Enhancing Semantic Segmentation with Detection Priors and Iterated Graph Cuts for Robotics

Morris Antonello[a,*], Sabrina Chiesurin[b,], Stefano Ghidoni[a,]

[a] *Intelligent Autonomous Systems Laboratory (IAS-Lab), Department of Information Engineering (DEI), University of Padova, Via Ognissanti 72, 35129, Padova, Italy*
[b] *School of Mathematical and Computer Sciences, Heriot Watt University, Edinburgh Campus, Boundary Rd N, EH14 4AS, Edinburgh, Scotland, United Kingdom*

Abstract

To foster human-robot interaction, autonomous robots need to understand the environment in which they operate. In this context, one of the main challenges is semantic segmentation, together with the recognition of important objects, which can aid robots during exploration, as well as when planning new actions and interacting with the environment. In this study, we extend a multi-view semantic segmentation system based on 3D Entangled Forests (3DEF) by integrating and refining two object detectors, Mask R-CNN and You Only Look Once (YOLO), with Bayesian fusion and iterated graph cuts. The new system takes the best of its components, successfully exploiting both 2D and 3D data. Our experiments show that our approach is competitive with the state-of-the-art and leads to accurate semantic segmentations.

Keywords: Semantic Scene Understanding, Object Detection, Segmentation and Categorization, Mapping

1. Introduction

Semantic segmentation is the task of decomposing a scene into its meaningful parts. It received great attention in recent years within the research community because of its importance in scene understanding, robotics and

---

[*] Corresponding author

Email addresses: `morris.antonello@dei.unipd.it` (Morris Antonello), `sc186@hw.ac.uk` (Sabrina Chiesurin), `stefano.ghidoni@dei.unipd.it` (Stefano Ghidoni)

Preprint submitted to Engineering Applications of Artificial Intelligence March 31, 2020

autonomous vehicles [1, 2, 3]. In general, this task is non-trivial given the high level of variability in the world and the limits of vision sensors; however, when dealing with moving robots, the same scene can be framed multiple times from different locations, which can make the task easier. In [4, 5, 6, 7], visual recognition techniques, which are usually applied to a single view at a time, are combined with a Simultaneous Localization and Mapping (SLAM) algorithm, which incrementally builds a global map. This allows to find correspondences between multiple views, which can be exploited to improve the semantic segmentation. Both single-view and multi-view problems have received attention in different contexts and at different scales: indoor and outdoor scenes, scaling up to entire cities [8]. Semantic segmentation can be the sensory input fed to systems reasoning about contents and their representation in the domain of natural language [9]. These systems can learn about the inter-modal correspondences between language and visual data so that they can describe the content of images, e.g. by means of rich and descriptive captions. Also, semantic segmentation can help robots and autonomous cars in a variety of tasks, including object detection and picking [10] and autonomous navigation [11].

Prior work includes many approaches, based both on plain 2D RGB data [12, 4] and RGB-D (or 3D) data [13, 7, 3]. In this work, we contribute to the problem of segmenting objects, humans and coarse scene elements, e.g. walls, floor and ceiling, on RGB-D data, showing that some components of the proposed system can be used also when only RGB data is available. Our approach can be successfully used in the context of service robotics [14, 15], including applications like social companion and health care: the proposed system can enhance navigation, planning and interaction thanks to an improved perception. Industrial applications can also be positively impacted by the proposed methods. In [16], semantic segmentation is proposed to detect the key elements involved in production and automatically sand boat components. Since high reliability is required to perform challenging manufacturing operations, all sources of information, in particular multiple views and contextual cues, are exploited.

Another interesting application of the proposed system is the automatic annotation of datasets [17]. Indeed, real products, that must satisfy accuracy and safety requirements, need huge labeled datasets if based on data-driven methods. Making the annotation process faster and less expensive is of utmost importance.

In this work, we build upon a setting consisting of a single-view semantic segmentation method for indoor scenes called 3D Entangled Forest classifier (3DEF), previously presented in [13], and a multi-view frame fusion scheme, previously presented in [18] and in [16] for industrial applications.

3DEF is a 3D semantic segmentation approach which works on single camera views of indoor environments and relies on an extension of the Random Forest. Given a single-view image, this approach is able to model its complex contextual features in a single pass in about one second. The semantic segmentation problem is tackled in two stages. First, the scene is over-segmented in such a way that each segment contains at most one object. Being an over-segmentation, objects can be split in many segments. Second, the semantic label of each segment is inferred by means of the 3DEF classifier. In particular, the classification of each segment depends on learned geometric relations of neighbouring segments. Finding correspondences between multiple views can further enhance the semantic segmentation thanks to the various vantage points, namely the good observations points.

Despite the good results with coarse scene elements, e.g. walls, floor and ceiling, this approach often struggle when dealing with objects: semantic segmentation does not rely on any high-level prior, but focuses on local geometry and texture. In this context, object detection can be seen as a complementary approach: it is based on strong priors about a given set of objects that need to be recognized in a scene. This leads object detectors to accurately detect and localize such objects, neglecting all the background, that is, the main part of an image. In this work, we study how to exploit both approaches, extending a state-of-the-art object detector with iterated graph cuts [19, 20] to output accurate segmentation masks and then using Bayesian fusion to combine such segmentations with 3DEF and the multi-view frame fusion scheme. While many approaches have been developed over the last years, we focus on Mask R-CNN [21] and You Only Look Once (YOLO) [22, 23, 24]. Mask R-CNN is a deep neural network used to detect objects in images while generating a segmentation mask for each object detected. YOLO is also a deep neural network but it does not generate any segmentation mask. In contrast to prior works, these methods do not need object proposals to reduce the search space; rather, they apply a neural network to the full image so predictions are informed by global image context. These methods are fast: they process images in real-time with a GPU acceleration and, using the lightest models, they run in a few seconds per image on a CPU. Even with limited computational resources, they can be successfully used to refine lighter and less precise methods if executed asynchronously alongside them.

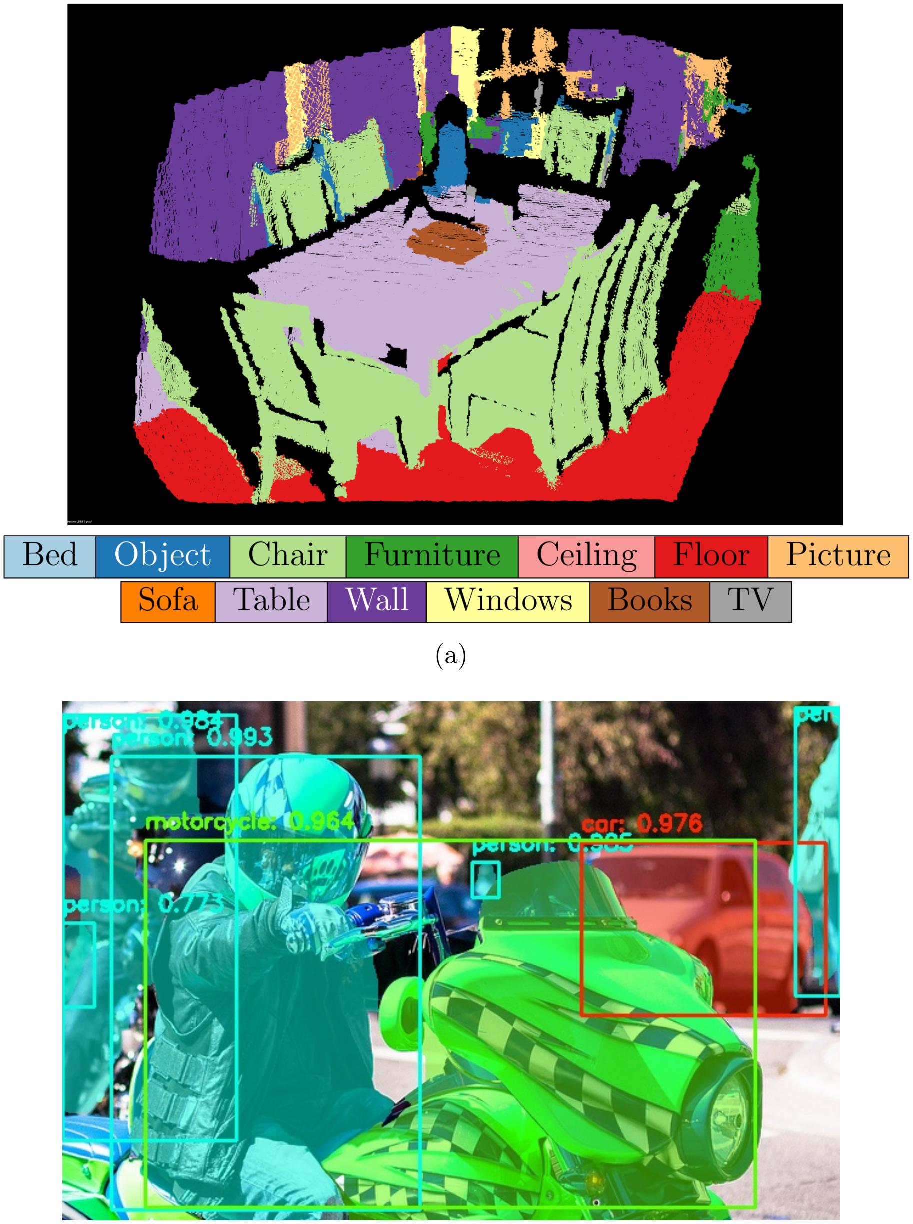
An example of the final result achieved by the proposed system is reported in Figure 1: (a) shows a dining room annotated pixel per pixel, (b) shows an outdoor scene with refined segmentation masks for each object.

The main contributions of this paper are:

- the introduction of an object detector into our multi-view semantic segmentation pipeline, in order to deal with complex objects as well as coarse scene elements like walls;
- the Bayesian approach for incorporating the top-down cues of an object detector into the bottom-up semantic segmentation process, which achieves a good balance between the two systems;
- the extension of state-of-the-art object detector like Mask R-CNN and YOLO with graph cut optimization for accurate object detection and contour segmentation.

Our novel approach proved to be competitive with respect to the state-of-the-art. It can handle the multiple, sometimes overlapping, bounding boxes and segmentation masks returned by the object detector. Furthermore, it takes advantage of the confidences provided by the detection and semantic segmentation systems to consider the best of the two predictions. The 3D multi-view frame fusion technique further refines the semantic segmentation.

The remainder of the paper is organized as follows. Section 2 overviews the state-of-the-art in object detection, single-view semantic segmentation and multi-view semantic segmentation. Section 3 introduces both the single-view and multi-view approach for semantic segmentation. Special attention is paid to the description of the process of creating accurate segmentation using the detection priors and iterated graph cuts. Then, the fusion of Mask R-CNN and You Only Look Once Detector (YOLO) with the 3D Entangled Forests (3DEF) is also described in depth. In Section 4, our methods are thoroughly evaluated on the NYU Depth Dataset V2 [2]. Further tests are performed on the Microsoft Common Objects in COntext (MS COCO) dataset [25] showing that the 2D component of our method can be useful even for computer vision applications lacking 3D data, both indoor and outdoor. Finally, in Section 5, our achievements are recapped and future directions of research identified.



(a)

(b)

Figure 1: Example of (a) multi-view semantic segmentation with object priors obtained on the NYU dataset and (b) refined segmentation masks obtained on the COCO dataset.

2. Related Work

Nowadays, Deep Neural Networks (DNNs) are boosting many fields. Convolutional Neural Networks (CNNs) already revolutionized semantic segmentation. One of the early attempts belongs to Couprie *et al.* [3, 26], who proposed a multiscale CNN architecture to combine information at different perceptive field resolutions. They were among the first to train a CNN with depth information for this task. Later, many other approaches have been proposed [7, 12, 27, 28, 29, 30]. The work by L. P. Tchapmi *et al.* [28] proposes a deep neural network called SEGCloud able to work with point clouds, instead of regular 3D voxel grids or collections of images. The method combines the advantages of neural networks, trilinear interpolation and fully connected Conditional Random Fields to enforce global consistency. For robotic or mobile applications, for which computational power is often constrained, the trade-off between speed and accuracy have been further explored [31, 13, 32]. To reduce the computational power required, other non CNN-based approaches also exist in this scenario, like the two works by D. Wolf *et al.* [31, 13]. Interestingly, in [13], D. Wolf *et al.* outperform [31] introducing the 3D Entangled Forest, an extension to the standard Random Forest. This classifier is able to model complex contextual features in one single pass in less than one second per frame on a standard CPU, without relying on complex graphical models, random fields or other post-processings as e.g. in [33]. In this work, the capabilities of this approach are further explored. First, it is coupled with an object detector. Then, to get the best out of the two methods, Bayesian fusion and a refinement step working in 3D are proposed.

In applications with moving robots, recognition techniques can be enhanced by observing the environment from several points of view. This problem is a particular instance of semantic mapping, described in [34] as the problem of identifying and recording the signs and the symbols that contain meaningful concepts for humans. These can be coarse scene elements [35], objects [35, 36, 37, 38, 39], places [40, 37] and other elements of interest [41]. In the literature, the creation of such representation is tackled at different scales, indoor and outdoor, and using a reference system that can be either local, (e.g. with respect to the sensor), or global. In this work we focus on multi-view semantic segmentations of indoor scenes in the camera reference system. Solutions to this problem have been proposed by J. Stückler *et al.* [42], A. Hermans *et al.* [4] and J. McCormac *et al.* [5]. They differ because of the adopted registration system and semantic segmentation method. For registration, they use a Multi-Resolution Surfel Map-based SLAM, a camera tracking system without explicit loop closure and Elastic Fusion [5], respectively. For semantic segmentation, they use random decision forests, a combination of random decision forests and conditional random fields, and a CNN, respectively. They all adopt a Bayesian framework for combining the multiple views. In [43], a new method for incrementally building a dense, semantically annotated 3D map in real-time is studied. It assigns class probabilities to each region, not each element, of the 3D map, which is built up through a robust SLAM framework and incrementally segmented with a geometric-based segmentation method. Alternative multi-view approaches incorporating multi-view information into state-of-the art convolutional networks have been proposed in [44, 45, 46]. Another multi-view frame fusion scheme was introduced by Antonello *et al.* [18]. This method is tested with a light SLAM algorithm like RGB-D SLAM [47], which finds the correspondences between the views. The multi-view semantic fusion considers the neighbourhood of each point and adds a geometrical verification step, useful for improving the semantic segmentation of the single-frames. Wrong contributions due to lens distortions or alignment errors are filtered out. In this work, this method is further studied. With respect to the previous work, the single-view contributions are enhanced by detection priors refined with iterated graph cuts. As discussed in [48], the lack of a uniform representation, as well as standard benchmarking suites, prevents the direct comparison of many semantic mapping algorithms. Here, since our focus is more the classification task, we cast the problem as multi-view semantic segmentation and, as in [4, 5, 43], evaluate each single frame after taking into account the multiple points of view.

In the past, the most successful approaches to object detection utilized a sliding window paradigm, in which a computationally efficient classifier tests for object presence in every candidate image window [49, 50, 51]. The steady increase in complexity of the classifiers has led to improved detection quality, but at the cost of significantly increased computation time per window. Thus, in order to reduce the search space, many top performing object detectors [52, 53, 54] work on detection proposals [55, 56], i.e. only a small subset of all the possible windows. Two in-depth reviews can be found in [57, 58]. In contrast to prior works, the state-of-the-art family of object detectors known as You Only Look Once (YOLO) [22, 23] does not need object proposals and applies a single neural network to the full image, so its predictions are informed by global context in the image. This network divides the image into regions and predicts bounding boxes and related detection probabilities for each region. These bounding boxes are weighted by the predicted probabilities. Such methods are fast: they process images in real-time with GPU acceleration and, using a lighter model, they run on a CPU at a few seconds per image. In recent years, object detectors capable of generating a high-quality segmentation mask for each instance have been proposed, e.g. Mask R-CNN [21]. Mask R-CNN extends Faster R-CNN by adding a branch for predicting an object mask in parallel with the existing branch for bounding box recognition. Given an image as input, Mask R-CNN generates proposals about the regions where there might be an object and predicts its class. Based on the proposal, it then generates a mask of the object. The boxes and masks returned by these methods can be coarse and benefit from a further refinement. In the literature, there exists methods for segmenting foreground and background given some initial hints, e.g. boxes, incomplete segmentation masks [19, 20] and extreme points [59]. In this work, we prefer boxes and segmentation masks over extreme points, i.e. left-most, right-most, top, bottom pixels, to better cope with imperfect boxes and mask. In addition to refining the detected objects in the multiple, likely overlapping, priors, we also study how to combine these priors with a multi-view semantic segmentation system.

3. Methods

Our approach tackles the fusion of a bottom-up semantic segmentation with top-down object detection priors and the preliminary refinement of the object detector priors. The semantic segmentation and object detection approaches are fused with the aim of leveraging the best of the two algorithms, which have different properties as they assume different prior knowledge about the observed scene, and they are based on 3D data (semantic segmentation) and 2D data (object detection). Such a combination needs to handle multiple, likely overlapping, object priors returned by the detector. This will be achieved by integrating the object priors in the right order, fusing the two contributions in a Bayesian way and smoothing the results in 3D. For improved results, the object detection priors are refined before fusion. The obtained single-view semantic segmentation is further improved by means of our multi-view fusion scheme. An overview of both the single-view and multi-view algorithms is reported in Figure 2. The existing setting

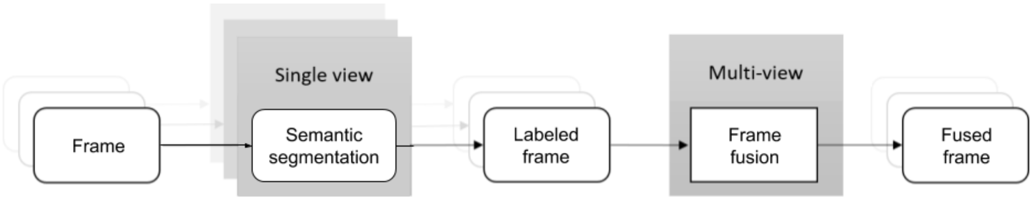


Figure 2: Overview of the proposed approach. The single view approach can be 3DEF or our combination of 3DEF with an object detector, Mask R-CNN or YOLO. The multi-view frame fusion technique is based on the multiple frame fusion scheme introduced in [18]. The number of frames can be configured. Here, for visualization purposes, just three frames are visualized.

is presented from Subsection 3.1 to 3.3. Our contributions are thoroughly discussed in Subsection 3.4.

3.1. 3D Entangled Forest Classifier

The 3DEF approach in [13] operates on 3D point clouds, which can be acquired with an RGB-D sensor. The approach comprehends three phases:

- supervoxel over-segmentation in 3D patches;
- fusion of similar adjacent segments into larger, mostly planar segments;
- segment classification.

The input point cloud is over-segmented into homogeneous 3D patches by means of the Voxel Cloud Connectivity Segmentation (VCCS) [60]. This solution aims at preserving the edges by finding patches not crossing object boundaries and, at the same time, it reduces the noise and the amount of data. This is a region growing method which incrementally expands patches, in particular supervoxels, i.e. volumetric over-segmentations of 3D point cloud data, from a set of seed points distributed evenly in space on a grid of fixed resolution R_{seed}. Expansion from the seed points is governed by a distance measure D calculated in a feature space consisting of spatial extent, color, and normals:

$$D = \sqrt{w_c D_c^2 + \frac{w_s D_s^2}{3R_{seed}^2} + w_n D_n^2}\,,$$

in which the spatial distance D_s is normalized by the seeding resolution, the color distance D_c is the euclidean distance in normalized RGB space, and the normal distance D_n measures the angle between surface normal vectors. Three weights can be controlled by the user: w_c, w_s and w_n. This method was proved to be more effective than existing 2D solutions.

In the subsequent step, this approach applies a region growing algorithm, which recursively merges two adjacent segments c_i and c_j into larger ones. The underlying idea is that bigger segments are better since the classifier features tend to be more reliable. This merging step is performed evaluating a distance function $d(c_i, c_j)$. In particular, given a threshold τ_{merge}, the constraint $d(c_i, c_j) < \tau_{merge}$ must hold. This distance function is a linear combination of the color, surface normal and point-to-plane distance between the segments:

$$d(c_i, c_j) = w_c d_c(c_i, c_j) + w_n d_n(c_i, c_j) + w_p d_p(c_i, c_j)\,,$$

in which d_c is the color distance in Lab CIE 94 color space, d_n the surface normal difference indicated by the dot product $(1 - n_i n_j^T)$, d_p is the max of the point-to-plane distance from c_i to c_j and viceversa. The user can control three weights: w_c, w_n and w_p, normalized to sum up to 1. The algorithm stops if there are no more adjacent segments to be merged and returns the final set of segments $\mathcal{S}$.

For each segment generated by the over-segmentation, a feature vector x of length 18 is calculated. Besides simple color features, it includes fast geometric features. Some of them are calculated from the eigenvalues of the scatter matrix of the segment, which represent the variance magnitudes in the main directions of the spread of the segment points. Others are calculated from the Oriented Bounding Box (OBB) including all the segment points. A complete list of features is given in Table 1. Then, for each segment s_t, a set of close-by-segments s_i is selected on the basis of three constraints: point-to-plane distance, enclosed angles and Euclidean distance. During training and inference, this set can be used to evaluate five binary tests defining the entangled features, which are capable of describing complex geometrical relationship between segments in a neighbourhood. A complete list is given in Table 2. They are briefly explained as follows:

- *Existing Segment Feature*: this evaluates to true if the set of close-by-segments s_i is nonempty;

Table 1: List of unary features calculated for each 3D segment and their dimensionality.

Unary features	Dimensionality
Color mean and std. dev.	2
Compactness (λ_0)	1
Planarity ($\lambda_1 - \lambda_0$)	1
Linearity ($\lambda_2 - \lambda_1$)	1
Angle with floor (mean and std. dev.)	2
Height (top and bottom point)	2
OBB dimensions	3
OBB face areas	3
OBB elongations	3
Total dimensionality	18

Table 2: List of entangled features calculated for each 3D segment and their dimensionality.

Entangled features	Dimensionality
Existing segment	4
TopN segment	6
Inverse TopN segment	6
Node descendant	5
Common ancestor	5
Total dimensionality	26

- *TopN Segment Feature* and *Inverse TopN Segment Feature*: these features take into account the class label distributions of the current tree nodes, which the candidate segments s_i have reached so far during classification. Two parameters are learned: a label l and the bound N. In particular, they evaluate to true if a certain label l is among the most frequent N labels;

- *Node Descendant Feature* and *Common Ancestor Feature*: these features consider the path a target segment s_t or candidate segment s_i took through the tree during classification. Two parameters are learned: a label l and the bound M. They evaluate to true if a certain label l is

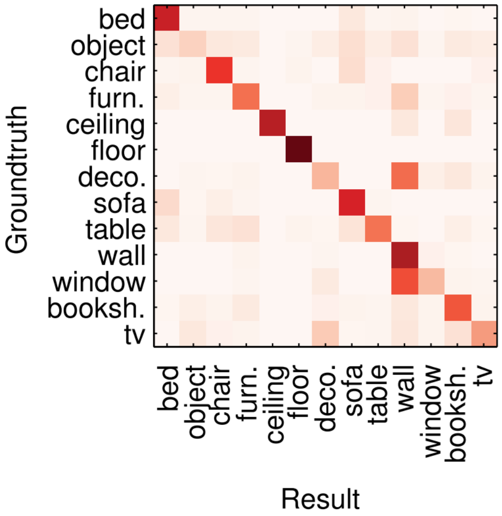


Figure 3: Confusion matrix of 3DEF on the NYUv2 dataset. Two challenging classes are the labels *Object* and *Furniture*, which comprehend many different objects of different sizes and shapes. The main confusion values appear between *Wall/Wall Decoration*, *Wall/Wall Window* and *Wall Decoration/TV*.

encountered within M steps.

For further details, we refer to [31]. In our tests, we stuck to the original parameters for the sake of comparison.

The shortcomings of the 3DEF classifier can only be mitigated by the availability of multiple points of view, as found out in [18]. To quantitatively analyze its main weaknesses, we calculated its confusion matrix on the NYUv2 dataset, see Figure 3. Two challenging classes are the generic labels *Object* and *Furniture*, which comprehend many different objects of different sizes and shapes making it hard for a classifier to capture any distinct properties. Also, the class *Chair* is often confused with the class *Sofa*. Finally, the classes *TV*, *Decoration* and *Window* are challenging since they all are objects located/mounted on walls so their segmentation can rely mainly on color cues. Given that a multi-view method can only slightly improve over these underlying issues, we further studied how to combine the strengths of 3DEF with those of a state-of-the-art object detector. A semantic segmentation approach like 3DEF can accurately segment many coarse scene elements and relatively big objects like *Floor*, *Ceiling*, *Wall*, *Bed*, *Sofa*, *Chair* or *Book-*

shelves. Instead, an object detector like Mask R-CNN or YOLO is trained to detect a variety of objects with clear boundaries.

3.2. Multi-view Frame Fusion Scheme

The multi-view frame fusion scheme presented in [18] operates on sequences of RGB-D frames, which may be acquired during normal robot operations (consider, for example, a typical patrolling task). These frames may overlap and contain different views of the same entity (object or scene element) from different angles and distances. This module is composed of three steps which can potentially run in parallel: the 3D reconstruction step, the semantic segmentation step and the multi-view frame fusion step. The 3D reconstruction step, here based on RGB-D SLAM [47], takes a new frame from a sequence of RGB-D frames and registers it to the 3D reconstruction returning its rigid transformation with respect to the reference frame. The semantic segmentation step can be the original 3DEF approach applied to each frame or our combination of 3DEF with Mask R-CNN or YOLO. The multi-view frame fusion step, which is the focus of this section, fuses together the semantic information for each point in order to exploit the availability of multiple points of view.

Given a sequence S of RGB-D frames I_i with i varying from 1 to N, a reference frame I_{ref} can be selected, e.g. with $\text{ref} = N/2$. Every 3D point P^{xy}, where x and y are the coordinates in the image reference system, belonging to it can be forward-projected to all the other frames in S. This way, the optimal label of each point P^{xy} can be estimated after considering all the contributions from all the N points of view. Figure 4 shows that the optimal label of $P^{xy}_{N/2}$ can be selected after considering also the contributions from forward-projected points FP^{xy}_i in the frames I_1 and I_N while Figure 5 shows that not always a forward projection exists so the contribution from some frames can be missing.

Anyway, due to lens distortions and SLAM errors like double walls or chairs, we cannot be sure that each point $P^{xy} \in I_{\text{ref}}$ truly coincides with the 3D points corresponding to each forward projection $\{FP^{xy}_i\}$. Hence, we introduced a geometrical validation step: each FP^{xy}_i is transformed to the reference coordinate system and can contribute only if:

$$\left| FP^{xy}_i.z - P^{xy}_{\text{ref}}.z \right| < \epsilon \,. \tag{1}$$

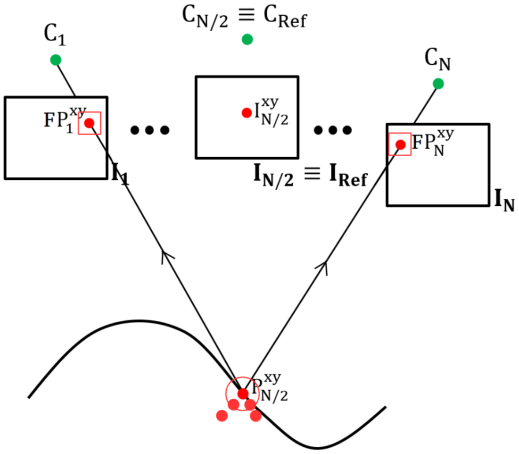


Figure 4: Forward projection from 3D to I_i, $i \neq$ ref. The red boxes around FP_1^{xy} and FP_N^{xy} denote the Moore neighbourhood. The red circle around $P_{N/2}^{xy}$ the geometric validation step: only the points side it can contribute.

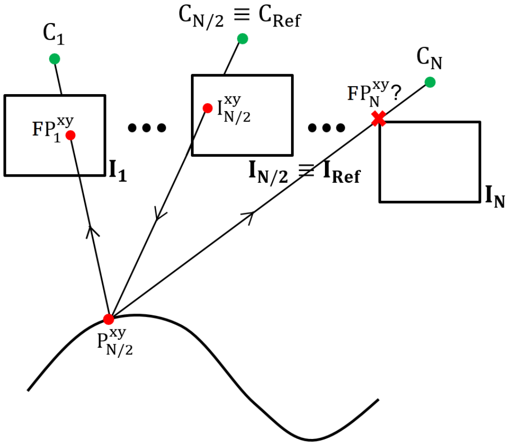


Figure 5: Example of missing forward projection.

A good ϵ proved to be 0.05 m since just the contributions of truly coinciding 3D points are of interest.

To consider the contributions from the other frames, an approach based on the Bayesian fusion at the pixel level is considered. Not only this method operates on labels but it takes in input also the classifier confidences. Given a point $P^{xy}_{\text{ref}} \in I_{\text{ref}}$ and the respective forward projected points $\{FP^{xy}_i\}$ with $i \in \{1, ..., N\} \bigwedge i \neq \text{ref}$, let j be a semantic label and $z^{\text{ref}} = \{z_1, ..., z_{\text{ref}}, ..., z_N\}$ its measurements in each frame I_i, i.e. the labels assigned to the point P^{xy}_{ref} (z_{ref}) and its forward-projections FP^{xy}_i (z_i, with $i \neq \text{ref}$). According to Bayes' rule:

$$p(j|z^{ref}) = \frac{p(z_{\text{ref}}|j, z^{\overline{\text{ref}}})p(j|z^{\overline{\text{ref}}})}{p(z_{\text{ref}}|z^{\overline{\text{ref}}})} \,,$$

where $z^{\overline{\text{ref}}} = z^{\text{ref}} \setminus \{z_{\text{ref}}\}$, i.e. the labels assigned to the forward-projections only. Under the assumptions of i.i.d. condition (independent and identically distributed condition) and equal a-priori probability for each class, it can be simplified to:

$$p(j|z^{\text{ref}}) = \tau_j \prod_i p(z_i|j) \,,$$

where τ_j is a normalization factor such that:

$$\sum_{j=1...N} \tau_j p(j|z^{\text{ref}}) = 1 \,.$$

In particular τ_j is calculated as:

$$\tau_j = \frac{1}{\sum_{k=1...N} p(k|z^{\text{ref}})} \,.$$

Parity cases are important and must be addressed appropriately. In the event of parity, the label from the reference frame is kept.

Finally, the forward projection is improved by means of a smoothing step. This step takes into account the pixel context so as to improve robustness with respect to errors in the forward projection process, which can be due to noise or locally imprecise registration. Each forward-projected point FP^{xy}_i does not contribute with its label only but with the most frequent label in its Moore neighbourhood, which comprehends itself and the eight neighbours, NP^{xy}_{ik} with $1 \leq k \leq 8$, see the red boxes enclosing them in Figure 4. Formally, let $d_{FP^{xy},j}$ denote whether the classifier selects the label j on point FP^{xy}_{ref} or not, and let $d_{NP^{xy}_{ik},j}$ denote whether the classifier selects the label j on point NP^{xy}_i or not. The majority label combination leads to the class J receiving the largest total vote:

$$d_{FP^{xy}_{\text{ref}},J} + \sum_{k\in 1\ldots 8 \bigwedge i\neq \text{ref}} d_{NP^{xy}_{ik},J} = \max_{j=1,\ldots,c}\left(d_{FP^{xy}_{\text{ref}},j} + \sum_{k\in 1\ldots 8 \bigwedge i\neq \text{ref}} d_{NP^{xy}_{i}k,j} \right) .$$

In addition, each forward-projected point does not contribute with its label confidences but with those of the neighbour pixel with the most frequent label J in the Moore neighbourhood. Nevertheless, without any geometrical verification step, this method could introduce noise in the labelling results. To be sure that each point in the 2D Moore neighbourhood is a real neighbour in 3D, only the points passing the geometrical verification step previously introduced in Equation 1 can contribute, in this case:

$$\left| NP^{xy}_{ij}.z - P^{xy}_{\text{ref}}.z \right| < \epsilon .$$

3.3. Object Detector

We selected two state-of-the-art real-time one-shot object detectors, Mask R-CNN [21] and You Only Look Once (YOLO) [22], more precisely the second version YOLOv2 [23].

Mask R-CNN generates bounding boxes and segmentation masks for each instance of an object in the image. Mask R-CNN extends Faster R-CNN [53] by adding a branch for predicting an object mask in parallel with the existing branch for bounding box recognition. Given an image as input, Mask R-CNN generates proposals about the regions where there might be an object and predicts its class. Based on the proposal, it then generates a mask of the object. The implementation used in this work [61] is based on Feature Pyramid Network (FPN) and a ResNet101 backbone. For a full description, we refer to [21].

In contrast to Mask R-CNN, YOLO generates only the bounding boxes. It feeds a single neural network with a full RGB frame so that its predictions can be informed by the global frame context. The network divides the image into regions and predicts bounding boxes and probabilities for each region. These bounding boxes are weighted by the predicted probabilities. The network architecture of the first version YOLOv1 is inspired by the GoogLeNet model [62] for image classification. The network has 24 convolutional layers followed by 2 fully connected layers. Instead of the inception modules

(a)

(b)

Figure 6: (a) Mask R-CNN finds a set of bounding boxes as well segmentation masks, for each of which a label and a confidence are associated (b) Similarly, YOLO finds a set of bounding boxes.

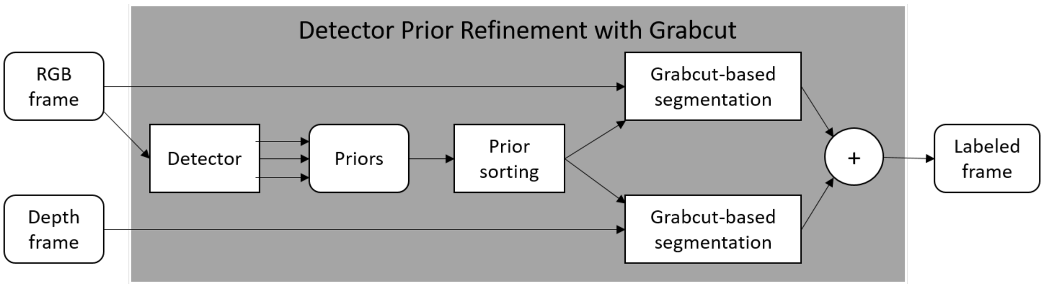
used by GoogLeNet, it uses 1 × 1 reduction layers followed by 3 × 3 convolutional layers, similar to Lin *et al.* [63]. The detection framework of YOLOv2 improves in speed and accuracy thanks to various design choices making it competitive with respect to region-based approaches like Faster R-CNN or Mask R-CNN. For a full description, we refer to [23].

For both detectors, we selected a model trained on the COCO detection dataset [25], containing over 200 000 images with 80 different object classes. The annotations of this dataset are accurate and the models learned from it can be reused in other contexts, as shown also in this work. These classes, which do not include coarse or large scene elements like *Wall*, *Ceiling* and *Floor*, can be easily mapped to the other classes of the semantic segmentation problem: most of the COCO classes simply falls in the *Object* class. For our tests, we considered the proposals with a high confidence threshold, greater than 0.5. The output of the detectors on two sample images is shown in Figure 6.

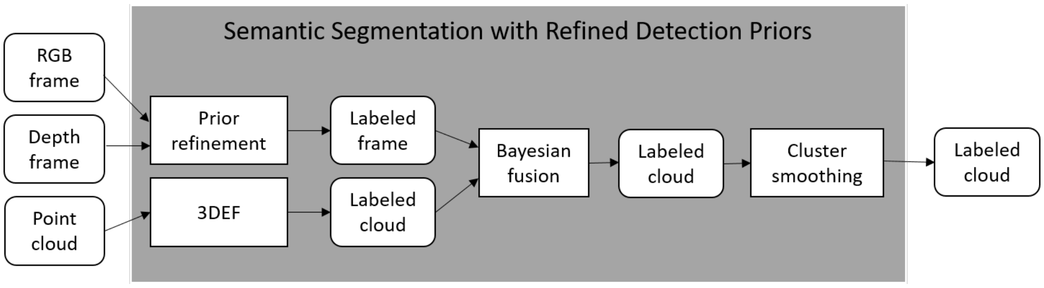
3.4. Object Detection and Semantic Segmentation Fusion

Two steps are required to integrate the detector into our semantic segmentation pipeline:

- refinement of the object detection priors with Grabcut;



(a)



(b)

Figure 7: (a) Overview of the algorithm performing semantic segmentation with an object detector. In this scheme, for ease of visualization, the detector generates only three priors. (b) Overview of the algorithm to combine 3DEF and an object detector. The Bayesian fusion leverages on the strengths of both methods. The cluster smoothing is a final refinement.

- fusion of the refined detection priors with the semantic segmentation.

The two steps are illustrated in Figure 7 and detailed as follows.

A straightforward implementation of the first step consists in labeling all the pixels in the detection prior, i.e. the segmentation mask returned by Mask R-CNN and the bounding box returned by YOLO. Instead, we further refine these priors with the approach illustrated in Figure 7(a) and formally described in Algorithm 1. The approach exploits both 2D and 3D data and handles overlapping priors. For each RGB frame, the detector proposes a set of detection priors associated with a label and a confidence. Given each detection prior, the detected object is segmented with a method based on Grabcut, a state-of-the-art unsupervised segmentation algorithm [19]. It can be initialized in three ways using:

Algorithm 1 Detector Prior Refinement with Grabcut

```
procedure REFINE_PRIORS($I_{RGB}$, $I_{depth}$)                ▷ Input images
    $priors \leftarrow$ DETECT($I_{RGB}$)                      ▷ Mask R-CNN or YOLO
    $sorted\_priors \leftarrow$ SORT($priors$)                 ▷ Decreasing size order
    $new\_priors \leftarrow \emptyset$
    for all $prior : prior \in sorted\_priors$ do
        $new\_prior_{RGB} \leftarrow$ REFINE($prior$, $I_{RGB}$)          ▷ Grabcut
        $new\_prior_{depth} \leftarrow$ REFINE($prior$, $I_{depth}$)      ▷ Grabcut
        $new\_prior \leftarrow new\_prior_{RGB} \vee new\_prior_{depth}$
        $new\_priors \leftarrow new\_priors \cup \{new\_prior\}$
    $I_{labeled} \leftarrow$ LABEL_IMAGE($new\_priors$)
    return $I_{labeled}$                        ▷ With objects classes and confidences
```

- a mask with pixels labeled as foreground, background, probable foreground and probable background;
- a bounding box around the foreground region;
- both the mask and bounding box.

For Mask R-CNN, we exploit the first option. The third option did not prove helpful since the bounding box is too coarse to help refining the mask. In particular, we set the border of the original mask as probable foreground, the inner area as foreground and the outer area as background. We determine the border thickness t as a fraction f of the radius r of a circle with perimeter p as long as the bounding box perimeter:

$$t = fr = f\frac{w+h}{\pi},$$

where f was set to 0.1 in our experiments, w is the bounding box width and h the bounding box height. For YOLO, we exploit the second option since YOLO does not provide any segmentation mask. This option corresponds to marking the outer area as background and the inner area as probable foreground. Given the labeled masks in input, Grabcut creates the background/foreground segmentation by solving a max-flow min-cut problem. A weighted graph is created based on the pixel neighbouring and the labeled masks. In particular, given the label α, the color z and some parameters θ describing foreground and background color distributions, the cost function

$E(\alpha, \theta, z)$, that Grabcut minimises with iterated graph cuts, is defined by a data term $U(\alpha, \theta, z)$ and a smoothness term $V(\alpha, z)$:

$$E(\alpha, \theta, z) = U(\alpha, \theta, z) + V(\alpha, z)\,.$$

The two terms describe how well the pixels fit the background/foreground color distributions and how smooth the labeling is over similar/a-similar neighboring pixels. The optimization is followed by border matting to deal with blur and mixed pixels along smooth object boundaries on which both Mask R-CNN and 3DEF struggle. For robustness, given that not always a segmentation can be found, Grabcut is run on both RGB and depth frames. This way, the segmentations obtained from RGB and depth frames can be fused using a pixel-per-pixel OR operation. We run the graph cut optimization for 5 iterations; if Grabcut cannot return any segmentation, we consider the initial object detection priors as foreground. This solution does not penalize labels like *Object* and *Book*, which can be characterized by tight bounding boxes. Then, a label and confidence is assigned to each pixel.

Since detection priors can overlap, the order with which the bounding boxes are processed may negatively impact the results. For instance, depending on the processing order of Grabcut, an object on a table may be segmented before the table itself, so the subsequent table segmentation may override the previous object segmentation, see examples in Figure 6. Because of this, a straightforward method running Grabcut on each bounding box is not ideal. Here, with a heuristic, detection priors are sorted in decreasing order of size. This way, bigger boxes are segmented before smaller ones. Indeed, big boxes might be supporting surfaces like tables while small boxes may contain objects lying on them. This component already improves the semantic segmentation of 3DEF.

Given that the detector does not support the detection of all the 13 classes (e.g. it cannot detect coarse scene elements like floor, walls and ceiling, because they do not have clear boundaries) the output it provides is incomplete and needs to be fused with a semantic segmentation approach. An overview of the fusion process is illustrated in Figure 7(b) and formally described in Algorithm 2. For each frame pixel, the predictions of 3DEF and of the detector are fused in a Bayesian way. The two contributions can be easily retrieved in 2D by iterating over the output of 3DEF and of our semantic segmentation method based on the detector. Indeed, both outputs are semantic images, encoding the most likely label and the probability distribution over the set

Algorithm 2 Semantic Segmentation with Refined Priors

```
procedure SEMANTIC_SEGMENTATION($cloud$, $I_{RGB}$, $I_{depth}$)   ▷ Input point cloud and images
    $I_{labeled}$ ← REFINE_PRIORS($I_{RGB}$, $I_{depth}$)   ▷ With confidences
    $cloud\_labeled$, $clusters$ ← 3DEF($cloud$)   ▷ With confidences
    $cloud\_labeled$ ← BAYESIAN_FUSION($I_{labeled}$, $cloud\_labeled$)
    $cloud\_labeled$ ← SMOOTH_CLUSTERS($cloud\_labeled$, $clusters$)
    return $cloud\_labeled$   ▷ Labeled point cloud
```

of labels. For simplicity, we assume that the two semantic segmentations are independent and identically distributed. This is reasonable since the detector and semantic segmentation rely on different features, 2D and 3D, therefore they have different strengths and weaknesses. Given a frame I and a frame pixel $P^{xy} \in I$, let j be its semantic label, z_{3DEF} the semantic label returned by 3DEF and z_{Det} the semantic label returned by the detector. According to Bayes' rule and under the assumption of i.i.d. condition, confidences can be accumulated as follows:

$$p(j|z_{\text{3DEF}} \wedge z_{\text{Det}}) = \tau_j p(z_{\text{3DEF}}|j) \times p(z_{\text{Det}}|j) \, ,$$

where $p(z_{\text{Det}})$ is the confidence returned by 3DEF, $p(z_{\text{Det}})$ is the confidence returned by the detector and τ_j is a normalization factor such that:

$$\sum_{j=1 \ldots N} \tau_j p(j|z_{\text{3DEF}} \wedge z_{\text{Det}}) = 1 \, .$$

The selected label J is the one with the highest probability:

$$J = \arg\max_j p(j|z_{\text{3DEF}} \wedge z_{\text{Det}}) \, .$$

Nevertheless, errors in the detector prior location or in the Grabcut-based segmentation may lead to the assignment of wrong labels and confidences to the pixels close to the object borders. To alleviate this, a subsequent cluster smoothing step is performed. In contrast with previous steps, this one exploits the point cloud, in particular the 3D preliminary segmentation based on the the Voxel Cloud Connectivity Segmentation (VCCS) [60] and the subsequent region growing, see Section 3.1. Given each unlabeled cluster C, which is the output of the preliminary segmentation phase in the 3DEF approach, the most frequent label of the points in C is considered. Each point in C is labelled consistently with the most voted label in the cluster. In the same way, the respective confidences are propagated inside the cluster to all the other points.

The performance of the presented methods will be extensively discussed in the following section.

4. Experiments

4.1. Datasets

We assessed the performance of our methods on the popular NYU Depth dataset NYUv2 [2] and further evaluated the detection refinement on the Microsoft Common Objects in COntext (MS COCO) dataset [25].

The NYUv2 dataset contains 1449 pixel-wise labeled RGB-D frames which are commonly split into a subset of 795 frames for training/validation and 654 for testing. It was recorded with a Kinect v1 sensor. In contrast to its predecessor NYUv1, the annotation quality is higher and it does not wrap the class *Object* in the class *Background*. In particular, we tested our methods on the 13-class semantic segmentation problem. The 13 classes include objects, furniture and coarse scene elements, e.g. walls, ceiling and floor.

MS COCO is a large-scale dataset object detection and segmentation dataset containing about $200k$ labeled RGB images. The object detection and segmentation problem considers 80 class labels of common objects in everyday scenes from all around the world. The dataset is split into a subset of $155k$ training images, $5k$ validation images and $40k$ test images. The labels of the test set are not public available and the evaluation is performed in a test server.

4.2. Experiments on NYUv2

Similarly to the other approaches evaluated on this dataset, we used two performance indicators: pixelwise recall (in the following: Global Accuracy – GA) and classwise recall (in the following: Class Accuracy – CA). In addition, we also reported a third performance indicator, the classwise precision (in the following: Class Precision – CP), useful to further compare the variants of our methods. Considering a label set with n class labels and based on the elements of the confusion matrix (true positives tp, false positives fp and false negatives fn), the metrics are defined as follows. GA is calculated as the overall portion of correctly labeled points:

Table 3: Evaluation of the fusion of 3DEF with Mask R-CNN and YOLO on the NYUv2. The methods are reported in increasing order of class-wise accuracy CA. The best result are in bold. Integrating an object detector always improves over the baseline 3DEF. 3DEF+YOLO+Grabcut performs slightly better than 3DEF+Mask R-CNN. Using the depth image improves Grabcut segmentations.

Method	CA	GA	CP
3DEF [13]	55.7	65.0	53.3
3DEF+YOLO+Grabcut (rgb)	60.9	67.4	56.0
3DEF+Mask R-CNN+Grabcut (rgb)	61.2	67.3	56.1
3DEF+Mask R-CNN+Grabcut (rgb and depth)	61.2	67.3	56.2
3DEF+Mask R-CNN	61.2	67.4	56.2
3DEF+YOLO+Grabcut (rgb and depth)	**61.3**	**67.6**	**56.3**

$$GA = \frac{\sum_{i=1}^{n} tp_i}{\sum_{i=1}^{n}(tp_i + fn_i)} .$$

CA is the average class recall:

$$CA = \frac{1}{n}\frac{\sum_{i=1}^{n} tp_i}{\sum_{i=1}^{n}(tp_i + fn_i)} .$$

CP is the average class precision:

$$CP = \frac{1}{n}\frac{\sum_{i=1}^{n} tp_i}{\sum_{i=1}^{n}(tp_i + fp_i)} .$$

The last two indicators are less biased towards frequent classes. In the following, we will analyze the different combinations of 3DEF and object detector, the multi-view contribution and how our best approaches do in comparison with other state-of-the-art approaches.

We compared different ways to integrate 3DEF with Mask R-CNN and YOLO. Table 3 shows that integrating an object detector always improves over the baseline 3DEF, up to +5.6% in CA, +2.6% in GA and +2.0% in CP. 3DEF+YOLO+Grabcut performs slightly better than 3DEF+Mask R-CNN. Indeed, even if Mask R-CNN segmentations are precise, the method is penalized by misclassifications. Experimental results do not highlight any benefits in using Grabcut with Mask R-CNN: they report a situation of substantial

(a) Mask R-CNN (b) Mask R-CNN+Grabcut

(c) Mask R-CNN (d) Mask R-CNN+Grabcut

Figure 8: Examples of Mask R-CNN masks refined by Grabcut.

parity with a small detriment (-0.1%) in GA. Nevertheless, inspecting the generated masks, we found out that Grabcut refines the segmentations, as shown by a couple of examples in Figure 8. This improvement is counterbalanced by misclassified objects: in other words, the negative impact of misclassified objects increases if their masks are refined. To further investigate the combination of Mask R-CNN with Grabcut, we detail additional tests on the COCO dataset in Section 4.3, which better show the benefits of using Grabcut both quantitatively and qualitatively. In Figure 9, we present additional qualitative results for 3DEF+YOLO+Grabcut. We report the initial output of 3DEF in Figure 9(a). The integration of YOLO without Grabcut, see Figure 9(b), generates a semantic labeling clearly less accurate than the integration of YOLO with Grabcut, see Figure 9(c). We also re-

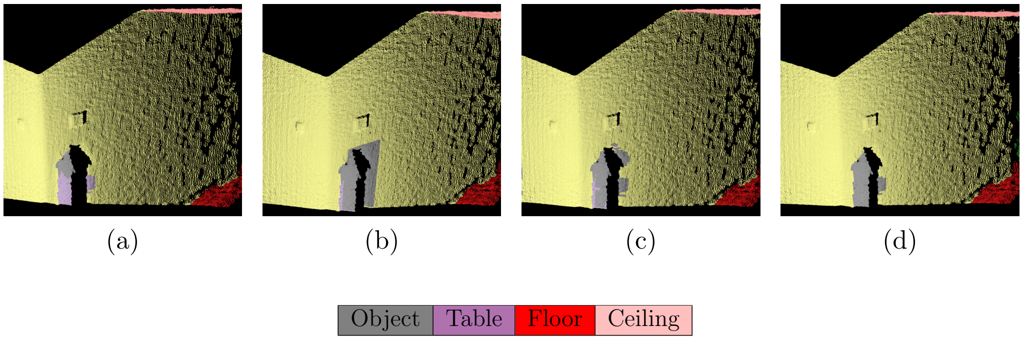

(a) (b) (c) (d)

Figure 9: Semantic segmentation of a fire extinguisher on the wall: (a) 3DEF: the object is mainly confused with a table; (b) YOLO-based semantic segmentation without Grabcut: the object is correctly classified but many points on the wall are misclassified; (c) YOLO-based semantic segmentation: many points are correctly classified but the object is still partially labeled as table and the wall as object; (d) 3DEF+YOLO with Bayesian fusion and the final cluster smoothing: there are no wrong labels on the object and only a few points of the wall are still labeled as object because of the imperfect initial segmentation of the 3DEF framework.

Table 4: Evaluation of the multi-view approaches on the NYUv2. The methods are reported in increasing order of class-wise accuracy CA. The best result are in bold. Using multiple views lead to the best results in CA, GA and CP.

Method	CA	GA	CP
3DEF [13]	55.7	65.0	53.3
MV-3DEF [18]	56.1	65.3	53.7
3DEF+Mask R-CNN (best)	61.2	67.4	56.2
3DEF+YOLO (best)	61.3	67.6	56.3
MV-3DEF+YOLO	61.5	**67.7**	56.4
MV-3DEF+Mask R-CNN	**64.0**	66.0	**56.5**

port the improved output after Bayesian fusion and clustering smoothing in Figure 9(d).

We selected the best approaches in the previous experiment and tested the multi-view frame fusion scheme in [18] on them. For simplicity, we refer to 3DEF+YOLO+Grabcut as 3DEF+YOLO (the best approach). Table 4 shows that using multiple views does not have the same effect on all meth-

Table 5: Performance comparison on the NYUv2. The methods are reported in increasing order of class-wise accuracy CA. The class performance improvements with respect the baselines 3DEF and MV-3DEF are in boxes. The best result are in bold. Combining 3DEF with a detector makes the approach more competitive with respect to existing approaches.

Method	CA	GA	CP
Couprie *et al.* [3]	36.2	52.4	-
Hermans *et al.* [4]	48.0	54.2	-
3DEF [13]	55.7	65.0	53.3
MV-3DEF [18]	56.1	65.3	53.7
SEGCloud [28]	56.4	66.8	-
Nakajima *et al.* [43]	58.5	70.7	-
Eigen [12, 5]	59.9	66.5	-
3DEF+MaskRCNN (best)	61.2	67.4	56.2
3DEF+YOLO (best)	61.3	67.6	56.3
MV-3DEF+YOLO	61.5	67.7	56.4
Eigen-SF [5]	63.2	69.3	-
Eigen-SF-CRF [5]	63.6	69.9	-
MV-3DEF+MaskRCNN	64.0	66.0	**56.5**
MVCNet-MaxPool [45]	**69.5**	**77.7**	-

ods. In particular, MV-3DEF+YOLO slightly improves over all the coefficients (+0.2%, +0.1%, +0.1%) while MV-3DEF+Mask R-CNN improves in classwise recall and precision (+3.5% and +0.1%) but deteriorates the global accuracy (−1.4%). This difference is expected since different methods have different success and failure models, and different confidence distributions. On this dataset, the average number of labelled frames per scene is 2.74. As shown in [18], this reduces the performance benefit of the multi-view method, which improves with the number of forward-projected frames.

In Table 5 and Table 6, we compare our methods with state-of-the-art methods for single-view and multi-view semantic segmentation. In Table 5, we report the results of single-view methods working on both RGB-D data, Couprie *et al.* [3] and Eigen *et al.* [12, 5], and 3D point clouds, 3DEF [13] and SEGCloud [28]. We also report the results of different multi-view methods, Hermans *et al.* [4], Eigen-SF-CRF [5], MV-3DEF [18], Nakajima *et al.* [43] and MVCNet-MaxPool [45]. These works are evaluated at full resolution

Table 6: Class performance comparison on the NYUv2. The class performance improvements with respect the baselines 3DEF and MV-3DEF are in boxes. The best result are in bold. Combining 3DEF with a detector makes the approach more competitive with respect to existing approaches.

Method	Bed	Object	Chair	Furniture	Ceiling	Floor	Picture	Sofa	Table	Wall	Window	Books	TV
Couprie *et al.* [3]	38.1	8.7	34.1	42.4	62.6	87.3	40.4	24.6	10.2	86.1	15.9	13.7	6.05
Hermans *et al.* [4]	68.4	8.6	41.9	37.1	83.4	91.5	35.8	28.5	27.7	71.8	46.1	45.4	38.4
3DEF [13]	74.2	17.2	63.4	48.1	80.3	**98.7**	26.5	71.0	46.5	84.0	25.4	55.1	34.1
MV-3DEF [18]	73.2	17.5	64.5	48.8	80.2	**98.7**	27.2	74.5	50.4	84.2	29.5	56.0	42.7
SEGCloud [28]	75.1	39.3	62.9	61.8	69.1	95.2	34.4	62.8	45.8	78.9	26.4	53.5	28.5
Nakajima *et al.* [43]	83.7	52.5	56.7	**76.1**	24.4	83.3	40.8	**77.7**	53.0	75.3	**64.4**	15.6	57.3
Eigen [12, 5]	42.3	46.5	72.4	60.8	73.1	85.7	57.3	38.9	42.1	85.5	55.8	49.1	68.5
3DEF+Mask R-CNN	85.2	18.5	82.8	57.8	79.2	97.4	23.8	76.7	**55.1**	80.1	22.2	61.3	55.8
3DEF+YOLO	86.9	17.7	82.4	55.0	79.2	96.8	24.1	71.6	51.4	82.7	25.0	66.3	57.5
MV-3DEF+YOLO	87.8	17.7	82.3	54.8	81.3	96.6	23.0	71.6	51.2	82.7	25.8	66.7	57.3
Eigen-SF-CRF [5]	48.3	**46.9**	74.7	63.5	79.0	90.8	**63.6**	46.5	45.9	**89.4**	55.6	51.5	**71.5**
MV-3DEF+Mask R-CNN	**95.3**	18.9	**85.9**	62.8	**89.4**	96.2	22.6	75.9	53.7	79.8	14.5	**68.8**	67.7

(640 × 480) with the exception of the approaches presented in [5, 43] which report the result when working at half resolution (320 × 240). In Table 6, we compare the methods class by class. We do not report the results for MVCNet-MaxPool [45] since they are not available and we report the results of Eigen-SF-CRF over Eigen-SF since it is the best performing among the two.

As reported in both tables, a significant boost in performance is obtained by combining the 3DEF classifier and a detector, both Mask R-CNN and YOLO. In particular, our best single-view 3DEF+YOLOs outperform the baselines based on 3DEF (+5.2% in CA, +2.3% in GA and +2.5% in CP) as well as SEGCloud [28] (+4.9% in CA and +0.8% in GA) and Eigen [5, 43] (+1.3% in CA and +1.1% in GA). 3DEF+YOLO outperforms also Nakajima *et al.* [43] in CA (+3.0%) but not in GA (-3.0%) since our method offers better performance class by class but not on classes with more samples in the dataset. Using multi-views highlights the strengths of our methods: MV-3DEF+YOLO gets closer to Eigen-SF, Eigen-SF-CRF and MVCNet-MaxPool while MV-3DEF+Mask R-CNN outperforms Eigen-SF and Eigen-SF-CRF, and gets closer to MVCNet-MaxPool. In particular, MV-3DEF+Mask R-CNN outperforms Eigen-SF-CRF in CA (+0.4%) but not in GA (−3.9%). The method is stronger class by class but penalized by the performance with the classes with more samples in the dataset, in particular the class *Wall*. Neither the integration of the object detector nor the multi-view allow to outperform MVCNet-MaxPool [45], (−5.5% in CA

Table 7: Class performance differences between the two best methods on the NYUv2. MV-3DEF+YOLO and MV-3DEF+Mask R-CNN outperform MV-3DEF in 8 and 9 out of 13 classes, respectively. Improvements are in bold.

Method vs MV-3DEF [18]	Bed	Object	Chair	Furniture	Ceiling	Floor	Picture	Sofa	Table	Wall	Window	Books	TV
MV-3DEF+YOLO	**+14.6**	**+0.2**	**+17.8**	**+6.0**	**+1.1**	-2.1	-4.2	-2.9	**+0.8**	-1.5	-3.7	**+10.7**	**+14.6**
MV-3DEF+Mask R-CNN	**+22.1**	**+1.4**	**+21.4**	**+14.0**	**+9.2**	-2.5	-4.6	**+1.4**	**+3.3**	-4.4	-15.0	**+12.8**	**+25.0**

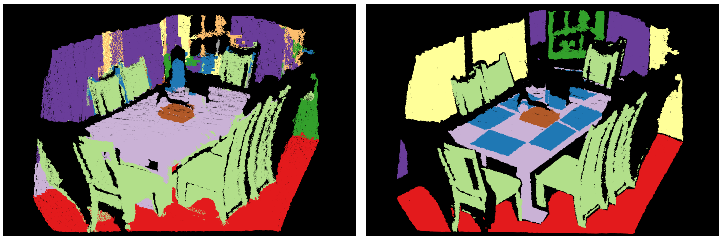
Table 8: Class performance differences between the two best methods on the NYUv2. MV-3DEF+YOLO and MV-3DEF+Mask R-CNN outperforms Eigen-SF in 7 out of 13 classes. MV-3DEF+Mask R-CNN and Eigen-SF-CRF are almost equivalent in 2 other classes. Improvements are in bold.

Method vs Eigen-SF-CRF [5]	Bed	Object	Chair	Furniture	Ceiling	Floor	Picture	Sofa	Table	Wall	Window	Books	TV
MV-3DEF+YOLO	**+39.5**	-29.2	**+7.6**	-8.7	**+2.3**	**+5.8**	-40.6	**+25.1**	**+5.3**	-6.7	-29.8	**+15.2**	-14.2
MV-3DEF+Mask R-CNN	**+47.0**	-28.0	**+11.2**	-0.7	**+10.4**	**+5.4**	-41.0	**+29.4**	**+7.8**	-9.6	-41.1	**+17.3**	-3.8

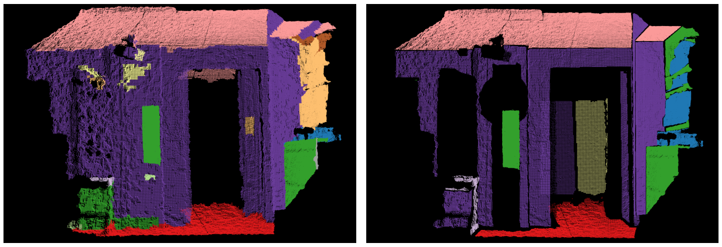
and −11.7% in GA). This approach already exploits multiple views and it would be interesting to study how to combine it with an object detector.

Class by class performance is further investigated comparing our best methods against the baseline MV-3DEF [18] in Table 7 and against Eigen-SF-CRF [5] in Table 8. MV-3DEF+YOLO and MV-3DEF+Mask R-CNN outperform MV-3DEF [18] in 8 and 9 out of 13 classes, respectively. The improved classes are *Bed*, *Object*, *Chair*, *Furniture*, *Ceiling*, *Sofa*, *Table* and *Bookshelf*. MV-3DEF+YOLO and MV-3DEF+Mask R-CNN outperform Eigen-SF-CRF [5] in 7 out of 13 classes, *Bed*, *Chair*, *Ceiling*, *Floor*, *Sofa*, *Table* and *Bookshelf*. MV-3DEF+Mask R-CNN and Eigen-SF-CRF [5] are almost equivalent in 2 other classes, *Furniture* and *TV*. Both tables show that our methods suffer when classifying *Wall*, *Picture* and *Window*. This is a weakness of 3DEF that cannot be compensated by the detectors since they are not trained on those classes. This could be further investigated by training the detector on the classes *Picture* and *Window* or by improving the preliminary region growing segmentation in 3DEF. Indeed, the region growing can erroneously merge the three classes in a single cluster making it impossible for 3DEF to classify them correctly.

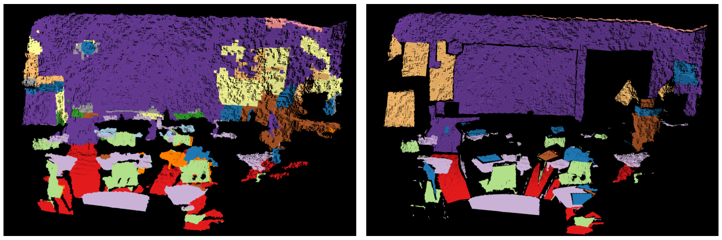
Additional qualitative results are reported in Figure 10. For each scene, the predicted semantic segmentation and its ground truth are reported side



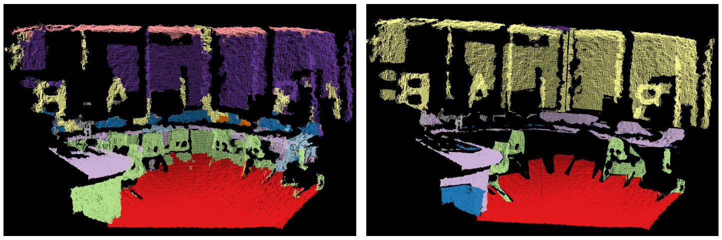
(a) MV-3DEF+Mask R-CNN (b) Groundtruth



(c) MV-3DEF+Mask R-CNN (d) Groundtruth



(e) MV-3DEF+Mask R-CNN (f) Groundtruth



(g) MV-3DEF+Mask R-CNN (h) Groundtruth

Figure 10: Qualitative results on the NYUv2 dataset: (a)(c)(e)(g) multi-view semantic segmentation obtained with the best of our methods, MV-3DEF+Mask R-CNN and (b)(d)(f)(h) groundtruth semantic segmentation.

Table 9: Average precision comparison on the COCO dataset. The performance improvements with respect to the baseline Matterport Mask R-CNN [61] are enclosed in boxes. The best results are in bold.

Method	AP	AP$_{50}$	AP$_{75}$	AP$_{\mathrm{S}}$	AP$_{\mathrm{M}}$	AP$_{\mathrm{L}}$
Matterport Mask R-CNN [61]	28.2	47.1	30.0	12.7	30.0	38.0
Mask R-CNN+Grabcut	28.4	47.7	29.9	12.5	29.9	39.1
FAIR Mask R-CNN [21]	**43.8**	**68.8**	**47.1**	**23.7**	**46.4**	**61.4**

Table 10: Average recall comparison on the COCO dataset. The performance improvements with respect to the baseline Matterport Mask R-CNN [61] are enclosed in boxes. The best results are in bold.

Method	AR$_{1}$	AR$_{10}$	AR$_{100}$	AR$_{\mathrm{S}}$	AR$_{\mathrm{M}}$	AR$_{\mathrm{L}}$
Matterport Mask R-CNN [61]	24.6	34.3	34.9	15.9	37.2	47.9
Mask R-CNN+Grabcut	25.0	34.9	35.5	15.7	37.5	49.8
FAIR Mask RCNN [21]	**34.7**	**55.0**	**58.0**	**40.7**	**62.1**	**73.3**

by side. Generally, our approach successfully classifies several classes, e.g. *Chair*, *Furniture*, *Table* and *Books* in the reported scenes. Also some correct instances of *Object* are visible. Nevertheless, as previously discussed, the method struggles with *Picture*, *Wall* and *Windows*.

4.3. Experiments on COCO

We further investigate the performance of the 2D component of our approach on the COCO dataset [25]. Similarly to other approaches evaluated on this dataset, we characterized the performance of our method using the 12 metrics proposed by the authors. They capture the average precision at different Intersection over Unions (IoU), i.e. with loose or strict detection versus groundtruth matching criteria, and across scales, i.e. evaluating the performance separately when dealing with small objects and large objects. They capture also the average recall given a maximum number of objects per frame and across scales. Each metric is described in the following:

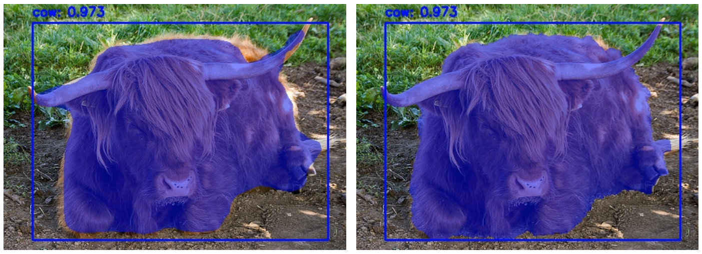
- average precision with IoUs from 0.50 to 0.95 with a step of 0.05 (AP);

- average precision at IoU 0.50 (AP_{50});
- average precision at IoU 0.75 (strict metric) (AP_{75});
- average precision for small objects with an area less than $32^2\,px^2$ (AP_S);
- average precision for medium objects with an area greater than $32^2\,px^2$ and less than $96^2\,px^2$ (AP_M);
- average precision for large objects with an area greater than $96^2\,px^2$ (AP_L);
- average recall given one detection per image (AR_1);
- average recall given 10 detections per image (AR_{10});
- average recall given 100 detections per image (in the following: AR_{100});
- average recall for small objects with an area less than $32^2\,px^2$ (AR_S);
- average recall for medium objects with an area greater than 32^2 and less than $96^2\,px^2$ (AR_M);
- average recall for large objects with an area greater than $96^2\,px^2$ (AR_L).

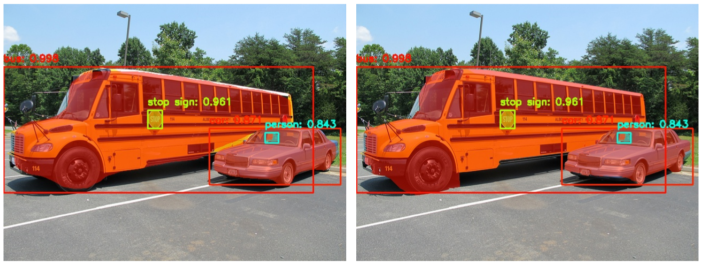
In Table 9 and 10 we compare our method against Matterport Mask R-CNN [61] and FAIR Mask R-CNN [21]. Matterport Mask R-CNN [61] is an open-source implementation of Mask R-CNN we use as baseline for developing our method Mask R-CNN+Grabcut. FAIR Mask R-CNN [21] is an ensemble of 30 Mask R-CNN methods. This method is the best performing one. As reported in Table 9 and 10, our approach obtains better results in both AP and AR with respect to the baseline Matterport Mask R-CNN [61]. The performance improvement with respect to the baseline is enclosed in boxes. Most of the metrics (AP, AP^{50}, AP^{L}, AR^{1}, AR^{10}, AR^{100}, AR^{M} and AR^{L}) are improved while the two approaches are almost equivalent with respect to the remaning ones (AP^{75}, AP^{S}, AP^{M}, AR^{S}).

Qualitative results are shown in Figure 11. Using our method, the object contours are better defined, as it is visible comparing Figure 11(a)(b) with Figure 11(b)(d). Nevertheless, the mask can get worse if the color model is not captured by Gaussian mixture model used by Grabcut. An example of this behaviour in shown in Figure 11(g)(h) in which Grabcut is confused by the square pattern of the shirt.

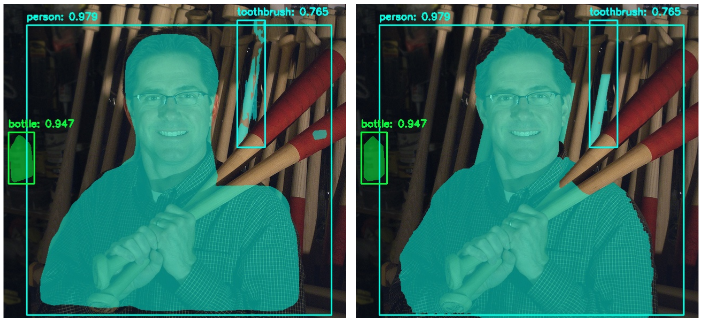
(a) Mask R-CNN (b) Mask R-CNN+Grabcut



(c) Mask R-CNN (d) Mask R-CNN+Grabcut



(e) Mask R-CNN (f) Mask R-CNN+Grabcut



(g) Mask R-CNN (h) Mask R-CNN+Grabcut

Figure 11: Qualitative results on the COCO dataset: (a)(c)(e)(g) segmentation masks obtained with Matterport Mask R-CNN [61] and (b)(d)(f)(h) refined segmentation masks obtained with Mask R-CNN+Grabcut. Our approach refines the mask contours.

Table 11: Running times of our system on the laptop Dell Inspiron 15 7000 installed on our mobile robot [14].

Method	fps
Semantic segmentation with 3DEF	0.53
Mask R-CNN detector	0.94
YOLO detector	4.20
Mask R-CNN refinement with Grabcut	0.19
YOLO refinement with Grabcut	0.90
Multi-view frame fusion scheme	2.27
Full system with Mask R-CNN	0.12
Full system with YOLO	0.27

4.4. Runtime Analysis

We tested our system on a standard laptop Dell Inspiron 15 7000 installed on our mobile robot [14]. It runs Ubuntu 18.04 and is equipped with an Intel Core i7-6700HQ CPU with 4 cores clocked at 2.60 GHz, the graphic card NVIDIA GeForce GTX 960M and 16 GB of DDR3 RAM. We worked at full resolution (640×480 px). The running times evaluated on the NYUv2 dataset are reported in Table 11. The proposed approach makes use of a technique for semantic segmentation, which requires approximately 0.53 fps on the CPU. The object detectors Mask R-CNN and YOLO work on the GPU at 0.94 fps and 4.20 fps, respectively. The combinations of the detectors with Grabcut work at an average speed of 0.19 fps when using masks and 0.90 fps when using boxes. The multi-view works at an average speed of 2.27 fps leading to a total runtime of approximately 0.12 fps with Mask R-CNN and 0.27 fps with YOLO. The current system requires more work to be used in real-time on a standard laptop. Nevertheless, it is suitable in less demanding applications requiring occasional accurate decisions or for offline processing.

5. Conclusions

In this work, we extended a multi-view semantic segmentation system based on 3D Entangled Forests (3DEF) by integrating and refining two object detectors, Mask R-CNN and You Only Look Once (YOLO), with Bayesian fusion and Grabcut. The new system takes the best of its components, successfully exploiting both 2D and 3D data. Our experiments on two popular datasets, NYUv2 and COCO, show that our approach is competitive with the state-of-the-art and leads to accurate semantic segmentations. In particular, the 2D component of our method can be useful even for computer vision applications lacking 3D data, both indoor and outdoor. In the future, we would like to explore other semantic segmentation techniques and study how to perform accurate detection and segmentation of both objects and coarse scene elements limiting the number of separate components.

Acknowledgments

Part of this work was supported by IAS-Lab in the Department of Information Engineering of the University of Padua, Italy, and MIUR (Italian Minister for Education) under the initiative Departments of Excellence (Law 232/2016).

References

[1] Nathan Silberman and Rob Fergus. Indoor scene segmentation using a structured light sensor. In *IEEE International Conference on Computer Vision Workshops (ICCV Workshops), 2011*, pages 601–608. IEEE, 2011.

[2] N. Silberman, D. Hoiem, P. Kohli, and R. Fergus. Indoor segmentation and support inference from rgbd images. In *European Conference on Computer Vision (ECCV)*, pages 746–760. Springer, 2012.

[3] Camille Couprie, Clément Farabet, Laurent Najman, and Yann Lecun. Indoor semantic segmentation using depth information. In *First International Conference on Learning Representations (ICLR 2013)*, pages 1–8, 2013.

[4] Alexander Hermans, Georgios Floros, and Bastian Leibe. Dense 3d semantic mapping of indoor scenes from rgb-d images. In *IEEE International Conference on Robotics and Automation (ICRA), 2014*, pages 2631–2638. IEEE, 2014.

[5] John McCormac, Ankur Handa, Andrew Davison, and Stefan Leutenegger. Semanticfusion: Dense 3d semantic mapping with convolutional neural networks. *arXiv preprint arXiv:1609.05130*, 2016.

[6] Renato F Salas-Moreno, Richard A Newcombe, Hauke Strasdat, Paul HJ Kelly, and Andrew J Davison. Slam++: Simultaneous localisation and mapping at the level of objects. In *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)*, pages 1352–1359, 2013.

[7] Saurabh Gupta, Pablo Arbeláez, Ross Girshick, and Jitendra Malik. Indoor scene understanding with rgb-d images: Bottom-up segmentation, object detection and semantic segmentation. *International Journal of Computer Vision*, 112(2):133–149, 2015.

[8] Kenneth Vanhoey, Carlos Eduardo Porto de Oliveira, Hayko Riemenschneider, András Bódis-Szomorú, Santiago Manén, Danda Pani Paudel, Michael Gygli, Nikolay Kobyshev, Till Kroeger, and Dengxin Dai. Varcity-the video: the struggles and triumphs of leveraging fundamental research results in a graphics video production. In *ACM SIGGRAPH 2017 Talks*, page 48. ACM, 2017.

[9] Andrej Karpathy and Li Fei-Fei. Deep visual-semantic alignments for generating image descriptions. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015.

[10] Markus Vincze, Markus Bajones, Markus Suchi, Daniel Wolf, Astrid Weiss, David Fischinger, and Paloma da la Puente. Learning and detecting objects with a mobile robot to assist older adults in their homes. In *European Conference on Computer Vision (ECCV)*, pages 316–330. Springer, 2016.

[11] Paul Sturgess, Karteek Alahari, Lubor Ladicky, and Philip HS Torr. Combining appearance and structure from motion features for road scene understanding. In *BMVC 2012-23rd British Machine Vision Conference*. BMVA, 2009.

[12] David Eigen and Rob Fergus. Predicting depth, surface normals and semantic labels with a common multi-scale convolutional architecture. In

Proceedings of the IEEE International Conference on Computer Vision (ICCV), pages 2650–2658, 2015.

[13] D. Wolf, J. Prankl, and M. Vincze. Enhancing semantic segmentation for robotics: the power of 3-d entangled forests. *IEEE Robotics and Automation Letters*, 1(1):49–56, 2016.

[14] M. Carraro, M. Antonello, L. Tonin, and E. Menegatti. An open source robotic platform for ambient assisted living. *Artificial Intelligence and Robotics (AIRO)*, 2015.

[15] D. Fischinger, P. Einramhof, K. Papoutsakis, W. Wohlkinger, P. Mayer, P. Panek, S. Hofmann, T. Koertner, A. Weiss, and A. Argyros. Hobbit, a care robot supporting independent living at home: First prototype and lessons learned. *Robotics and Autonomous Systems*, 75:60–78, 2016.

[16] Matteo Terreran, Morris Antonello, and Stefano Ghidoni. Boat hunting with semantic segmentation for flexible and autonomous manufacturing. In *European Conference on Mobile Robotics (ECMR)*, pages 1–8. IEEE, 2019.

[17] Bryan C Russell, Antonio Torralba, Kevin P Murphy, and William T Freeman. Labelme: a database and web-based tool for image annotation. *International journal of computer vision*, 77(1-3):157–173, 2008.

[18] Morris Antonello, Daniel Wolf, Johann Prankl, Stefano Ghidoni, Emanuele Menegatti, and Markus Vincze. Multi-view 3d entangled forest for semantic segmentation and mapping. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1855–1862. IEEE, 2018.

[19] C Rother, V Kolmogorov, and A Blake. Grabcut: Interactive foreground extraction using iterated graph cuts. *ACM Transactions on Graphics (TOG)*, 23(3):309–314, 2004.

[20] Victor Lempitsky, Pushmeet Kohli, Carsten Rother, and Toby Sharp. Image segmentation with a bounding box prior. In *12th International Conference on Computer Vision (ICCV)*, pages 277–284. IEEE, 2009.

[21] K. He, G. Gkioxari, P. Dollár, and R. Girshick. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision (ICCV)*, pages 2961–2969, 2017.

[22] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 779–788, 2016.

[23] Joseph Redmon and Ali Farhadi. Yolo9000: better, faster, stronger. *arXiv preprint arXiv:1612.08242*, 2016.

[24] Joseph Redmon and Ali Farhadi. Yolov3: An incremental improvement. *arXiv preprint arXiv:1804.02767*, 2018.

[25] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014.

[26] Clement Farabet, Camille Couprie, Laurent Najman, and Yann LeCun. Learning hierarchical features for scene labeling. *IEEE transactions on pattern analysis and machine intelligence*, 35(8):1915–1929, 2013.

[27] Ankur Handa, Viorica Patraucean, Vijay Badrinarayanan, Simon Stent, and Roberto Cipolla. Understanding real world indoor scenes with synthetic data. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4077–4085, 2016.

[28] Lyne Tchapmi, Christopher Choy, Iro Armeni, JunYoung Gwak, and Silvio Savarese. Segcloud: Semantic segmentation of 3d point clouds. In *2017 International Conference on 3D Vision (3DV)*, pages 537–547. IEEE, 2017.

[29] Liang-Chieh Chen, Yukun Zhu, George Papandreou, Florian Schroff, and Adam Hartwig. Encoder-decoder with atrous separable convolution for semantic image segmentation. In *Proceedings of the European conference on computer vision (ECCV)*, pages 801–818, 2018.

[30] Jingdong Wang, Ke Sun, Tianheng Cheng, Borui Jiang, Chaorui Deng, Yang Zhao, Dong Liu, Yadong Mu, Mingkui Tan, and Xinggang

Wang. Deep high-resolution representation learning for visual recognition. *arXiv preprint arXiv:1908.07919*, 2019.

[31] D. Wolf, J. Prankl, and M. Vincze. Fast semantic segmentation of 3d point clouds using a dense crf with learned parameters. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4867–4873. IEEE, 2015.

[32] Mennatullah Siam, Mostafa Gamal, Moemen Abdel-Razek, Senthil Yogamani, Martin Jagersand, and Hong Zhang. A comparative study of real-time semantic segmentation for autonomous driving. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops(CVPR Workshop)*, pages 587–597, 2018.

[33] Abhishek Anand, Hema Swetha Koppula, Thorsten Joachims, and Ashutosh Saxena. Contextually guided semantic labeling and search for three-dimensional point clouds. *The International Journal of Robotics Research*, 32(1):19–34, 2013.

[34] Ioannis Kostavelis and Antonios Gasteratos. Semantic mapping for mobile robotics tasks: A survey. *Robotics and Autonomous Systems*, 66:86–103, 2015.

[35] Andreas Nüchter and Joachim Hertzberg. Towards semantic maps for mobile robots. *Robotics and Autonomous Systems*, 56(11):915–926, 2008.

[36] Ioannis Kostavelis and Antonios Gasteratos. Learning spatially semantic representations for cognitive robot navigation. *Robotics and Autonomous Systems*, 61(12):1460–1475, 2013.

[37] Ioannis Kostavelis and Antonios Gasteratos. Semantic maps from multiple visual cues. *Expert Systems with Applications*, 68:45–57, 2017.

[38] Niko Sünderhauf, Trung T Pham, Yasir Latif, Michael Milford, and Ian Reid. Meaningful maps with object-oriented semantic mapping. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5079–5085. IEEE, 2017.

[39] Yoshikatsu Nakajima and Hideo Saito. Efficient object-oriented semantic mapping with object detector. *IEEE Access*, 7:3206–3213, 2018.

[40] Ioannis Kostavelis and Antonios Gasteratos. On the optimization of hierarchical temporal memory. *Pattern Recognition Letters*, 33(5):670–676, 2012.

[41] Andrzej Pronobis and Patric Jensfelt. Large-scale semantic mapping and reasoning with heterogeneous modalities. In *2012 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3515–3522. IEEE, 2012.

[42] Jörg Stückler, Nenad Biresev, and Sven Behnke. Semantic mapping using object-class segmentation of rgb-d images. In *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*, pages 3005–3010. IEEE, 2012.

[43] Yoshikatsu Nakajima, Keisuke Tateno, Federico Tombari, and Hideo Saito. Fast and accurate semantic mapping through geometric-based incremental segmentation. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 385–392. IEEE, 2018.

[44] Yang He, Wei-Chen Chiu, Margret Keuper, Mario Fritz, and Saarland Informatics Campus. Std2p: Rgbd semantic segmentation using spatio-temporal data-driven pooling. In *International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7158–7167, 2017.

[45] Lingni Ma, Jörg Stückler, Christian Kerl, and Daniel Cremers. Multi-view deep learning for consistent semantic mapping with rgb-d cameras. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 598–605. IEEE, 2017.

[46] Jingdao Chen, Yong Kwon Cho, and Zsolt Kira. Multi-view incremental segmentation of 3-d point clouds for mobile robots. *IEEE Robotics and Automation Letters*, 4(2):1240–1246, 2019.

[47] Felix Endres, Jurgen Hess, Jurgen Sturm, Daniel Cremers, and Wolfram Burgard. 3-d mapping with an rgb-d camera. *IEEE Transactions on Robotics*, 30(1):177–187, 2014.

[48] Roberto Capobianco, Jacopo Serafin, Johann Dichtl, Giorgio Grisetti, Luca Iocchi, and Daniele Nardi. A proposal for semantic map represen-

tation and evaluation. In *2015 European Conference on Mobile Robots (ECMR)*, pages 1–6. IEEE, 2015.

[49] Constantine Papageorgiou and Tomaso Poggio. A trainable system for object detection. *International Journal of Computer Vision*, 38(1):15–33, 2000.

[50] Paul Viola and Michael J Jones. Robust real-time face detection. *International journal of computer vision*, 57(2):137–154, 2004.

[51] Pedro F Felzenszwalb, Ross B Girshick, David McAllester, and Deva Ramanan. Object detection with discriminatively trained part-based models. *IEEE transactions on pattern analysis and machine intelligence*, 32(9):1627–1645, 2010.

[52] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Computer Vision and Pattern Recognition (CVPR)*, 2014.

[53] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster R-CNN: Towards real-time object detection with region proposal networks. In *Advances in Neural Information Processing Systems (NIPS)*, 2015.

[54] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg. Ssd: Single shot multibox detector. In *European conference on computer vision*, pages 21–37. Springer, 2016.

[55] Jasper RR Uijlings, Koen EA Van De Sande, Theo Gevers, and Arnold WM Smeulders. Selective search for object recognition. *International journal of computer vision*, 104(2):154–171, 2013.

[56] Asako Kanezaki and Tatsuya Harada. 3d selective search for obtaining object candidates. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 82–87. IEEE, 2015.

[57] Jan Hosang, Rodrigo Benenson, and Bernt Schiele. How good are detection proposals, really? *arXiv preprint arXiv:1406.6962*, 2014.

[58] Jan Hosang, Rodrigo Benenson, Piotr Dollár, and Bernt Schiele. What makes for effective detection proposals? *IEEE transactions on pattern analysis and machine intelligence*, 38(4):814–830, 2016.

[59] K.-K. Maninis, S. Caelles, J. Pont-Tuset, and L. Van Gool. Deep extreme cut: From extreme points to object segmentation. In *Computer Vision and Pattern Recognition (CVPR)*, 2018.

[60] J. Papon, A. Abramov, M. Schoeler, and F. Worgotter. Voxel cloud connectivity segmentation-supervoxels for point clouds. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2027–2034, 2013.

[61] Waleed Abdulla. Mask r-cnn for object detection and instance segmentation on keras and tensorflow. `https://github.com/matterport/Mask_RCNN`, 2017.

[62] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)*, pages 1–9, 2015.

[63] Min Lin, Qiang Chen, and Shuicheng Yan. Network in network. *arXiv preprint arXiv:1312.4400*, 2013.