Chapter 3

Object Instance Re-Identification (re-OBJ)

Summary

Conventional approaches to object re-identification rely on matching appearances of the target objects among a set of frames. However, learning appearances of the objects alone might fail when there are multiple objects with similar appearance or multiple instances of same object class present in the scene. In this chapter, we propose that partial observations of the background can be utilized to aid in the object re-identification task for a rigid scene, especially a rigid environment with a lot of reoccurring identical models of objects. Using an extension to the Mask R-CNN architecture, we learn to encode the important and distinct information in the background jointly with the foreground (Section 3.2) relevant to rigid real-world scenarios such as an indoor environment where objects are static and the camera moves around the scene. We demonstrate the effectiveness of our joint visual feature in the re-identification of objects in the ScanNet dataset and show a relative improvement over the state-of-the-art method.

Figure 3.1: Similar looking objects in rigid, indoor scenes from ScanNet dataset. Multiple instances of the same object class, chair, in this case, are hard to differentiate with each other. In such cases, background can be highly useful to re-identify a particular instance in multiple views.

Consider a static indoor scene, where the environment is cluttered with several objects including objects that can be near-identical or objects that are just different instances of a particular object category. This makes it challenging to identify and track a particular instance of an object among a number of such similar-looking objects present in the scene, e.g. see Fig. 3.1. This specific task of matching and correctly re-identifying an instance of a specific object category in the scenario of a rigid scene is referred to here as the problem of object instance re-identification (re-OBJ). This problem is particularly interesting from the perspective of various different tasks such as query-based object instance level image retrieval or an object-oriented Simultaneous Localization and Mapping (SLAM). Also, a good object tracking algorithm, for example, would be able to match the correct instances of an object across the frames to be able to track them throughout the sequence. Thus, identifying and matching the already seen instances of a particular object class in multiple views can be understood as the problem of object instance re-identification. Most of the standard object matching or tracking algorithms in the literature rely on building a model of the target

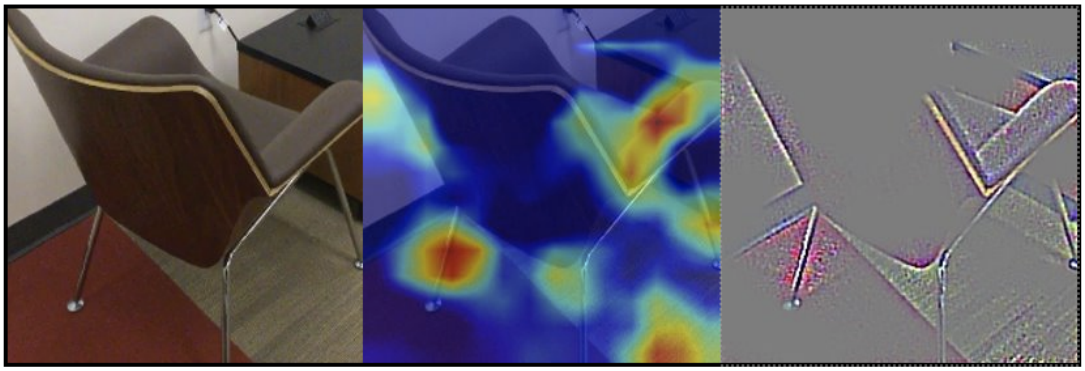

Figure 3.2: Left: The image of an object instance in a static scene. Centre: The image of the instance viewed through Grad-CAM [10] highlighting gradient weighted high importance regions. Right: Guided Grad-CAM image to show the pixel-space gradient visualizations to better understand the discriminative features present in the background.

objects by learning the appearance of only the target objects in the foreground and by applying the learned appearance model try to match the target objects across multiple views. But, if only the objects in the foreground are considered, then any affine-invariant descriptor would learn to find similarities between two incorrect instances simply because they look just the same as only learnt from their foreground appearances. On the other hand, if we focus also on the background around a particular object instance, it can provide us more discriminative information. Consider, two different views of the same static scene where the objects are stationary and only the camera is moving, the background might contain extremely important cues like border between the wall and the floor, other objects in vicinity and other useful discriminative features present in the scene (see Fig. 3.2).

These background cues might be really helpful, in addition to the foreground, for building a unique representation of a particular object instance in the rigid scene which would be different for the other instance despite the similarity of the two instances. To include the background information, the first step in our ap-

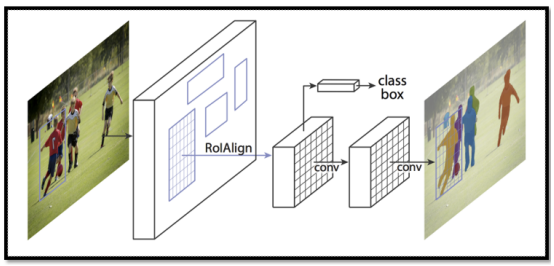

Figure 3.3: A schematic diagram of Mask R-CNN [5] architecture

proach is to use an off-the-shelf object detector, i.e. Mask R-CNN (Section. 3.1), and obtain foreground masks of the objects with the bounding boxes that are expanded (see Section. 3.3.2) in order to include a substantial background around the object within the bounding boxes. Encodings from the separated masked foregrounds and the masked backgrounds are extracted using ResNet50 (Section. 3.3), which are concatenated to obtain joint embeddings. These embeddings then are sampled into triplets $\{positive, negative, anchor\}$ and fed to a triplet-based network architecture consisting of identical ConvNets (see Figure 3.6 and Figure 3.7) with the pairwise ranking model to learn image similarity for a triplet-based ranking loss function.

3.1 Object Detection

Our approach uses a State-of-the-Art instance segmentation algorithm, Mask R-CNN [5] which uses region-based object detector like Faster R-CNN to detect objects. It is an instance segmentation method which provides a mask representing a set of pixels belonging to different objects and a bounding box around them. However, Mask R-CNN could be replaced by any other method that proposes better bounding boxes and the masks for objects.

As shown in Figure 3.3, a Region Proposal Network (RPN) is used to generate a number of region proposals followed by a position-sensitive RoI pooling layer

to warp them into a fixed dimension. Finally, it is fed into fully-connected layers to produce class scores and the bounding box predictions. A parallel branch of two additional fully-connected layers provides the mask. Using the output from the Mask R-CNN, we extract each bounding box including masks as separate images and resize them into images of a fixed size in order to train our network to learn a visual encoding of the objects' mask and the background surrounding them within the bounding boxes (see first column, Figure 3.4).

3.2 Foreground-Background Embeddings

For each object of the input images, we create two sets of images $F = \{I_f, I_b\}$. Using the detections obtained from Mask R-CNN, one set is created by extracting masks representing objects in the foreground (I_f). The other set only contains the background with the subtracted foreground (I_b). As shown in Figure 3.4, a pair of images is taken from each set to pass through two identical streams to learn an encoding between the masked foreground and the background.

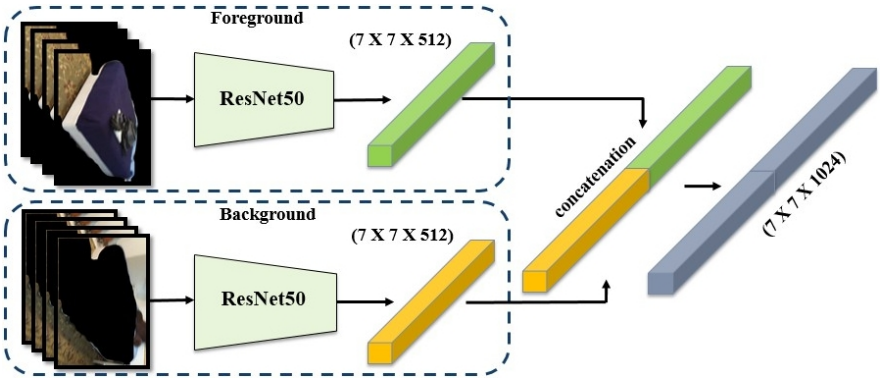

Figure 3.4: As input, our network takes expanded bounding boxes (see Section 3.3.2) which construct a pair of images for masked foreground and masked background (seen on left of image). Each of the pair of images is passed through a ResNet50 where we take an intermediary representation $7 \times 7 \times 512$ providing spatial information, which is concatenated to provide a joint representation of $7 \times 7 \times 1024$.

Each of the images, the masked background and the masked foreground is fed into a ResNet50 [42] deep model pre-trained on ImageNet [122] dataset to extract the features. We take from an intermediary layer of the network providing $I_{(.)} \in \mathbb{R}^{7\times7\times512}$ representation of the two images retaining spatial context, the tensors are then concatenated to provide an embedding $F \in \mathbb{R}^{7\times7\times1024}$.

3.2.1 Triplet loss

An effective algorithm for object instance re-identification should be able to distinguish not only between the images of different objects but also between different instances of the same object class. Especially, in the indoor scenes where multiple instances of the same object category are present, e.g. an office with multiple tables and chairs; it is highly challenging to re-identify a particular object instance amongst others.

We define a triplet of images that has three kinds of images: an *anchor* which acts like a query template, a *positive* and a *negative* image. In order to ensure an effective reidentification at the instance level, it is important also to consider the intra-class variations and different instances of the same object as negative examples. For example, a backpack and a chair are definitely an example of *anchor-negative* pairs but two different instances of the same chair (with a different background) should also be considered an *anchor-negative* pair. This can be better understood with the help of a visual example. From Figure 3.5, we can see that any other view of an object instance, a trashcan in this case, can be considered as a *positive* image and an instance of a different object which is a bucket of a different shape and color is a *negative* image even though the broader object category might be the same (both are type of buckets). On the other hand, the another instance of an object similar in the semantic class but present at a different location in the scene would also be considered a *negative* example,

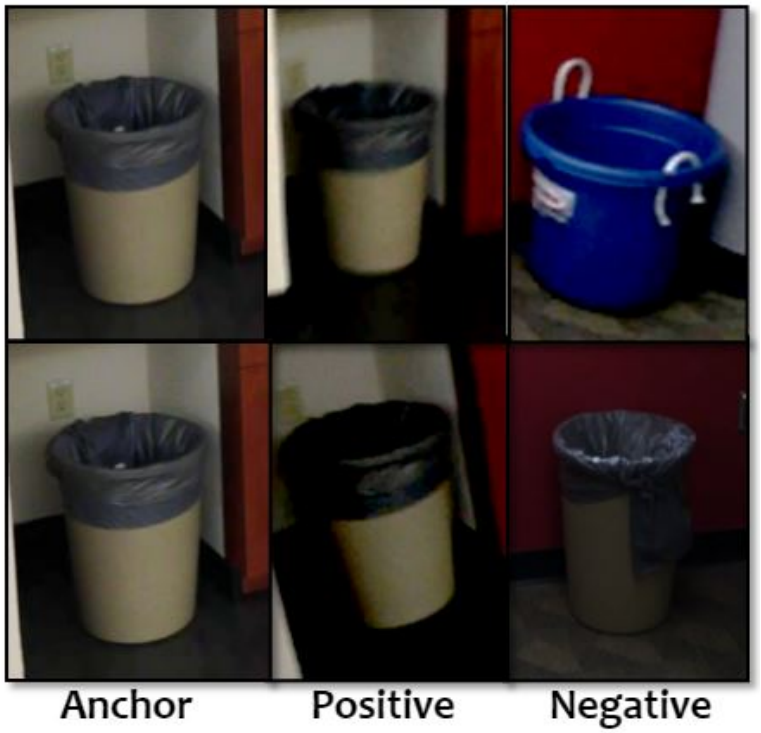

Figure 3.5: Left: Image of an object instance considered an *anchor* image (white background); Center: Another view of the same instance as seen in the *anchor* image considered here as a *positive* image (white background); Right: Top-right shows a *negative* image in the form of intra-class variation while bottom-right shows a *hard negative* image as it is another instance similar in shape and color to the *anchor* one but present at a different location in the scene (red background).

a *hard negative* example. A *hard negative* example can be distinguished from an *anchor* and a *positive* image only by looking at the background which can be seen in the column on the bottom-right in Figure 3.5.

We take inspiration from a triplet-based network architecture with the pair-wise ranking model to learn image similarity for the triplet loss function presented in the work by Wang et al. [8]. Let us consider, a set of $F = f_1, ..., f_F$ images out of which we can sample out triplets based on the similarity (distance between them) between any two images. We can have a triplet $t_i = (f_{iA}, f_{iP}, f_{iN})$ where f_{iA}, f_{iP} and f_{iN} are the anchor, positive and negative images, respectively. The

goal of the training would be to learn an embedding function such that:

$$D(f_{iA}, f_{iP}) < D(f_{iA}, f_{iN}) \tag{3.1}$$

where $D(.)$ is the squared Euclidean distance in the embeddings space. A triplet incorporates a relative ranking based on the similarity between the anchor, positive and the negative images.

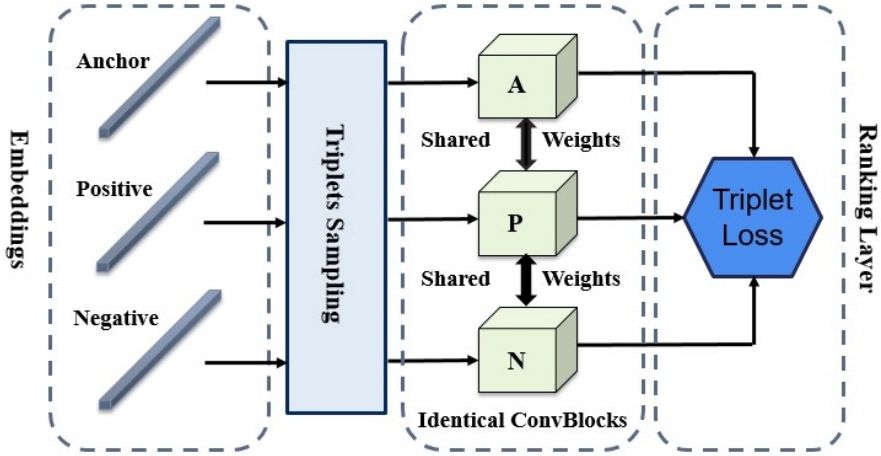

Figure 3.6: Triplet of input tensors corresponding to images. Each tensor contains an embeddings of the anchor image A, positive image P and a negative Image N which are fed into three identical deep neural networks independently with shared weights where the triplet loss is optimized.

The triplet ranking loss function is given as:

$$l(f_{iA}, f_{iP}, f_{iN}) = max\{0, M + D(f_{iA}, f_{iP}) - D(f_{iA}, f_{iN})\} \tag{3.2}$$

where M is *margin* that regulates the gap between the pairwise distance: (f_{iA}, f_{iP}) and (f_{iA}, f_{iN}). The model learns to minimize the distance between more similar images and maximize the distance between the dissimilar ones.

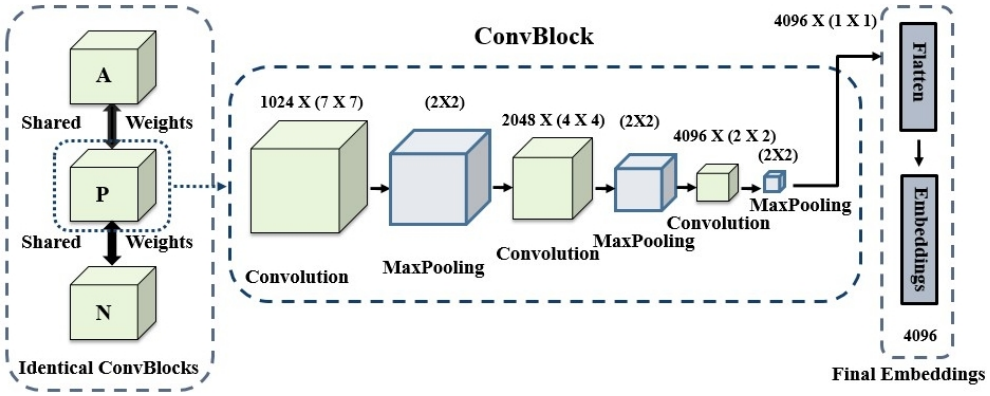

Figure 3.7: Our ConvBlock takes in the encoding from fig. 3.4. The ConvBlock consists of a network of convolutional and maxpooling layers, which pool the spatial information and merge the foreground and background encodings to obtain final embeddings.

3.3 Experimental Details

3.3.1 Dataset

ScanNet We use ScanNet dataset [7] for our experiments which consists of 1500 indoor RGBD scans annotated with 3D camera poses, surface reconstructions, and mesh segmentation related to several object categories. These annotations allowed us to evaluate the accuracy of Mask R-CNN on the ScanNet images to be used in the proposed pipeline. To generate our training data, we ran Mask R-CNN over a subset of 863 scenes randomly selected from the whole ScanNet dataset. In total, the Mask R-CNN provided 646, 156 object detections with masks belonging to 29 object classes (see Table 3.1).

Matterport3D We also perform experiments on another large-scale RGB-D dataset called Matterport3D [123] that contains 10,800 panoramic views obtained from 194,400 RGB-D images of 90 building scenes with indoor and outdoor objects annotated with surface reconstructions, camera poses, and 2D and 3D semantic segmentations. The dataset contains a total of 52410 objects appearing

in multiple views where we use Mask R-CNN again to get the object detections with masks.

3.3.2 Experimental Setup

Since not all the objects in both the datasets are annotated, we computed the bounding box overlap ratio between the ground truth (GT) bounding boxes and the detections provided by Mask R-CNN to select only the *valid* detections. If the overlap ratio was higher than 60% and the label of the detected object matches with the GT label, it was considered a *valid* detection.

After mapping each detection obtained from the Mask R-CNN with the corresponding 2D ground truth (GT) from the ScanNet dataset, we found 9.11% of the total, i.e. around 58876 detections to be considered fit for the experiments. The regions indicated by the bounding boxes were extended by an additional 10 pixels-wide border in order to allow loosely-fitted bounding boxes around the objects and thus, allowing a more significant background around each object's mask within the bounding boxes. These regions were then extracted out of the full images, resized to 224×224 and categorically stored based on the object's class and it's observed instances. Finally, for each object image, the foreground masks and the background masks were extracted and stored as separate images. The data obtained from the ScanNet dataset is split into a 2-fold cross-validation manner with 39250 images for training and 19626 images for test more than 1701 instances of objects as shown in Table 3.1. Likewise, the data from the Matterport3D dataset is split into 34940 images for training and 17470 images for test more than 1529 instances of objects as shown in Table 3.2.

To the check the feasibility of our intuition, we performed our experiments in three different setups. In all the experimental setups, we used pretrained ResNet50 [42] on the ImageNet [122] dataset as the backbone model to extract

features from the images of the objects. **no-train:** In this setup, the features extracted from full images were matched against each other by using an $L2$ distance-based metric, without any training. **full:** In another setup, our model is trained on the embeddings obtained using the full images without extracting separate foreground and background masks. **concat:** The third type of experimental setup is the approach proposed in this thesis where the model is trained on the embeddings obtained by concatenating the features from masked foregrounds and the backgrounds. In *concat* setup, the model learns to minimize the difference between the anchor f_{iA} and the positive f_{iP} images while also learning to maximize the difference between the anchor f_{iA} and the negative f_{iN} images by employing the triplet-loss based training.

3.3.3 Performance Evaluation

Most re-ID algorithms use Cumulative Matching Characteristic (CMC) curve as a standard metric to measure their performance which compares the identification rate vs rank. Each probe image is compared with all the gallery images and a score is calculated, which is generally, the euclidean distance. All the images are ranked in ascending order by their similarity measures, which is again the matching distance usually, with the anchor/probe image. A rank is calculated where the maximum score occurs and the proportions of good matches of the probe image with the set of images in rank-1 would indicate a good or bad performance of the algorithm. A CMC curve is computed for all these individual ranks. However, in our evaluation procedure, we compare the performance of our method with the performance of deepSort [113] tracking algorithm which is used here as a rank-1 re-ID method, which is why we cannot compare with a CMC curve. Also, it will not be fair to compare recall and precision values between the deepSort and our method. Thus, we compute the rank-1 accuracy by measuring

the percentage of correctly identified objects.

Analysis Evaluated using the aforementioned experimental setup, the proposed method achieves the best performance on the ScanNet dataset in regards to both the rank-1 accuracy as shown in Table 3.3. Figure 3.8 shows that the proposed method, *concat* was able to find the best match with the probe image. In the bottom row, *no-train* and *full* tried to match with an image which either had an object of the same color or the shape. However, the proposed method, *concat* could not always correctly identify the images and was performing occasionally poor as can be seen in Figure 3.9. Overall, the results from Table 3.3 show that the *concat* method was able to improve the rank-1 accuracy by 22.19% and 17.1% against *no-train* and *full*, respectively.

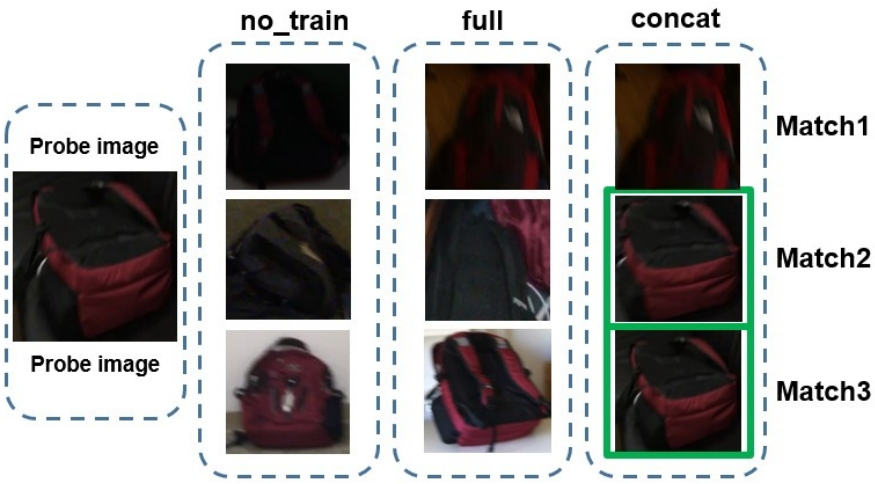

Figure 3.8: The visualizations show some examples of the matches found in *no-train*, *full* and *concat* setups. The right matches with the probe image are highlighted in green color.

Table 3.4 shows that the proposed method, *concat* was able to find the best match with the probe image with Matterport3D also. In the bottom row, *no-train* and *full* tried to match with an image which either had an object of the same color or the shape. However, the proposed method, *concat* could not always correctly identify the images and was performing occasionally poor as can be seen

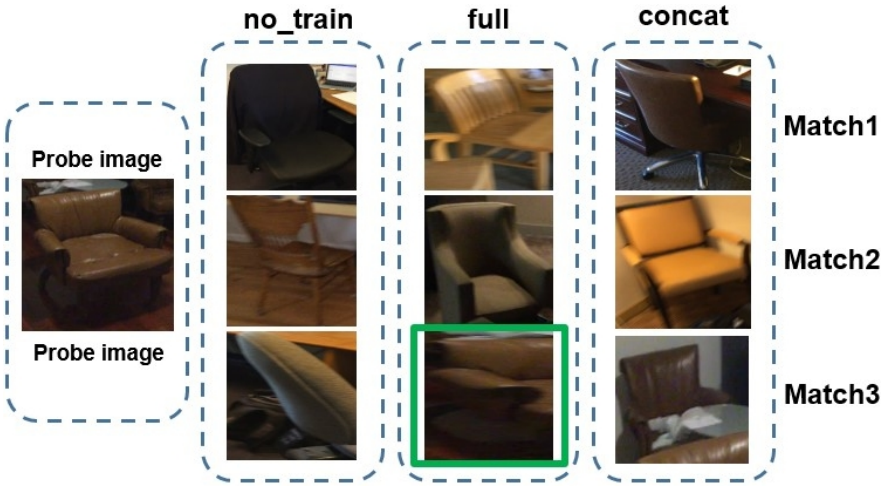

Figure 3.9: Examples of the matches found in *no-train*, *full* and *concat* setups. The right matches with the probe image are highlighted in green color.

in Figure 3.9. Overall, the results from Table 3.3 show that the *concat* method was able to improve the rank-1 accuracy by 24.48% and 12.07% against *no-train* and *full*, respectively.

Comparison with deepSort deepSort [113] is an open-source implementation of the original SORT [124] algorithm which employs deep appearance descriptors to improve the performance in multiple object tracking. deepSort learns discriminative feature embeddings offline in order to obtain a deep association metric for a person re-identification dataset in the original work. For our experiments, we provided two random sets of image pairs obtained from the ScanNet and Matterport3D scenes to the algorithm to identify multiple objects ensuring that an image pair is not consisting of images from two different scenes. We computed the performance by measuring the percentage of matched object instances across all the image pairs. Figure 3.10 shows the possible problems that standard object matching or tracking algorithms might face in re-identifying objects. The figure shows that the deepSort was able to match an object (in yellow bounding box) in multiple frames but lost an object (in red bounding box) when

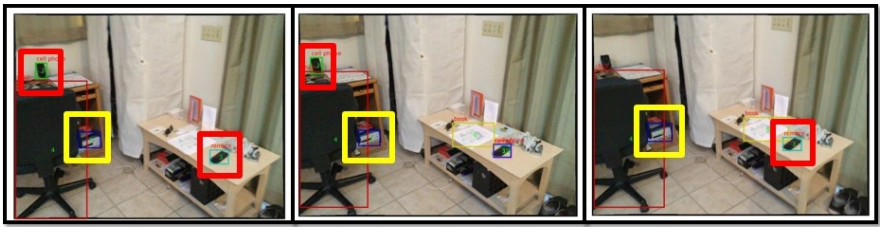

Figure 3.10: An example object being matched by the deepSort algorithm inside the yellow bounding box and the lost object in the red bounding box.

the camera revisits a similar view later. deepSort achieved a rank-1 accuracy of 49.60% and 46.92% against the rank-1 accuracy of 77.85% and 54.40% obtained with our method on ScanNet and Matterport3D datasets, respectively.

3.3.4 Experiments with person re-ID Baseline

To compare the performance of our method with other existing state-of-the-art methods in re-ID, we performed some experiments with the methods we consider as baseline person re-identification methods. The results in comparison to the proposed approach are given in Table 3.6.

OSNet We used one such baseline re-ID method like Omni-Scale Feature Learning for Person Re-Identification (OSNet) [125] on our data without any fine-tuning. OSNet tries to address person re-identification (ReID) as an instance-level recognition problem by not only capturing different spatial scales but also encapsulating a combination of multiple scales. The authors define a combination of features of both homogeneous and heterogeneous scales as omni-scale features. For example, features of variable homogeneous scale while identifying a person in multiple cameras would be a combination of global-scale features like the whole body region including the clothes and the corresponding local-scale features like the shoes, glasses etc. But, these features could be shared among different individuals like two different people wearing the similar clothes and shoes. Thus,

apart from these features of variable homogeneous scales, more complicated and richer features would be required. For example, when two people are wearing the clothes of same color (say, a white shirt), a specific logo in the front could be the distinguishing factor between the two. Although, the logo alone won't be distinctive enough on its own without the consideration of the clothes as a context which might otherwise be confused with several other patterns in the scene. Thus, the unique combination of small-scale features like the logo size on the shirt and the medium scale features like the upper-body size of the person could constitute the features of variable heterogeneous scales. In this work, a novel deep ReID CNN is proposed for learning these so called omni-scale features. This is achieved by designing a residual block composed of multiple convolutional streams, each detecting features at a certain scale. Importantly, a novel unified aggregation gate is introduced to dynamically fuse multi-scale features with input-dependent channel-wise weights.

DGNet Another baseline method we used for our experiments is Joint Discriminative and Generative Learning for Person Re-identification (DGNet) [126]. DGNet proposes a joint framework to couple the discriminative and generative learning to improve the learned re-id embeddings by leveraging the generated data. The generative module separately encodes each target (person) into an appearance code and a structure code, and a discriminative module shares the appearance encoder with the generative module. The appearance space encodes appearance of the person and other related semantic features including color of clothing, shoes, texture and style etc. while the structure space encode geometrical and spatial information including body size, volume of hair, pose, background and viewpoint, etc. By switching the appearance or structure codes, the generative module is able to generate high-quality cross-id composed images, which are then fed back online to the appearance encoder which is further used to improve

the discriminative module.

3.3.5 Conclusions

In this chapter, we showed that the information obtained from the background around the target objects in a static, rigid scenes could be highly useful in providing discriminative features to differentiate between two near-identical objects or two instances of the same object class. The discriminative features learned from the explicit concatenated foreground and background can be utilized to re-identify objects at the instance-level. Our experiments have shown that the proposed method performs well even in the case of highly cluttered rigid environments like the indoor scenes obtained from ScanNet and Matterport3D datasets.

Table 3.1: For ScanNet dataset, Number of views after mapping with GT for *valid* detections, selected based on object's label and the bounding box overlap ratio and the number of unique instances for each object category.

No. of views and unique instances per object class		
Object category	No. of views	No. of instances
bicycle	110	6
bench	27	4
backpack	1563	117
handbag	486	32
suitcase	377	30
sports ball	379	21
bottle	903	27
cup	278	25
chair	38203	508
couch	1371	75
potted plant	1294	55
bed	83	17
bowl	121	8
dining table	1853	185
toilet	1755	103
tv	562	46
laptop	600	41
mouse	59	6
keyboard	1879	67
microwave	667	61
oven	72	6
toaster	11	4
sink	2694	157
refrigerator	60	11
book	3124	65
clock	25	6
person	260	8
teddy bear	47	8
vase	13	2

Table 3.2: For Matterport3D dataset, Number of views after mapping with GT for *valid* detections, selected based on object's label and the bounding box overlap ratio and the number of unique instances for each object category.

No. of views and unique instances per object class		
Object category	No. of views	No. of instances
bicycle	68	28
bench	2545	89
backpack	12	8
handbag	57	26
suitcase	93	39
sports ball	1	1
bottle	440	25
cup	36	20
chair	23362	89
couch	5020	85
potted plant	9169	83
bed	3754	87
bowl	97	38
dining table	5371	89
toilet	467	75
tv	1795	84
laptop	47	26
mouse	2	2
keyboard	18	14
microwave	144	53
oven	1495	83
toaster	2	1
sink	1467	82
refrigerator	7336	90
book	6874	68
clock	336	68
person	1175	78
teddy bear	97	18
vase	1229	80

Table 3.3: Scores on our ScanNet validation data split with Rank-1,-5,-20 and -50 accuracy values. The best performing type of setups is highlighted in bold.

type	**Rank-1**(%)	**Rank-5**(%)	**Rank-20**(%)	**Rank-50**(%)
no-train	55.66	66.67	77.46	89.67
full	60.75	69.61	80.90	95.21
concat	**77.85**	**91.55**	**98.36**	**99.80**

Table 3.4: Scores on our Matterport3D validation data split with Rank-1,-5,-20 and -50 accuracy values. The best performing type of setups is highlighted in bold.

type	**Rank-1**(%)	**Rank-5**(%)	**Rank-20**(%)	**Rank-50**(%)
no-train	29.92	39.23	60.71	84.25
full	42.33	65.23	82.04	91.2
concat	**54.40**	**76.43**	**86.69**	**93.08**

Table 3.5: Comparison of Rank-1 accuracy for deepSort and our method

dataset	**method**	**Rank-1**(%)
ScanNet	deepSort	49.60
ScanNet	concat	**77.85**
matterport3D	deepSort	46.92
matterport3D	concat	**54.40**

Table 3.6: Scores on our re-OBJ data with Rank-1,-5,-10 and -20 accuracy values. The best performing type of setups is highlighted in bold.

method	**Rank-1**(%)	**Rank-5**(%)	**Rank-10**(%)	**Rank-20**(%)
OSNet	69	85.7	89	91.3
DGNet	58.3	76	83.7	92.4
re-OBJ	**77.85**	**91.55**	**-**	**98.36**

Chapter 4

3D Object Localization

Summary

In this chapter, we propose a probabilistic approach to recover affine camera calibration and objects position/occupancy from multi-view images using solely the information from image detections (Section 4.1). We show that remarkable object localisation and volumetric occupancy can be recovered by including both geometrical constraints and prior information given by objects CAD models from the ShapeNet dataset (Section 4.1.2 and 4.1.3). This can be done by recasting the problem in the context of a probabilistic framework based on PPCA that enforces both geometrical constraints and the associated semantics given by the object category extracted by the object detector. We present results on synthetic data and extensive real evaluation on the ScanNet datasets on more than 1200 image sequences to show the validity of our approach in realistic scenarios (Section 4.2). In particular, we show that 3D statistical priors are key to obtain reliable reconstruction especially when the input detections are noisy, a likely case in real scenes.

4.1 Quadrics Estimation from Conics

Consider a 3D scene with a set of rigid objects $i = 1...O$ detected in each of the frames $f = 1...F$. Each object i in every frame f is identified by a 2D bounding box B_{if} around the object obtained using a State-of-the-Art object detection algorithm such as YOLO [14]. We can define the bounding box as a set of parameters: $B_{if} = \{w_{if}, h_{if}, \mathbf{b}_{if}\}$, where w_{if} denotes the width of the bounding box, h_{if} denotes its height and $\mathbf{b}_{if}$ is a 2-vector defining the centre of the bounding box. The goal of the Localization-from-Detection (LfD) method [15] is to use these 2D bounding boxes for each object detected in multiple views to estimate the object's position and occupancy in the 3D scene. Rubino et al. in [15] formulated this problem as 3D quadrics estimation given the 2D detections in multiple views. In order to achieve this, a 2D ellipse $\hat{C}_{if}$ is fitted inside a 2D bounding box B_{if} around the object in each image. The LfD method aims to find a $\mathbb{R}^{4\times4}$ matrix representing a quadric Q_i whose projections onto the image planes best fit the 2D ellipses given by C_{if} to solve for both the 3D localization and the occupancy of each object detected in multiple views. C_{if} is a $\mathbb{R}^{3\times3}$ matrix with the homogeneous coordinates representing the 2D conics. Such an ellipse can be represented by a conic equation of a homogeneous quadratic form such as follows:

$$\mathbf{u}^\top C_{if}\ \mathbf{u} = 0, \tag{4.1}$$

where $\mathbf{u} \in \mathbb{R}^3$ is the homogeneous vector of a generic 2D point belonging to the conic defined by the symmetric matrix C_{if}. Similarly to the conics, the 3D ellipsoids in the 3D space can be represented with the homogeneous quadratic form of a quadric equation:

$$\mathbf{x}^\top Q_i\, \mathbf{x} = 0, \tag{4.2}$$

where $\mathbf{x} \in \mathbb{R}^4$ represents an homogeneous 3D point belonging to the quadric defined by the symmetric matrix Q_i. So, each conic $C_{if} \in \mathbb{R}^{3\times3}$ would be the projection of each quadric $Q_i \in \mathbb{R}^{4\times4}$ on the image plane.

However, the relationship between the conic C_{if} and the quadric Q_i is not straightforward in the Euclidean space of 3D points also called the primal space (2D points in the images), it is mathematically convenient to formulate them in the dual space, which is the space of the planes (lines in the images) [127]. In the dual space, a 2D conic can be represented as an envelope of all the lines tangent to the conic curve called as dual conic $C^*_{if} = adj(C_{if})$ and the 3D quadrics as an envelope of all the planes tangent to the quadric surface called dual quadrics $Q^*_i = adj(Q_i)$ [25]. The relationship between Q^*_i and C^*_{if} is defined by the orthographic projection matrix $P_f \in \mathbb{R}^{3\times4}$ as:

$$P_f = \begin{bmatrix} \mathbf{R_f} & | & \mathbf{t_f} \\ \mathbf{0}_3^\top & | & 1 \end{bmatrix} = \begin{bmatrix} p_{11} & p_{12} & p_{13} & p_{14} \\ p_{21} & p_{22} & p_{23} & p_{24} \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{4.3}$$

where $\mathbf{R_f} \in \mathbb{R}^{2\times3}$ is an orthographic rotation matrix such that $\mathbf{R_f}\mathbf{R_f}^\top = \mathbf{I}$ and the vector $\mathbf{t_f}$ represents the camera translation vector. The dual conic C^*_{if} and dual quadrics Q^*_i are defined upto an overall scale factor that can be normalized by fixing the $C_{3,3}$ and $Q_{4,4}$ elements to the value -1. Hence, the relationship between the dual quadrics and dual conics can be given as:

$$C^*_{if} = P_f Q^*_i P_f^\top \tag{4.4}$$

Now, in order to obtain the dual quadrics Q^*_i from Eq.(4.4), it is required to rearrange the equation into a factorization problem and solving by the vectoriza-

tion of symmetric matrices Q_i^* and C_{if}^* into $\dot{v}_i = vech(Q_i^*)$ and $C_{if}^* = vech(C_{if}^*)$, respectively. Then, arranging the products of the matrices P_f and $P_f^\top$ in a unique matrix $G_f \in \mathbb{R}^{6\times 10}$ given as follows:

$$G_f = A(P_f \otimes P_f^\top)B \tag{4.5}$$

where $\otimes$ is the Kronecker product and matrices $A \in \mathbb{R}^{6\times 9}$ and $B \in \mathbb{R}^{16\times 10}$ are two matrices such that $vech(X) = Avec(X)$ and $vec(X) = Bvech(X)$ respectively, where X is a symmetric matrix.The Eq.(2) can now be rewritten as:

$$\dot{\mathbf{c}}_{\mathbf{if}} = G_f \dot{\mathbf{v}}_{\mathbf{if}} \tag{4.6}$$

where $\dot{\mathbf{c}}_{\mathbf{if}}$ and $\dot{\mathbf{v}}_{\mathbf{if}}$ are $6F$ and $10F$ dimension vectors, respectively, containing the conics and the ellipsoids in all the frames, G_f is $6F \times 10$ matrix which contains the camera for each view. Rubino et al. [15] showed that it is possible to obtain a direct unique solution for the ellipsoids $\dot{\mathbf{v}}_{\mathbf{if}}$ given atleast three frames. By stacking the Eq. (4.6) column-wise for the frames $f = 1 \dots F$, with $F \geq 3$, the equation can be re-written as:

$$\mathtt{M}_i \mathbf{w}_i = \mathbf{0}_{6F}, \tag{4.7}$$

where $\mathbf{0}_l$ denotes a column vector of zeros of length l, and the matrix $\mathtt{M}_i \in \mathbb{R}^{6F\times(10+F)}$ and the vector $\mathbf{w}_i \in \mathbb{R}^{10+F}$ are defined as follow:

$$\mathtt{M}_i = \begin{bmatrix} G_1 & -\dot{\mathbf{c}}_{\mathbf{i1}} & \mathbf{0}_6 & \mathbf{0}_6 & \dots & \mathbf{0}_6 \\ G_2 & \mathbf{0}_6 & -\dot{\mathbf{c}}_{\mathbf{i2}} & \mathbf{0}_6 & \dots & \mathbf{0}_6 \\ G_3 & \mathbf{0}_6 & \mathbf{0}_6 & -\dot{\mathbf{c}}_{\mathbf{i3}} & \dots & \mathbf{0}_6 \\ \vdots & \mathbf{0}_6 & \mathbf{0}_6 & \mathbf{0}_6 & \ddots & \mathbf{0}_6 \\ G_F & \mathbf{0}_6 & \mathbf{0}_6 & \mathbf{0}_6 & \dots & -\dot{\mathbf{c}}_{\mathbf{iF}} \end{bmatrix}, \quad \mathbf{w}_i = \begin{bmatrix} \dot{\mathbf{v}}_{\mathbf{if}} \\ \boldsymbol{\beta}_i \end{bmatrix}, \tag{4.8}$$

with $\boldsymbol{\beta}_i = [\beta_{i1}, \beta_{i2}, \cdots, \beta_{iF}]^\top$. Since in the real world scenarios, a general purpose

object detector might be inaccurate in localizing the objects and consequently, the bounding box location and the window size and thus, the ellipses $\hat{C}_{if}$ computed might be inaccurate. In a similar manner, this will have an effect on the ellipse fitting inducing an error on the $\mathtt{C}_{if}$. For this reason, for $\tilde{\mathtt{M}}_i$, the matrix obtained from the object detections, a solution can be achieved by minimising:

$$\tilde{\mathbf{w}}_i = \arg\min_{\mathbf{w}} \|\tilde{\mathtt{M}}_i \mathbf{w}\|_2^2 \quad s.t. \ \|\mathbf{w}\|_2^2 = 1, \tag{4.9}$$

where the equality constraint $\|\mathbf{w}\|_2^2 = 1$ avoids the trivial zero solution. Using SVD on the $\tilde{\mathtt{M}}_i$ matrix, the minimisation problem from Eq. (4.9) can be solved by taking the right singular vector associated to the minimum singular value. The first 10 entries of $\tilde{\mathbf{w}}_i$ are the vectorised elements of the estimated dual quadric denoted by $\tilde{\mathbf{v}}^*_{if}$. To get back the estimated matrix of the quadric in the primal space, we obtain first the dual estimated quadric by:

$$\tilde{\mathtt{Q}}^*_i = vech^{-1}(\tilde{\mathbf{v}}^*_{if}) \tag{4.10}$$

and subsequently apply the following relation:

$$\tilde{\mathtt{Q}}_i = adj^{-1}(\tilde{\mathtt{Q}}^*_i) \tag{4.11}$$

where adj^{-1} denotes the inverse of the adjoint operator. To further overcome the inaccuracies in the bounding boxes leading to the ill-conditioning of the $\tilde{\mathtt{M}}_i$ matrix, the LfD method uses the fact that the translation parameters of the quadrics can be solved separately. In practice, it consists in solving a classical SfM problem where the 2D measurements correspond to the centres of the ellipses. The solution provides both the centre $\mathbf{t}^{\mathbf{e}}_{\mathbf{i}}$ of each ellipsoid $\dot{\mathbf{v}}_{\mathbf{if}}$ and the camera matrix G_f.

To solve for the remaining parameters, namely the shape and the orientation,

the ellipsoids are registered to the origin. Let us denote by $\mathbf{w_{if}}$ the 6 dimension vectorised quadric which has been registered from the non-translated quadric $\dot{\mathbf{v}}_{\mathbf{if}}$. It can be shown that with w_{if} and its corresponding centred conics $\dot{\mathbf{c}}_{\mathbf{if}}{}^{s}$, the factorization equation Eq. (4.6) becomes:

$$\dot{\mathbf{c}}_{\mathbf{if}}{}^{s} = G_f^s \mathbf{w_{if}} \tag{4.12}$$

where the $\dot{\mathbf{c}}_{\mathbf{if}}{}^{s}$ is a $(3F)$-vector containing all the centred conics for the object i observed in F frames and G_f^s is a $3F \times 6$ matrix obtained from G_f. In practice, it consists simply in selecting the rows and the columns related to the quadric shape and disregarding the ones related to the translation (that are equal to zero given the centring). At this point, the SfMO method consists in multiplying by the pseudo inverse of G_f^s to obtain the remaining quadric parameters $\mathbf{w_{if}}$. Our PSfMO approach differs from SfMO on this last step where we use a probabilistic factorisation to include our object prior.

4.1.1 Object pose and shape decomposition

We seek to perform an optimization constrained by a priory knowledge on the semi-axis lengths of each object. Let us first show how these lengths appear explicitly in the factorization problem defined by Eq. (4.6). A generic ellipsoid in dual space $\mathtt{Q}^*$ can be written as:

$$\mathtt{Q}^* = \mathtt{Z}\breve{\mathtt{Q}}^*\mathtt{Z}^\top \tag{4.13}$$

where $\breve{\mathtt{Q}}^*$ is an ellipsoid centred on the origin and with the axes aligned to the 3D coordinates and $\mathtt{Z}$ is an homogeneous transformation accounting for an arbitrary

rotation and translation. Then $\mathtt{Z}$ and $\breve{\mathtt{Q}}^*$ can be written as:

$$\mathtt{Z} = \begin{bmatrix} \mathtt{R}(\boldsymbol{\theta}) & \mathbf{t} \\ \mathbf{0}_3^\top & 1 \end{bmatrix}, \quad \breve{\mathtt{Q}}^* = \begin{bmatrix} a^2 & 0 & 0 & t_1^e \\ 0 & b^2 & 0 & t_2^e \\ 0 & 0 & c^2 & t_3^e \\ 0 & 0 & 0 & -1 \end{bmatrix}, \tag{4.14}$$

where $\mathbf{t}^e = [t_1,\, t_2,\, t_3]^\top$ is the translation vector, $\mathtt{R}(\boldsymbol{\theta})$ is the rotation matrix given by the Euler angles $\boldsymbol{\theta} = [\theta_1,\, \theta_2,\, \theta_3]^\top$ and $a,\, b,\, c$ are the three semi-axes of the ellipsoid.

Given the registered quadric $\mathbf{w_{if}}$ used in Eq. 4.12, the translation $\mathbf{t}$ can be then decoupled from the quadric matrix so we can register the ellipsoids to the origin by setting $\mathbf{t}^e$ to $\mathbf{0}$ and removing this variable from the equations. Now, let us examine the expression of the registered quadric $\mathbf{w_{if}}$ in more details. This vector can be expressed in terms of the remaining six quadric parameters. Defining the vector $\mathbf{e} \in \mathbb{R}^6$ as $\mathbf{e} = [\theta_1,\, \theta_2,\, \theta_3,\, a,\, b,\, c]^\top$, we can evaluate a functional form of the vector $\mathbf{w_{if}}(\mathbf{e})$ as follow:

$$\mathbf{w_{if}}(\mathbf{e}) = \begin{bmatrix} r_{11}(\boldsymbol{\theta})^2 a^2 + r_{12}(\boldsymbol{\theta})^2 b^2 + r_{13}(\boldsymbol{\theta})^2 c^2 \\ r_{11}(\boldsymbol{\theta}) r_{21}(\boldsymbol{\theta}) a^2 + r_{12}(\boldsymbol{\theta}) r_{22(\boldsymbol{\theta})} b^2 + r_{13}(\boldsymbol{\theta}) r_{23}(\boldsymbol{\theta}) c^2 \\ r_{11}(\boldsymbol{\theta}) r_{31}(\boldsymbol{\theta}) a^2 + r_{12}(\boldsymbol{\theta}) r_{32}(\boldsymbol{\theta}) b^2 + r_{13}(\boldsymbol{\theta}) r_{33}(\boldsymbol{\theta}) c^2 \\ r_{21}(\boldsymbol{\theta})^2 a^2 + r_{22}(\boldsymbol{\theta})^2 b^2 + r_{23}(\boldsymbol{\theta})^2 c^2 \\ r_{21}(\boldsymbol{\theta}) r_{31}(\boldsymbol{\theta}) a^2 + r_{22}(\boldsymbol{\theta}) r_{32}(\boldsymbol{\theta}) b^2 + r_{23}(\boldsymbol{\theta}) r_{33}(\boldsymbol{\theta}) c^2 \\ r_{31}(\boldsymbol{\theta})^2 a^2 + r_{32}(\boldsymbol{\theta})^2 b^2 + r_{33}(\boldsymbol{\theta})^2 c^2 \end{bmatrix} \tag{4.15}$$

From now, we will again denote this vector by $\mathbf{w_{if}}$ to simplify notations. We observe that it is possible to decompose it in the following way:

$$\mathbf{w_{if}} = R_{if}(\boldsymbol{\theta}) \mathbf{l}_{if}, \tag{4.16}$$

where $R_{if}(\boldsymbol{\theta})$ contains the orientation of the quadric and is of size 6×3:

$$R_{if}(\boldsymbol{\theta}) = \begin{bmatrix} r_{11}(\boldsymbol{\theta})^2 & r_{12}(\boldsymbol{\theta})^2 & r_{13}(\boldsymbol{\theta})^2 \\ r_{11}(\boldsymbol{\theta})r_{21}(\boldsymbol{\theta}) & r_{12}(\boldsymbol{\theta})r_{22}(\boldsymbol{\theta}) & r_{13}(\boldsymbol{\theta})r_{23}(\boldsymbol{\theta}) \\ r_{11}(\boldsymbol{\theta})r_{31}(\boldsymbol{\theta}) & r_{12}(\boldsymbol{\theta})r_{32}(\boldsymbol{\theta}) & r_{13}(\boldsymbol{\theta})r_{33}(\boldsymbol{\theta}) \\ r_{21}(\boldsymbol{\theta})^2 & r_{22}(\boldsymbol{\theta})^2 & r_{23}(\boldsymbol{\theta})^2 \\ r_{21}(\boldsymbol{\theta})r_{31}(\boldsymbol{\theta}) & r_{22}(\boldsymbol{\theta})r_{32}(\boldsymbol{\theta}) & r_{23}(\boldsymbol{\theta})r_{33}(\boldsymbol{\theta}) \\ r_{31}(\boldsymbol{\theta})^2 & r_{32}(\boldsymbol{\theta})^2 & r_{33}(\boldsymbol{\theta})^2 \end{bmatrix}, \tag{4.17}$$

and $\mathbf{l}_{if} = [a^2, b^2, c^2]^\top$ contains the three axis lengths. This provides the separation between the rotational pose and ellipse shape components thus making possible to impose the priors over the axes. In the following and unless stated otherwise, we will denote the orientation matrix with R_{if} to simplify the notation.

4.1.2 Object 3D prior

The decomposition in Eq. 4.16 shows that the shape of the ellipsoid is given by the square of the axis length $\mathbf{l}_i$ and we will now describe how to build a prior distribution on such axes. The prior is given by statistics on the dimensions of the objects collected from the ShapeNet dataset [44]. For each object i present in the dataset, we extract a 3 dimensional vector $\mathbf{h}_i = [a, b, c]^T$ containing the lengths along the main axes. These values have been normalised so that $|\mathbf{h}_i|^2 = 1$ and thus providing the information about the ratio of these axes. From the normalisation, we can deduce that the corresponding vector $\mathbf{l}_i$ lies in a 2-dimensional space. For each object category c, we first use PCA to compute a 3x2 projection matrix $\mathtt{V}_c$ and a mean $\boldsymbol{\mu}_c$ to project each vector $\mathbf{l}_i$ in the 2D space. Secondly, we fit a 2D Gaussian parameterized by $(0, \Sigma_c)$. This Gaussian encodes our *a priori* information about the ratio of the different dimensions of the object. Fig. 4.1 shows an example in the case of the bottle category where we can notice that the

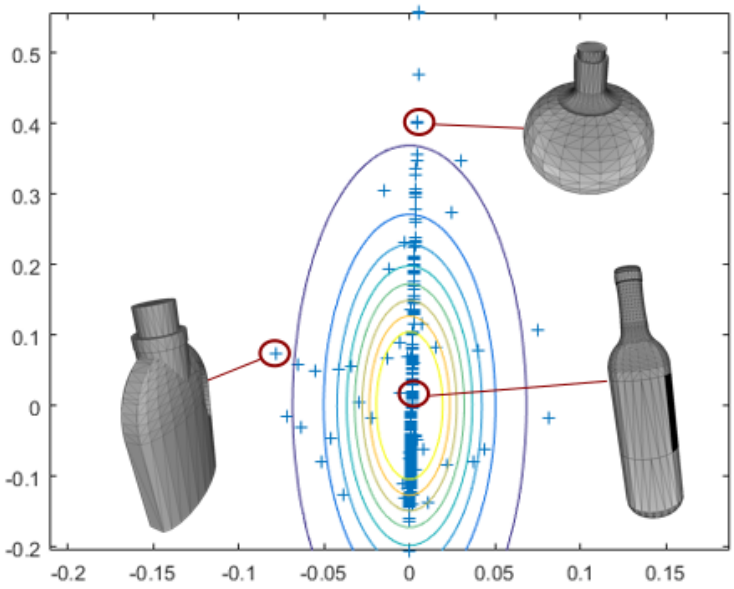

Figure 4.1: The contour lines are the Gaussian sections of the prior for the bottle category. Each blue cross corresponds to a CAD model extracted from the ShapeNet database.

horizontal dimension accounts for the ratio between x and y axes and the vertical one for the ratio between the bottle diameter and its height.

In order to apply the prior on test images, we must select one of the six possible correspondences between the axes of the estimated ellipsoids and the prior axes i.e. one of the six possible permutations of the rows of $\mathtt{V}_c$ and $\boldsymbol{\mu}_c$. In practice, we estimate the axis lengths for the 6 possible configurations and keep the one with the higher likelihood. Lastly, we must estimate a scale factor z to account for the scale difference between the normalised axes of the prior and the non normalised axes of the ellipsoids we wish to estimate. This last variable is estimated from the data as described in the following section.

The other parameters such as the camera translation and orientation are estimated by our method directly from the data using multi-view relations.

4.1.3 Probabilistic PCA with shape priors

From a generative point of view, our Probabilistic Principal Component Analysis (PPCA)[1] model assumes that each centred conic vector $\dot{\mathbf{c}}_{\mathbf{if}}^{s}$ is obtained by sampling a 2-dimensional latent random vector s_{if} and an additive Gaussian noise ϵ such that:

$$\dot{\mathbf{c}}_{\mathbf{if}}^{s} = G_f^s \, R_{if} \, z_{if} \, (\mathtt{V}_{c_{if}} s_{if} + \boldsymbol{\mu}_{c_{if}}) + \epsilon, \tag{4.18}$$

where $\epsilon \hookrightarrow N[0, \sigma \mathtt{I}d^{3F}]$. The hidden variable s_{if} encodes the axis lengths, and thus the ellipsoid shape, that we have to estimate. $\boldsymbol{\mu}_{c_i}$ and $\mathtt{V}_{c_i}$ are used to reproject the s_{if} into the 3 dimensional axis length vector, where c_i is the category of the object i. Conversely from the standard PPCA, we have a latent additional variable z_{if} which accounts for the scale difference between the prior and the reconstructed ellipsoid. As in the decomposition of Eq. (4.16), R_{if} is the rotation matrix defined in Eq. 4.17, and G_f^s is the camera matrix defined in Eq. 4.12. Inference and parameter estimation with PPCA is usually done with an *EM* algorithm. In the *E-step*, we estimate sufficient statistics from the posterior distribution $P(s_{if}|\dot{\mathbf{c}}_{\mathbf{if}}^{s})$. In the *M-step*, we compute the noise variance parameter σ.

The posterior over the latent variables can be written as:

$$P(\mathbf{s}_{if}, z_{if}|\dot{\mathbf{c}}_{\mathbf{if}}^{s}) \propto P(\dot{\mathbf{c}}_{\mathbf{if}}^{s}|s_{if}, z_{if}) P(s_{if}) \tag{4.19}$$

From Eq. 4.18, we can write that:

$$P(\dot{\mathbf{c}}_{\mathbf{if}}^{s}|s_{if}, z_{if}) = N(x_i, \sigma \mathtt{I}d^{3F}), \tag{4.20}$$

where we used the notation $x_i = G_f^s R_{if} z_{if} (\mathtt{V} s_{if} + \boldsymbol{\mu}_{c_i})$. It has been said in

[1] For a more general and extensive description we refer to [43].

Sec. 4.1.2 that $P(s_{if})$ is the Gaussian $N(0, \Sigma_{c_i})$ where c_i is the category of object i. Skipping the constants that do not depend on the latent variables, the logarithm of the posterior can be written as:

$$log(P(s_{if}, z_{if}|\dot{\mathbf{c}_{\mathbf{if}}}^s)) \propto (\mathbf{x}_i - \dot{\mathbf{c}_{\mathbf{if}}}^s)\frac{1}{\sigma} Id^{3F}(\mathbf{x}_i - \dot{\mathbf{c}_{\mathbf{if}}}^s) + s_{if}^T \Sigma_{c_i}^{-1} s_{if}, \tag{4.21}$$

It can be noticed from this formula that the log likelihood is simply a sum of two terms. The first one accounts for the reprojection error with respect to the observed conics $\dot{\mathbf{c}_{\mathbf{if}}}^s$, and the second one refers to the prior. In particular, the noise covariance parameter σ can be seen as a trade-off parameter between the two terms which is automatically estimated from the data. Since this distribution is intractable, we resort to Markov Chain Monte Carlo [128] (MCMC) to estimate the expectation of the latent variables $\{\hat{z}, \hat{s_{if}}\}$ under the posterior. The last element to estimate is the matrix containing the ellipsoid orientation $\mathtt{R}_i(\boldsymbol{\theta})$, defined in Eq. (4.17) for which we are now making explicit the dependence to the three Euler angles $\boldsymbol{\theta}$. One solution would be to include them as additional variables in the posterior, but we observe that the MCMC estimation becomes unreliable. Instead, we propose to estimate these angles by solving the following optimisation problem:

$$\hat{\boldsymbol{\theta}} = \arg\min_{\boldsymbol{\theta}} \|\hat{\mathbf{x}}_i - \dot{\mathbf{c}_{\mathbf{if}}}^s\|^2 \tag{4.22}$$

where $\hat{x}_i = G_f^s R_{if}(\boldsymbol{\theta}) \hat{z_{if}}(\mathtt{V}\hat{s_{if}} + \mu)$. We optimise the cost function with non-linear Least Squares by parameterizing rotations with quaternions. From the solution $\hat{\boldsymbol{\theta}}$, we can obtain our current quadric estimate as:

$$\hat{\mathbf{w}}_i = R_{if}(\hat{\boldsymbol{\theta}})\ \hat{s_{if}}\ \hat{z} \tag{4.23}$$

In the *M-step*, we use our estimate of $\hat{\mathbf{w}}_i$ to estimate the noise covariance

parameter σ given by:

$$\hat{\sigma} = \frac{1}{3F} \sum_{i=1}^{O} \{ ||\dot{\mathbf{c}}_{\mathbf{if}}{}^{s}||^2 - 2\hat{\mathbf{w}}_i^{\top} G_f^s \dot{\mathbf{c}}_{\mathbf{if}}{}^{s} + \text{trace}(\hat{\mathbf{w}}_i^{\top} \hat{\mathbf{w}}_i G_f^{s\top} G_f^s) \}. \tag{4.24}$$

Initialisation. Given the ellipses in multiple views, we first apply the SfMO method to obtain the camera matrix G_f^s and estimation of the ellipsoid orientation $\boldsymbol{\theta}$. We use these values as an initialisation for the PSfMO method.

4.2 Experimental Details

Our experiments aim at evaluating the proposed PSfMO approach against the prior-less SfMO method [6]. We test the behaviour of both methods in presence of noise in a synthetic setting and show results on real examples with the KINECT [69] and the ScanNet datasets [7]. We collected CAD models from the ShapeNetCore dataset.

4.2.1 Dataset

4.2.1.1 Experiments on synthetic data

We generated ellipsoids, randomly placed inside a cube of side 20 units. We generated a set of 5 objects drawn randomly from the categories present in our CAD collection. The length of the ellipsoid axes for each object category c were generated from Gaussian distributions obtained with the statistics $\hat{\boldsymbol{\mu}}_c$ and $\hat{\Sigma}_c$ that we computed from the CAD collection.

A set of 20 camera views were generated with a camera trajectory computed such that azimuth and elevation angles span the range $[0°, 10°]$. Given the orthographic camera matrix projection $\mathtt{P}_f$ at each camera frame, ground-truth ellipses were calculated from the exact projections of the 3D quadrics.

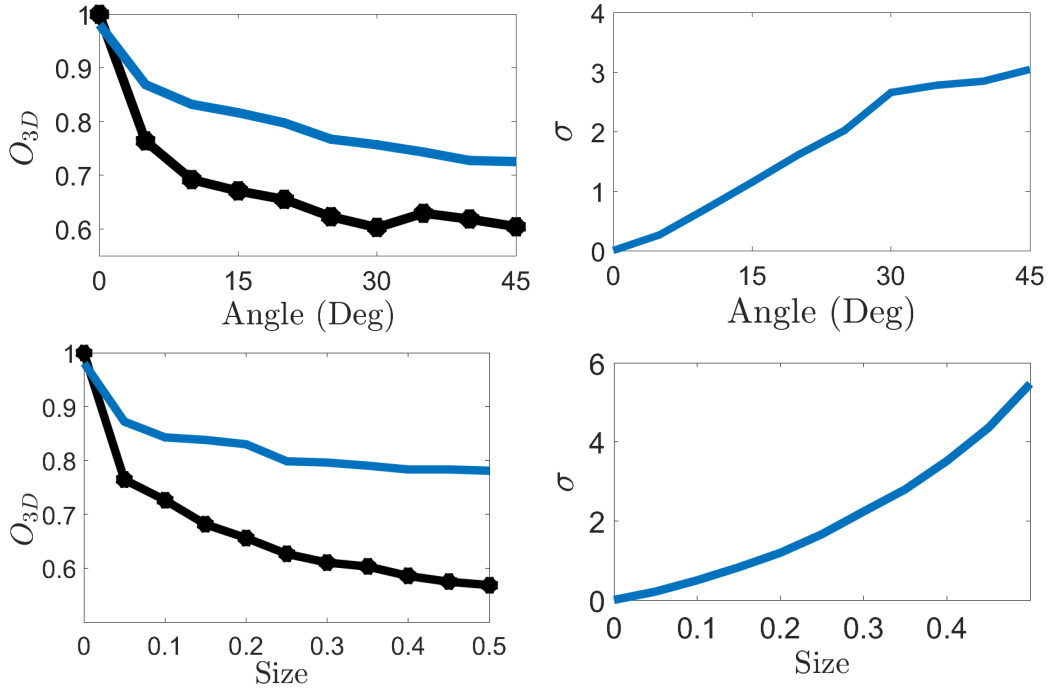

Figure 4.2: Top images: Comparison of the volume overlap O_{3D} and noise variance σ given rotation error of the ellipse for the SfMO (black dotted line) and the PSfMO (blue line) methods. Bottom images: values for O_{3D} and σ in the case of size errors in the ellipsoid.

The ellipses were corrupted with two types of errors on rotation and size. To impose such errors, the axis length l_1, l_2 and the orientation α of the first axis were perturbed as follows:

$$\hat{l}_j = l_j \left(1 + \nu^l\right), \qquad \hat{\alpha} = \alpha + \nu^\alpha, \tag{4.25}$$

where $\nu_j^{\mathbf{t}}$, ν^α and ν^l are random variables with uniform PDF and mean value equal to zero, and $\bar{l} = (l_1 + l_2)/2$.

4.2.1.2 Performance Evaluation

In order to highlight the specific impact of each error, they were evaluated separately. Error magnitudes were set tuning the boundary values of the uniform PDFs of $\nu_j^{\mathbf{t}}$, ν^{α} and ν^{l}. In detail, for each kind of error, we considered 10 different values of $\nu_j^{\mathbf{t}}$, ν^{a} and ν^{l}, with uniform spacing, and we applied the resulting error realisations to the ellipse reprojections related to all the ellipsoids. We run 100 trials for each setup, described by the number of objects and error on ellipses. The accuracy of the estimated 3D object position and pose was measured by the volume overlap $O_{\mathtt{3D}}$ given by the intersection between ground-truth (GT) and estimated (ES) ellipsoids respectively:

$$O_{\mathtt{3D}} = \frac{1}{N}\sum_{i=1}^{N}\frac{\mathcal{Q}_i \cap \tilde{\mathcal{Q}}_i}{\mathcal{Q}_i \cup \tilde{\mathcal{Q}}_i}, \tag{4.26}$$

where $\mathcal{Q}_i$ and $\tilde{\mathcal{Q}}_i$ denote the volume of GT and ES ellipsoids in the dual space respectively thus the metric evaluating an algebraic error. It might happen that the quadrics obtained by the methods do not correspond to a valid ellipsoid. In this case, we consider the test as failed. In Fig. 4.2, we reported the results for both methods SfMO and PSfMO in terms of 3D overlap $O_{\mathtt{3D}}$.

In the absence of noise, SfMO retrieves nearly perfect ellipsoids thanks to the closed-form solution. In this case, including prior information can only degrade slightly or produce similar performances. However, PSfMO is able to retrieve better ellipsoids when the ellipse orientations and sizes are corrupted. This effect is more important for the size error. This effect is expected, our prior information has a particular effect on the estimation of the axis lengths. We also observe that as the noise increases, the estimation of σ produces higher values, meaning that the method relies more on the prior. This indicates that the automatic trade-off between data and prior is effective and working as expected.

4.2.1.3 Experiments on ScanNet dataset

We provide exhaustive experimental evaluation over the ScanNet dataset [7] which consists of 1500 indoors RGBD scans annotated with 3D camera poses, surface reconstructions, and mesh segmentation related to several object categories. Such annotations enable us to construct 3D ground-truth (GT) data by fitting ellipsoids to each object's mesh so evaluating the accuracy of PSfMO in realistic environments. In more details, the GT ellipsoid of an object is computed as the one which includes all the labelled 3D points in the mesh and has minimal volume. Given the available camera matrices, we can then reproject each object's mesh onto the image plane so localising the object as a 2D point cloud. Then, we draw a bounding box around the obtained 2D point cloud and consider this as a putative object detection for our system. From the bounding box, we finally construct an ellipse by taking the centre of the bounding box, its axis lengths as the width and high of the box and its orientation aligned to the axes of the box.

We also take into account occlusions when computing the bounding boxes. As occluded objects are typically missed by object detectors, we impose that at least 80% of the object surface should be visible, i.e. not occluded by another object, in the current frame to be detected. Moreover, we only keep images when more than three objects are appearing and we set the minimum length of a sequence to three frames. This experimental protocol provides us with 1217 sequences overall.

4.2.1.4 Performance Evaluation

As with the synthetic experiments, we compare the SfMO and the proposed PSfMO with the O_{3D} overlap measure. As it is in standard in 3D reconstruction, we use the Procrustes method to align the centres $\mathbf{t}_i^e$ of the estimated ellipsoids with the centres of the GT ellipsoids and then proceed with the error evaluation.

In Fig 4.3, we show the O_{3D} for the ten object categories most represented in

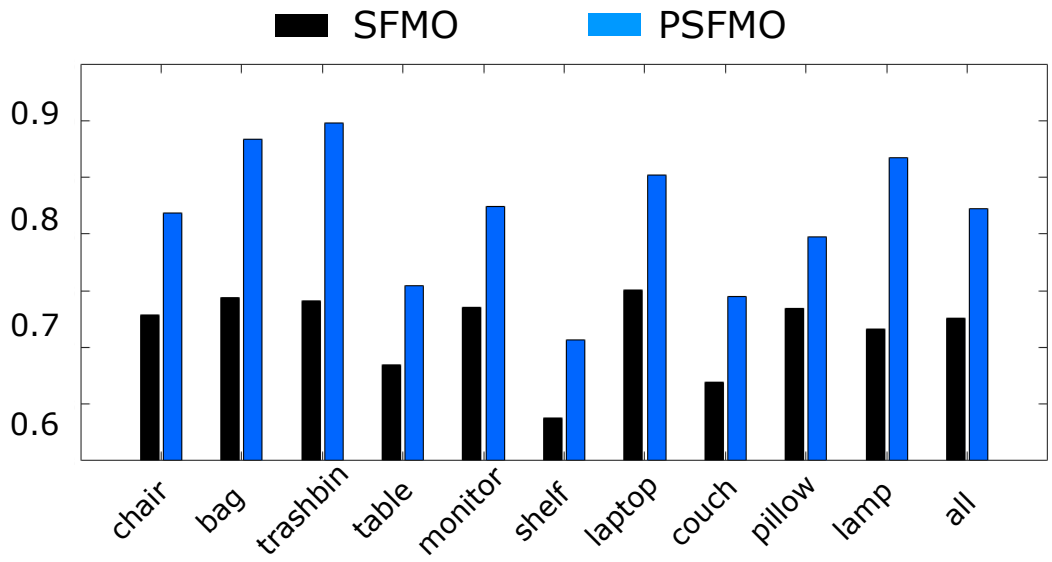

Figure 4.3: Each couple of bars shows the accuracy for both SfMO and PSfMO methods for the top ten most prominent object categories in our dataset.

our sequences (83% of the objects) as well as the average for all categories. The results confirm the findings of the synthetic experiments: The PSfMO method shows the best performances by using the prior built on the ShapeNet dataset. The average O_{3D} for all the 1217 sequences is 0.72 for SfMO and 0.82 for PSfMO. We can see that the prior is especially useful for category with low intra-class variability. For instance the dimensions of a trash bin category is rather constrained as x and y-axes have usually equal lengths. On the opposite, categories such as table and couch have higher variability.

We have also evaluated the camera estimation by using the GT perspective camera matrices provided by the ScanNet dataset. To recover camera poses in the affine case we solved a PnP problem [129] given the affine camera intrinsic, the 3D GT ellipsoid centres and the 2D noisy conic centres. We then align the estimated camera poses with the given GT poses. The relative average translation error (normalised by the sequence length) is 20% and the rotation error is 32 degree on average. These errors might be relevant for some applications and future work

on better camera estimation could improve the overall results.

The figure 4.4 shows an illustration of the results. We can see that SfMO has difficulties when estimating the ellipsoid axis which is more parallel to the optical camera axis. By using prior knowledge, PSfMO retrieves more accurate ellipsoids and constrain the shape to the right occupancy. Due to the ambiguity of the problem, there are many ellipsoid configurations which can project closely to the observed ellipses. Our PSfMO method selects the one which is the closest to the prior. Similar results can be noticed in Fig. 4.5 where the camera has a minimal baseline and rotation, thus making more challenging the accurate estimation of the objects size in the scene.

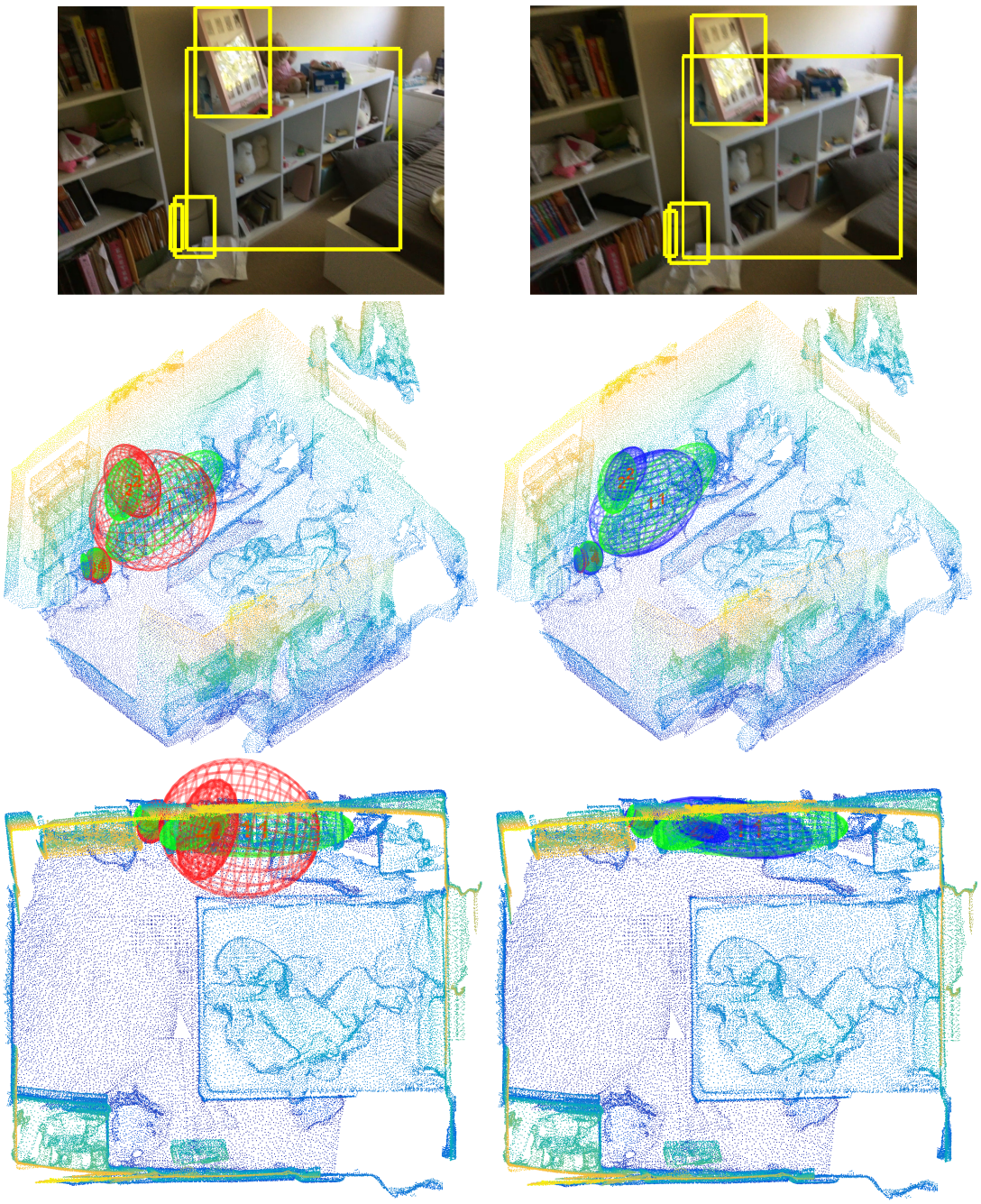

Figure 4.4: Illustration of results with ScanNet data. The top rows are color frames with object detections. The midddle and bottom rows display the scan with the ellipsoids. The green are the ground-truth, the red ones on the left are obtained with SfMO and the blue ones on the right with our method PSfMO.

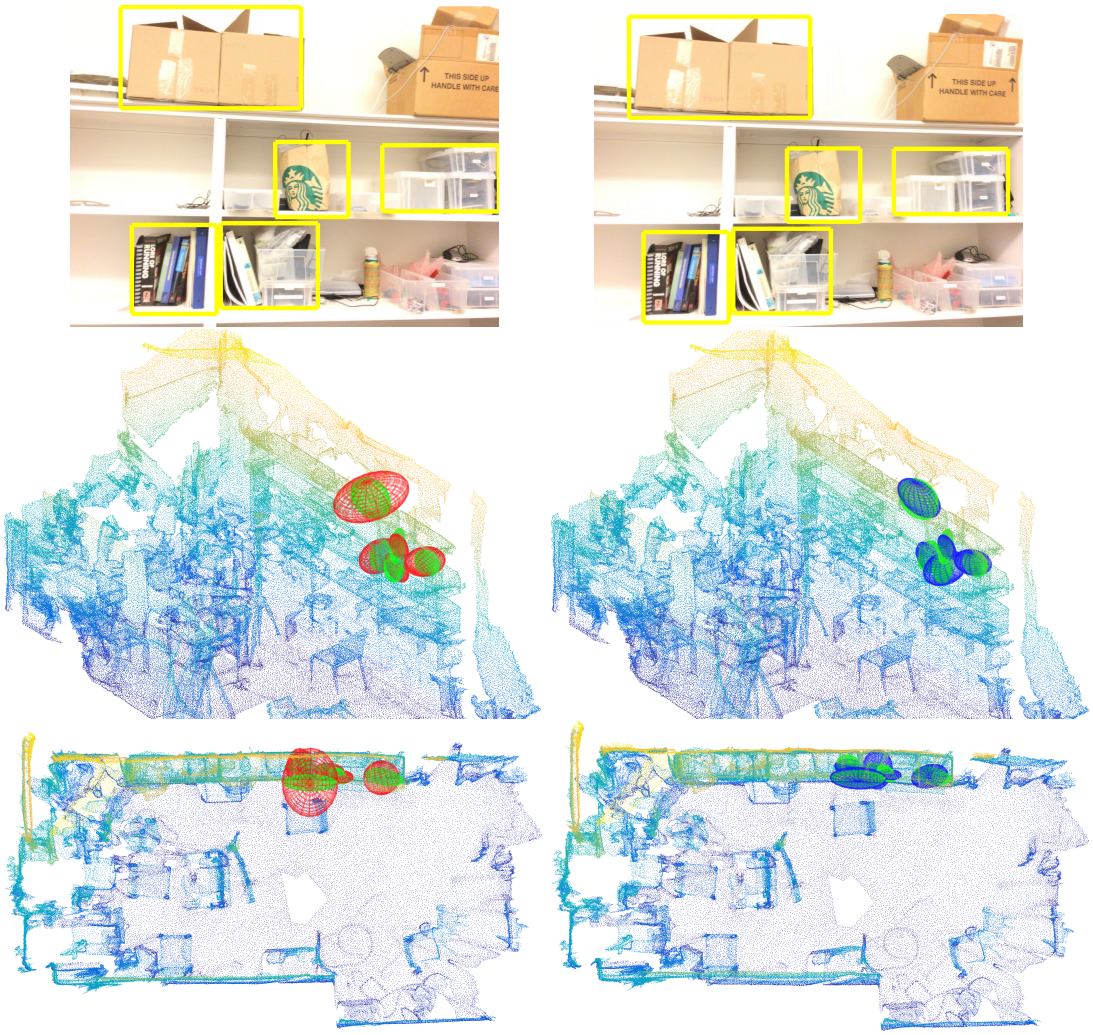

Figure 4.5: Illustration of results with ScanNet data. The top rows are color frames with object detections. The middle and bottom rows display the scan with the ellipsoids. The green are the ground-truth, the red ones are obtained with SfMO and the blue ones with PSfMO.