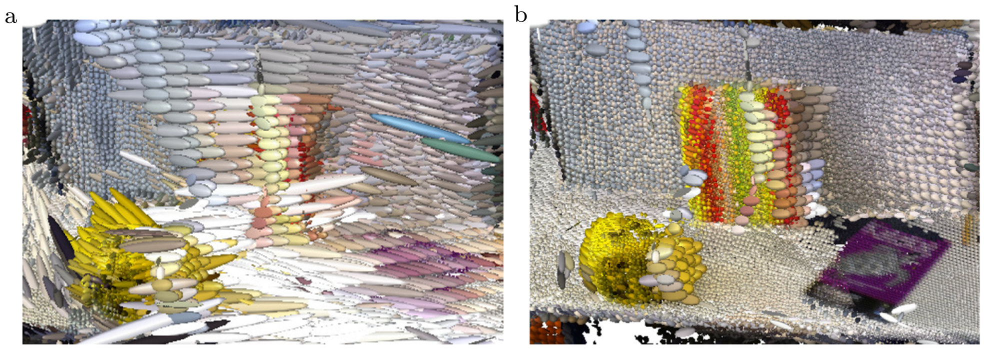


Fig. 10 Occluding ellipsoids removal procedure which allows keeping ellipsoids which represent objects with the highest precision: before (**a**) and after filtering (**b**)

In the third experiment, we show the results of the occluding removal procedure. The example scene is presented in Fig. 10. In Fig. 10a, we show all ellipsoids computed for the local maps and transformed into the global map frame. Some measurements are obtained from distant views, in other cases, the camera was closer to the objects. As a result, the ellipsoids have various sizes. The goal of the occluding ellipsoids removal procedure is to keep only the ellipsoids that represent the object with the highest accuracy. The results of the procedure are presented in Fig. 10b. When the procedure is applied, the uncertain measurements of the scene are removed and the most accurate data is used to represent parts of the scene.

We also show the global map obtained for the `fr3_long` sequence from the TUM dataset [40]. The visualization of the global set of ellipsoids is presented in Fig. 11. The global map contains 358258 ellipsoids after merging local maps and removing occluding ellipsoids. The obtained number is smaller than the number of pixels in a single RGB image which shows the compression rate for the proposed method. Despite the reduced number of ellipsoids we still keep detailed information about the objects. In Fig. 11, we show the enlarged regions of the global map. Some areas of the map are registered with higher precision because they were measured from a closer distance than other regions of the environment.

In Table 1, we show the properties of the global maps obtained on the three sequences from the TUM dataset [40]. The initial number of ellipsoids in the global map Σ_e is 892232 for the model presented in Fig. 11. After merging the ellipsoids and removing occlusions the number of ellipsoids Σ'_e is more than two times smaller. A similar reduction rate is obtained for other sequences presented in Table 1. We also check the time needed to obtain a global map. The maximum value is 14.2 s which is obtained for the graph with 161 local maps. For `fr2_desk` sequence, where the number of local maps is 88, the global map is generated in less than 4 s. The results presented in Table 1 are obtained on the computer with i5-8250U CPU.

In the next experiment, we compare three mapping methods: OctoMap [19], NDT-OM, and the proposed view-dependent approach. We test the method on the `freiburg1_room` sequence from the TUM dataset [40]. In Fig. 12a, we show the single RGB image from the sequence which shows the configuration of the environment. The obtained maps are shown in Fig. 12b-d (OctoMap, NDT-OM, and the proposed method). The voxel size in the OctoMap and NDT-OM is set to 0.1 m. When the voxel size is decreased the accuracy of the map increases. Also, the number of updated voxels and computation time increases. The shape of the objects is better represented by NDT-OM than OctoMap, but it comes with a higher computational cost. Our approach, presented in Fig. 12d, provides a 3D map with varying resolution. The details of the objects included in the map are much better visible.

Table 1 Parameters of the maps obtained on the `fr3_long_office`, `freiburg1_room`, and `fr2_desk` sequences from the TUM dataset

sequence	`fr3_long`	`fr1_room`	`fr2_desk`
submaps no.	161	88	99
Σ_e	892232	405137	390435
Σ'_e	358258	189434	90719
t_g [s]	14.2	4.8	3.9

We compare the total number of ellipsoids in the local maps Σ_e, ellipsoids after clustering and filtering Σ'_e, and time required to generate global map t_g

The last experiment related to the proposed mapping system is performed to compare the accuracy of the view-dependent ellipsoids and the OctoMap [19]. The results are presented in Table 2. We use the `living_room_kt3` sequence from the synthetic ICL-NUIM dataset [15], because it contains the ground truth model of the environment. Also, the sensor data does not contain the noise and we can compare the influence of the mapping methods on the accuracy of the model. For each experiment, we use the ground truth trajectory, raw RGB, and depth images to update the map. We compare the obtained map of the environment with the reference mesh model. To this end, we create a point cloud from the OctoMap and 3D ellipsoids by taking the centers of these geometric structures. Then, we compute the accuracy of the model using CloudCompare tool[2]. To this end, we align the obtained map with the model manually and then we use Iterative Closest Point (ICP) to align precisely the point cloud obtained from the map to the model. For the obtained alignment, we compute the RMSE values between the point cloud and the reference model (Table 2).

In Table 2 we show reconstruction results for the OctoMap [19] and we compare the results to our approach. We show the results for the set of local maps and results for the global map obtained after merging local maps and removing occluding ellipsoids. In Table 2, we show that decreasing the size of the voxel in the OctoMap improves the model of the environment. Obviously, the number of voxels significantly increases (to 328660 when the voxel size is 0.02 m). The Root Mean Squared Error (RMSE) is 15.27 mm for the OctoMap with 0.02 m voxel size. Further decreasing the voxel size increases significantly the memory consumption. The reconstruction error is much smaller when the proposed view-dependent model is used. Even when the cell size is set to 15×15 px the RMSE is 9.8 mm and the number of ellipsoids is only 23620. The

[2]CloudCompare, https://github.com/cloudcompare

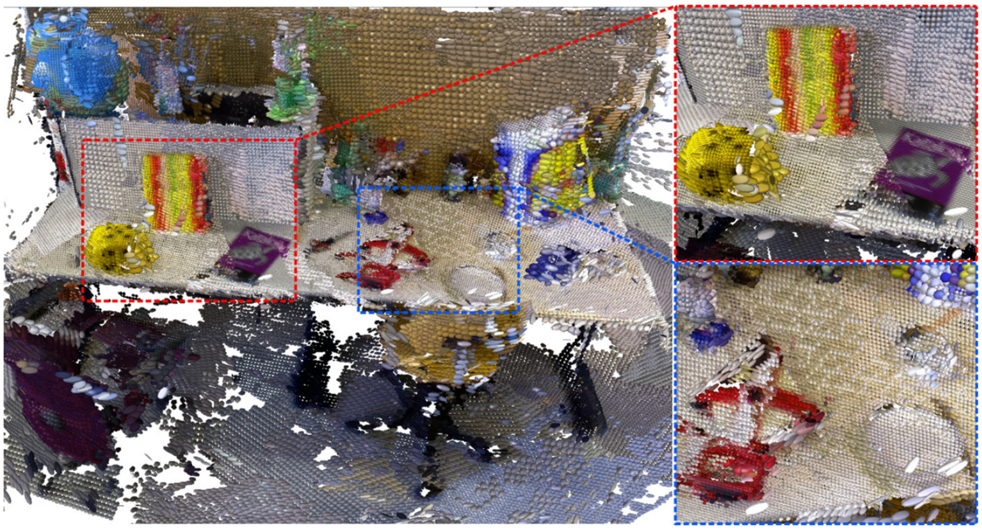


Fig. 11 Global map obtained for the `fr3_long` sequence from the TUM dataset [40]. The enlarged regions of the map are in the red and blue frames

number of ellipsoids is almost 14 times smaller than the voxels in the OctoMap and RMSE is 0.64 times smaller for the view-dependent ellipsoids than for the OctoMap. However, the view-dependent map with 15×15 px cell size and OctoMap with 0.1 m voxel size is sparse and many details about the environment are missing.

The better results for our system results also from the fact that the number of points measured with higher accuracy is higher in the map. The objects which are closer to the camera are densely sampled by the proposed method. On the other hand, the distant objects which are naturally measured with smaller precision are represented by a small number of points in the view-dependent local maps. In OctoMap the environment is sampled uniformly and independently on the measurement accuracy. We can also observe that the global model is more accurate after merging local maps. However, we also observe that merging local maps reduces the accuracy of the model on the edges and corners of the rooms because the ellipsoids belonging to two different surfaces are merged.

The next part of the results section describes the experiments with the dense mapping system integrated with the SLAM method. We show the advantages of the proposed method in the reconstruction task of the new environment. Because the ground truth model of the environment is not available for the TUM dataset, we show the experiments on the ICL NUIM dataset. In this dataset, the model of the `living_room` is publicly available. Thus, we can compare the obtained model with the ground truth data and compare results. The results of the experiments with the whole pipeline performed on the ICL NUIM `living_room` $kt0$, ..., $kt3$ sequences are presented in Table 3.

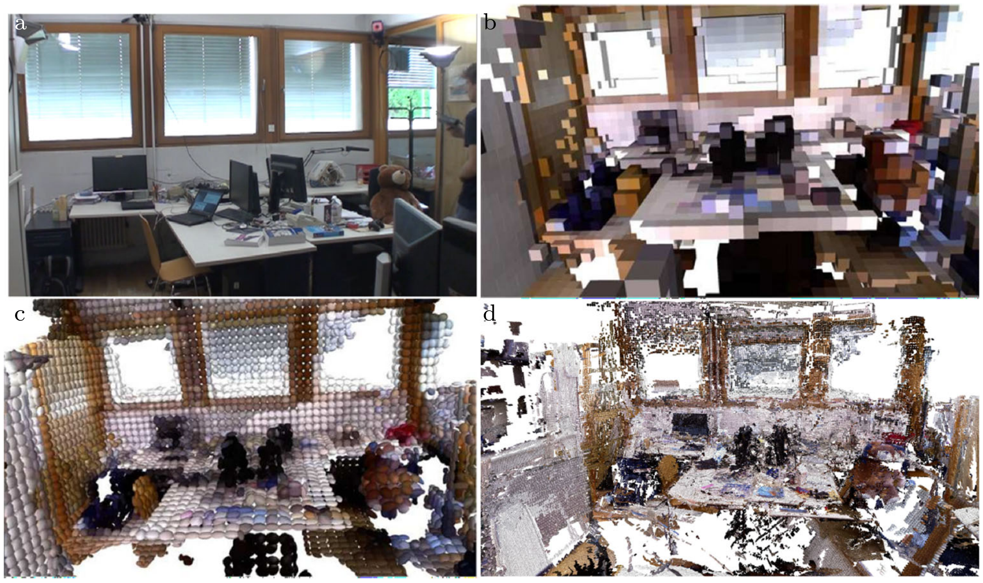


Fig. 12 Global maps obtained for the `freiburg1_room` sequence from the TUM dataset [40]: example RGB image (**a**), OctoMap (**b**), NDT-OM (**c**) and our approach (**d**)

Table 2 Comparison of OctoMap and proposed VD approach in the reconstruction task for the ICL-NUIM `kt3` sequence

	`OctoMap`					`VD`			
	0.1 m	0.075 m	0.05 m	0.035 m	0.02 m	15×15 px	9×9 px	5×5 px	3×3 px
RMSE	31.19	29.27	22.44	18.61	15.27	10.03	7.75	7.92	6.84
$[mm]$						**9.80**	**7.41**	**6.46**	**6.35**
Σ_v	7858	15089	37117	80799	328660	56511	207771	724514	1244615
						23620	**83821**	**171112**	**596759**

Values in bold are obtained after local maps merging and removing occluding ellipsoids

For the $kt0, ..., kt2$ sequences we are able to show only the reconstruction accuracy (Table 3). The ORB-SLAM2 localization system does not detect loop closure. As a result, the pose graph and the dense maps are not corrected. The most interesting results are obtained for the `kt3` sequence. In this case, the ORB-SLAM2 detects the loop closure at the end of the sequence. We can register the reconstruction error before and after the loop closure detection. Before the loop closure, the RMSE and MAE are 4.12 cm and 3.45 cm, respectively. After loop closure detection the error is reduced for the RMSE and MAE, respectively. The experiments presented in Table 3 are performed on the synthetic data. We can compare the accuracy for the obtained results, but the environment is not really challenging and the localization drift is relatively small. Thus the loop closure, if detected, does not improve the results significantly. Thus, we introduced the noise to the parameters of the camera which represents the imperfect camera calibration and more realistic scenario. In this case, the trajectory obtained from the ORB-SLAM2 differs from the ground truth trajectory more than for the perfect camera parameters. In this case, the loop closure detection and trajectory correction reduces the reconstruction error by 10% and 25% for the RMSE and MAE, respectively.

The visualizations of the obtained maps before and after loop closure detection are presented in Fig. 13. The model of the room presented in Fig. 13a and in Fig. 13c is deformed due to the localization drift. After the correction of the pose graph, the dense model can be also corrected. The results are presented in Fig. 13b and in Fig. 13d. In Fig. 14, we show the error histogram computed for the global map before and after loop closure detection. The number of places where the error is high and the maximum error of the model is significantly limited. When we increase the localization drift by introducing imperfect camera parameters the localization drift (Fig. 1a and Fig. 1c), correction of the camera trajectory (Fig. 1b), and improvement of the global dense model after loop closure detection are better visible (Fig. 1d).

We've compared the proposed method to state-of-the-art methods in the reconstruction task on the ICL-NUIM `living_room` dataset. The ICL-NUIM `living_room` dataset contains information about the ground truth trajectory of the camera and the real model of the room used to generate synthetic images. However, the ORB-SLAM2 used as a backbone of our method does not perform well in this dataset. However in the future, it can be replaced, e.g. by a new version of ORB-SLAM which utilizes inertial measurements [5]. ORB-SLAM2 loses tracking and creates local maps that are not positioned properly in the global frame (results in Table 3). It

also does not detect loop closures and does not correct these errors for all trajectories. Thus, in the comparison experiment, we use the ground truth trajectory of the camera. Our main contribution is the system that can improve the accuracy of the obtained dense map after loop closure detection. This is not possible when global maps like Octomap, NDT-OM, or TSDF are used. The system used in this comparison that localizes the camera and simultaneously reconstructs the scene is ElasticFusion [46].

Table 3 Mean Absolute Error (MAE) and Root Mean Squared Error (RMSE) reconstruction error computed for the ICL NUIM `living_room` $kt0, ..., kt3$ sequences

sequence	`kt0`	`kt1`	`kt2`	`kt3_NLC`	`kt3_LC`	`kt3*_NLC`	`kt3*_LC`
RMSE [cm]	3.22	7.32	5.06	4.12	**3.35**	4.2	**3.76**
MAE [cm]	3.03	6.74	3.84	3.45	**3.10**	4.15	**3.11**

The error values for the $kt3\_NLC$ and $kt3\_LC$ sequences are obtained before and after the loop closure, respectively. The values for the $kt3^*$ trajectory are obtained for the SLAM with reduced accuracy

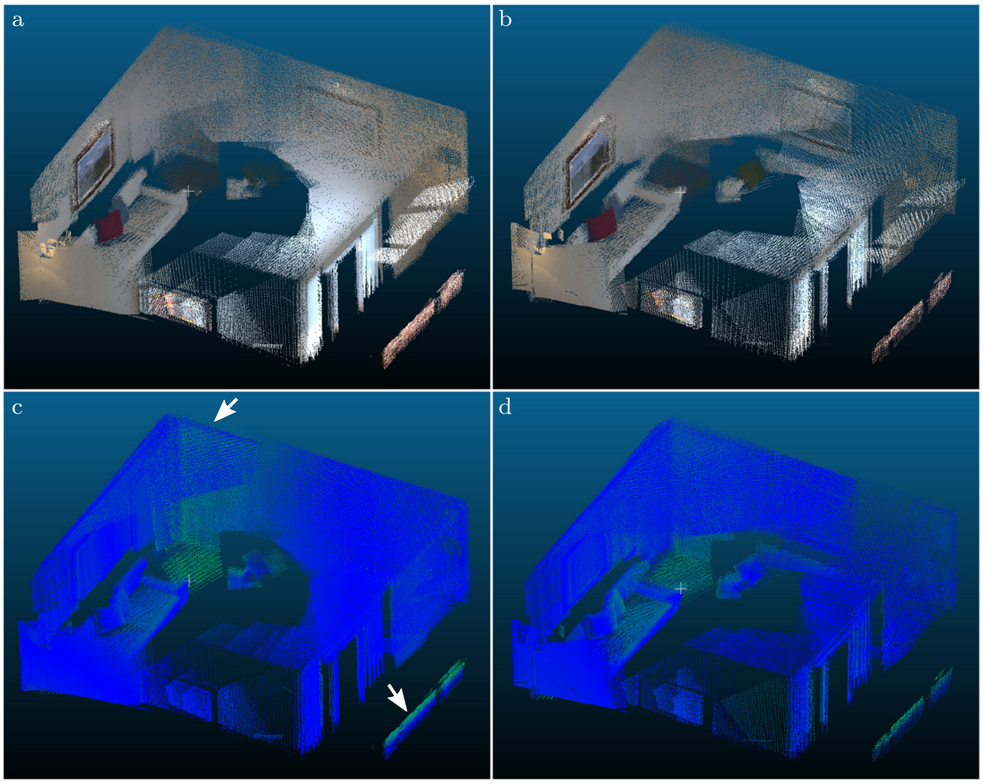


Fig. 13 Global map obtained for the ICL NUIM `living_room` kt3 sequence before (a,c) and after loop closure (b,d): global point cloud (a,b), and comparison between the obtained and reference models (c,d) (green color represents larger error). Arrows indicate regions corrected after loop closure detection

Fig. 14 Error histogram obtained for the ICL NUIM `living_room` kt3 sequence before and after loop closure

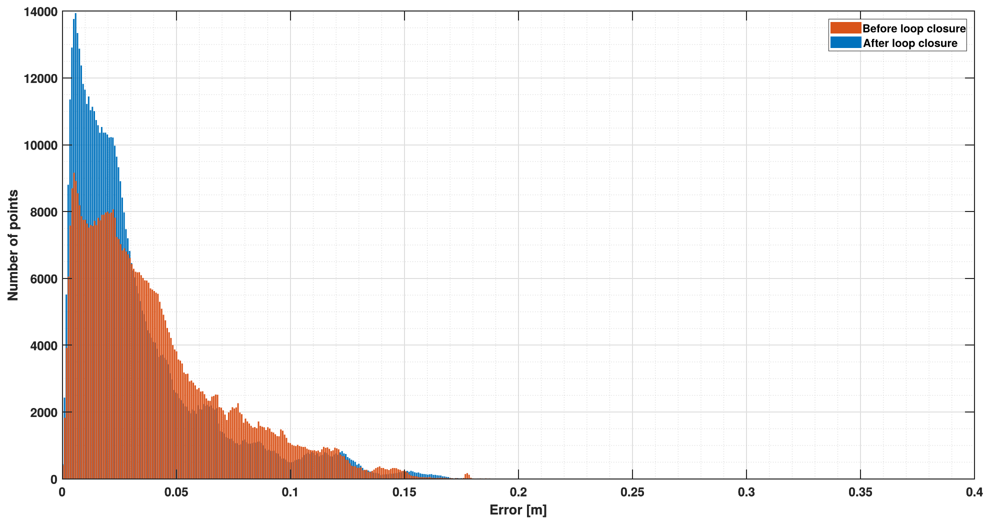


Table 4 Comparison of the reconstruction error computed for the ICL NUIM `living_room` $kt0$, ..., $kt3$ sequences

	`RMSE (m)`			
`System`	kt0	kt1	kt2	kt3
ElasticFusion	**0.006**	0.009	**0.010**	0.048
Voxblox (GT Poses)	0.010	0.017	0.014	0.011
Our (GT Poses)	0.007	**0.008**	**0.010**	**0.006**
Our (GT Poses + merged maps)	0.008	**0.008**	**0.010**	0.007

Table 5 Root Mean Square Error computed for the maps obtained in the experiments on the mobile manipulating robot in the laboratory

		head		base	
	LC	RMSE [cm]	std	RMSE [cm]	std
seq1	no	9.39	10.70	18.22	18.63
	yes	**8.67**	**10.43**	**16.87**	**16.00**
seq2	no	9.57	9.88	13.65	14.50
	yes	**7.51**	**9.21**	**13.49**	**13.65**
seq3	no	14.56	17.13	19.31	17.38
	yes	**13.92**	**15.67**	**18.69**	**17.63**

The results are presented in Table 4. The ElasticFusion [46] performs the best even though it has to localize the camera. However, the ElasticFusion also "bends" the surface of objects to improve the accuracy of the reconstruction. Our system performs similarly when the ground truth poses are used to reconstruct the environment. The accuracy of the reconstruction does not change significantly when the proposed procedure of merging local maps is applied. We compared the results to Voxblox that uses TSDF for reconstruction [31]. In this case, the ground truth poses are also used to reconstruct the scene. The reconstruction results are 29-53% better when the proposed local maps with NDT are used (Table 4).

The next experiments are performed on our mobile-manipulating platform Robot 4.0 [23]. The robot is equipped with two RGB-D cameras (Kinect Xbox One). The first camera is installed in the head of the robot and it is tilted down by 45°. The camera is used to observe the table in front of the robot and to support manipulation tasks. The second camera is mounted on the base of the robot and is used for navigation and collision avoidance. In the experiment, we teleoperate the robot and register the data from both cameras. The three trajectories of the robot estimated by the ORB-SLAM2 and the point cloud representing the environment are presented in Fig. 16. Later, data from the cameras are used to build the model of the room and verify the proposed system. The reference model of the room is built using a Surphaser 100HSX 3D scanner and is presented in Fig. 15.

The results obtained on the mobile-manipulating platform are presented in Table 5. For each sequence, we register the RMSE before and after loop closure detection and correction of the pose graph. We also show the results for the camera mounted on the head and the base of the robot. For each registered trajectory, the error computed after loop closure detection is reduced. With the mean error, the standard deviation of the error also decreases. The reduction of the reconstruction error is not significant, because the experiment is performed in a room that is relatively small and the trajectory of the robot is relatively short. The localization drift, which results from the fame-to-frame motion estimation, is not high. Thus, the reduction of the reconstruction error after LC detection is also not high.

In the last two experiments, we show the possible applications of the proposed mapping method. First, we present a possibility to add information about detected objects to the local maps. In Fig. 17, the example objects (displayed in yellow) are added to the map. In the map, we store information about the category and the instance of the object and the 3D pose w.r.t the local coordinate system. The objects can be detected using the RGB-D images associated with the local map or a container of local ellipsoids. After successful object detection, the ellipsoids in the map can be replaced with 3D mesh models of the known objects. Moreover, searching for the object in the map is computationally efficient because we use the graph-like structure.

In the last experiment, we show the application of the global map in collision-free path planning. The set of local maps is converted to the global map which is later used for collision avoidance and motion planning. The overlapping regions of local maps are merged into a single model. The result is presented in Fig. 18. We remove the ellipsoid which belongs to the ground from the global map because we plan the motion for the differential-drive mobile robot. We use the mesh model of the ellipsoids and mesh model of the robot to detect collisions because our global map, in contrast to local maps, does not provide constant access time to the cells like OctoMap or NDT-OM. To this end, we applied the FCL collision detection library [32]. To find the path between the current and goal position of the robot we use

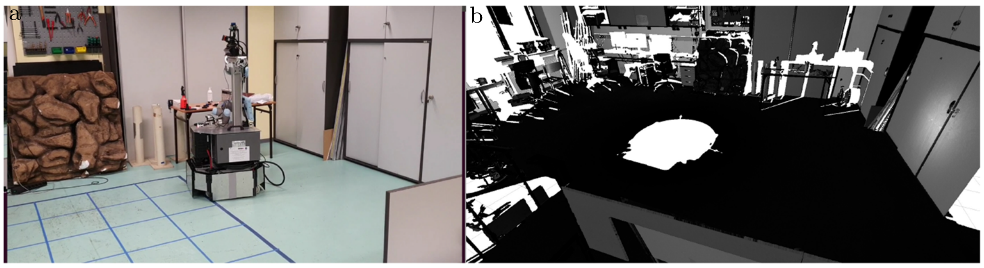


Fig. 15 Mobile manipulating platform Robot 4.0 during the experiment in the laboratory (**a**) and the reference model of the room (**b**)

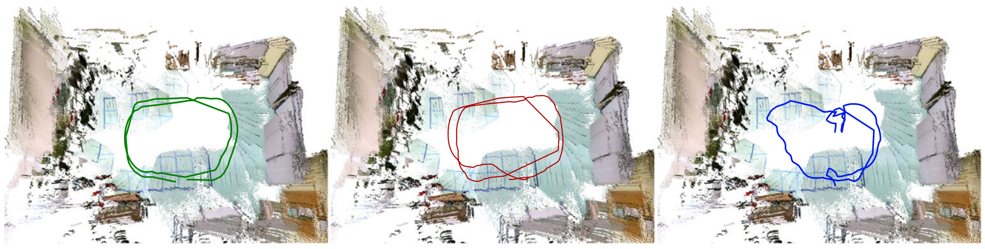


Fig. 16 Trajectories of the robot estimated by the ORB-SLAM2 during the experiment in the laboratory

Fig. 17 Local map obtained for the `freiburg2_desk` sequence from the TUM dataset [40] and three objects stored in the map: an office chair, two coffee cups, and a pen

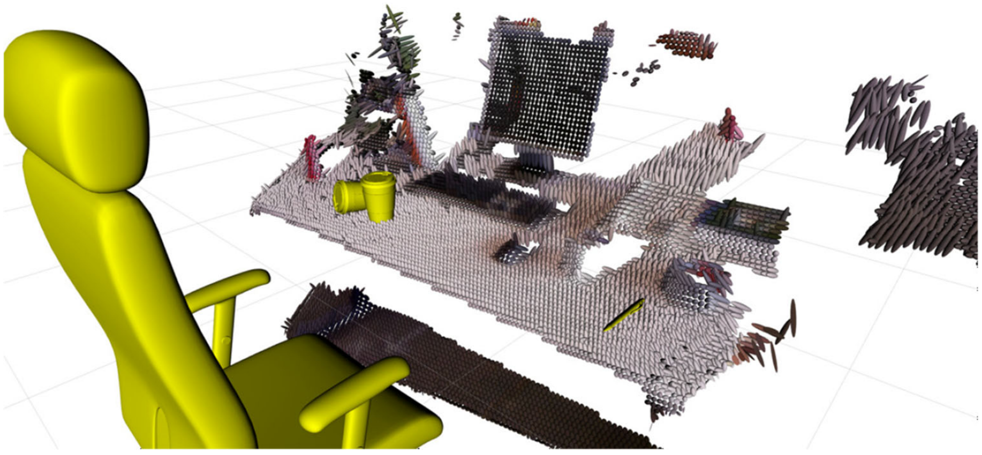
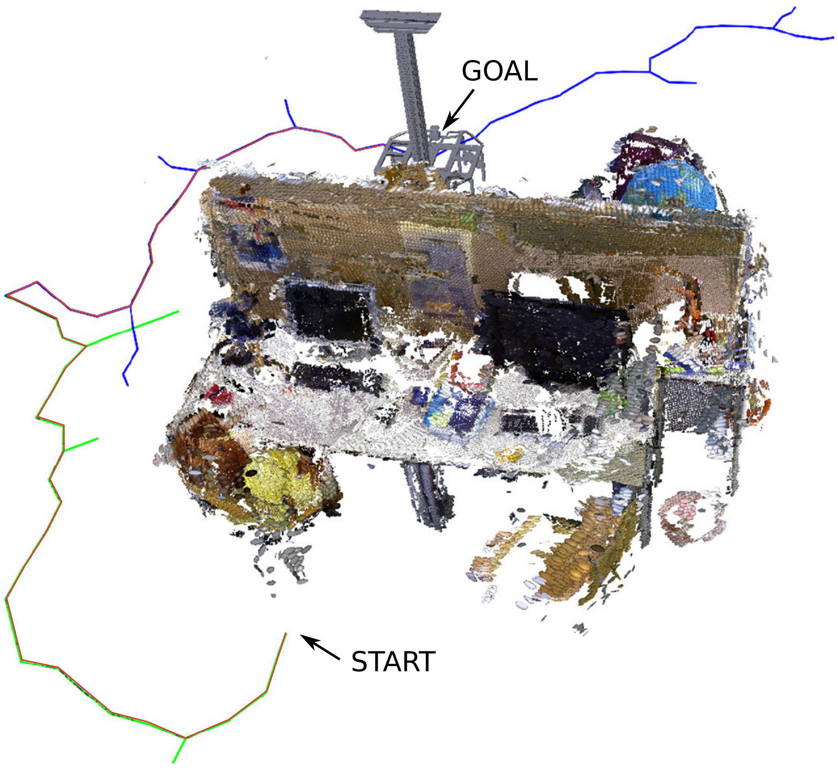


Fig. 18 Path planning with RRT-Connect algorithm on the global map (set of ellipsoids)



the RRT-Connect method. The obtained path is presented in Fig. 18. The planning time in the range 5 m is below 0.5 s.

5 Conclusions

In this article, we focus on the dense mapping problem of a new environment explored by a mobile robot. The main contribution of this work lies in:

- a new view-dependent representation of a local map that utilizes the properties of the sensor. The proposed representation stores NDT transforms in the 2D grid. The measurements directly represent the uncertainty model of the sensor;
- an improved method for local map filtering which utilizes the reference uncertainty model of the camera. The proposed method removes cells that lie on the edges of the objects and contain measurements from multiple surfaces;
- a new architecture of the mapping system which utilizes the properties of the pose graphs. The local maps are organized in a graph-like structure that defines the pose of each local map;
- integration of the graph-based dense mapping system with the keyframe-based localization system (ORB-SLAM2). In the proposed architecture local maps are created when the new keyframe is added to the ORB-SLAM2. Local maps are updated from the neighboring poses of the camera and use the pose estimated by the localization subsystem. When the scene changes the new local map is created;
- procedures that create a global map of the environment from the graph of local maps. The proposed procedure utilizes the forward and inverse model of the camera to project the measurements from neighboring local maps;
- capability of the system to correct the global map when the loop closure is detected;

We also show the practical properties of the proposed system in the experiments on the publicly available datasets and the experiments with the real robot in the laboratory. We also compare the method to the OctoMap to show the advantages of the view-dependent representation. We also provide the application of the system in collision-free path planning and we also show that the map can be used to store the information about the poses of the detected objects. The graph-based architecture also eases searching for objects which are stored in the map. To summarize, we show that with the small number of the ellipsoids we can represent accurately the environment (Table 2). We also show that the accuracy of the reconstruction can be improved after loop closure detection (Table 3, Figs. 11, 14, Table 5). We compare the proposed environment model to the Octomap (Table 2) and TSDF (Table 4). Finally, we show that the local maps can be quickly merged into a global map and used to find a collision-free path of the mobile-manipulating robot. The planning time for the proposed environment model in the range of 5 m is below 0.5 s.

However, this method has some limitations. It inherits the properties of the localization system. If the loop closure is not detected the global map is not corrected. As a result, the local maps which represent the same part of the environment do not overlap. In the visualization of the global map, we can observe two or more layers of the same surface. This behavior will be corrected when the drift is reduced and the surface is close enough to be merged by the proposed methods which generate a local map. Also, the localization method influences the number of local maps. Sometimes, it creates a high number of keyframes. This happens especially when the camera rotates. We see the potential of new neural-based methods which should improve the quality of the loop closure detection methods and keyframes selection.

In the future, we are going to work on the procedures which merge local maps. We are going to improve methods that align the matching surfaces independently on the current distance like in ElasticFusion [46]. We are also going to simultaneously provide feedback to the localization system about the results of the local maps matching and improve the camera pose estimation. The utilization of information about the uncertainty of the measurements should also improve the matching accuracy [2]. In the future, we are also going to use GPU to improve the frame rate and time needed to generate the global map.

Supplementary Information The online version contains supplementary material available at https://doi.org/10.1007/s10846-021-01476-1.

Author Contributions Dominik Belter contributed to the study conception and design and was responsible for funding acquisition. Rafal Staszak was working with the initial version of the mapping system. Local view-dependent NDT-based mapping was implemented by Krzysztof Zieliński. Mikolaj Nowaczyk integrated the local mapping system with ORB-SLAM2. Data collection and analysis were performed by all authors. All authors prepared the first draft, read and approved the final manuscript.

Funding This work was supported by the National Centre for Research and Development (NCBR) through project LIDER/33/0176/L-8/16/NCBR/2017

Code Availability Modified version of ORB-SLAM2 is available here: https://github.com/mikolajnowaczyk/ORB_SLAM2

Declarations

Consent for Publication The authors and relevant institutions approve publishing this research.

Conflict of Interests The authors have no conflict of interests to declare.

References

1. Amigoni, F., Yu, W., Andre, T., Holz, D., Magnusson, M., Matteuci, M., Moon, H., Yokotsuka, M., Biggs, G., Madhavan, R.: A standard for map data representation. IEEE Robot. Autom. Mag. **25**(1), 65–76 (2018)
2. Belter, D., Nowicki, M., Skrzypczyński, P.: Modeling spatial uncertainty of point features in feature-based RGB-d SLAM. Mach. Vis. Appl. **29**(5), 827–844 (2018)
3. Belter, D., Piaskowski, K., Staszak, R.: Keyframe-Based local normal distribution transform occupancy maps for environment mapping. In: IEEE 23rd International Conference on Emerging Technologies and Factory Automation (ETFA), pp. 706–712 (2018)
4. Belter, D., Wietrzykowski, J., Skrzypczyński, P.: Employing natural terrain semantics in motion planning for a multi-legged robot. J. Intell. Robot. Syst. **93**, 723–743 (2019)
5. Campos, C., Elvira, R., Rodríguez, J.J.G., Montiel, J.M.M., Tardós, J.D.: ORB-SLAM3: An accurate open-source library for visual visual-inertial and multi-map SLAM (2020)
6. Canelhas, D.R., Stoyanov, T., Lilienthal, A.J.: Sdf Tracker: A parallel algorithm for on-line pose estimation and scene reconstruction from depth images. In: IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 3671–3676 (2013)
7. Dai, A., Nießner, M., Zollhöfer, M., Izadi, S., Theobalt, C.: Bundlefusion: Real-time globally consistent 3D reconstruction using on-the-fly surface reintegration. ACM Trans. Graph. 36(3) (2017)
8. Droeschel, D., Schwarz, M., Behnke, S.: Continuous mapping and localization for autonomous navigation in rough terrain using a 3D laser scanner. Robot. Auton. Syst. **88**, 104–115 (2017)
9. Dryanovski, I., Morris, W., Xiao, J.: Multi-volume occupancy grids: an efficient probabilistic 3D mapping model for micro aerial vehicles. In: Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 1553–1559 (2010)
10. Engel, J., Schöps, T., Cremers, D.Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T. (eds.): LSD-SLAM: large-scale direct monocular SLAM. Springer, Cham (2014)
11. Fankhauser, P., Bloesch, M., Gehring, C., Hutter, M., Siegwart, R.: Robot-centric elevation mapping with uncertainty estimates. In: Kozłowski, K. (ed.) Robotics, Mobile Service, pp. 433–440. World-Scientific (2014)
12. Fankhauser, P., Bloesch, M., Hutter, M.: Probabilistic terrain mapping for mobile robots with uncertain localization. IEEE Robot. Autom. Lett. **3**(4), 3019–3026 (2018)
13. Grinvald, M., Furrer, F., Novkovic, T., Chung, J.J., Cadena, C., Siegwart, R., Nieto, J.: Volumetric instance-aware semantic mapping and 3D object discovery. IEEE Robot. Autom. Lett. **4**(3), 3037–3044 (2019)
14. Halmetschlager-Funek, G., Suchi, M., Kampel, M., Vincze, M.: An empirical evaluation of ten depth cameras. IEEE Robot. Autom. Mag. **26**(1), 67–77 (2018)
15. Handa, A., Whelan, T., McDonald, J., Davison, A.J.: A Benchmark for RGB-D Visual Odometry, 3D Reconstruction and SLAM. In: 2014 IEEE International Conference on Robotics and Automation (ICRA), pp. 1524–1531 (2014)
16. Hebert, M., Caillas, C., Krotkov, E., Kweon, I.: Terrain mapping for a roving planetary explorer. In: Proceedings of IEEE International Conference on Robotics and Automation, pp. 997–1002 (1989)
17. Hess, W., Kohler, D., Rapp, H., Andor, D.: Real-time loop closure in 2D LIDAR SLAM. In: Proceedings of IEEE International Conference on Robotics and Automation, pp. 1271–1278 (2016)
18. Ho, B.J., Sodhi, P., Teixeira, P., Hsiao, M., Kusnur, T., Kaess, M.: Virtual occupancy grid map for submap-based pose graph SLAM and planning in 3D environments. In: Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 2175–2182 (2018)
19. Hornung, A., Wurm, K.M., Bennewitz, M., Stachniss, C., Burgard, W.: Octomap: an efficient probabilistic 3D mapping framework based on octrees. Auton. Robot. **34**(3), 189–206 (2013)
20. Kähler, O., Adrian Prisacariu, V., Yuheng Ren, C., Sun, X., Torr, P., Murray, D.: Very high frame rate volumetric integration of depth images on mobile devices. IEEE Trans. Vis. Comput. Graph. **21**(11), 1241–1250 (2015). https://doi.org/10.1109/TVCG.2015.2459891
21. Kähler, O., Prisacariu, V.A., Murray, D.W.Leibe, B., Matas, J., Sebe, N., Welling, M. (eds.): Real-time large-scale dense 3D reconstruction with loop closure. Springer International Publishing, Cham (2016)
22. Konolige, K., Bowman, J., Chen, J., Mihelich, P., Calonder, M., Lepetit, V., Fua, P.: View-based maps. Int. J. Robot. Res. **29**(8), 941–957 (2010). https://doi.org/10.1177/0278364910370376
23. Kulecki, B., Mlodzikowski, K., Staszak, R., Belter, D.: Practical aspects of detection and grasping objects by a mobile manipulating robot. Industrial Robot. https://doi.org/10.1108/IR-10-2020-0242 (2021)
24. Kweon, I., Kanade, T.: High-resolution terrain map from multiple sensor data. IEEE Trans. Pattern Anal. Mach. Intell. **14**(2), 278–292 (1992)
25. McCormac, J., Clark, R., Bloesch, M., Davison, A., Leutenegger, S.: Fusion++: Volumetric Object-Level SLAM. In: 2018 International Conference on 3D Vision (3DV), pp. 32–41 (2018). https://doi.org/10.1109/3DV.2018.00015
26. McCormac, J., Handa, A., Davison, A., Leutenegger, S.: Semanticfusion: Dense 3D semantic mapping with convolutional neural networks. In: 2017 IEEE International Conference on Robotics and Automation (ICRA), pp. 4628–4635 (2017). https://doi.org/10.1109/ICRA.2017.7989538
27. Meilland, M., Comport, A.I.: On unifying key-frame and voxel-based dense visual SLAM at large scales. In: Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 3677–3683, Tokyo, Japan (2013)
28. Millane, A., Taylor, Z., Oleynikova, H., Nieto, J., Siegwart, R., Cadena, C.: C-Blox: A scalable and consistent Tsdf-based dense mapping approach. In: 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 995–1002 (2018). https://doi.org/10.1109/IROS.2018.8593427
29. Mur-Artal, R., Tardós, J.: Visual-inertial monocular SLAM with map reuse. IEEE Robot. Autom. Lett. **2**(2), 796–803 (2016). https://doi.org/10.1109/LRA.2017.2653359

30. Mur-Artal, R., Tardós, J.: ORB-SLAM2: An open-source SLAM system for monocular, stereo and RGB-d cameras. IEEE Trans. Robot. **33**(5), 1255–1262 (2017)
31. Oleynikova, H., Taylor, Z., Fehr, M., Siegwart, R., Nieto, J.: Voxblox: Incremental 3D Euclidean signed distance fields for on-board MAV planning. In: 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 1366–1373 (2017). https://doi.org/10.1109/IROS.2017.8202315
32. Pan, J., Chitta, S., Manocha, D.: Fcl: A general purpose library for collision and proximity queries. In: 2012 IEEE International Conference on Robotics and Automation, pp. 3859–3866 (2012)
33. Park, J.H., Shin, Y.D., Bae, J.H., Baeg, M.H.: Spatial uncertainty model for visual features using a Kinect sensor. Sensors **12**(7), 8640–8662 (2012)
34. Pfaff, P., Triebel, R., Burgard, W.: An efficient extension to elevation maps for outdoor terrain mapping and loop closing. Int. J. Robot. Res. **26**(2), 217–230 (2007)
35. Saarinen, J., Andreasson, H., Stoyanov, T., Ala-Luhtala, J., Lilienthal, A.: Normal distributions transform occupancy maps: Application to large-scale online 3D mapping. In: Proceedings of the IEEE International Conference on Robotics and Automation, pp. 2233–2238 (2013)
36. Saarinen, J., Andreasson, H., Stoyanov, T., Lilienthal, A.J.: 3D normal distributions transform occupancy maps: an efficient representation for mapping in dynamic environments. Int. J. Robot. Res. **32**(14), 1627–1644 (2013)
37. Steinbruecker, F., Sturm, J., Cremers, D.: Volumetric 3D mapping in real-time on a CPU. In: Proceedings of the IEEE International Conference on Robotics and Automation, pp. 2021–2028 (2014)
38. Strom, J., Olson, E.: Occupancy grid rasterization in large environments for teams of robots. In: Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 4271–4276 (2011)
39. Stückler, J., Behnke, S.: Multi-resolution surfel maps for efficient dense 3D modeling and tracking. J. Vis. Commun. Image Represent. **25**(1), 137–147 (2014)
40. Sturm, J., Engelhard, N., Endres, F., Burgard, W., Cremers, D.: A benchmark for the evaluation of RGB-d SLAM systems. In: Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 573–580 (2012)
41. Sungjoon, C., Zhou, Q., Koltun, V.: Robust reconstruction of indoor scenes. In: 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 5556–5565 (2015). https://doi.org/10.1109/CVPR.2015.7299195
42. Wang, J., Song, J., Zhao, L., Huang, S.Hutter, M., Siegwart, R. (eds.): A Submap joining based RGB-D SLAM algorithm using planes as features. Springer International Publishing, Cham (2018)
43. Weilharter, R., Schenk, F., Fraundorfer, F.: Globally consistent dense real-time 3D reconstruction from RGBD data. In: Welk, M., Urschler, M., Roth, P. (eds.) Proceedings of the OAGM Workshop 2018, pp. 121–127. Verlag der Technischen Universität Graz (2018). https://doi.org/10.3217/978-3-85125-603-1-25
44. Whelan, T., Leutenegger, S., Salas-Moreno, R., Glocker, B., Davison, A.J.: Elasticfusion: Dense SLAM without a Pose Graph. In: Robotics: Science and Systems (2015)
45. Whelan, T., McDonald, J., Kaess, M., Fallon, M., Johannsson, H., Leonard, J.: Kintinuous: Spatially Extended KinectFusion. In: RSS Workshop on RGB-D: Advanced Reasoning with Depth Cameras. Sydney, Australia (2012)
46. Whelan, T., Salas-Moreno, R.F., Glocker, B., Davison, A.J., Leutenegger, S.: Elasticfusion: Real-time dense SLAM and light source estimation. Int. J. Robot. Res. **35**(14), 1697–1716 (2016). https://doi.org/10.1177/0278364916669237
47. Zieliński, K., Belter, D.: Keyframe-based dense mapping with the graph of view-dependent local maps. In: Proceedings of the IEEE International Conference on Robotics and Automation, pp. 10,744–10,750 (2020)

Krzysztof Zieliński graduated from Poznan University of Technology with a bachelor of engineering in Automatic Control & Robotics in 2019. Currently, he is pursuing a master's degree in the same field. He is writing his thesis at Universal Robots in Odense where he is also working as a student software developer. His research interests include computer vision, robot control, and collaborative robots.

Rafał Staszak is a third-year Ph.D. student and he takes part in two scientific projects held at Poznan University of Technology. The main focus of his research revolves around flexible perception mechanisms to allow robots to interact with an environment independently and autonomously. In his work, he makes use of modern programming tools for developing deep neural networks and robotic systems. He holds a master's degree in Automatic Control and Robotics from the Poznan University of Technology.

Mikołaj Nowaczyk graduated from Poznan University of Technology in 2020 with a master's degree. His master's thesis was focused on the integration of the graph-based mapping system with the localization system (ORB SLAM2). His research interests include computer vision, mobile robotics, and machine learning.

Dominik Belter graduated from the Poznan University of Technology (2007). He received a Ph.D. degree in robotics from the same University in 2012. Since 2012, he has been an Assistant Professor at the Institute of Control and Information Engineering of the Poznan University of Technology. He spent a year working as a postdoc in the Intelligent Robotics Laboratory at The University of Birmingham in the years 2013-2016. Dominik Belter has been taking part as an investigator in 3 EU and 6 national projects. Currently, he leads two projects related to mobile manipulation and deep learning for robot perception. He serves as a member of the scientific board of the CLAWAR and EMCR conferences. He is the author or co-author of over 70 technical papers in the fields of robotics and computer science. His research interests include walking robots, machine learning, vision, robot manipulation, and soft computing.