SkiMap++: Real-Time Mapping and Object Recognition for Robotics

Daniele De Gregorio
University of Bologna, Italy
d.degregorio@unibo.it

Tommaso Cavallari
University of Oxford
tommaso.cavallari@eng.ox.ac.uk

Luigi Di Stefano
University of Bologna, Italy
luigi.distefano@unibo.it

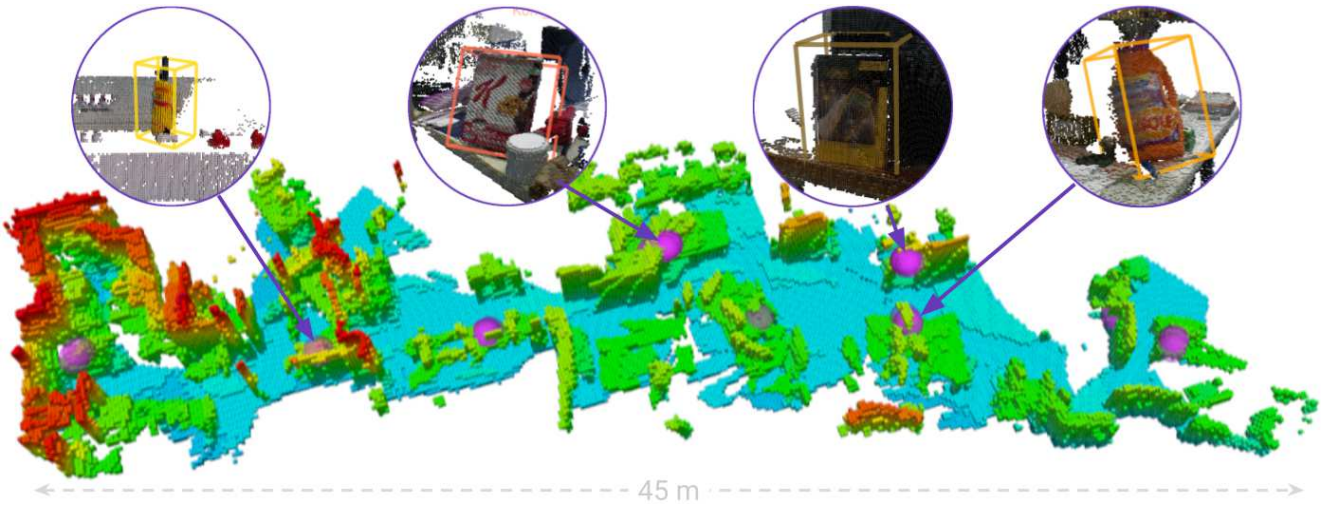

Figure 1. Large-scale map reconstructed online by SkiMap++ through a mobile robot equipped with an head-mounted RGB-D camera. Purple spheres represent areas found alongside with reconstruction which are likely to contain object instances. Magnified circles represent outcomes of the final *Instance Estimation Algorithm*, which is performed in the aforementioned areas only. The whole map is acquired by relying on the robot's own odometry in order to track camera poses over time.

Abstract

We introduce SkiMap++, an extension to the recently proposed SkiMap mapping framework for robot navigation [1]. The extension deals with enriching the map with semantic information concerning the presence in the environment of certain objects that may be usefully recognized by the robot, e.g. for the sake of grasping them. More precisely, the map can accommodate information about the spatial locations of certain 3D object features, as determined by matching the visual features extracted from the incoming frames through a random forest learned off-line from a set of object models. Thereby, evidence about the presence of object features is gathered from multiple vantage points alongside with the standard geometric mapping task, so to enable recognizing the objects and estimating their 6 DOF poses. As a result, SkiMap++ can reconstruct the geometry of large scale environments as well as localize some relevant objects therein (Fig.1) in real-time on CPU. As an additional contribution, we present an RGB-D dataset featuring ground-truth camera and object poses, which may be deployed by researchers interested in pursuing SLAM alongside with object recognition, a topic often referred to as Semantic SLAM. [1]

1. Introduction and related works

Autonomous robots rely on suitable mapping modules to navigate within an unknown space. Different kinds of mapping approaches have been proposed in literature in order to fulfill different requirements and best trade between richness/accuracy of the representation and computational complexity. For example, a simple 2D occupancy grid allows for efficiently planning a path through a large environment but might turn out much less effective than richer representations (such as a 2.5 height map or a 3D voxel grid) in helping the robot to avoid different types of obstacles. To tackle some of the research issues associated with map-

[1] https://vision.disi.unibo.it/research/110-skimap-pp

ping for robot navigation, we have recently introduced the SkiMap framework [1], which turns out both time and memory efficient thanks to a novel core data structure organized as a *Tree of SkipLists* and, peculiarly, features a multi-level querying system capable to obtain rapidly representations as diverse as a 3D voxel grid, a 2.5D height map and a 2D occupancy grid.

In this paper we are concerned with extending SkiMap to enrich the representation to the level of semantics, i.e. so as to go beyond pure geometric mapping and incorporate information to enable detection of certain objects as well as estimation of their 6 DOF poses in the world space. This novel type of semantic mapping might be useful, e.g., to support an autonomous robot that would navigate within an unknown environment while seeking for certain objects that, if found, should be picked.

In literature, many works tackled the problem of 3D object detection and pose estimation from RGB-D images. Such techniques can be split between those relying on a single view to perform detection [2, 3, 4], and those deploying multiple images from several vantage points to ascertain whether an object is located in the observed scene and compute its pose. Multi-view object detection pipelines have been presented by Thomas *et al.* [5], who track feature points across views to determine the location and pose of the objects of interest jointly. Collet and Srinivasa [6] propose to handle each view independently and then perform a global refinement. Civera *et al.* [7] detect object instances in a sequence of images by means of SURF correspondences [8] and insert such objects into a map refined by a SLAM (Simultaneous Localization and Mapping) algorithm. This work introduced the idea of integrating the object detection into a SLAM pipeline to increase resilience of object localization with respect to the partial occlusions occurring in a single view: by reconstructing a consistent model of the 3D scene, and performing the detection therein, one can robustly identify the instances of interest by exploiting -possibly partial- evidence accumulated over time.

Object detection and 6DOF pose estimation from 3D data may be achieved by detecting 3D keypoints, then computing and matching 3D descriptors between the current scene and a set of 3D models [9, 10, 11, 12]. A different approach is due to Lai *et al.* [13, 14], who project per-pixel object probabilities from RGB-D frames onto voxels to obtain a semantic labeling of the scene, though in this work object poses are not estimated.

Alternatively, one can augment a SLAM pipeline to account for the task of object instance detection: [15, 16] were among the first papers to propose leveraging on recognized object instances as a means to improve the consistency of the SLAM process and vice-versa. Tateno *et al.* [17] propose to rely on a framework that simultaneously deploys a SLAM algorithm (used to obtain a reconstruction of the scene), a segmentation algorithm and an object recognition algorithm, so as to match descriptors to such segments to provide accurate and stable 6DOF poses for the objects of interest. Li *et al.* [18] also rely on the idea of synergistic exploitation of SLAM reconstructions for object detection, in order to improve scene understanding. This task is accomplished by fusing object hypotheses from single frames (possibly depicting partially-occluded instances) into a Global Semantic Map, as introduced in [19].

This paper follows the above-mentioned line of work: we propose a novel framework, referred to as SkiMap++, which allows for simultaneous recognition of objects and reconstruction of the environment as explored by a mobile agent. By accumulating evidence for objects into an extensible, real-time, mapping system, SkiMap++ can detect the presence of objects of interest in a 3D reconstruction of the scene and estimate their 6DOF pose.

2. Overview

The SkiMap++ framework is based on the SkiMap structure, which we introduced in [1] to realize efficient real-time mapping of large scale environments. Among the key features of this proposal are:

- Suited for large scale environments, thanks to a low memory footprint.
- Fast random voxel access $O(\log n)$.
- Equipped with several components implementing the Fusion/Erosion technique from [20], so as to optimize the map on-line alongside with reconstruction.
- Ability to perform radius-based search with better performance than *Octree* [21] and *Kd-Tree* [22].

The flexibility of the SkiMap data structure allow us to adopt it in SkiMap++ pipeline in order to store not only the map of the explored workspace but also the kind of data instrumental to the Object Recognition task, *i.e.*, in our proposal, *2D Features*, *Object Hypotheses* and *Guessed Instances*.

These additional data-types need to be queried and updated in real-time constraints and at the same time they need to be stored in a map as large as the mapped environment. Thus, SkiMap++ relies on continuous update of these heterogeneous maps and schedules queries on them so as to speculate on 6-DOF Objects Poses.

In the next two Sections we describe first how to train our system with *N* generic target objects (*Offline Pipeline*); then how the system employs the outcome from this training procedure to perform object recognition and pose estimation during real-time operation (*Online Pipeline*).

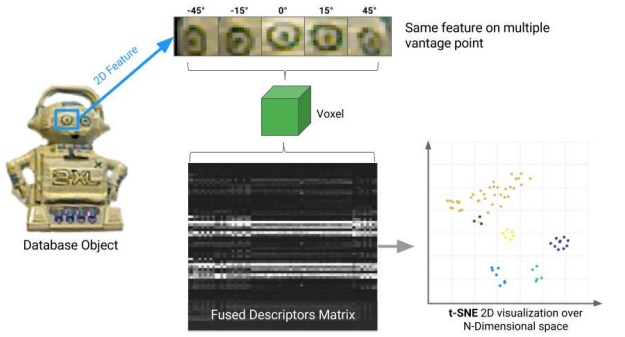

Figure 2. Each object feature looks differently depending on the vantage point. Experimental results show that fusing together, inside the same voxel, multiple descriptors computed from different viewpoints yields a *Descriptor Matrix* representing a multi-modal distribution in the descriptors space $\mathbb{R}^n$. A 2D visualization of descriptors obtained by *t-SNE* [23] highlights how these different descriptors tend to concentrate into a few clusters.

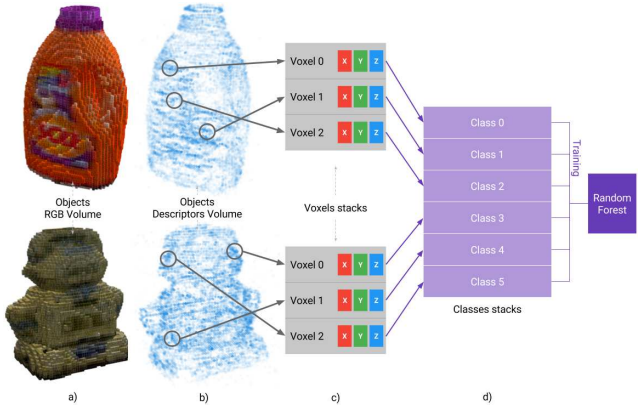

Figure 3. The stacking procedure used in SkiMap++ to create the Object Dataset and train the associated Classifier which can then be used on-line to perform object recognition. Column *a)* shows the reconstructed RGB Volumes of two objects. Column *b)* depicts the Descriptors Voxels Volume containing descriptions of multiple appearances as in Fig. 2. Column *c)* shows equally sized voxels stacks, ordered by cardinality, for each object. Finally, in *d)*, voxels stacks are merged into a global Classes Stack that will represent the prediction *target* for the forest training process.

3. Offline pipeline

The SkiMap++ object recognition approach is based on detection of *sparse 2D features*, that are then matched against a pre-trained Object Database, to achieve full 6-DOF pose estimation of target objects alongside with reconstruction of the environment. The matching component is built upon a Random Forest Classification system able to predict, given a 2D feature descriptor, to which object the feature belongs together with the coordinates of the voxel – in the object reference frame – in which the said feature was found during the training phase.

Brachmann *et al.* [24] investigated the Decision Forest approaches to predict, from a given feature, the object class and its position in the model reference frame. Formally:

$$p(c|d) \qquad p(\mathbf{y}|c) \tag{1}$$

namely the decision forest predicts the class $c \in C$ – given a feature $d \in \mathbb{R}^n$, the n-dimensional feature space – as well as the probability of object point $\mathbf{y}$ – in the object coordinate system – given class c. Indeed, the prediction $p(\mathbf{y}|c)$ in the second step is achieved by storing, at training time, all feature positions in the leaf of the decision trees, filling a multi-modal distribution in $\mathbb{R}^3$ discretized over a $5 \times 5 \times 5$ fixed grid. Furthermore, by adopting a *dense feature* approach, the technique introduced in [24] can predict during the online phase, the eligible object class and its internal point given a generic *pixel* (input images are densely described).

In SkiMap++ we exploited a similar approach, but focused on the analysis of the multi-modal distribution in the descriptor space $\mathbb{R}^n$, a distribution that grows during the database acquisition of each region of the target object model. As can be seen in Fig. 2, if we observe a target object from different vantage points, the same keypoint (*e.g.* the one belonging to the *eye* of the robot) is likely to show different appearances depending on the point of view. Multiple appearances in the *descriptor* space yield different descriptor vectors $d \in \mathbb{R}^n$ modeled as a *Mixture of Multi-Variate distributions* over $\mathbb{R}^n$. In the experiment outlined in Fig. 2 we use SURF features [8] to detect and describe keypoints, resulting in a $\mathbb{R}^{64}$ descriptor space. As shown in in Fig. 2, given an object feature we employed the *t-SNE* technique *et al.* [23] to analyze the distribution of descriptors dealing with different vantage points and found that these tend to from a small number of clusters.

We deploy a classifier trained to predict from keypoint descriptors the 3D voxel – in the object coordinate system – wherein the 3D feature originating is likely to falls within. To achieve this, we exploit SkiMap also during the database creation phase, so to build multiple voxel maps for each object: one fuses RGB data (to obtain an user-friendly object representation) while another one fuses together descriptors in such a way that each voxel stores a *Descriptor Matrix* (we will use interchangeably the terms *Descriptor Voxel* and *Descriptor Matrix Voxel*). Fig. 6 illustrates this dualism between RGB voxels and the corresponding Descriptor Voxels. We denote a *class* $c \in C$ for each Descriptor Voxel, determining a mapping function that from c allows us to easily compute the corresponding point in the object coordinate frame. We then train a Random Decision Forest to predict, given a target 2D feature, the object class as well as the voxel containing it:

$$p(c|d) \qquad v(c) = \mathbf{v}_j \tag{2}$$

we replace the second part of Eq. (1) with a deterministic function to compute the exact voxel given a predicted class c. To avoid confusion we need to define the difference between a predicted *class* and the *labels* used in the semantic labeling procedure: the *label* is $l \in \{0, 1, ...m-1\}$ where m is the number of training objects; a *class*, instead, predicts both the object as well as the voxel therein, $c \in \{0, 1, ..., m*k-1\}$ where k is the size of subset of Descriptor Voxels choosen among all object's voxels.

As the number of Descriptor Voxels could be very large we need to choose k carefully: in our system we choose to stack the Descriptor Voxels of each object ordered by their cardinality, then keeping those containing more cues. Fig. 3 shows a sample object database (containing only two models) to clarify the process to convert each voxel into a class and vice-versa:

$$v(c) = \mathbf{v}_j \qquad o(\mathbf{v_j}) = o_i \tag{3}$$

with o_i the index of the object containing voxel $\mathbf{v_j}$. Interestingly, this process might be thought of as a Local-to-Global indexing conversion. Having obtained a set of classes, each of which originated by a $d \in \mathbb{R}^n$ vector, we can train a decision forest classifier according to the standard procedure described in [25]. Without any loss of information, we reduce the memory footprint of Descriptor Voxels by computing the means of the relative mixture of gaussian (*e.g.* centroids in $\mathbb{R}^n$ of each eligible cluster, as highlighted in the *t-SNE* representation shown in Fig.2). As the number of clusters is not known a-priori, we adopt the *Mean Shift* [26] clustering approach.

It is worth pointing out that the proposed SkiMap++ framework is detector-agnostic. In fact, in our implementation the adopted 2D feature detector-descriptor is just a parameter of the system, as the number of *classes* per object. In Section. 5 we show some results while varying these parameters.

3.1. Built-in Model Database Compression

In object recognition pipelines based on 2D feature detection, the recognition rates turn out quite negatively affected by discrepancies between off-line and on-line conditions such as different light sources. So to strengthen the algorithm, we deemed useful to provide more evidence into the database: in our experiments, for example, we evaluated the performance and memory footprint of whole system while adding more amd more evidence. Intuitively, in SkiMap++ more visual cues only leads to increasing cardinality of each voxel in the Descriptor Volume, without increasing the memory footprint (shown in Fig. 4) – thanks to *Mean Shift* clusters analysis – at the same time increasing the amount of training informations fed to the random forest.

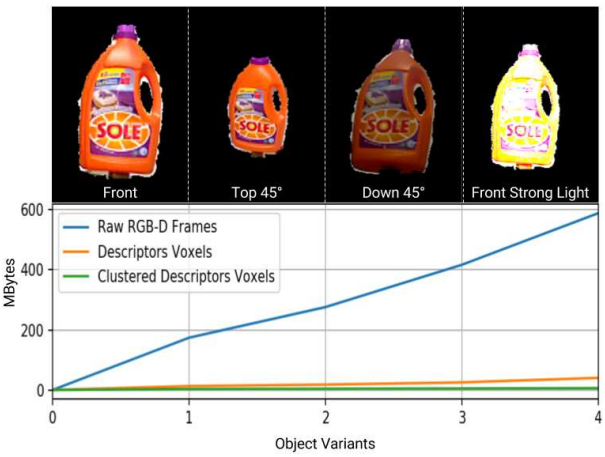

Figure 4. For an object in the dataset many variants may be acquired. In the figure each variant is intended as a full rotation around the object with the RGB-D camera in different conditions, *e.g.* in this figure from the *Front* or from the *Top* with an angle of $45°$, and so on. Each variant enriches the object description by filling Descriptor Voxels with additional evidences. Clustering the descriptors ensures the further decrease of memory footprint compared to the usage of all descriptors computed from the raw RGB-D frames. Storing the clustered representation of Descriptors is necessary to verify Classifier prediction (*i.e.* compute Euclidean distance between input descriptor and the centroids of the clusters stored in the predicted *Descriptor Voxel*, and verify that it is under a certain threshold).

4. Online pipeline

In this section we examine the SkiMap++ pipeline from acquisition of a the new frame to the final object instance recognition and pose-estimation. As depicted in Fig.5, the work-flow is subdivided into three macro blocks each of which affecting a different 3-dimensional SkiMap volume. In the next subsections each block will be explained in detail. To summarize: the input data for the SkiMap++ recognition pipeline is a generic pair *(RGB-D frame, camera pose)*, regardless of how they are generated; RGB-D data are integrated into a *SkiMap* RGB Volume according to the associated pose in order to reconstruct the environment. Simultaneously, sparse *2D features* are extracted from the current frame and processed by the *Random Forest*, built with the procedure described in Section. 3, in order to predict *Labels* which will be fused into an associated semantic SkiMap Volume: Fig. 6 illustrated the dualism between RGB and Labels volumes. For each *Camera Pose* an *Active Sphere* can be derived, so as to outline a local area of interest whose object hypotheses will in turn be fused into a further SkiMap. Finally we can perform a 3D query on the last Hypotheses Map to retrieve final object instances, resulting from the aggregation of Hypotheses in the neighbourhood of the queried point.

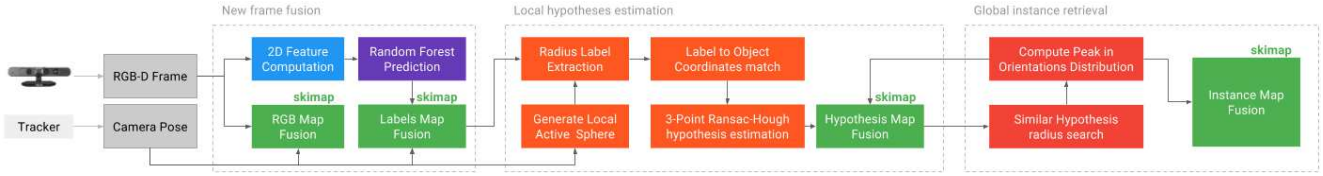

Figure 5. SkiMap++ online pipeline. First, new frames are integrated in two separate maps, one for RGB data and the other for labels. A *local active sphere* is generated according to the current camera pose, this sphere is used to query the label map to obtain objects matches and compute local hypotheses. Such hypotheses will be fused into another map. Given a target region inside the Hypotheses Map, the last phase of the pipeline entails the identification of the hypothesis with highest score, performing a radius search of similar hypotheses and merging them together to estimate and refine the final 6-DOF pose. Refined hypotheses shall be considered as object *Instances* and will be fused again in a global instances map.

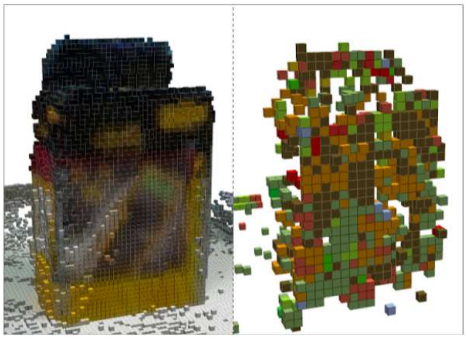

Figure 6. The left part depicts a portion of environment reconstructed through SkiMap++, fusing RGB-D Frames captured from multiple vantage points. On the right, the corresponding semantic map obtained by fusing *Labels* is shown instead. For visualisation purposes, voxels in the latter representation are coloured depending on object to which they belong. In this case, *brown* voxels are those belonging to the object being considered.

4.1. Frame Integration Module

The first stage of SkiMap++ is the *Frame Integration* block: this sub-component of the system is responsible of two main, independent, mapping tasks.

The first task is to build the RGB Volume, just as in the original SkiMap approach [1]. Accordingly, colour information is fused into a *RGBWeightedVoxel* data structure implementing weighted *sum/subtraction* operations. As mentioned in our previous work [1], this is a peculiar trait of SkiMap, which allows the system to *integrate* new sensor measurements or *de-integrate* past data marked as invalid.

The second, and more important in this paper, task is the computation of a semantic map, which leverages again on the SkiMap data structure. This semantic map is fed with *Labels* predicted trough the Random Forest described in Section 3, using as input data the 2D Sparse Features detected in the current frame. Fig. 6 illustrates a portion of the semantic map in which voxels are coloured according to the predominant *Label* within. In this case, in fact, unlike a *RGBWeightedVoxel* that could be merged by mixing colours, labels cannot be naively aggregated because of their categorical nature. Thus, the semantic map adopts as base element a *MultiLabelVoxel* implementing the *sum/subtraction* operations, as follows:

$$V_i = \{\langle l, w_i \rangle : l \in L, w_i \in \mathbb{Z}_0^+\} \tag{4}$$

$$V_1 \pm V_2 = V_3 \rightarrow V_3 = \{\langle l, w_1 \pm w_2 \rangle\} \tag{5}$$

Eq. 4 describes a generic *MultiLabelVoxel* as a *MultiSet*, namely a *Set* with repetitions, of classes $l \in L$ with cardinality w_i (*i.e.* repetition counter). This notation is necessary to allow a Label to be fused into a Voxel repeatedly, without losing the cardinality information; furthermore we allow two Voxels to be merged together through a lossless procedure (Eq. 5). Such kind of Voxel shall be equipped with a method to retrieve the maximal ordered pair: $\langle l, w_i \rangle$ in order to fetch the largest Label and its weight during the matching phase.

The main purpose of the subsystem described above is to provide a – searchable – semantic map through which we can retrieve all labels belonging to a given object. In the next section we will detail how to speculate on object 6-DOF pose hypotheses starting from a local map of labels.

4.2. Local Hypotheses Estimation Module

The second module of SkiMap++ depends on the *Camera Pose* from which it derives an *Active Sphere*. In particular, the goal is to compute a reference frame ${}^{M}T_S$ (**M**ap to **S**phere) relative to the camera frame ${}^{M}T_C$ (**M**ap to **C**amera) falling into its frustum:

$${}^{M}T_S = {}^{M}T_C \cdot {}^{C}T_{t_z} \tag{6}$$

where ${}^{C}T_{t_z}$ is a translation along the z-axis. Such translation could be fixed (for example in our experiments to $1m$) or could be computed by analysing scene conditions (*e.g.* by computing the nearest point in depth image). This global reference frame ${}^{M}T_S$ is used to perform a radius search on the *Labels Map*, allowing the system to build a *per-frame Local Space* on which to perform Hypotheses Estimation, rather than attempt this task on the whole map, thus ensuring bounded time complexity for subsequent stages of the process. The outcome of a radius search on the Labels Map is a *set* of Voxels of type *MultiLabelVoxels* as seen in Eq. 4. SkiMap allows, during a search, to enrich *User Data* stored

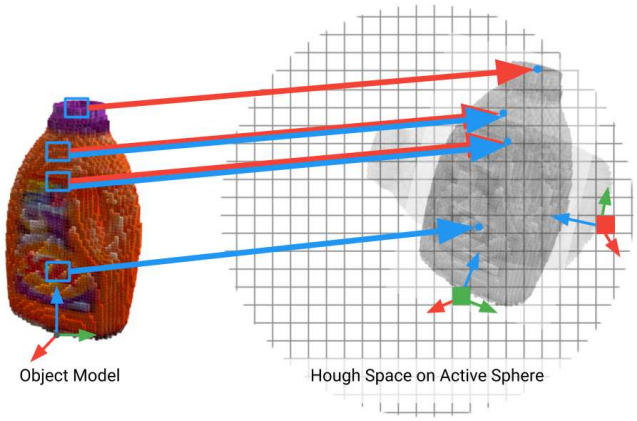

Figure 7. The right part of the image depicts the *Active Sphere* on which we build the 3D Hough space. For each three random correspondences we can project an Object Reference Frame ${}^{Obj}T$ inside the sphere computing the relative Reference Frame ${}^{H}T_{Obj}$ in the Hough coordinate space and cast votes for the object base in the relative bin: in the figure the first match (blue arrows) and the second match (red arrows) project the object base in the same bin (green square). The other bin (red box) represents an hypotheses brought in by a false positive. Had the three hypotheses voted for their own centroid instead of the base, the associated bin would have accounted a distorted number of votes taking into account as inliers the relatively rotated matches, coming from the false positive hypotheses.

in Voxels with their coordinates, by obtaining a subset:

$$H = \bigcup_{C_i \subset C} H_i = \{(c_i, \mathbf{p}) : c_i \in C_i\} \tag{7}$$

where H is the set of pairs $(c, \mathbf{p})$ containing a predicted class c and a point $\mathbf{p}$ representing the center of the associated voxel. As it can be seen from Eq. 7, the set H can be split in many similar subsets, one for each object, with C_i containing only the predicted classes from the *i-th* object. For each pair $(c, \mathbf{p})$, we can apply Eq. 3 to retrieve:

$$o(v(c)) \rightarrow (o_i, \mathbf{v}_j) \tag{8}$$

generating the new pair $(o_i, \mathbf{v}_j)$, where $\mathbf{v}_j \in \mathbb{R}^3$ represents the center of the *j*-th voxel of the *i*-th object. By iterating this approach, we estimate a set of 3D matches $\{(c, \mathbf{p}), (o_i, \mathbf{v}_j)\} \rightarrow \{(\mathbf{p}, \mathbf{v}_j)\}$ suited to 6-DOF pose estimation of the target object within the Local Space.

We employ a fast and robust technique to estimate 6-DOF pose of the objects under occlusion and clutter. This problem was addressed also by Tombari *et al.* [27] with promising results in comparison with other competitors involved in 3D free-form object recognition problem. The most important feature in this approach is to reduce the Hough-problem complexity from 6 to 3 dimensions, discarding the rotational part of pose estimation implicitly covered by the voting process. Unfortunately the scheme in [27] requires the estimation of a Local Reference Frame per each feature, since each feature should cast a vote independently from others. The computation of a Local RF is a computationally onerous procedure, not suited to real-time application, but at the same time the approach has been proven valid. We thus built a new, hybrid, approach drawing inspiration also from the work of Hollander *et al.* [28]. This approach relies on the 3-Point Ransac/Hough-voting scheme described as follows: given $M_i = \{(\mathbf{p}, \mathbf{v}_j)\}$, the set of matches involving object i in the current Local Space, we pick m random subsets $R_{M_i} \subset M_i$ with $|R_{M_i}| = 3$. Each set of such 3 points $\mathbf{p} \in R_{M_i}$ outlines a Reference Frame through which we can project an instance of object i in the Local Space (i.e. in the **M**ap reference frame) ${}^{M}T_{Obj}$ and cast a vote for its centroid (or any rigid point attached to it, for example its base) in the associated bin of a *3D Hough Space* built right on the *Active Sphere*. To increase the performance of this voting scheme, differently from the approach in [27], we do not vote for the centroid of the projection of the object in the scene, but for its base, thus ensuring that every false positive with inferred centroid near the real object pose will score far from the correct bin. In Fig.7 the voting scheme procedure is depicted graphically.

As mentioned above, the final aim of the *Local Hypotheses Estimation* component is to fuse hypotheses in a new map, so as to store spatial information about these guesses. The base unit of this new SkiMap structure is a *HypothesesVoxel* consisting of a simple set $V = \{(\mathbf{t}_h, \mathbf{R}_h, o_i)\}$ of tuples where $\mathbf{t}_h$ represents the position of the hypothesis in map reference frame, $\mathbf{R}_h$ its rotation in the same space and o_i the identifier of the object being considered. More precisely, a 6-DOF Hypothesis should be stored at least in a $\mathbb{R}^6$ space, but following the logical approach of [27] we store them in a $\mathbb{R}^3$ space via their translational component, preserving each orientation $\mathbf{R}_h$ as is, to deploy them in a further refinement step.

4.3. Global Instance Retrieval Module

The last module of SkiMap++ is mainly dedicated to refining the hypotheses coming out from the previous stage. Starting from the ${}^{M}T_S = ({}^{M}\mathbf{R}_S, {}^{M}\mathbf{p}_S)$ *Active Sphere* reference frame, we can perform again a neighborhood search on *Hypotheses Map* based on a target object o_i:

$$r({}^{M}\mathbf{p}_S, o_i) = \{(\mathbf{t}_{h_k}, \mathbf{R}_{h_k}, o_i)\}, k \in [0, m) \tag{9}$$

where the result of the search operation $r(\cdot)$ is a set of k hypotheses in the neighborhood of the point ${}^{M}\mathbf{p}_S$. Since this resulting subset may contain some outliers, we cannot simply average the outcome in a $\mathbb{R}^6$ space. Therefore, once again, we can adopt a statistical approach analyzing the orientation distribution of these hypotheses in the $SO(3)$

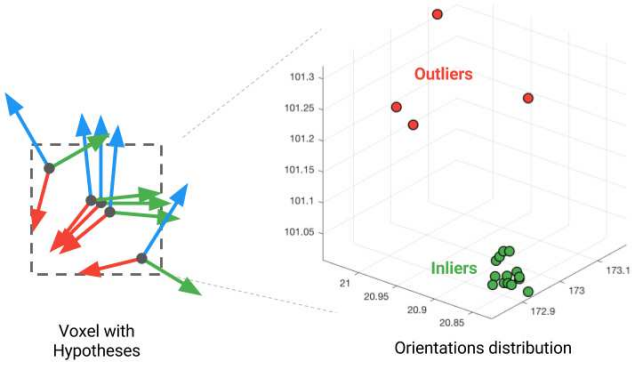

Figure 8. On the left, a sample of a classical *Hypotheses Voxel* containing many computed Reference Frames. On the right, the distribution of their orientations represented with *axix angle* notation in a $SO(3)$ group. With high probability inliers will be grouped together in a cluster, the centroid of which can be inferred as the best candidate for the final resulting orientation of the instance.

group of the *axis angle* representation of them (Fig.8 clarifies this issue). To infer the final *Instance* $(\mathbf{t}_I, \mathbf{R}_I, o_i)$ of the object o_i in the scene we can simply average positions $\mathbf{t}_I = \frac{1}{m}\sum_{k=1}^{m} \mathbf{t}_{h_k}$ and choose as orientation $\mathbf{R}_I$ the centroid of the largest cluster in the orientation space.

Finally, also the found *Instances* will be integrated into a further SkiMap structure in units called *InstancesVoxels*, a simple extension of *HypothesesVoxels*. For clarity reasons Fig.9 shows graphically, by means of a real frame captured from SkiMap++ execution, the difference between *Hypotheses* and *Instances*.

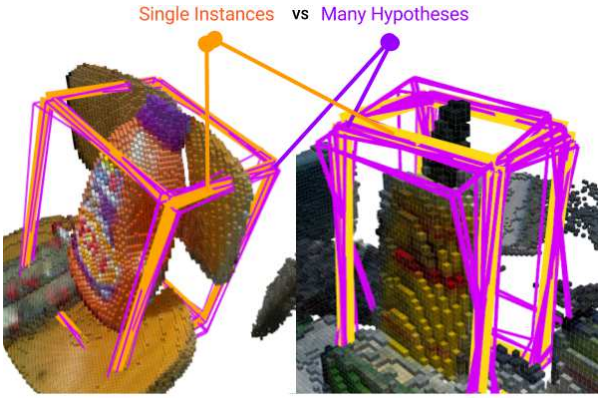

Figure 9. This figure portrays a frame taken from a real-time scan of one scene of the dataset. RGB Reconstruction is carried out with two different resolutions: $0.005m$ for left object and $0.01m$ for right object. *Hypotheses* (purple boxes) are contained with *HypothesesVoxels* of $0.03m$ instead. Finally the *Instances* (orange boxes) are grouped in $0.05m$ *InstancesVoxels*.

5. Experimental results

In this section we first describe the dataset used in our experiments. Next we show some quantitative results obtained from the aforementioned dataset while varying some key parameters involved in SkiMap++ pipeline. Finally we present some qualitative evaluation of the algorithm.

5.1. Dataset SK17

To test the whole SkiMap++ framework we have tried to find suitable public dataset without success. The same type of search performed by Li*et al.* [18] ended up in the same conclusion. The main reasons are the absence of 6-DOF ground truth of objects or the low frame density. Also the proposed dataset in the latter study is not still fully available to be used with our approach (missing RGB-D frames for objects). For this reason we developed a brand new dataset comprising 12 Objects and 5 Medium/Large Scale Scenes. The Camera, an Asus Xtion, is tracked by the Vicon Motion Capture System *(Vicon Motion Systems, LA, USA)* by ensuring a precision of the order of magnitude of *1cm*. The 6-DOF ground truth of the objects in each scene was manually labelled in a global reference frame centered on the floor of each scene (Precision of ground truth has the same resolution of the smallest RGB map attainable by the SkiMap approach (*i.e.* about $0.005m$).

5.2. Quantitative results

In this subsection we examine the performance of SkiMap++ in terms of precision/recall computed on all the 5 scenes of the dataset by taking into account all the objects instances present within the environment and by counting a *True Positive* when we do find the instance with a maximum translational error under $0.03m$ and maximum rotational error of $10°$ in comparison with their ground truth. Fig.10 shows the accuracy of the system by varying some key parameter. As observed from the first plot, by increasing the number of *classes* trained per-object (*i.e.* the number of Descriptors Voxels taken into account to grow decision trees) we can achieve higher performance, but what also stands out is that after a certain number of classes the accuracy starts to decrease. Intuitively this is due to the under-fitting problem in the Random Forest that would need to grow in size to learn more labels to be classified. The second plot shows a simple comparison between 2D Features that reflects their intrinsic characteristics. The last plot addresses the inverse problem seen in the first graph: the curves show that keeping the detector fixed and increasing the size of the Random Forest (*e.g.* 24x24 means a forest of 24 trees with depth 24) we can slightly increase accuracy at the expense of prediction time. Overall, Fig.10 shows a precision/recall index over 80/80 in the best position within parameters space. Moreover Fig.11 highlights the key feature of this Multi-View Object Recognition approach: the higher is

Table 1. Here a complete list of all objects in our dataset. The table shows a comparison between two best sets of parameters, tuned during our tests, varying 2D Descriptor. Percentages represent the recall after a complete round within every scene.

	Boxgreen	Boxmouse	Boxred	Chamo	Cupgreen	Cupred	Detergent	Glue	Korn	Multimeter	Robot	Talc	Fps
SIFT	**33%**	**100%**	35%	**100%**	**75%**	**75%**	**100%**	**100%**	**100%**	**100%**	**78%**	**75%**	6
SURF64	0%	50%	**87%**	**100%**	0%	0%	**100%**	**100%**	**100%**	**100%**	75%	**75%**	**20**

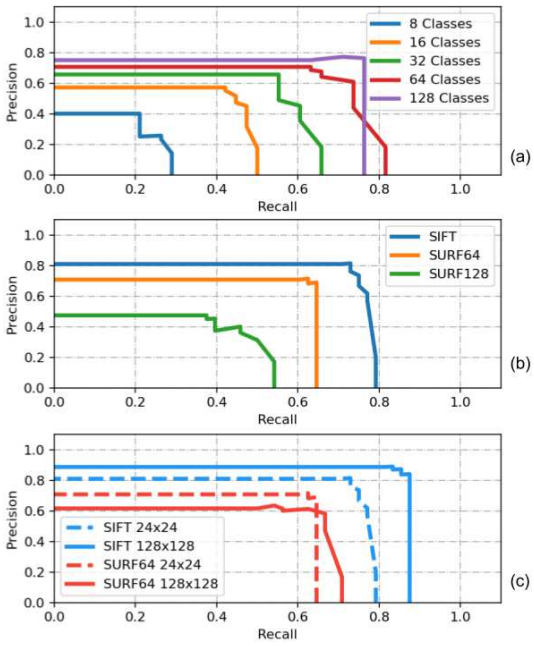

Figure 10. These plots show precision/recall while varying some key parameters like: (a) Number of trained classes per object; (b) differences between 2D Feature detectors; (c) size of the random forest while using the same descriptor. The overall result shown here is over 80.0/80.0 precision/recall index.

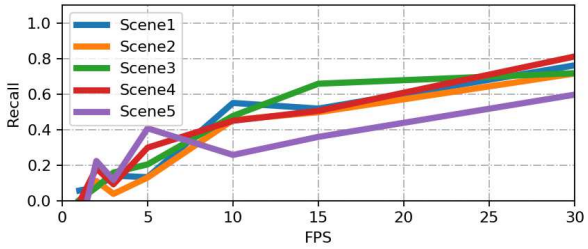

Figure 11. This plot shows the importance of Multi-View based Object Recognition. Decreasing the number of vantage points(*i.e.* decreasing fps and so the number of frames) the accuracy falls down to zero.

the density of frames (and therefore the number of vantage points), the larger is the accuracy of the system. The latter result is justified by a series of important factor, such as: an high frame rate helps to merge an higher number of evidences; many vantage points can be lost if the frame rate is low when camera movements are fast and without allowing for the fact that the multi-view approach is the only solution for occlusion. Finally, Table1 lists an overview on accuracy based on single objects among all scenes; it should be noted that the score of the object *Boxgreen* is very low because it

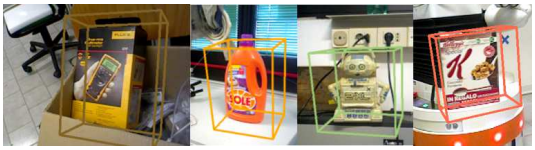

Figure 12. Real-time Augmented Reality enabled by stable 6-DOF pose.

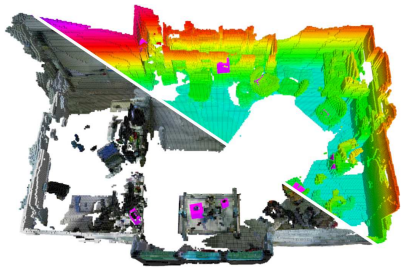

Figure 13. Large scale map (dataset scene) fully reconstructed by SkiMap++ with Object Instances identified by purple bounding boxes.

is the only object with an overall dimension comparable to Voxels Resolutions (about *5cm* on the longest side), which renders pose estimation very challenging.

5.3. Qualitative results

We provide here also some qualitative results to show that SkiMap++ is suited to real settings. Fig.1 shows a large scale environment fully reconstructed by means of a mobile robot based on its odometry system. In this environment we placed some objects, taken from our dataset, and the robot successfully found them while mapping the workspace. Other qualitative samples are shown in Fig.13,12: the latter illustrates also the interesting Augmented Reality attainable thanks to the high stability of the object hypotheses in the global reference frame yielded by SkiMap++. More qualitative results are available in the supplementary material, which shows the system running in real-time. [1]

6. Concluding remarks

We have described an extension of our previous work [1] aimed at enriching the map of the workspace with semantic information related to the presence and 6 DOF poses of object instances. Thanks to the inherent efficiency and versatility of the SkiMap data structure, we plan to pursue further extensions towards richer semantic perception, in particular by endowing the map also with labels related to major object categories (e.g. floor, walls, table, chair..). This will allow SkiMap++ to efficiently reconstruct in real-time both the geometry and the semantic of large scale environments.

References

[1] D. D. Gregorio and L. D. Stefano, "Skimap: An efficient mapping framework for robot navigation," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*, May 2017, pp. 2569–2576.

[2] P. Wohlhart and V. Lepetit, "Learning descriptors for object recognition and 3D pose estimation," *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, no. 1, pp. 3109–3118, 2015.

[3] Chi Li, J. Bohren, E. Carlson, and G. D. Hager, "Hierarchical semantic parsing for object pose estimation in densely cluttered scenes," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, may 2016, pp. 5068–5075.

[4] E. Brachmann, F. Michel, A. Krull, M. Y. Yang, S. Gumhold, and C. Rother, "Uncertainty-Driven 6D Pose Estimation of Objects and Scenes from a Single RGB Image," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, jun 2016, pp. 3364–3372.

[5] A. Thomas, V. Ferrar, B. Leibe, T. Tuytelaars, B. Schiel, and L. Van Gool, "Towards Multi-View Object Class Detection," in *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition - Volume 2 (CVPR'06)*, vol. 2. IEEE, 2006, pp. 1589–1596.

[6] A. Collet and S. S. Srinivasa, "Efficient multi-view object recognition and full pose estimation," in *2010 IEEE International Conference on Robotics and Automation*, vol. 2, no. d. IEEE, may 2010, pp. 2050–2055.

[7] J. Civera, D. Galvez-Lopez, L. Riazuelo, J. D. Tardos, and J. M. M. Montiel, "Towards semantic SLAM using a monocular camera," in *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, sep 2011, pp. 1277–1284.

[8] H. Bay, T. Tuytelaars, and L. Van Gool, "Surf: Speeded up robust features," in *European conference on computer vision*. Springer, 2006, pp. 404–417.

[9] R. B. Rusu, G. Bradski, R. Thibaux, and J. Hsu, "Fast 3D recognition and pose using the Viewpoint Feature Histogram," in *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, oct 2010, pp. 2155–2162.

[10] S. Salti, F. Tombari, and L. Di Stefano, "SHOT: Unique signatures of histograms for surface and texture description," *Computer Vision and Image Understanding*, vol. 125, no. PART 3, pp. 251–264, aug 2014.

[11] A. Aldoma, F. Tombari, R. B. Rusu, and M. Vincze, "OUR-CVFH: Oriented, Unique and Repeatable Clustered Viewpoint Feature Histogram for Object Recognition and 6DOF Pose Estimation," *Pattern Recognition*, vol. 7476, pp. 113–122, 2012.

[12] A. Aldoma, F. Tombari, J. Prankl, A. Richtsfeld, L. Di Stefano, and M. Vincze, "Multimodal cue integration through Hypotheses Verification for RGB-D object recognition and 6DOF pose estimation," *Proceedings - IEEE International Conference on Robotics and Automation*, pp. 2104–2111, 2013.

[13] K. Lai, L. Bo, X. Ren, and D. Fox, "Detection-based object labeling in 3D scenes," in *2012 IEEE International Conference on Robotics and Automation*. IEEE, may 2012, pp. 1330–1337.

[14] K. Lai, L. Bo, and D. Fox, "Unsupervised feature learning for 3D scene labeling," in *2014 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, may 2014, pp. 3050–3057.

[15] R. F. Salas-Moreno, R. a. Newcombe, H. Strasdat, P. H. Kelly, and A. J. Davison, "SLAM++: Simultaneous Localisation and Mapping at the Level of Objects," *2013 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1352–1359, jun 2013.

[16] N. Fioraio and L. Di Stefano, "Joint detection, tracking and mapping by semantic bundle adjustment," *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 1538–1545, jun 2013.

[17] K. Tateno, F. Tombari, and N. Navab, "When 2.5D is not enough: Simultaneous reconstruction, segmentation and recognition on dense SLAM," *Proceedings - IEEE International Conference on Robotics and Automation*, pp. 2295–2302, 2016.

[18] C. Li, H. Xiao, K. Tateno, F. Tombari, N. Navab, and G. D. Hager, "Incremental scene understanding on dense SLAM," in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, oct 2016, pp. 574–581.

[19] K. Tateno, F. Tombari, and N. Navab, "Real-time and scalable incremental segmentation on dense SLAM," *IEEE International Conference on Intelligent Robots and Systems*, pp. 4465–4472, 2015.

[20] N. Fioraio, J. Taylor, A. Fitzgibbon, L. D. Stefano, and S. Izadi, "Large-scale and drift-free surface reconstruction using online subvolume registration," in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015, pp. 4475–4483.

[21] D. J. Meagher, "Geometric modeling using octree encoding," in *Computer Graphics and Image Processing*, 1982, pp. 129–147.

[22] J. L. Bentley, "Multidimensional binary search trees used for associative searching," *Communications of the ACM*, vol. 18, no. 9, pp. 509–517, 1975.

[23] L. v. d. Maaten and G. Hinton, "Visualizing data using t-sne," *Journal of Machine Learning Research*, vol. 9, no. Nov, pp. 2579–2605, 2008.

[24] E. Brachmann, A. Krull, F. Michel, S. Gumhold, J. Shotton, and C. Rother, *Learning 6D Object Pose Estimation Using 3D Object Coordinates*. Cham: Springer International Publishing, 2014, pp. 536–551. [Online]. Available: http://dx.doi.org/10.1007/978-3-319-10605-2_35

[25] A. Criminisi and J. Shotton, *Decision forests for computer vision and medical image analysis*. Springer Science & Business Media, 2013.

[26] Y. Cheng, "Mean shift, mode seeking, and clustering," *IEEE transactions on pattern analysis and machine intelligence*, vol. 17, no. 8, pp. 790–799, 1995.

[27] F. Tombari and L. Di Stefano, "Hough voting for 3d object recognition under occlusion and clutter," *IPSJ Transactions on Computer Vision and Applications*, vol. 4, pp. 20–29, 2012.

[28] R. J. Den Hollander and A. Hanjalic, "A combined ransac-hough transform algorithm for fundamental matrix estimation," in *BMVC*, 2007, pp. 1–10.