# Object-centric Mapping

Kejie Li
School of Computer Science
University of Adelaide

A thesis submitted for the degree of
*Doctor of Philosophy*

Supervised by:
*Prof. Ian D. Reid*
*Prof. Tat-jun Chin*

July 2021

I acknowledge the support I have received for my research through the provision of an Australian Government Research Training Program Scholarship.

# Acknowledgements

This thesis would not have been possible without the support of many people. Firstly, I would also like to extend my deepest gratitude to my principal supervisor, Prof. Ian Reid, for giving me this opportunity of working with him and his support and invaluable guidance in the past four years.

I am also grateful to many brilliant and kind people that have led me through this journey in Adelaide. Tat-Jun Chin has offered insightful discussions, especially at the beginning of my PhD. Dr. Trung Pham guided me through my first research project in this journey. Dr. Ravi Garg was generous with his time during our collaboration. Dr. Hamid Rezatofight kindly spent long hours on the whiteboard to explain mathematical concepts. Many thanks to Dr. Yasir Latif, and Dr. Saroj Weerasekera. I often would have a new perspective, leading to creative solutions to a problem even in our casual coffee talks. I would also like to express my gratitude to my labmates and friends, Huangying Zhan, Ming Cai, Mehdi Hosseinzadeh, Rafael Felix, Chin Fang Ch'ng, Chee Kheng Ch'ng, for the joyful company during the night hours and crunch mode. Again, I thank them all for their patience, support, and help.

During my 6-month internship at Facebook Reality Lab, I was fortunate to meet many brilliant minds. I would like to extend my deepest gratitude to my mentor at Facebook Reality Lab, Dr. Julian Straub, for all his guidance during the internship. His trust and support were fundamental to my success in my internship. I would also like to extend my sincere thanks to Martin Runz, Meng Tang, Chen Kong, Lingni Ma, Tanner Schmidt, and Steven Lovegrove for many fruitful discussions and insightful advice. Many thanks also go to Carl Ren, Chen Liu, and Xiaqing Pan for supporting at the beginning of my internship.

2020 is undoubtedly an unusual and challenging year for everyone in the world. I would like to express my gratitude to my housemates, Huangying Zhan, and Peishen Liu, who gave me tremendous support during the lockdown. Your companionship would always have a special place in my memory.

I'm extremely grateful to my parents and family for all their unconditional love throughout my life. Lastly, words cannot describe how grateful I am to my wife, Shuman Liu, to whom I dedicate this thesis, for all the love and support. I thank you for being the source of happiness and positive energy even during the most challenging times in this journey.

# Abstract

This thesis focuses on building an object-centric 3D map given an RGB image sequence, in which the basic elements are object instances. This is fundamentally different from conventional visual Simultaneous Localisation and Mapping (SLAM) that describes the geometry of an environment using geometric entities, such as 3D points, voxels or surfels. Representing an environment at the level of objects captures both the semantic and geometric information of an environment. It is more natural, compact, and closer to how human beings perceive the environment. Specifically, we investigate methods where well-studied geometry and deep learning can be combined to achieve object-centric mapping in general scenes.

We first build upon recent advances in deep learning for single-view object reconstruction, a task of recovering full 3D object shape from a single RGB image. An open question when using deep networks to solve this question is how to generate object shape efficiently. To this end, we propose a novel multi-view representation to generate dense point cloud efficiently. Although this pure deep learning paradigm shows impressive results on synthetic data, the lack of a large amount of annotated real images leads to a domain gap when inference on real images.

We then introduce a new single-view object reconstruction method by combining well-studied geometry and a deep learned shape prior. This new approach optimises an object's shape and pose using both 2D image cues, such as object silhouette, and constraints on learned object shape prior at inference time. Although we only address the single-view object reconstruction in this work, the online refinement makes it straightforward to incorporate more observations.

We introduce our first object-centric mapping system – FroDO (From Detections to Objects) based on our works on single-view object reconstruction. It takes as input an RGB image sequence and infers object location, pose, and shape in a coarse-to-fine manner, meaning that we reconstruct an object-centric map starting from a set of 2D object detections, through a 3D bounding box, to a sparse point cloud, and a dense mesh progressively. Although FroDO shows promising results on general and cluttered indoor scenes, it is neither an online system nor capable of handling object motions.

To address the limitations of FroDO, we subsequently present MO-LTR (Multiple Object Localisation, Tracking, and Reconstruction). It combines a monocular object

detector for object pose and scale prediction, a shape embedding network for shape modelling, and an IMM filter for tracking. Although each component's contribution is relatively incremental, as a system, it achieves dynamic object-centric mapping for both indoor and outdoor scenes.

# Contents

# List of Figures

# List of Tables

# 1 Introduction

## Contents

## 1.1 Background

Simultaneous Localisation and Mapping (SLAM) is the problem of localising a moving agent in a previously unknown environment and reconstructing the surrounding simultaneously given sensor information. There is a wide range of sensors that can be used for SLAM, but in the domain of this thesis, we focus on visual SLAM that employs a single camera as the input sensor. One of many challenges of SLAM is that the agent has no *a priori* knowledge of the environment. To localise itself, it has to incrementally create a persistent representation (*i.e.* mapping) of the environment as it moves.

In early SLAM systems, the mapping is to serve localisation. To make the system work more efficiently given limited computation power at the time, a common choice of map representation is a set of sparse 3D feature points as landmarks for tracking

(Davison et al., 2007; Klein et al., 2007). However, the sparse reconstruction cannot provide information about free space in the environment, which is a limitation for applications involving path planning and obstacle avoidance. By leveraging increased computation power from GPU, a number of dense SLAM systems are introduced where dense TSDF (Newcombe et al., 2011a) or surfels (Whelan et al., 2015) is used for reconstruction.

The study of SLAM has achieved a level of sophistication leading to a few commercial products in our daily life – autonomous vacuum cleaners where SLAM techniques are used to assist path planning, and AR/VR headsets where SLAM is used to track device poses to name a few. However, advanced intelligent systems require a deeper understanding of the environment not only from the geometric perspective but also from the semantic perspective. For instance, a service robot that does housework will need to recognise objects and understand their shape for interaction. In the AR/VR domain, when a user interacts with an object, the system has to track the dynamic element as a whole (by understanding it is an object). Furthermore, parsing the surroundings semantically will be able to facilitate semantic-based reasoning, which exploits the relationship between semantic entities (*i.e.* object co-existence, affordance) for better mapping, navigation, and interaction with human beings.

To bring the aforementioned futuristic applications closer to reality, a critical component of an intelligent system is the capability of understanding the environment semantically at an object level. This involves both recognition and reconstruction, where the system needs to identify semantically meaningful entities in the environment and recover their 3D structure. The task of building an object-centric reconstruction that consists of separate and meaningful object entities is termed as object-centric mapping in this thesis.

## 1.2 Motivation

This thesis aims to develop an object-centric mapping system in a relatively simple sensor setup (*i.e.* a monocular RGB camera). Despite early attempts, with the rise of deep learning, we believe it is the right time to explore ways, where traditional geometry and modern deep learning can be combined, for a robust and generic object-centric mapping system. There are two strong themes in this thesis. Firstly, we investigate how deep learning can be used in the task of object shape reconstruction. The second is building an object-centric mapping system that leverages the deep-learning recognition and reconstruction modules.

The capability of object recognition is a distinctive feature of object-centric mapping from geometric-only mapping. In recent years, we have witnessed impressive progress of deep learning in 2D recognition tasks(*e.g.* image classification, object detection, and instance segmentation), and deep networks (Convolution Neural Networks (CNN) to be precise) have become the standard approach for these tasks. An increasingly popular research trend is applying deep learning to geometry problems such as single view depth estimation, surface normal estimation, and visual odometry.

From the perspective of object-centric mapping, we are specifically interested in object shape inference using deep networks. Humans can effortlessly infer the 3D structure of objects from incomplete viewpoints or even just from a single observation. This leads us to the first question we would like to answer in this thesis: can we use a deep network to infer the complete 3D shape of objects from limited viewpoints or even just a single image? Furthermore, given that shape reconstruction using a deep network is a mapping function from image space to shape space, what constraints would be needed to ensure that the reconstructed shape is consistent with multi-view geometry, and how to update the shape when more observations are available?

Moving from the task of shape reconstruction to an object-centric mapping system, we here highlight a few key features that we aim to achieve with our new paradigm of object-centric mapping:

- It employs deep nets as "sensors" for object recognition and reconstruction. By processing the raw sensor data (*i.e.* RGB images), deep nets are used to identify meaningful objects and provide an initial guess for object reconstruction.

- It is an online system, which means the system can only use data received up to the current state, and the map is incrementally built. Therefore, it is important for an online system to be able to refine object reconstruction when new observations become available.
- It is capable of modelling object motion independently because humans could move objects during an interaction.

## 1.3 Contributions and Thesis Outline

We first dedicate Chapter 2 to review related works in object shape reconstruction from image(s), semantic SLAM, and object-centric mapping.

In Chapter 3, we present our first attempt to applying deep learning on object shape reconstruction from a single view observation. This work comes at a time when researchers started to use deep networks to learn a mapping from image(s) to object shape. An open question is: "what is an efficient object shape representation for network training?" To this end, we propose a novel and highly efficient deformed multi-view depth map representation for object shape. We also show how a neural network can be trained with this novel representation with a multi-view consistency loss. Given a single RGB image of an object, the network can predict a dense point cloud that models an object surface by fusing depth maps from multiple canonical viewpoints.

Although the work described in Chapter 3 shows promising results on object shape reconstruction from a single image, the object shape reconstruction by a feed-forward only neural network is undesirable for an object-centric mapping system for two reasons. Firstly, the shape reconstruction cannot be incrementally refined when more images are available. Secondly, a well-known problem of deep nets is the lack of transparency (*i.e.* a function approximator in a black box). It is not bounded by any geometry constraints used in geometry-based reconstruction methods (*e.g.* photometric consistency or silhouette consistency). Chapter 4 aims to develop a better shape reconstruction module to overcome these limitations. Instead of using a deep network as a direct mapping from raw image observation(s) to an object shape, we leverage the non-linear dimension reduction of deep networks to learn a shape embedding. We cast the single-view object shape reconstruction as an optimisation problem where we minimise the discrepancy between the reconstructed 3D shape and visual observations by changing the latent variable (*i.e.* shape code) in the embedding. To enforce prior knowledge of object shape in the shape embedding, we show that a simple Gaussian Mixture Model (GMM) prior fitted in the space can help predicting plausible 3D object shapes from just single-view observation.

We present our first object-centric mapping solution — FroDO in Chapter 5 based on the optimizable shape reconstruction proposed in the previous chapter. FroDO takes as input RGB sequences of real-world multi-object scenes and infers an object-centric map, leveraging 2D recognition, learning-based 3D reconstruction, and multi-view optimisation with shape priors. We also introduce a novel deep joint shape embedding that allows simultaneous decoding to sparse point cloud and continuous SDF representations and enables faster shape optimisation in a coarse-to-fine manner.

Although FroDO can generate object object-centric maps, it is limited to static environments, and its data association module works in an offline fashion. It is clear that object motions would exist in many robotic applications. Unlike some traditional SLAM approaches that consider object motions as outliers, it is crucial to capture these motions in object-centric mapping. In Chapter 6, we explore ways to handle dynamic objects and develop a truly online system. To this end, we propose MO-LTR, a unified framework for object-centric mapping, which can localise, track, and reconstruct multiple objects in an online fashion given monocular RGB image sequences. The proposed system, by combining deep learned 3D detection and shape prior, and multiple model Bayesian filter, achieves superior performance in object localisation, tracking, and reconstruction.

We summarise the research presented in the thesis and discuss future research direction in Chapter 7.

## 1.4 Publications

The research described in this thesis has led to the following articles that have been published, or under review.

- **Kejie Li**, Trung Pham, Huangying Zhan, Ian Reid. "Efficient Dense Point Cloud Object Reconstruction Using Deformation Vector Fields" Proceedings of the European Conference on Computer Vision (ECCV) 2018
- **Kejie Li**, Ravi Garg, Ming Cai, Ian Reid. "Single-view Object Shape Reconstruction Using Deep Shape Prior and Silhouette" 30th British Machine Vision Conference (BMVC) 2019
- Hosseinzadeh Mehdi, **Kejie Li**, Yasir Latif, Ian Reid. "Real-Time Monocular Object-Model Aware Sparse SLAM" International Conference on Robotics and Automation (ICRA) 2019
- **Kejie Li***, Martin Rünz*, Meng Tang, Lingni Ma, Chen Kong, Tanner Schmidt, Ian Reid, Lourdes Agapito, Julian Straub, Steven Lovegrove, Richard Newcombe. "FroDO: From Detections to 3D Objects" Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) 2020
- **Kejie Li**, Hamid Rezatofighi, Ian Reid. "MO-LTR: Multiple Object Localisation, Tracking and Reconstruction from Monocular RGB Videos". Accepted in IEEE Robotics and Automation Letters (RAL)

* indicates equal contribution

# 2

# Literature Review

## Contents

*The purpose of this chapter is to provide a broad overview of techniques that are relevant to object-centric mapping, examine the advantages and disadvantages of previous approaches, and highlight the novelties of our works. We start with approaches that use deep learning for recognition and geometry problems, followed by a detailed discussion on object reconstruction from images. We then discuss the evolution of geometric mapping systems from sparse to dense representations. Motivated by the limitation of geometric-only mapping, we discuss recent developments in semantic mapping and object-centric mapping.*

## 2.1 An Overview of Deep Learning

Deep learning has revolutionised the field of computer vision in recent years. This section covers significant developments of deep learning and its relation to the works described throughout the thesis.

### 2.1.1 Deep learning for recognition

AlexNet (Krizhevsky et al., 2017) first shows the potential of deep learning on image classification when a large amount of labelled data is available for training in the early 2010s. Deeper and more powerful network architectures (generally referred to backbones), such as VGG (Simonyan et al., 2014), and ResNet (He et al., 2016), further improve deep network performance.

Soon after its success in image classification, deep learning has been applied to more recognition tasks. Object detection is a task to classify and localise the extent of objects of interest within an image via bounding boxes. RCNN (Girshick et al., 2015) proposes a selective search algorithm to generate regions of interest within an image. Object detection can then be achieved by image classification in each region of interest. Fast RCNN (Girshick, 2015) introduces an ROI pooling operation, where features for each region proposal are processed in a single forward pass, to achieve faster inference and more surpassing performance than RCNN. Faster RCNN (Ren et al., 2016), taking a step forward to integrate region proposal into the deep network, is differentiable from predictions to the input image, and thus can be trained end-to-end. They showed that this end-to-end approach outperformed previous methods in terms of both speed and accuracy. Mask RCNN (He et al., 2017), tackling another recognition task – instance segmentation – is based on Faster RCNN. It generates a binary mask as an object silhouette for each detected object. We use Mask RCNN as our recognition module in Chapter 5.

More recently, DETR (Carion et al., 2020) introduces a new pipeline for object detection that does not rely on region proposals. A transformer-like network (Vaswani et al., 2017) predicts a set of bounding boxes and their classes directly. Our monocular 3D detector in Chapter 6 extends DETR by adding more prediction heads for additional attributes (*e.g.* depth, object size, and rotation).

Deep learning based approaches have also achieved state-of-the-art performance on dense pixel-wise prediction task, such as semantic segmentation (Long et al., 2015), image generation (Goodfellow et al., 2014), and style transfer (J. Zhu et al., 2017).

In addition to the applications mentioned above that take as input an image(s), extracting semantic information from geometric input(*e.g.* point cloud, voxel, and mesh) has attracted researchers' attention recently.

PointNet (Qi et al., 2017), OctNet (Riegler et al., 2017) have been proposed to classify objects given a 3D shape of an object. While works above focus on a single object, there are many works for semantic segmentation (Dai et al., 2018), object detection (Qi et al., 2019), and instance segmentation (Hou et al., 2019) given a point cloud or voxel grid of a scene. More recently, an object-centric mapping solution using point cloud as input data — RfD-Net (Nie et al., 2020) have been proposed. It first segments object instances given an input point cloud and then completes each segmented object's full 3D shape. A key difference from our works presented in Chapter 5 and 6 is the input modality. RfD-Net takes as input a complete geometric reconstruction, while in contrast, we use sequences of images. A major disadvantage of taking as input a 3D geometric reconstruction for object-centric mapping is the inability to model object dynamics. In Chapter 6, we show how our video-based method can effectively model object dynamics using a Bayesian filter.

It is worth mentioning ScanNet (Dai et al., 2017a), a large-scale RGBD video dataset with semantically annotated reconstructions. It contains about 1500 scans in different indoor environments, each of which has instance-level annotations on the reconstructed mesh. Thanks to ScanNet, the deep learning approaches working with point cloud and voxels can be trained and evaluated on a massive amount of labelled data. We use ScanNet to benchmark our object-centric mapping approaches presented in Chapter 5 and 6.

### 2.1.2 Deep learning for geometry

Deep learning has also gained attraction in geometry and 3D vision tasks following its success in recognition tasks. It has shown promising results in applications ranging from low-level feature extraction and matching to high-level reconstruction and tracking.

While classic feature descriptors, such as SIFT (Lowe, 2004), SURF (Bay et al., 2008), and ORB (Rublee et al., 2011), are handcrafted, SuperPoint (DeTone et al., 2018) demonstrates how a deep network could be trained to discover salient feature points, whose features are used for matching between two frames. In subsequent work, SuperGlue (Sarlin et al., 2020) takes a step further toward an end-to-end data-driven image matching framework by learning the matching between two images using a graph neural network.

Researchers have applied CNNs to dense pixel-wise geometric prediction, such as monocular depth estimation (Eigen et al., 2014), normal estimation (X. Wang et al., 2015), optical flow (Fischer et al., 2015), and stereo reconstruction (Kendall et al., 2017). While most early works rely on labelled data for training, which is expensive to collect, an increasingly popular trend is to use well-studied geometry constraints as a training signal. Garg et al. (2016) proposes to train a deep network for monocular depth estimation supervised by a photometric loss between stereo pairs. Rather than learning depth from stereo image pairs, Zhou et al. (2017) shows how both camera trajectory and depth could be optimised jointly using temporal pairs using photometric loss. Photometric loss is also used extensively in Chapter 5 for object shape and pose optimisation at inference time.

A deep network can also be used for recognition and geometry jointly. Eigen et al. (2015) shows that predicting geometry and semantic information jointly is complementary. It is worth noting that the detector used in Chapter 6 that predicts not only 2D bounding boxes but also objects' 3D location from a single image is another example of using deep learning for recognition and geometry jointly.

## 2.2 Object Reconstruction

Object reconstruction aims at recovering object 3D shape from an image(s). In this section, we review the literature on object reconstruction from a single view and multiple views using either traditional geometry-based or deep learning based methods. We also discuss different representations for object shape.

### 2.2.1 Single-view reconstruction

3D object reconstruction from a single image is an extremely ill-posed problem. That is to say, there are infinite combinations of object shape and pose that can result in the same image observation. Therefore, an important research direction is to use prior knowledge to reduce the search space of object shapes. If objects to be reconstructed can be categorised into classes, prior knowledge of a category – which often captured via a low-dimensional shape embedding – can be leveraged to address the task.

Various dimension reduction techniques have been used to this end. Principal Component Analysis (PCA) have been utilised to capture the variance in object shapes represented by Signed Distance Function (SDF) (Tsai et al., 2003). Dambreville et al. (2008a) introduced kernel PCA to learn a non-linear shape embedding and showed that it outperformed linear PCA. Prisacariu et al. (2012) learned a probabilistic and non-linear shape embedding via Gaussian Process Latent Variable Models (GP-LVM). However, the techniques above are limited to object categories with low shape variance, such as cars, aeroplanes, and human bodies, and the reconstructions can only capture low-frequency details. Instead, we use a deep autoencoder to learn shape embedding, which we demonstrate in this thesis can work with object categories with high shape variance or even multiple object classes.

More recently, similar to other computer vision tasks, end-to-end deep learning based approaches have been proposed to learn a mapping from a single RGB image to complex and generic object shapes. Early works involving deep networks aimed to map an RGB image to a sparse point cloud (Fan et al., 2017a) or a low-resolution voxel grid (Choy et al., 2016a). These representations have their advantages and disadvantages, which is discussed in section 2.2.3.

Girdhar et al. (2016) is a pioneering work with the insight of learning a shape embedding explicitly to improve single-view object reconstruction. They propose to train two networks for single-view object reconstruction. The first network is an AutoEncoder trained to map an object shape represented by a voxel grid to a shape embedding, and then reconstruct the input shape using the decoder. The second network is a Convolution Neural Network that mapps an input image to the shape embedding. When reconstructing an object shape from a single image, the trained CNN maps the image to a shape code in the embedding, decoded by the decoder to an object shape. A low-dimensional embedding as an intermediate representation is a distinctive feature of this work over previous works that train a network to map an image to an object shape directly. This idea of training a deep shape embedding has inspired many works, including ours. We develop our probabilistic shape embedding in Chapter 4 and the joint embedding that works with both sparse point cloud representation and DeepSDF, a dense implicit representation, in Chapter 5. An active research direction in single-view object reconstruction currently is investigating what representation to be used in a deep learning framework, which is covered in Section 2.2.3.

An important research direction of single-view object reconstruction is weakly supervised learning that uses multi-view geometry instead of 3D ground-truth shape as training signal for deep networks. The motivation is twofold: 1) It is time-consuming and labour-intensive to acquire ground-truth object shapes from real images. 2) As a result, deep learning based approaches often resort to using CAD models of objects, such as ShapeNet (Chang et al., 2015). A large number of synthetic images are rendered from different viewpoints of a CAD model. Rendered images and the CAD models can then be used as input and target for network training, respectively. However, the domain gap between synthetic images in the training and real images in testing deteriorates network performance at inference time.

The main principle in weakly supervised learning is to minimise the inconsistency between a predicted 3D object shape and the corresponding image observations. The observations can be object silhouette (Tulsiani et al., 2017a; Yan et al., 2016) or texture (Kanazawa et al., 2018). However, inference using the trained network still solely relies on the mapping function controlled by network weights and is not bounded by any geometric constraint. In contrast, our approaches in Chapter 4 and 5 enforce the geometric consistency explicitly at inference time.

Instead of using a single RGB image, some works propose to complete a full 3D object shape from a single depth observation. Dai et al. (2017b) first completes a low-resolution full 3D shape from a single depth observation, followed by an up-sampling procedure by leveraging a shape database with high-resolution models. B. Yang et al. (2018) proposes to use a conditional Generative Adversarial Network (cGAN) to complete a high-resolution voxel reconstruction directly from a single depth view to bypass the need of a CAD model database.

### 2.2.2 Sparse multi-view reconstruction

While classic SfM and SLAM approaches require a dense number of viewpoints for feature matching and correspondence estimation, another set of approaches focuses on reconstructing object shapes from a sparse set of viewpoints arbitrarily distributed. In the scope of sparse multi-view reconstruction, the baseline between observations is often too large to apply feature matching for reconstruction as done in traditional SfM/SLAM. We dedicate this section to methods in a sparse multi-view setup, and SfM/SLAM based approaches for object reconstruction are discussed in 2.4.2.

Shape from silhouette (Pujari, 1993) is a classic algorithm in sparse multi-view reconstruction. Each contour image of an object (*i.e.* silhouette) was unprojected to 3D using known camera intrinsic parameters to form a visual cone. A visual hull, the 3D reconstruction, was obtained by intersecting multiple visual cones from different viewpoints. Shape from silhouette is simple in the sense that it does not require any knowledge on lighting condition or texture. However, a reconstruction degenerates when the number of viewpoints is too few to capture the entire shape, or the 3D shape is concave.

To handle concavity, Seitz et al. (1999) and Kutulakos et al. (2000) extended visual hull by reasoning texture and colour assuming Lambertian reflection. Shape from shading (Zhang et al., 1999) is able to recover concave surfaces but requires prior knowledge of the lighting direction. To some extent, the work we describe in Chapter 4 can be considered as a shape from silhouette approach with a deep learnt shape prior. Unlike the traditional shape from silhouette that requires viewpoints with a large baseline to recover a full 3D shape, we show that if a category-wise shape prior is available, the full 3D shape can be recovered given only a single view.

Several deep learning approaches are introduced in recent years to recover a 3D object shape given a sparse set of viewpoints. 3D-R2N2 (Choy et al., 2016a) proposes to fuse information from multiple images via a Recurrent Neural Network (RNN) to predict an object shape. Instead of using an RNN, Sridhar et al. (2019) and Wiles et al. (2017) use permutation invariant operations, such as max pooling or average pooling, to fuse multi-view information. In Learnt Stereo Machine (Kar et al., 2017), 2D image features are unprojected and transformed into a 3D feature volume using the known camera intrinsic and extrinsic parameters. A 3D CNN takes as input the 3D feature volume and generates the final 3D reconstruction. LSM shows that the 3D feature volume is crucial for multi-view fusion.

### 2.2.3 Shape representation

Memory consumption of different representations must be taken into account carefully when training deep networks for shape generation. Choy et al. (2016a) and Girdhar et al. (2016) use a stack of 3D deconvolution layers to generate 3D voxel grids to represent object shapes, which is analogous to the 2D deconvolution operation producing dense pixel-wise predictions. However, a reconstructed voxel grid is limited to low resolution due to the memory growth in cubic order. Efficient data structures, such as Octree (Tatarchenko et al., 2017) have been proposed to alleviate this problem. Recognising object shapes can be parameterised by a set of bases, Johnston et al. (2017) propose to map an image to the bases of the Discrete Cosine Transform. An inverse DCT operation replaces the expensive deconvolution layers to reconstruct voxel grids at the desired resolution.

A limitation of voxel-based representation despite extensions mentioned earlier for higher resolution is that it needs voxels to indicate occupancy for the entire discretised space. In contrast, mesh and point cloud are two popular choices besides voxel for efficiency because only the surface of an object shape needs to be modelled. Early works with mesh representation (N. Wang et al., 2018; Henderson et al., 2018) learn to deform a simple template (*e.g.* ellipsoid and cube) or a retrieved shape template (Pontes et al., 2018) to model different object shapes. These methods, however, fail to reconstruct shapes that have different topology from the templates. Although MeshRCNN (Gkioxari et al., 2019) still uses mesh deformation, it was initialised by a low-resolution voxel reconstruction, which is likely to have a similar topology with the target shape, and thus improved the final mesh reconstruction.

Fan et al. (2017a) introduces a method for single-view object reconstruction using point clouds. However, it is confined to a sparse set of points because the number of learnable parameters grows proportional to the point cloud. Moreover, the loss functions working with point clouds (Chamfer Distance (CD) or Earth Mover Distance (EMD)) become intractable for dense point clouds. Instead of directly predicting a dense point cloud, Lin et al. (2018a) propose to predict multiple depth maps from a set of fixed viewpoints. A dense point cloud can be generated by unprojecting the depth maps and transforming into the pre-defined canonical pose. Nevertheless, this representation is not as efficient as possible because not every pixel in a depth map is used to model object shape. They predict a binary mask to filter out background pixels in a depth map that do not intersect with the object surface after unprojection. As a result, a substantial amount of computation for the background pixels are wasted, and it also leads to the sparsity of the fused point cloud. Our work in Chapter 3 presents a novel framework to generate dense point cloud more efficiently. The novelty lies in the use of our deformable vector field, which alters the direction of unprojection rays of each pixel such that every ray can intersect with the object surface.

More recently, Implicit representations introduced in DeepSDF (Park et al., 2019a) and Occupancy Net (Mescheder et al., 2019a) have attracted significant attention in the community. The core of these representations is a learned mapping function from a 3D coordinate to a classification result of whether this 3D coordinate is inside or outside the object shape boundary. However, a large number of inquiries (by passing a 3D coordinate to the network) are needed to recover the entire object shape. Therefore, although these representations have shown to be more expressive for shape modelling and can potentially represent shapes to arbitrarily fine resolution, they come with the cost of many network forward operations. Methods like (S. Liu et al., 2019) that use geometry constraints to optimise an object shape represented by a DeepSDF embedding requires forward and backward through the network for hundreds of iterations. The computation cost is prohibitively expensive for any online systems. To alleviate this issue, we introduce a novel joint embedding, where a shape code can be decoded to either a sparse point cloud or a dense DeepSDF representation in Chapter 5. Therefore, our method can use the sparse point cloud representation for efficient optimisation before switching to deepSDF for further dense refinement.

## 2.3 Geometric-only Mapping

Geometric mapping from image observations is most commonly formulated as Structure from Motion (SfM) or Simultaneous Localisation And Mapping (SLAM) problem. While both SfM and SLAM involve solving camera poses and scene geometry jointly, SfM is often solved in a batch mode via a technique called Bundle Adjustment (BA) (Triggs et al., 1999), but SLAM is solved *online* and incrementally, which is more attractive to many robotic and AR/VR applications. In this section, we focus mainly on the mapping part of SLAM, particularly for the interest of this thesis.

Early SLAM approaches, such as MonoSLAM (Davison et al., 2007) and PTAM (Klein et al., 2007), with the main focus on localising camera in a previously unknown environment, use a sparse set of salient feature points for feature matching. At this stage, the sparse feature points that describe the geometry of the environment come as a side product for camera tracking. Although sparse feature points do not provide a dense reconstruction, which is crucial for obstacle avoidance, they are lightweight and still used in modern SLAM systems (Mur-Artal et al., 2017). While PTAM (Klein et al., 2007) uses sparse feature points for mapping, the idea of separating tracking and mapping has facilitated significant developments on mapping, such as denser reconstruction and incorporating semantic information.

Unlike feature-based approaches described above that only use a subset of pixels for mapping, direct methods utilise all image pixels' gradient. DTAM (Newcombe et al., 2011b) is a novel work on this direction. It estimates the depth map of each RGB image by reprojecting the depth values into a cost volume where the photometric cost is minimised. A smoothing term is applied to help reconstruction in texture-less regions. Camera poses are tracked by optimising the per-pixel photometric cost in a Lucas-Kanade style warp.

Researchers have tried to incorporate a higher level of entities than 3D points in the map. Lovegrove (2012) proposes a plane mosaicing algorithm to fuse multiple visual observation into a single plane. More recently, the work from Hosseinzadeh et al. (2017), building upon Orb-SLAM (Mur-Artal et al., 2017), takes 3D planes as an additional geometric entity for mapping. The addition of planes generates a denser mapping and enables point-plane constraint and plane-plane constraint to improve accuracy. Similarly, S. Yang et al. (2016) demonstrate that 3D planes can improve dense mapping, especially in low-texture environments.

The rise of commodity RGBD sensors like Kinect from Microsoft and PrimeSense has promoted the development of RGBD mapping systems. The RGB-D mapping proposed by Henry et al. (2014) uses a combination of depth observations and feature point matching in a joint optimisation for both camera pose and mapping, represented by surfels. KinectFusion (Newcombe et al., 2011a), while using a similar technique for tracking, adopts the Truncated Signed Distance Field (TSDF) as a volume representation for reconstruction where depth measurements could be fused into a volume by weighted averaging. KinectFusion demonstrates how noisy depth reading can be fused for accurate mapping and subsequently has encouraged a series of “fusion” approaches.

While the TSDF used in KinectFusion limits its scalability, ElasticFusion (Whelan et al., 2015) adopts a surfel-based representation, which extends the mapping to room-scale environments. It achieves globally consistent mapping by using loop closure and non-rigid surface deformations. While all aforementioned systems assume a static environment, DynamicFusion (Newcombe et al., 2015) is proposed to handle non-rigid objects presented in a scene.

More recently, Bloesch et al. (2018a) present a key frame based SLAM, where the depth map of each keyframe is represented by a compact low-dimensional latent code. The depth map can be optimised using photometric consistency with nearby views. Our works in Chapter 4, 5, and 6 share common ground with this work in using a deep network to learn a compact embedding for geometry.

## 2.4 Semantic Mapping

While dense reconstruction has enabled obstacle avoidance and path planning, it is increasingly clear that semantic information is critical for more complex tasks. Furthermore, human beings perceive the surrounding semantically, and it would benefit interactive applications if the robots can also parse the environment as we do. Semantic mapping, referring to associating semantic concepts to geometric entities in a robot’s surrounding (Cadena et al., 2016), has been proposed to fill in the gap. Semantic mapping can be divided into two groups: one that parses the semantic information at a category level, and the other that takes the concept of object instance into account, which we refer to object-centric mapping in this thesis.

### 2.4.1 Category-level semantic mapping

Early works in semantic mapping adopt a two-stage framework, where the first stage is building a geometric mapping, followed by the second stage attaching semantic class labels to the geometric mapping. Kahler et al. (2013) use KinectFusion (Newcombe et al., 2011a) to build a geometric reconstruction, which is then labeled semantically by Decision Tree Field (Nowozin et al., 2011) and Regression Tree Field (Jancsary et al., 2012). Similarly, Pham et al. (2015) also rely on KinectFusion for geometric reconstruction, but they use a hierarchical Conditional Random Field to label category-level semantic information.

Unlike the off-line approaches where geometric reconstruction and semantic labelling is decoupled, online approaches reconstruct and label semantically simultaneously. Given a new incoming frame, Hermans et al. (2014) infer 2D semantic segmentation map using Randomised Decision Forests. The 2D segmentation map is unprojected to 3D using the depth observations and fused to the 3D semantic model via a Bayesian update. A dense pairwise Conditional Random Field regularises the update on the 3D point cloud. Instead of automatically using a trained model for semantic information inference, SemanticPaint (Valentin et al., 2015) labels the environment semantically guided by a user input while an RGBD sensor scans the environment.

More recently, an extensive and ongoing body of work uses deep networks for semantic segmentation when building a semantic mapping pipeline. A famous work on this direction is SemanticFusion (McCormac et al., 2017) that is built upon ElasticFusion, a geometric only SLAM framework. It uses a semantic segmentation network to infer a 2D semantic segmentation map from each incoming RGB image. The predicted semantic segmentation map is unprojected to 3D using the depth map from the depth sensor. This new prediction is fused into the reconstruction probabilistically.

Similarly, CNN-SLAM (Tateno et al., 2017), using a deep network to infer 2D semantic segmentation map, explores the possibility of using a monocular depth estimation network to replace the “depth” channel of an RGBD camera. The semantic representation of both CNN-SLAM and SemanticFusion is incrementally fused into the 3D map from 2D semantic segmentation predictions. A recent work, SceneCode (Zhi et al., 2019), built upon a latent code based CodeSLAM, proposes to jointly embed depth map and semantic segmentation map in a learned latent space.

### 2.4.2 Object-centric mapping

While associating category-level semantic information to a geometric reconstruction is a critical step toward more "meaningful" reconstruction, not all advanced robotic tasks can be facilitated by category-level semantic information. For instance, fetching an object requires not only pixel-level semantic labelling but also the concept of an object instance. Therefore, another direction researchers have begun to explore focus on creating a reconstruction with instance-level semantic information, namely, object-centric mapping.

SLAM++ (Salas-Moreno et al., 2013a) is an important milestone in the development of object-centric mapping. It relies on a pre-constructed object database. When a hand-held depth camera traverses in a scene, it employs Point-Pair Features to match depth observations with object reconstructions in the database. If matched, the reconstruction is placed into the scene using the object pose estimated by Hough voting using the Point-Pair Features. However, the apparent drawback of SLAM++ is the need for the pre-defined object database that limits its practicality to an arbitrarily new scene with unregistered object instances.

Some following works leverage shape prior knowledge to bypass the need for a pre-defined object database. Dame et al. (2013) combines a dense SLAM with object pose and shape optimization. The authors proposed to capture shape prior knowledge in low-dimensional embedding using a Gaussian Process Latent Variable Machine (GPLVM). Photometric consistency and depth information used in the dense SLAM system can be utilised to optimise object shape represented in this embedding and object pose. It is one of the earliest works to show that shape prior information can enhance the completeness and accuracy of geometric reconstruction.

In a similar formulation, Bao et al. (2013a) solve an off-line Structure from Motion problem with shape prior information.

The main theme in this thesis is leveraging deep learning to learn a better shape prior. GP-LVM used in Dame et al. (2013) does not scale well to be trained on large datasets or learn priors on high-dimension shape representations. This forces Dame et al. (2013) to use frequency-based compression techniques (*i.e.* Discrete Cosine Transformation) leading to loss of thin and other high-frequency object structures. Compared to Dame et al. (2013), our object-centric mapping system in Chapter 5 is more practical because i) we can handle multiple objects by solving data association effectively; 2) the deep shape prior we used shows that we can model challenging object shapes with high shape variance in different categories.

Instead of densely modelling the shape of an object, there are works interested in representing objects in a coarse manner. Rubino et al. (2017) show that an object could be localised and represented by a 3D ellipsoid by reasoning its multiple 2D detections. Nicholson et al. (2018) replace the algebraic error function with a geometric error function and proposed to solve the problem in an online fashion. However, both systems require ground-truth data associations between 2D object detections. Hosseinzadeh et al. (2019) not only present a data association algorithm but also attempt to incorporate more accurate object shape represented by a point cloud using a monocular object shape prediction network. Orthogonal to the quadric representation, CubeSLAM (S. Yang et al., 2019a) represents each object using a 3D bounding box. In Chapter 5, we also leverage quadric representation to localise objects in a scene before further refining their pose and shape using point cloud and DeepSDF.

By replacing semantic segmentation networks used in (McCormac et al., 2017; Tateno et al., 2017) with an instance segmentation network Mask-RCNN (He et al., 2017), there are various approaches for dense object-centric mapping using RGBD camera. Fusion++ (McCormac et al., 2018a) uses Mask-RCNN to identify object instances, which were fused individually into an object-level volumetric map. Similarly, Mid-Fusion (Xu et al., 2019) extends Fusion++ by modelling dynamic objects and using efficient Octree representation for object shapes. MaskFusion (Runz et al., 2018) instead uses surfels to model object shape.

Almost all semantic mapping or object-centric mapping solutions using deep learning involve semantic/instance segmentation mask prediction to identify semantic entities. In contrast to these approaches, Sünderhauf et al. (2017) employ an object detection network. They then leverage depth information and 2D bounding boxes to generate instance segmentation masks without using an instance segmentation network.

More recently, shape prior based reconstruction has re-emerged by leveraging modern deep learning. A concurrent work of our work in Chapter 5, NodeSLAM (Sucar et al., 2020), learns a low-dimention shape embedding via a Variational AutoEncoder (VAE). Object shapes represented by latent code and pose are jointly optimised using the depth information.

In summary, there are mainly two types of approach to object-centric mapping. A line of works use 2D instance segmentation masks (obtained from a deep network) to associate instance-level semantic entities to a geometric reconstruction. This line of research has recently attracted a considerable amount of interest mainly because deep learning can provide impressive results on 2D instance segmentation. However, there is no direct connection between object geometry and semantic reasoning. Object geometry is obtained by running traditional geometry reconstruction algorithms, such as KinectFusion. Semantic identities are an additional layer on top of the geometry reconstruction. A significant drawback of these approaches is that it cannot model invisible parts. In contrast, human beings can effortlessly complete the full 3D shape of an object given partial observations using prior knowledge of object shapes.

In contrast, the second type adopts a tighter connection between geometry reconstruction and semantic reasoning. The object geometry is based on object shapes (using a pre-defined object database (Salas-Moreno et al., 2013b) or a low-dimension shape embedding (Dame et al., 2013)). This line of research is less explored in the era of deep learning because it is not clear how deep learning can be leveraged initially. Our works in Chapter 5 and 6 are pioneering works on investigating how deep learning can be used for both recognition and reconstruction jointly.

# 3
# Efficient Dense Point Cloud Reconstruction

## Contents

*In this chapter we introduce our work on reconstructing a 3D representation of an object given a single view of that object. To tackle this ill-posed problem, prior knowledge of object shape has been used extensively in the community. Recently, after witnessing the success of Convolution Neural Networks (CNN) on various image recognition tasks, a large body of works apply CNN to learn a mapping function from a single RGB image to an object shape. Shape prior information is embedded in the network weights after training with thousands of samples. A challenge of* *applying Convolution Neural Networks (CNN) to object shape reconstruction is that Euclidean convolution kernel operation cannot be used efficiently on traditional object representations, such as triangular meshes, voxel occupancy grid, and point cloud. To this end we develop a novel multi-view representation where each view representing the visible surface from this particular viewpoint comprises a depth map and the corresponding deformation field that ensures every pixel-depth pair in the depth map lies on the object surface. The deformed depth maps are then unprojected and fused to a dense point cloud that models the 3D shape. We show that a deep network using the proposed representation outperforms methods working with voxel and point cloud representation. Furthermore we test the generalization of the proposed representation on real images. The work described in this chapter has been published at European Conference on Computer Vision 2018 (K. Li et al., 2018).*

## 3.1 Introduction

Although humans can effortlessly infer an object's 3D structure from a single image, it is, however, an ill-posed problem in computer vision. To make it well-posed, researchers have been using hand-crafted 3D cues such as "Shape from X" (e.g., shading, texture) (Braunstein et al., 1993; Aloimonos, 1988; Prados et al., 2006) , and planarity (Saxena et al., 2008; M. Liu et al., 2014). More recently, there has been considerable interest in using deep networks to regress from an image to its depth (F. Liu et al., 2015; Garg et al., 2016; Godard et al., 2017; Zhan et al., 2018) for scene geometry reconstruction, and in particular from an image of an object to its 3D shape for object geometry reconstruction. There is no settled or the best way to represent 3D objects, with methods including meshes (Sinha et al., 2017; Kong et al., 2017), point clouds (Fan et al., 2016; Lin et al., 2018a; Tatarchenko et al., 2016), or voxel occupancy grids (Choy et al., 2016a; Girdhar et al., 2016; Tatarchenko et al., 2017) , each having both advantages and disadvantages in terms of the efficiency and convenience of the representation, and – importantly for our purposes – for learning.

For volumetric representation methods, most existing CNN-based methods simply extend 2D deconvolutions to 3D deconvolutions on 3D regular grids, as

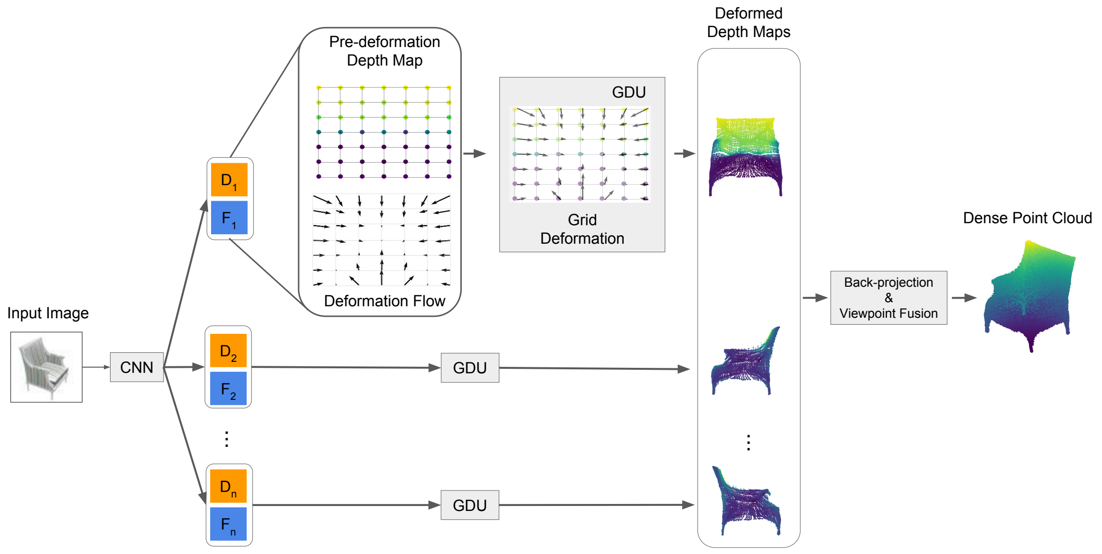


**Figure 3.1:** The overall pipeline of our approach. Given a single RGB image of an object from an arbitrary viewpoint, the CNN outputs a set of 2D pre-deformation depth maps and corresponding deformation fields at pre-defined canonical viewpoints. These are passed to a Grid Deformation Unit (GDU) that transforms the regular grid of the depth map to a deformed depth map. Finally, we transform the deformed depth maps into a common coordinate frame to fuse all 3D points into a single dense point cloud. The colours of points indicate different depth values, and the arrow direction represents the image grid deformation.

done in Choy et al. (2016a) and Girdhar et al. (2016). For each voxel, the network predicts the score of being occupied by the object. Thresholding the volumetric score map results in a 3D occupancy object representation. Nevertheless, 3D volumetric representations are very expensive in both computation and memory when working with deep networks — the required memory increases cubically with the grid resolution. Notably, only a small portion of the voxels are occupied by the object, leaving the rest wasteful. Consequentially, volumetric representation is limited to a low-resolution reconstruction (*e.g.* $64 \times 64 \times 64$), thus losing the surface granularity. Furthermore, it is generally non-trivial to find a suitable threshold to generate precise object surfaces for different object classes or even different objects in the same class.

Intuitively, it is more efficient to directly predict an object's surface, rather than the entire 3D space. Fan et al. (2016) propose Point Set Generation Net (PSGN) representing object surface by an orderless point cloud. However, because PSGN adopts a multi-layer perceptron to generate 3D points, the number of parameters grows linearly to the number of points, and thus it is not scalable to a dense point cloud. Furthermore, due to the orderless nature of this point cloud representation, the training loss needs to be permutation invariant, and thus it is costly — for instance, for $N$ prediction points along with $N$ ground-truth points, the complexity of the Chamfer Distance used in (Fan et al., 2016) is $O(N^2)$.

To tackle this issue, Lin et al. (2018a) and Tatarchenko et al. (2016) use a set of depth maps from different viewpoints relative to the object, which are then easily fused into a point cloud. The supervision (or losses) operates in 2D depth space instead of the 3D point cloud. However, the predicted depth maps from a deep neural net inherently cover not only object points but also unnecessary background. To classify foreground and background points, Lin et al. (2018a) and Tatarchenko et al. (2016) also predict a binary (i.e., foreground/background) mask for each predicted depth map, where each pixel is the score of being foreground. Background depth pixels that do not intersect with object surface after unprojection are discarded. This depth-mask representation is undesirable for several reasons. Firstly, it is inefficient because the computation for background depth pixels is wasted. Secondly, it leads to the sparsity of the fused point cloud. Lastly, our experiments show that it is non-trivial to find a suitable foreground/background threshold as it varies in different object instances and different object classes. A key research question we would like to answer in this work is how to better utilize the depth pixels that cannot intersect with an object surface to address the aforementioned issues.

Learning to regress depth from images often generates noisy points around the surface (Tatarchenko et al., 2016). The fusion of multiple partial-view point clouds escalates the noise. Lin et al. (2018a) propose to improve the quality of fused point clouds by a multi-view consistency supervision based on binary masks and depth maps. The idea is to project the predicted 3D point cloud to novel viewpoints to generate new depth maps and masks at these viewpoints, supervised by the corresponding ground-truth depth maps and binary masks. However, this supervision, being similar to the shape from silhouette technique (Martin et al., 1983) and voxel-based multi-view consistency supervision in deep learning based methods (Yan et al., 2016; Tulsiani et al., 2017a), encourages masking out the points projected to the background and further reduces the density of the predicted point clouds and harms the surface coverage.

In this work, we present a novel and highly efficient framework (shown in Fig. 3.1) to generate dense point clouds representing the 3D shape of objects. Given a single image of an object of interest taken from an arbitrary viewpoint, our network generates multiple partial surfaces of the object, each at a pre-defined canonical viewpoint. The critical difference to existing multi-view representation Tatarchenko et al. (2016) and Lin et al. (2018a) is that each surface is defined by a depth map and the corresponding deformation field (instead of a binary mask). In the Grid Deformation Unit (GDU), a point on the surface is obtained by first shifting a pixel on the depth map image grid by the amount given by the deformation field and then back-projecting (to the corresponding depth). The resulting set of points can then be considered a (dense) point cloud, though it is not an orderless one. The final 3D object representation is obtained by fusing the point-cloud surfaces into a single point cloud.

At training time, we use a combination of per-view and multi-view losses. Because each pixel in depth maps and deformation fields has its corresponding ground-truth value, the per-view loss can be evaluated in $O(n)$ time (where $n$ is the number of points). This in contrast to, for instance, Chamfer Distance usually required for unordered point-sets, leading to $O(n^2)$ complexity. The novel multi-view loss encourages the 3D points back-projected from a particular view to be consistent across novel views. More specifically when a predicted 3D point is re-projected into a novel viewpoint but falls outside of the object silhouette, our network incurs a loss based on the distance of the point to the boundary, rather than penalizing a binary cross entropy loss as done in (Lin et al., 2018a; Yan et al., 2016; Tulsiani et al., 2017a). Our extensive experiments demonstrate that using these combined per-view and multi-view losses yields more accurate and dense point cloud representations than any previous method.

Our contributions are summarized as follows:

- We propose a novel deformed depth map representation for 3D object reconstruction based on multiple canonical views that is efficient and bypasses foreground/background thresholding;
- We show how this representation can be effectively regressed using a deep network from a single view;
- We introduce a novel loss for our network that combines a per-view loss – that can be efficiently calculated thanks to our unique representation – with a novel multi-view loss based a distance field.
- We evaluate our method extensively showing more accurate and denser point clouds than previous methods. We include ablation experiments that demonstrate the value of the contributions above separately and together.

## 3.2 Related Work

Shape from X is a series of classic methods in single view reconstruction. “X” can be texture (Aloimonos, 1988), shading (Zhang et al., 1999), or silhouette (Martin et al., 1983). An object reconstruction is found by minimising an energy function that described the image formation process. To make the single-view object reconstruction less ill-posed, prior knowledge is often used. One of the simplest prior knowledge is smoothness constraints, which generally encourage adjacent pixels with similar intensity to be unprojected to nearby positions in 3D.

If objects can be categorised into different classes, shape regularity of a category can be captured as another form of prior knowledge, which improves resilience to incorrect feature matching and concavity (e.g., chairs should be concave between two arms). Kar et al. (2015) leverage the strong regularity of objects. For a specific object category, they learn deformable templates from a large collection of images with the object of interest presented and the corresponding segmentation masks. Dame et al. (2013) show how a low-dimensional shape embedding learnt by a GP-LVM can be used to improve reconstruction in a SLAM framework.

Together with the shape embedding as a shape prior, the object shape and pose are optimised by minimising the depth and photometric error given a sequence of images. Rather than learning a shape embedding, methods like Huang et al. (2015) and Kurenkov et al. (2017) use image features to retrieve similar 3D shapes, from which they deform to the target shapes.

Even though our method also uses deformation, the difference between our deformation and that of their approaches is twofold: Firstly, we perform 2D deformation on an image grid, such that the deformed image grid matches the object silhouette, while they deform the 3D shape directly. Secondly, they perform deformation on 3D basis models with small variants while ours deforms a regular grid into any 2D shape.

Since large repositories of CAD models become available (e.g., ShapeNet (Chang et al., 2015)), it is easy to render a large number of 2D images from CAD models. A large number of 2D-3D ground-truth pairs make it seamless to use a powerful yet data-hungry framework — deep networks. Several deep learning based methods to generate 3D object shapes from single images have been proposed recently.

The pioneering works are from Choy et al. (2016a) and Girdhar et al. (2016) that use 3D CNNs to produce voxel reconstruction that is limited to low-resolution voxel grids. Octree data structure (Tatarchenko et al., 2017; Häne et al., 2017) and Discrete Cosine Transform technique (Johnston et al., 2017) have been used to scale up the voxel grid. More recently, Fan et al. (2016) propose an alternative approach that predicts an orderless point cloud to model the surface of objects directly. Nevertheless, this method is limited to a sparse point cloud because 1) the number of learnable parameters increases linearly as the number of predicted points and 2) the direct 3D distance metrics (e.g., Chamfer Distance) are also intractable for dense point clouds. Thus, this method is not scalable in terms of memory and training time.

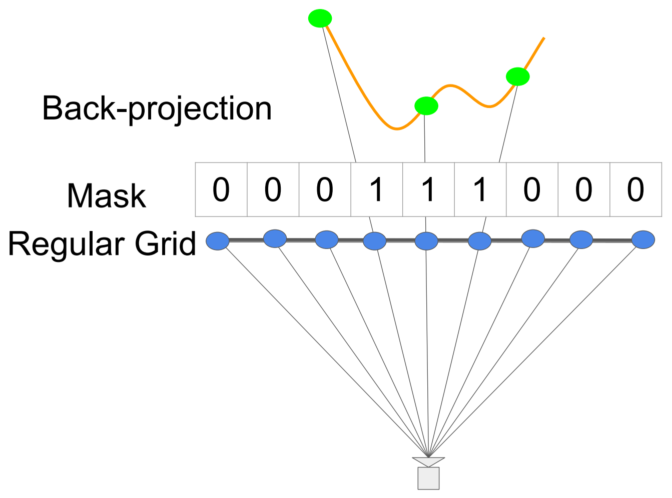
The most relevant works to us are Lin et al. (2018a) and Tatarchenko et al. (2016). We all advocate that, in order to generate dense point clouds, one should resort to partial surfaces each represented by a structured point cloud. However, the fundamental difference between our approach and Lin et al. (2018a) and Tatarchenko et al. (2016) is how to shape these surfaces. Their methods shape an object surface by predicting a binary mask along with the depth map to filter out background points that do not intersect to the object surface. The negative aspect of this multi-view representation is firstly, it is an enormous computation waste as pixels for the background are discarded, especially for objects with thin structures such as lamp, aeroplane and chair; secondly, foreground/background thresholding inherits the thresholding issue from 3D voxel grid representation. Instead, we predict the surface directly by deforming the regular depth map to circumvent issues above, as demonstrated in Fig. 3.2.

Moreover, although Lin et al. (2018a) have realised that the fusion of multiple partial surfaces generates noisy points and thus developed a multi-view consistency supervision based on binary masks and depths to address this issue. However, the binary cross entropy penalty leads more points to be discarded and thus, the surface coverage is sacrificed. In contrast, we develop a novel multi-view supervision framework based on a continuous distance field that does not suffer from the surface coverage trade-off.

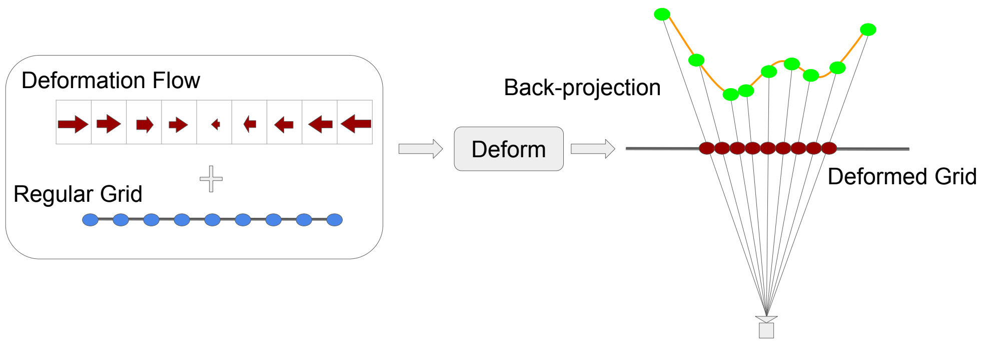
Subsequent to this work being published, some works propose novel implicit representations for object shapes (Park et al., 2019a; Mescheder et al., 2019b). Object shape surface is recovered by querying 3D positions whether they are inside/outside object surface. While implicit representations can convert to explicit representations (*e.g.* mesh, voxel and point cloud) in arbitrarily fine resolution, they are not as efficient because many queries are needed for high resolution. In Chapter 5, we show how explicit and implicit representations are complementary by representing object shapes in a joint embedding.

## 3.3 Method

Our goal is to train a CNN that is able to reconstruct a dense 3D point cloud to represent 3D object shape from a single RGB image. We first introduce how we



(a)



(b)

**Figure 3.2:** A 1D example of Depth-Mask back-projection and Deform-Depth back-projection. Blue points are pixel-depth pairs on a 1D regular grid. Dark red arrows are deformation flow. Dark red points are pixel-depth pairs shifted by the deformation flow. Orange lines are the target 2D surface. Green points are 2D points back-projected to reconstruct the 2D surface. (a) In previous work (Lin et al., 2018a), because pixels are filtered out by a binary mask, there are fewer points to reconstruct the surface. (b) In our deformed depth representation, the deformed grid is aligned with the surface, so that all pixels are used to reconstruct the surface.

represent a partial surface of an object using a deformed depth map and the per-view supervision for the deformed depth map, followed by a multi-view consistency supervision based on distance fields. Lastly, we detail the implementation details, including network architecture and training procedure.

### 3.3.1 Deformed Depth Map

One way to represent a view-dependent object surface is to use a depth map $\mathbf{D}$. For each pixel $p$ at location $(x_p, y_p)$ with a depth value $z_p$, we can unproject $p$ to a 3D point $\mathbf{p}$ through an inverse perspective transformation, *i.e.*,

$$\mathbf{p} = \mathbf{R}^{-1}(\mathbf{K}^{-1}\begin{bmatrix}x_p \ y_p \ z_p\end{bmatrix}^T - \mathbf{t}), \tag{3.1}$$

where $\mathbf{K}$, $\mathbf{R}$ and $\mathbf{t}$ are the camera intrinsic matrix, rotation matrix, and translation vector respectively. Learning a network to reconstruct the 3D object shape becomes learning to predict a set of depth maps, as done in Lin et al. (2018a) and Tatarchenko et al. (2016). Note that the size of the depth images need not be equal to the size of the input RGB image. The main issue of this representation is that not all pixels are back-projected to the object's surface, therefore the network must additionally predict a binary segmentation mask for each depth map to suppress background points. The abandoned points become wasteful.

Notice that in Eq. (3.1), the pixel locations $(x_p, y_p)$ are fixed in a regular image grid, which is not flexible to model the object's surface. Our insight is that regardless of depth values, the projection ray of every pixel should hit the object's surface. Specifically, for each pixel $p$ at pixel location $(x_p, y_p)$, our network predicts a deformation vector $[u_p, v_p]$ (besides its depth value $z_p$). The position of this pixel is shifted by the deformation flow. Then the new position of this pixel after deformation is $x'_p = x_p + u_p$, $y'_p = y_p + v_p$. An 1D illustration is presented in Fig. 3.2. We denote the entire deformation field for an entier image grid as $[\mathbf{U}, \mathbf{V}]$. The same inverse perspective transformation can be applied on the deformed depth map to back-project to 3D space,

$$\mathbf{p} = \mathbf{R}^{-1}(\mathbf{K}^{-1}\begin{bmatrix}x'_p \ y'_p \ z_p\end{bmatrix}^T - \mathbf{t}). \tag{3.2}$$

**Training losses for deformed depth map** During training, the deformation flow is supervised by a pseudo ground-truth (see the section below). The pixel-wise $L_1$ deformation flow losses for $x$ and $y$ directions are given below,

$$L_U = \|\mathbf{U} - \mathbf{U}_{\mathbf{gt}}\|_1 \quad L_V = \|\mathbf{V} - \mathbf{V}_{\mathbf{gt}}\|_1. \tag{3.3}$$

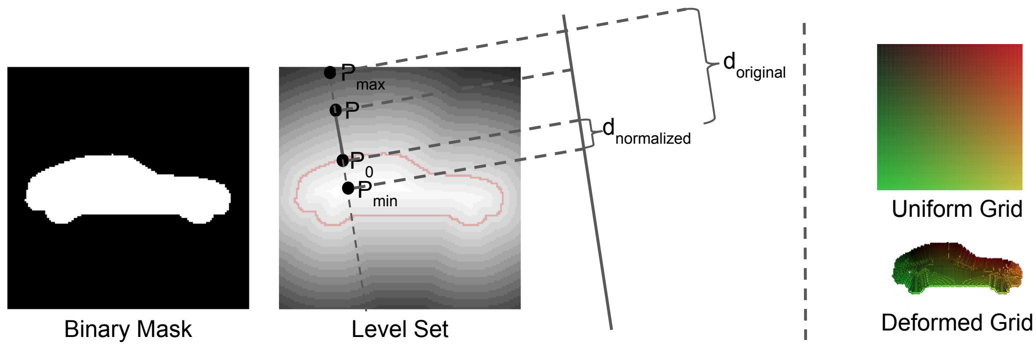


**Figure 3.3:** Left image is a binary mask; middle image is the corresponding level set where the silhouette boundary is shown in red line. To define the deformed position for point $p$, the closest point on the boundary $p_0$ is located, followed by finding the point at the maximum level $p_{max}$ and point at the minimum level $p_{min}$ along the line of $p$ and $p_0$. Then, $p$ should be normalized to the range between $p_0$ and $p_{min}$. Right images show a pair of uniform grid and its corresponding deformed grid. The color indicates point correspondence before and after deformation.

where the $\mathbf{U_{gt}}$ and $\mathbf{V_{gt}}$ are the ground-truth deformation fields on x-direction and y-direction respectively.

However, the direct pixel-wise loss cannot be used between a regular ground-truth depth map and a deformed depth map as pixel-wise correspondences have been changed due to deformation. To supervise the deformed depth map, we use the pseudo ground-truth deformation flow to deform the ground-truth depth map to obtain the deformed ground-truth depth map, which is given below,

$$L_d = \|f(\mathbf{D_{pred}}|\mathbf{U_{gt}}, \mathbf{V_{gt}}) - f(\mathbf{D_{gt}}|\mathbf{U_{gt}}, \mathbf{V_{gt}})\|, \tag{3.4}$$

where $f(\mathbf{D}|\mathbf{U_{gt}}, \mathbf{V_{gt}})$ is the deformation operation on a depth map $\mathbf{D}$ given the pseudo ground-truth deformation flow. The $\mathbf{D_{gt}}$ and $\mathbf{D_{pred}}$ are the ground-truth and predicted depth respectively.

**Pseudo ground-truth deformation flow** To train the network to predict deformation field, we need to define a ground-truth deformation vector for each pixel; however, the deformation vector is not unique. The criterion of an ideal deformation field is 1) every pixel should be shifted into the silhouette (*i.e.*, the regular grid should be deformed to fit the silhouette), 2) the *deformed* grid should be uniformly dense. To this end, we define a function that takes an object binary mask (silhouette as foreground and the rest as background) on a regular grid as input and outputs a vector field for deformation.

More specifically, we first convert the binary mask into a level set, where the inside silhouette is negative levels, the background is positive levels, and the silhouette boundary is at the zero level set (shown by the red line in Fig. 3.3). For each pixel $p$ at coordinate $(x_p, y_p)$ on the regular grid, it finds its closest pixel at the zero level set (silhouette boundary) called $p_0$. The deformation direction for a pixel outside of the silhouette is $\overrightarrow{\mathbf{pp_0}}$, and that of a point inside of the silhouette is $\overrightarrow{\mathbf{p_0p}}$.

After the direction is determined, we then calculate the magnitude for deformation. As illustrated in Fig. 3.3, along the line of $p$ and $p_0$, we find the local maximum point $p_{max}$ and the local minimum point $p_{min}$ in the level set. The deformation flow for pixel $p$ is defined below,

$$x'_p = x_p \frac{\|x_{min} - x_0\|}{\|x_{max} - x_{min}\|}, \quad y'_p = y_p \frac{\|y_{min} - y_0\|}{\|y_{max} - y_{min}\|} \tag{3.5}$$

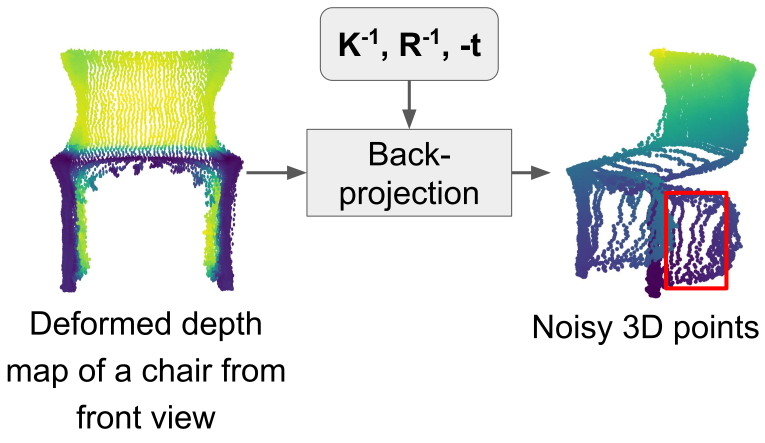
$$\mathbf{U}[x_p, y_p] = x'_p - x_p, \quad \mathbf{V}[x_p, y_p] = y'_p - y_p. \tag{3.6}$$

The Eq. (3.5) ensures that a point along the line between $p_{max}$ and $p_{min}$ moves to a point on the line between $p_0$ and $p_{min}$, such that the pixel is in the silhouette (the first criteria satisfied). Moreover, no pixels are collided (the second criteria satisfied).

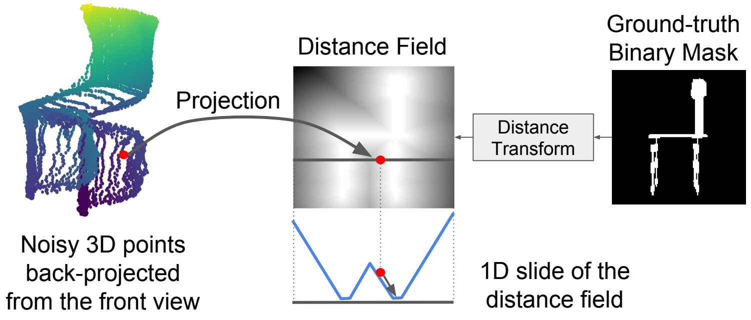
### 3.3.2 Distance Field Multi-view Consistency

As mentioned earlier, the 3D points unprojected from a predicted depth map are often noisy viewed from other viewpoints. Fig. 3.4(a) visualises this problem, where the point cloud back-projected from the front view of a chair contains many noisy points between the front and back legs of the chair. To alleviate this problem, we introduce a novel multi-view consistency supervision, which encourages the 3D points to project into the object silhouette (*i.e.*, foreground) but not the background at novel viewpoints.

To that end, we transform ground-truth binary masks (at novel viewpoints) into a distance field, where the foreground pixels' values are zero in the distance field (meaning no penalty), whereas the values of the background pixels are the distance



(a)


Ground-truth Binary Mask
Distance Field
Projection
Distance Transform
Noisy 3D points back-projected from the front view
1D slide of the distance field

(b)

**Figure 3.4:** (a) An example of a noisy reconstruction using only the depth as supervision. (b) Our distance field multi-view consistency. Given a ground-truth binary mask at a novel viewpoint, it is transformed to a distance field using distance transform Borgefors, 1986. 3D points are projected onto the distance field. The projection points are supervised by the multi-view consistency loss $L_{df}$ to move toward the object silhouette. Note that in the example above, the projected points move on the 1D surface only for the purpose of visualization. In reality, it moves on the 2D distance field surface.

to the closest boundary. Fig. 3.4(b) demonstrates an example of distance field. Such distance fields are used as the supervision signal to pull outliers (*i.e.*, points projected to the outside of the silhouettes) back to the object silhouettes (in 2D).

Technically, a 3D point $\mathbf{p}$ $(X_p, Y_p, Z_p)$ is projected to the distance field using the transformation and camera matrices at the novel viewpoint $n$, *i.e.*,

$$[x_p, y_p, 1]^T = \mathbf{K}_n(\mathbf{R}_n\mathbf{p} + \mathbf{t}_n), \tag{3.7}$$

where $[x_p, y_p]$ is the projection point coordinate in the distance field.

The multi-view consistency training loss $L_{df}$ becomes:

$$L_{df} = \sum_{n}^{N} \sum_{p}^{P} L_{df}^{(n,p)}, \tag{3.8}$$

where $N$ is the number of viewpoints and $P$ is the number of 3D points. $L_{df}^{(n,p)}$ is defined as:

$$L_{df}^{(n,p)} = \sum_{h}^{H} \sum_{w}^{W} \mathbf{F}^n[h,w] \max(0, 1 - |x_p - h|) \max(0, 1 - |y_p - w|), \tag{3.9}$$

where $H$ and $W$ are the height and width of the distance field respectively, $\mathbf{F}^n$ is the distance field at viewpoint $n$, and $\mathbf{F}^n[h,w]$ is the distance value at pixel location $[h,w]$.

Given a point at $[x_p, y_p]$, the values (distances) of the four neighbouring pixels are interpolated to approximate the corresponding distance field $\mathbf{F}[x_p, y_p]$. By minimising Eq. 3.9, this point is supervised to move toward the object silhouettes. This technique, called differentiable bilinear interpolation, was used in (Jaderberg et al., 2015) for differential image warping.

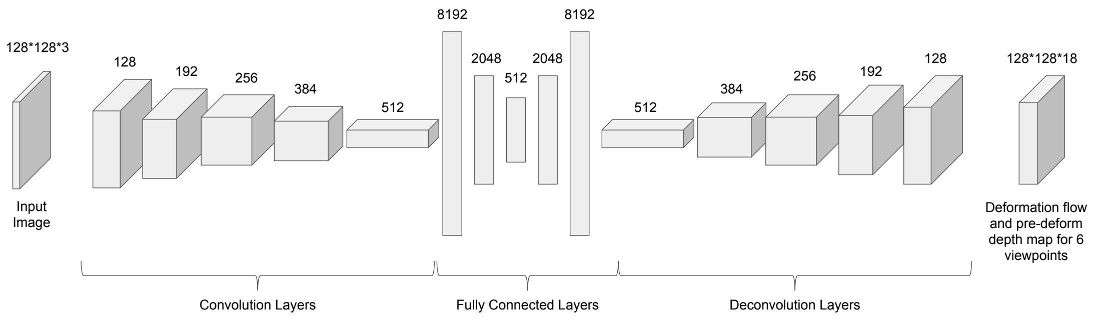


**Figure 3.5:** Network Architecture

### 3.3.3 Network Architecture and Training

**Training details** Our framework is implemented using the Tensorflow library (Martın Abadi et al., 2015). We train the network with the deformed depth map loss, deformation flow loss, and the distance field loss jointly. The final loss function is

$$L = \sum_{m}^{M} (L_d^m + L_U^m + L_V^m) + \lambda \sum_{n}^{N} L_{df}^n, \tag{3.10}$$

where $M$ is the six fixed viewpoints and $N$ is the number of distance field from novel viewpoints. We take a two-stage training procedure for all experiments. In the first stage, the network is trained to predict a coarse reconstruction with the per-view losses alone by setting the weight factor $\lambda$ of multi-view loss to zero. The second stage is a refinement stage using the multi-view loss and per-view losses jointly, where $\lambda = 1$. During training, the network is optimised using Adam (Kingma et al., 2014) with a momentum of 0.95 and initial learning rate of $1e^{-4}$.

**Network architecture** We use an autoencoder-like network where the encoder extracts image features and projects them into a latent space. The decoder comprises several 2D deconvolution layers to generate pairs of a pre-deformation depth map and a deformation flow map from 6 fixed viewpoints which are the faces of a cube centred at the object of interest. Details of the network architecture are depicted in Figure 3.5.

## 3.4 Experiments

We evaluate our proposed method beginning with ablation study of key components of our framework: deformed depth map and the distance field based multi-view consistency loss, followed by comparison to the prior art on single-view 3D object reconstruction. In addition, we test our method on a recently published real dataset to test its generalization ability to real images and compare with other methods reported.

### 3.4.1 Data Preparation

Following previous methods, we use a subset of ShapeNet, which contains objects in 13 categories, to train and evaluate our network. We render six depth maps along with the binary masks from fixed viewpoints of 6 faces of a cube where a 3D object is centred. The binary mask is used to construct the pseudo ground-truth deformation field. Additionally, we also render 24 RGB images along with its binary mask from arbitrarily sampling azimuth and elevation in $[0, 360)$, $[-20, 30]$ degrees respectively. The RGB images are input images to the network, and the binary masks are preprocessed to a distance field for multi-view consistency loss.

### 3.4.2 Quantitative Measurement

To evaluate results quantitatively, we use the average point-wise 3D Euclidean distance called Chamfer Distance between predicted and ground-truth point clouds.

$$D(S_1, S_2) = \sum_{p_i \in S_1} \min_{p_j \in S_2} \|p_i - p_j\|^2 + \sum_{p_j \in S_2} \min_{p_i \in S_1} \|p_i - p_j\|^2 \tag{3.11}$$

$$D_{S_1 \to S_2} = \sum_{p_i \in S_1} \min_{p_j \in S_2} \|p_i - p_j\|^2 \tag{3.12}$$

$$D_{S_2 \to S_1} = \sum_{p_j \in S_2} \min_{p_i \in S_1} \|p_i - p_j\|^2 \tag{3.13}$$

where $S_1$ is the predicted point cloud and $S_2$ is the ground-truth point cloud.

As demonstrated by Lin et al. (2018a), while the Chamfer Distance can evaluate the overall performance, it is also essential to report the prediction to ground-truth distance Eq. (3.12) and ground-truth to prediction distance Eq. (3.13) individually as they evaluate different aspects of the prediction point cloud. The former shows how far each prediction point to the closest ground-truth point (i.e., how accurate the prediction is), and the latter reports the distance from each ground-truth point to the closest prediction point indicating the surface coverage. Note that all numbers reported in the experiment section are scaled up by 100 for readability.

### 3.4.3 Ablation Study

In this section, we evaluate two key components of our framework: the deformed depth map and the distance field based multi-view consistency loss individually.

**Deformed depth map** We train two networks with our proposed deformed depth representation and the depth-mask representation (as a baseline) respectively in an identical training setting. This experiment shows that the depth-mask representation suffers from setting an appropriate threshold for foreground/background.

The networks are trained and evaluated on three categories (plane, car and chair). The per category results are reported in Fig. 3.6. It shows that the deformed depth

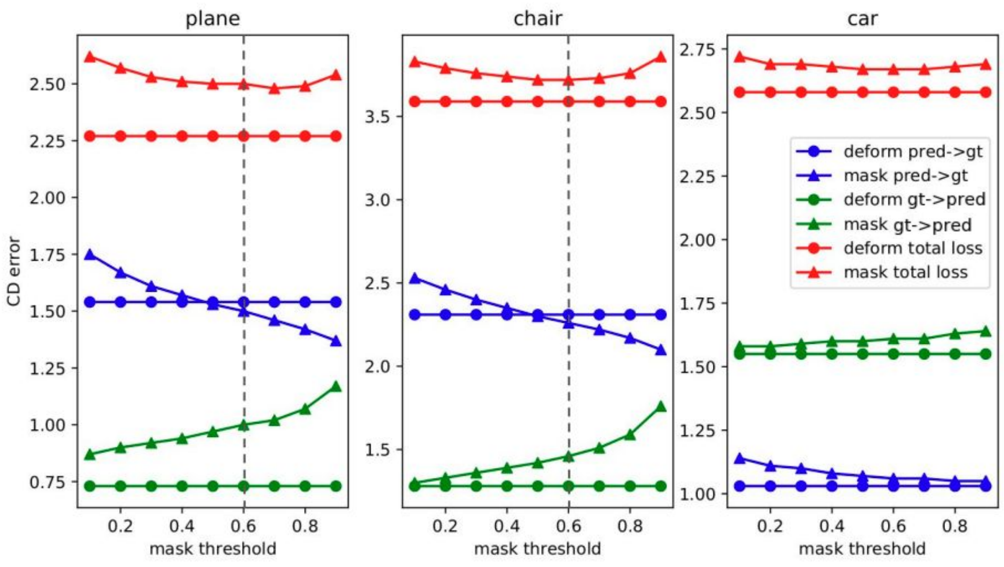


**Figure 3.6:** Chamfer Distance comparison with different foreground/background thresholds for the existing multi-view representation. The lines with triangle marks are the results of the proposed method, while the lines with round dots are the results of the depth-mask baseline. The lower gt $\rightarrow$ pred loss of our method demonstrates that our method provides better coverage than the baseline. Even though the baseline can achieve lower pred $\rightarrow$ gt loss when setting the threshold higher (e.g., threshold $\geq 0.6$) by only preserve points with high confidence, the penalty of coverage offset the accuracy increases leading to a higher overall loss.

representation consistently achieves lower loss under different threshold setting for the depth-mask baseline (shown in two red lines). There is a difficult trade-off between the prediction point accuracy and surface coverage in the baseline. When the $D_{pred \rightarrow gt}$ gradually decreases as the threshold rises (i.e., only good prediction points are preserved), the surface coverage loss increases significantly. More importantly, another disadvantage of depth-mask representation revealed from the experiment is that it is not trivial to select an optimal threshold for different object categories in a class-agnostic network, or even for different instances. To better visualize this issue, a few qualitative examples from Lin et al. (2018a) are given in Fig. 3.7.

**Distance field multi-view consistency** To evaluate our distance field based multi-view consistency loss, we compare to a baseline in which the loss is disabled, and a prior art Lin et al. (2018a) that use a binary mask multi-view consistency loss. The results reported in Table 3.1 show that after applying our distance field

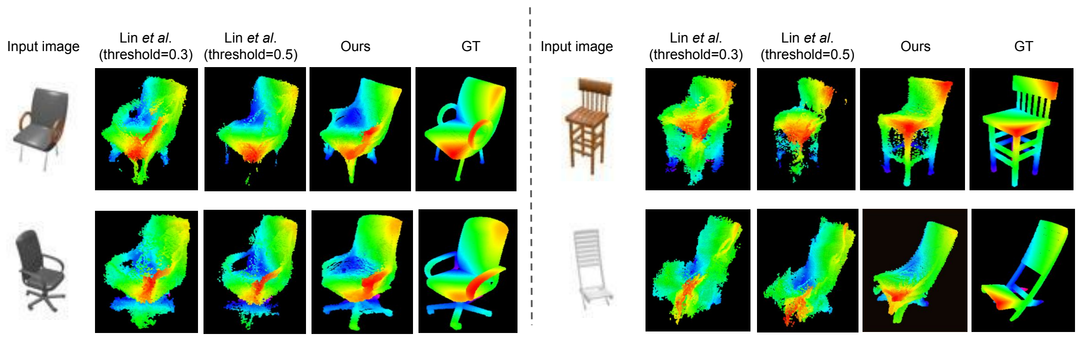

Input image
Lin et al. (threshold=0.3)
Lin et al. (threshold=0.5)
Ours
GT
Input image
Lin et al. (threshold=0.3)
Lin et al. (threshold=0.5)
Ours
GT

**Figure 3.7:** Visual comparison to Lin et al. (2018a)

| Methods | overall CD | prediction points |
|---|---|---|
| Lin et al. (2018a) (binary mask multi-view loss) | 3.240 / 3.531 | 31972 / 25401 |
| Ours (distance field multi-view loss) | 3.102 / **2.987** | **98304** / **98304** |

**Table 3.1:** Comparison on different multi-view consistency losses. The numbers reported in overall CD and the number of prediction points are: without multi-view consistency / with multi-view consistency

multi-view consistency, the network outperforms the baseline. Since our consistency loss does not simply mask out more points to reduce outliers, our method also outperforms the binary masked multi-view consistency Lin et al., 2018a.

### 3.4.4 Comparison with Prior Art

We compare our method against existing methods using point cloud or voxel grid representation in a synthetic setup and a real dataset in this section.

**Comparison to PSGN** The pre-trained model of PSGN (Fan et al., 2016) provided by the authors generates a point cloud aligned with the input image while ours is in a canonical pose. To be consistent in object pose, we use their published code to retrain their network to output point clouds in the canonical pose. Both PSGN and our network train on our rendered input images using the same training-test split. Our method outperforms PSGN in the overall performance in most of the categories (11 out of 13) as reported in Table 3.2.

To resolve the measurement bias to the density of predicted points, we also report the results of a densified PSGN, where we densify the point cloud size from

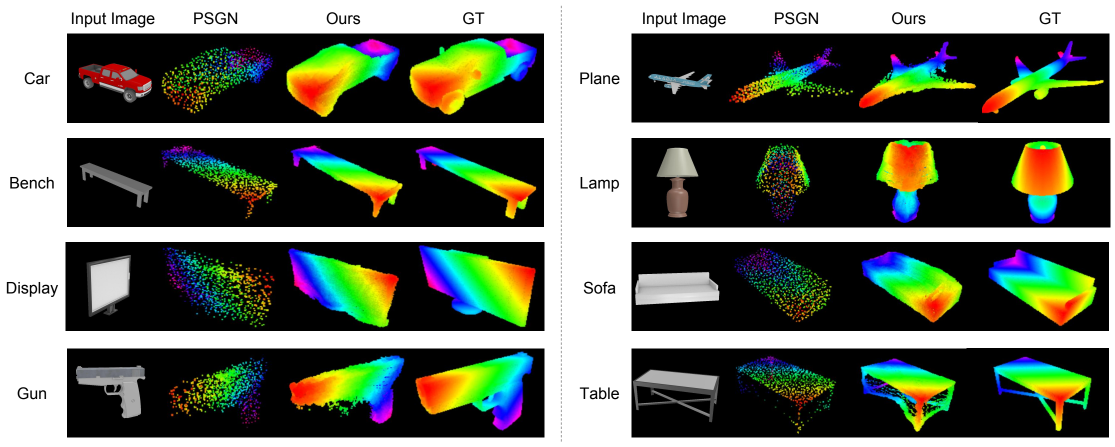


**Figure 3.8:** Visual comparison with PSGN

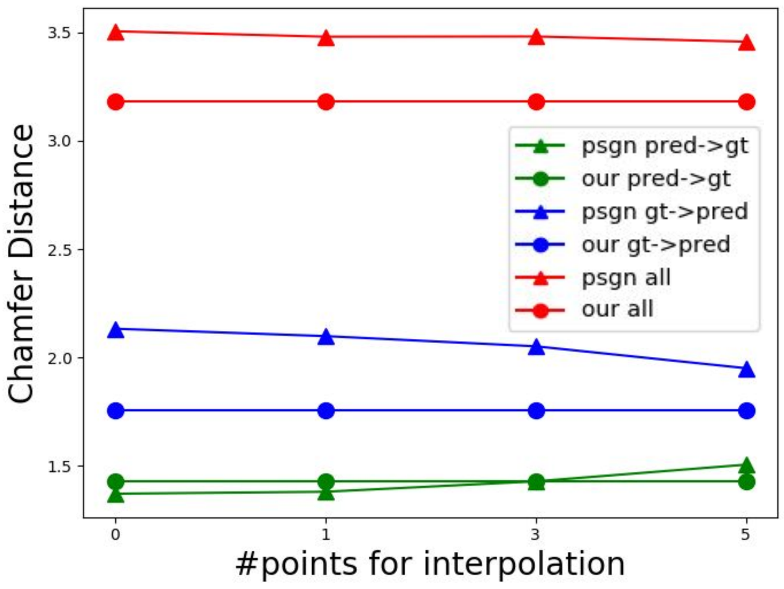


**Figure 3.9:** Densified PSGN Comparison

1024 to 98304 (the same size of our prediction) using linear interpolation between neighbouring points as a post-processing step. This densified PSGN is evaluated on five categories (chair, car, plane, bench and table) and the mean Chamfer Distance reported in Fig. 3.9. This graph shows that a densified/interpolated reconstruction cannot capture the finer 3D details that our method can. To contrast both methods visually, we present some qualitative examples in Fig.3.8.

**Comparison to Hierarchical Surface Prediction (HSP)** As PSGN has shown superior performance of point cloud representation over low-resolution voxel grid

| Category | PSGN (Fan et al., 2016) | Ours |
|---|---|---|
| plane | **2.582** | 2.66 |
| car | 3.253 | **2.897** |
| chair | 4.110 | **3.731** |
| table | 3.916 | **3.271** |
| bench | 3.660 | **3.357** |
| cabinet | 4.835 | **4.037** |
| display | 5.010 | **4.343** |
| lamp | 5.105 | **4.933** |
| speaker | 5.707 | **5.532** |
| gun | **2.949** | 3.259 |
| sofa | 4.644 | **4.267** |
| telephone | 3.999 | **3.588** |
| watercraft | 3.921 | **3.894** |

**Table 3.2:** Comparison with PSGN Fan et al. (2016) on 13 classes using Chamfer Distance.

| Methods | Chair | Plane | Car | Table | Bench |
|---|---|---|---|---|---|
| HSP | 4.716 | 3.878 | 3.487 | 4.072 | 4.467 |
| Ours | **3.731** | **2.660** | **2.897** | **3.271** | **3.357** |

**Table 3.3:** Chamfer Distance between ours and HSP

representation (e.g., 3D-R2N2), we provide a quantitative comparison between our method and HSP in Table 3.3. HSP up-scales the voxel grid to $512^3$ by using Octree data structure. Five categories of the ShapeNet are evaluated using input images and a pre-trained model provided by the author of HSP. To calculate Chamfer Distance for the HSP reconstructions, we generate a mesh from the voxels using the marching cube algorithm and sample uniformly the same number of points as ours from the mesh. Although the Octree method increases the resolution efficiently, it still suffers from the non-trivial occupancy thresholding and the “structurally unaware” cross entropy loss (i.e., missing thin structures), leading to poorer performance than ours.

### 3.4.5 Generalisation to Real Images

Pix3D (Sun et al., 2018), a large scale real dataset with high-quality image-shape pairs, has become available. To accommodate the varying background in real images, we train our method on synthetic images with backgrounds randomly selected from the SUN dataset (Xiao et al., 2016). After training, we evaluate

| Methods | CD ↓ | EMD ↓ |
|---|---|---|
| 3D-R2N2 | 0.239 | 0.211 |
| PSGN | 0.200 | 0.216 |
| 3D-VAE-GAN | 0.182 | 0.176 |
| DRC | 0.160 | 0.144 |
| MarrNet | 0.144 | 0.136 |
| AtlasNet | 0.125 | 0.128 |
| Pix3D | **0.124** | **0.124** |
| Ours | **0.124** | 0.125 |
| Pix3D (w/pose) | 0.118 | 0.119 |

**Table 3.4:** 3D shape reconstruction results on Pix3D dataset

on Pix3D images directly without finetuning on real images. Table 3.4 shows our method generalizes well to real data and achieves comparable performance to the state-of-the-art method then. Note that the best performance quoted in that paper uses joint training for object pose and 3D shape.

## 3.5 Conclusions

In this chapter, we introduced our work on using deep learning to reconstruct object shape from just a single-view observation. Specifically, we presented a novel framework to efficiently reconstruct 3D object surface via a dense point cloud fused by multiple deformed depth maps. Experiments have shown that our method can generate not only much denser but also higher quality point cloud than previous methods. Moreover, to refine the fused point cloud, we proposed a distance field based multi-view consistency loss during training. The ablation studies have shown the superior performance of the individual component over their baseline respectively. However, we also realized the limitations of a pure deep learning based shape reconstruction approach when we tried to integrate our method into an object SLAM system (Hosseinzadeh et al., 2019). Due to the lack of a large number of real images with ground-truth object shape annotations, we resort to synthetic images rendered from object CAD models for training, which causes a domain gap between training data and testing data. Furthermore, we also observed that the object reconstructions given the deep network might not satisfy well-defined geometry constraints used in traditional reconstruction pipelines. We show our approach to addressing these issues in the next chapter.

# 4

# Optimizable Object Reconstruction from a Single View

## Contents

*In this chapter we propose a new paradigm for single-view object reconstruction. Instead of learning an end-to-end mapping from an image to an object shape, inspired by traditional shape-prior-based approaches, we present a framework that brings together the best attributes of traditional geometry constraints and modern deep learning. We use an AutoEncoder and a Gaussian Mixture Model to learn a probabilistic low-dimension shape embedding to capture class specific shape prior information. During inference time, we first predict a shape code in this learnt* *shape embedding by mapping an image of an object of interest to the learnt shape embedding via a Convolution Neural Network. The shape code and object pose are iteratively refined by minimizing the silhouette and shape prior constraint. Although the scope of this work is still reconstructing object shape given a single observation, the joint reasoning using shape prior and geometry constraints paves the way to a multi-view setup (by including additional image evidences into the objective function). We show how the proposed method can be integrated in a object-centric mapping system in Chapter 5. Part of this chapter has been published at British Machine Vision Conference 2019 (K. Li et al., 2019).*

## 4.1 Introduction

The recent successes of deep learning have motivated researchers, including us, to depart from traditional multi-view reconstruction methods and directly learn the image to object-shape mapping using CNNs (Choy et al., 2016a; Fan et al., 2017b). These feed-forward networks show impressive results, with prior information about shape captured at training time and represented implicitly in the network's weights.

However, the drawbacks of these approaches became more apparent when we tried to integrate our single-view object shape reconstruction network into an object SLAM (Hosseinzadeh et al., 2019). First, the reconstruction given by a neural network is not optimised to satisfy well-studied geometry constraints (*e.g.* silhouette/photometric consistency). Second is the domain gap. Due to the lack of training data in real images for 3D object shape, it is common to train a neural network using synthetic images rendered from CAD models. Networks trained with synthetic images cannot reliably reconstruct objects given real images that are significantly different from its training data.

An alternative to an end-to-end mapping approach is to reason about the object shape and pose by combining 2D image cues (*e.g.* object silhouette) and prior knowledge of object shapes. This framework has been successfully used for inferring 3D shape from images before the resurgence of deep learning by Prisacariu et al. (2012) and Prisacariu et al. (2011) in single-view and Bao et al. (2013b) and Dame et al. (2013) in multi-view reconstruction frameworks. Prisacariu et al. (2012) and Prisacariu et al. (2011) propose to estimate the 3D object shape and pose given a single silhouette. To make this highly ill-posed problem well-posed, these works learn a probabilistic latent space embedding using a Gaussian Process Latent Variable Model (GPLVM). Object shape and pose inference is formulated as an optimisation problem to find the most likely shape (*i.e.* generated from the low variance area of the learned latent space) which can be projected using the optimal pose to the image plane. A silhouette consistency that minimises the discrepancy between the projection and the object silhouette is used to guide the optimisation. Using a non-linear dimensional reduction and an inference method which successfully accounts for the variance of the generated shape, Prisacariu et al. (2012) and Prisacariu et al. (2011) achieved the state-of-the-art reconstructions for a simple object like cars in the presence of heavy occlusion and background clutter. Furthermore, the iterative nature of these algorithms leads very naturally to an extension to a tracking framework of temporal refinement as pose or even shape change dynamically (Prisacariu et al., 2012).

These methods worked reasonably for object categories with small shape variations (*i.e.* lack of high-frequency details) – like cars or human bodies – and they can produce a dense and accurate volumetric reconstruction. However, more complex objects like chairs and tables, with larger shape variation and thin structures, were significant failure cases. Some of the limitations arise due to the use of GPLVM for dimensionality reduction. GPLVM does not scale well to be trained on large datasets or to learn priors on high-dimensional shape representations. This forces Prisacariu et al. (2012) to use frequency based compression techniques as a pre-processing step leading to loss of thin and other high-frequency object structures.

Deep learning has emerged as an attractive alternative to address these issues, and we explore the possibilities in this work. In particular, given a set of object CAD models, we learn a shape embedding represented by the bottleneck layer of an AutoEncoder. The shape embedding is given a probabilistic interpretation using a Gaussian Mixture Model (GMM) fitted on the shape codes. Given an

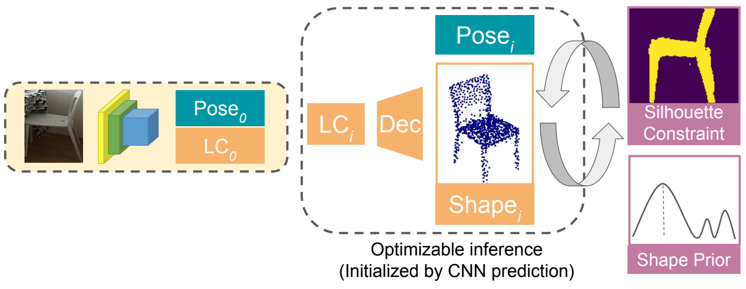


**Figure 4.1:** Given a single image, a deep neural net predicts the silhouette of the object, and also initializes the object pose and latent code of the object, both of which are then optimized by fitting the silhouette while remaining to be plausible explained by the shape prior.

image of an object of interest at inference time, we use a CNN to initialise object shape code and pose with respect to the input image, which are optimised in a manner that is analogous to traditional iterative approaches; *i.e.* by optimising the alignment between the image silhouette and the expected projection of the object, guided by the GMM shape prior. The image silhouette is obtained using a conventional segmentation network, and the learned shape prior helps avoid unrealistic shapes (because these correspond to low confidence regions of the GMM). Figure 4.1 encapsulates our framework.

In summary, our contributions are as follows:

- By leveraging the power of modern deep learning for non-linear dimensionality reduction, we show that a simple GMM fitting into this latent space can effectively be a probabilistic shape prior that plays a crucial role in our optimisation framework

- Instead of using a deep network as a black box to learn an end-to-end mapping from an image to an object shape for single-view object reconstruction, we present the first fully deep shape-prior aware inference method for single-view object reconstruction. We show that our framework – essentially a deep version of Prisacariu et al. (2011) – improves previous methods by a large margin.

## 4.2 Related Work

A large body of literature on object shape reconstruction has been covered in Chapter 2 and Section 3.2. We focus on shape-prior-based approaches and the differences in our proposed method from previous methods in this section.

Shape prior information has been used extensively in the object shape reconstruction community. An important realisation is that valid 3D shapes of objects belonging to a specific category are highly correlated. Deformable shape model, an idea of representing any object shape within an object category by deforming a mean shape using a set of shape bases, has been explored in Kar et al. (2015).

Dimensionality reduction also emerges as a prominent tool to model object shapes. For example, Gaussian Process Latent Variable Model or (Kernel) Principal Component Analysis is employed to learn a compact latent space of object shapes in Dambreville et al. (2008a), Prisacariu et al. (2012), Dame et al. (2013), and Engelmann et al. (2016). Recently deep AutoEncoders (Hinton et al., 2006; Vincent et al., 2008), Variational AutoEncoders (VAE) (Kingma et al., 2013), and Generative Adversarial Networks (GAN) (Goodfellow et al., 2014; Wu et al., 2016a) are alternatives based on deep learning.

With the resurgence of deep learning, there are many end-to-end systems proposed to solve the object shape reconstruction problem from a single RGB image directly (Choy et al., 2016a; Girdhar et al., 2016; Gwak et al., 2017; Tulsiani et al., 2017a; Wu et al., 2017a). However, a purely data-driven approach does not follow any well-studied geometry constraints and suffers from generalisation to real images. Tulsiani et al. (2017a) and Yan et al. (2016) aim to embed silhouette consistency in a network by using a projection consistency loss where the discrepancy between the projection of a reconstructed shape and the ground-truth silhouette is penalised for training a neural network. Nevertheless, it is still not guaranteed that the silhouette consistency is minimised at inference time.

More recently, there are several methods that take advantage of 2D image cues during inference to refine one-shot prediction from a deep neural net. For the scene-level geometry reconstruction, CodeSLAM (Bloesch et al., 2018a) uses nearby

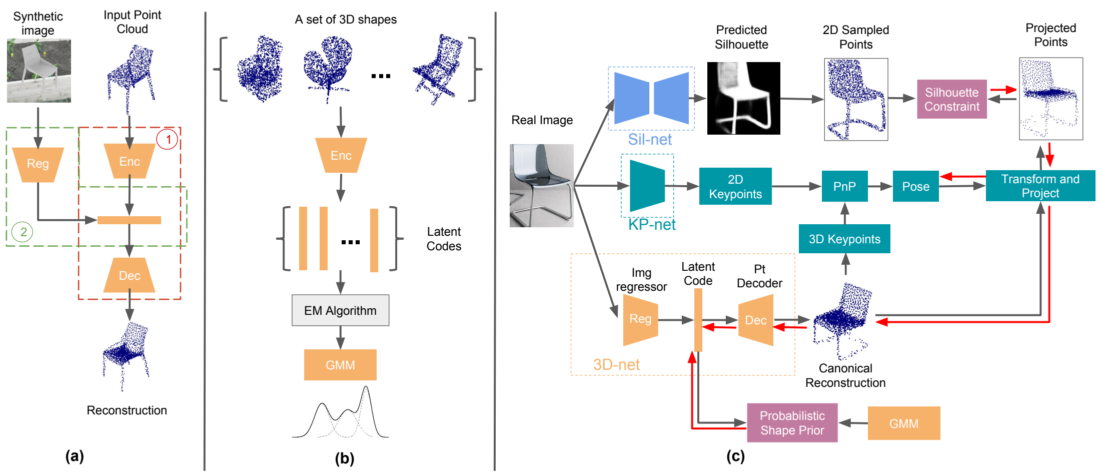


**Figure 4.2:** (a) We train the point cloud AutoEncoder to learn a latent space for 3D shapes (b) A set of 3D point clouds are mapped to the latent codes by the trained point cloud encoder. A GMM as a probabilistic model of the latent space is learned by fitting on this set of latent codes. (c) At inference time, we jointly estimate the shape and pose guided by the probabilistic shape prior via the learned GMM and the silhouette-shape constraint. The **orange**, **green**, **blue**, and **purple** blocks correspond to the components for shape, pose, silhouette, and loss terms respectively. The red arrows $\longrightarrow$ represent the gradient flow with respect to the latent code and the object pose during inference.

views to form a photometric loss, minimised by searching in the learned latent space of depth maps. MarrNet (Wu et al., 2018) and R. Zhu et al. (2018) enforce the 2.5D (silhouette, depth, or normal) - 3D shape constraint and photometric loss respectively to search in the latent space of object shapes at inference time. However, a common disadvantage of these works, which we address in our work, is that they do not consider a probabilistic prior in the latent space explicitly when they perform optimisation, which we show is essential for the optimisation in this work.

## 4.3 Method

This work aims to reconstruct a 3D shape of an object and its pose with respect to the camera from a single image. Our framework closely follows traditional object prior based 3D reconstruction methods by framing the reconstruction as an optimisation problem with respect to the object shape and pose assuming that a deep neural network provides us with the object silhouette. This inference method is described in the section 4.3.1. To facilitate the inference, we require a latent space which encodes the object shape in low dimensions, a learned probabilistic prior on this latent space providing us with the likelihood of any point in the latent space, and a single view object silhouette prediction to guide the shape and pose optimisation and initialisation for object shape and pose. How we learn the relevant prerequisites for inference is elaborated in section 4.3.2. Figure 4.2 illustrates the pipeline of our method.

## 4.3.1 Inference as Optimization

At inference, we are given $\boldsymbol{S_{sil}}$, $\boldsymbol{l_0}$, $\boldsymbol{R_0}$, and $\boldsymbol{t_0}$, which are sampled points from the predicted silhouette, initial latent code of shape, initial pose (rotation and translation) by neural nets respectively. Additionally, the learned probabilistic shape prior is a GMM on the latent space. There are two terms in our objective function to minimize: the silhouette-shape constraint $L_{sil}$ and the probabilistic shape prior $L_{shape}$, as shown in Equation (4.1). Within this optimization, we want to minimize the objective function with respect to $\boldsymbol{l}$, $\boldsymbol{R}$ and $\boldsymbol{t}$.

$$L(\boldsymbol{l}, \boldsymbol{R}, \boldsymbol{t}) = L_{sil} + \lambda L_{shape}, \tag{4.1}$$

where $\lambda$ is the weighting factor between these two loss terms.

**Silhouette-shape constraint** The projection of a 3D object shape in the camera frame should be consistent with the object's silhouette on the image. This can be achieved in the form of point cloud by defining a 2D Chamfer loss between the projection of the transformed 3D point cloud and sampled points of the predicted silhouette shown in Equation (4.2).

$$\begin{aligned} \boldsymbol{S} &= dec(\boldsymbol{l}) \\ L_{sil}(\boldsymbol{P}, \boldsymbol{S}, \boldsymbol{S_{sil}}) &= \sum_{\boldsymbol{s_i} \in \boldsymbol{S}} \min_{\boldsymbol{s_j} \in \boldsymbol{S_{sil}}} \|\boldsymbol{P}\boldsymbol{s_i} - \boldsymbol{s_j}\|^2 \\ &+ \sum_{\boldsymbol{s_j} \in \boldsymbol{S_{sil}}} \min_{\boldsymbol{s_i} \in \boldsymbol{S}} \|\boldsymbol{P}\boldsymbol{s_i} - \boldsymbol{s_j}\|^2, \end{aligned} \tag{4.2}$$

where $\boldsymbol{S}$ is a point cloud reconstruction in a predefined canonical pose generated by a latent code $\boldsymbol{l}$ through the decoder. $\boldsymbol{P}$ is a transformation matrix that is composed of $\boldsymbol{K}[\boldsymbol{R}|\boldsymbol{t}]$, camera intrinsic matrix $\boldsymbol{K}$, object rotation $\boldsymbol{R}$ and its translation $\boldsymbol{t}$. $\boldsymbol{S_{sil}}$ is a set of 2D sampled points from a predicted silhouette.

**Shape prior constraint** Because of the highly unconstrained nature of the silhouette-shape constraint, we devise this shape prior term to interpret the likelihood of a latent code given a probabilistic model – GMM in our case. This can mitigate the problem of generating unrealistic 3D shapes. Specifically, the shape prior is defined as to minimise the negative log likelihood of a latent code in the GMM, which can be written as follows.

$$L_{shape}(\boldsymbol{l}) = -log(\sum_{k=0}^{K} \pi_k \mathcal{N}(\boldsymbol{l}|\boldsymbol{\mu_k}, \boldsymbol{\Sigma_k})), \tag{4.3}$$

where $\pi_k$, $\boldsymbol{\mu_k}$, and $\boldsymbol{\Sigma_k}$ are the weight, mean, and covariance matrix of the $k^{th}$ Gaussian component, and $K$ is the number of Gaussian components.

## 4.3.2 Learning

As explained in section 4.3.1, our inference method requires learning a probabilistic low-dimensional representation of object shape from data and three networks predicting object silhouette, initial object pose and object shape in the form of a latent code. In this section we describe how we learn these components.

### Learning shape latent space and prior

Our inference method heavily relies on learning a low-dimensional latent space of shape from which a decoder network can map a latent code $\boldsymbol{l}$ to a unique 3D point cloud $\boldsymbol{S}$. We train an AutoEncoder network to learn this latent embedding of the object shapes. The encoder $enc(.)$ maps a point cloud $\boldsymbol{S}$ to a latent code $l$, followed by the decoder $dec(.)$ reconstructing a point cloud $\hat{\boldsymbol{S}}$. This network is trained using 3D Chamfer loss as defined below:

$$L_{cd}(\hat{\boldsymbol{S}}, \boldsymbol{S}) = \sum_{\hat{\boldsymbol{s}}_i \in \hat{\boldsymbol{S}}} \min_{\boldsymbol{s_j} \in \boldsymbol{S}} \|\hat{\boldsymbol{s}}_i - \boldsymbol{s_j}\|^2 + \sum_{\boldsymbol{s_i} \in \boldsymbol{S}} \min_{\hat{\boldsymbol{s}}_j \in \hat{\boldsymbol{S}}} \|\hat{\boldsymbol{s}}_j - \boldsymbol{s_i}\|^2, \tag{4.4}$$

To learn a probabilistic model as the shape prior, we fit a GMM to the learned latent space. Technically, we map point clouds $\{\boldsymbol{S}_1 ... \boldsymbol{S}_N\}$ in the training set to the latent space using the trained point cloud encoder, to get a set of latent codes $\{\boldsymbol{l}_1 ... \boldsymbol{l}_N\}$. A GMM is learned by fitting this set of codes with the Maximum Likelihood Estimate (MLE) of its parameters (Equation (4.5)), where the Expectation-maximisation algorithm is used.

$$\min_{\substack{\pi_1 ... \pi_K, \\ \boldsymbol{\mu_1} ... \boldsymbol{\mu_K}, \\ \boldsymbol{\Sigma_1} ... \boldsymbol{\Sigma_K}}} - \sum_{i=0}^{N} log(\sum_{k=0}^{K} \pi_k \mathcal{N}(\boldsymbol{l_i} | \boldsymbol{\mu_k}, \boldsymbol{\Sigma_k})) \tag{4.5}$$

where $\boldsymbol{\pi_k}$, $\boldsymbol{\mu_k}$, and $\boldsymbol{\Sigma_k}$ is the weight, mean and covariance matrix of the $k^{th}$ Gaussian component, and $N$ and $K$ are the number of observations and the number of Gaussian components. The number of Gaussian component $K$ is chosen by maximizing the GMM likelihood on a holdout set.

There are a number of generative models that can be used as a shape prior. We employ GMM with a simple AutoEncoder following the recent study of Achlioptas et al. (2018) which empirically shows that, in the form of the point cloud, among variants of GAN, and GMM, the latter outperforms other models in terms of generalisation and diversity of shapes, indicating that the GMM captures a better distribution of object shapes.

### Learning segmentation and initialization

Our inference requires three networks, each taking an RGB image containing the object of interest as the input. These networks are: (i) The 3D-net predicts the latent code $\boldsymbol{l}_0$ corresponding to an image which can be used to initialise the object shape using the decoder $dec(.)$. (ii) The Keypoint (KP)-net predicts the projection locations of 8 corner points of the object 3D bounding box, which are used to recover the initial object pose $\boldsymbol{R}_0$ and $\boldsymbol{t}_0$ with respect to the camera. (iii) the Sil-net predicts the object's silhouette in the image that guides the inference. In this section, we outline the procedure adopted in this work to train these components.

**3D-net**. This network consists of two parts. An image regressor learns to map an image to a latent code, which is decoded by the learned $dec(.)$ introduced in Section 4.3.2 to generate a 3D point cloud. During training, the image regressor is trained by minimising the $L_1$ loss between its prediction and the ground-truth latent code produced by the trained encoder. At inference, the image regressor maps the input image to a latent code $\boldsymbol{l}_0$ as initialisation of object shape in the latent space. While this process looks similar to Chapter 3, where we map an image to an object shape using a deep network, the difference is twofold: i) We have built a low-dimensional shape embedding. An image is first mapped to the shape embedding and then decoded to an object shape. ii) Instead of using the network prediction as the reconstruction, the shape and pose are further refined using the silhouette constraint.

**KP-net**. We define eight corner points of the object's 3D bounding box in the predefined canonical pose as 3D keypoints. The KP-net learns to predict these eight points' projection locations onto the image plane using the ground-truth object pose available at the training time. $L_1$ loss between the ground-truth 2D locations of the keypoints and the prediction is employed to supervise the training.

The 2D-3D keypoint correspondence is used to recover the object pose with respect to the camera, which is an initialisation for inference. More specifically, during inference time, we retrieve the 3D corner points $\boldsymbol{X_{3d}}$ from the point cloud reconstruction of the 3D-net, together with the KP-net prediction $\boldsymbol{X_{2d}}$ to form the correspondence, which is formulated as a PnP problem as shown in Equation (4.6) and solved iteratively using the Levenberg Marquardt algorithm (Moré, 1978).

$$L(\boldsymbol{R}, \boldsymbol{t}) = \|\boldsymbol{K}(\boldsymbol{R}\boldsymbol{X_{3d}} + \boldsymbol{t}) - \boldsymbol{X_{2d}}\|_2, \tag{4.6}$$

**Sil-net**. A straight-forward U-net structure is adopted to map an RGB image to an object silhouette, from which we sample points used in the silhouette-shape constraint. The network training is supervised by the binary cross entropy loss. Instead of training the Sil-net from scratch, our framework can also use an off-the-shelf semantic segmentation network (He et al., 2017; Chen et al., 2018) or finetune from such a network with pretrained network weights.

### 4.3.3 Implementation Details

**Network architecture** We denote the convolution and transposed convolution operation as $Conv(k, s, f)$ and $Deconv(k, s, f)$ respectively. The $k$, $s$, $f$ denote the kernel size, stride, and the number of output feature channels respectively. Additionally, we use $FC(f)$ to represent a fully connected layer, where $f$ is the number of output channels. All learnable layers are connected via a non-linear activation layer – Rectified Linear Unit (ReLU) except for the last layer.

The network architectures of four networks in our framework are given as follows:

**Point cloud AutoEncoder** We adapt the network architecture from PointNet (Qi et al., 2017). Features of each 3D point are extracted independently using $1D$ convolutions. Max pooling on the features of each 3D point is used to extract global features of the entire point cloud.

Input Point Cloud $(2048, 3)$ - $Conv(1, 1, 128)$ - $Conv(1, 1, 128)$ - $Conv(1, 1, 256)$ - $Conv(1, 1, 512)$ - $Max\ Pooling$ - $FC(256)$ - $FC(256)$ - $FC(512)$ - $FC(2048 \times 3)$ - $Reshape(2048, 3)$.

**Image regressor**

Input Image $(128, 128, 3)$ - $Conv(3, 2, 32)$ - $Conv(3, 1, 32)$ - $Conv(3, 1, 32)$ - $Conv(3, 2, 64)$ - $Conv(3, 1, 64)$ - $Conv(3, 1, 64)$ - $Conv(3, 2, 128)$ - $Conv(3, 1, 128)$ - $Conv(3, 1, 128)$ - $Conv(3, 2, 256)$ - $Conv(3, 1, 256)$ - $Conv(3, 1, 256)$ - $Conv(3, 2, 256)$ - $Conv(3, 1, 256)$ - $Conv(3, 1, 256)$ - $Conv(3, 2, 512)$ - $FC(1024)$ - $FC(512)$ - $FC(256)$.

**Sil-net**

Input Image $(128, 128, 3)$ - $Conv(3, 2, 32)$ - $Conv(3, 1, 32)$ - $Conv(3, 1, 32)$ - $Conv(3, 2, 64)$ - $Conv(3, 1, 64)$ - $Conv(3, 1, 64)$ - $Conv(3, 2, 128)$ - $Conv(3, 1, 128)$ - $Conv(3, 1, 128)$ - $Conv(3, 2, 256)$ - $Conv(3, 1, 256)$ - $Conv(3, 1, 256)$ - $Conv(3, 2, 256)$ - $Conv(3, 1, 256)$ - $Conv(3, 1, 256)$ - $Conv(3, 2, 512)$ - $Deconv(5, 2, 384)$ - $Deconv(5, 2, 384)$ - $Deconv(5, 2, 384)$ - $Deconv(5, 2, 192)$ - $Deconv(5, 2, 96)$ - $Deconv(5, 2, 1)$.

**KP-net**

Input Image $(128, 128, 3)$ - $Conv(3, 2, 32)$ - $Conv(3, 1, 32)$ - $Conv(3, 1, 32)$ - $Conv(3, 2, 64)$ - $Conv(3, 1, 64)$ - $Conv(3, 1, 64)$ - $Conv(3, 2, 128)$ - $Conv(3, 1, 128)$ - $Conv(3, 1, 128)$ - $Conv(3, 2, 256)$ - $Conv(3, 1, 256)$ - $Conv(3, 1, 256)$ - $Conv(3, 2, 256)$

- $Conv(3,1,256)$ - $Conv(3,1,256)$ - $Conv(3,2,512)$ - $FC(1024)$ - $FC(512)$ - $FC(256)$ - $FC(16)$.

**Training details** Data to train all networks is generated from the ShapeNet dataset (Chang et al., 2015). We render synthetic images from the textured 3D CAD models from random viewpoints and overlay random background on the rendered objects to simulate the background clutter present in the real images. Background images are acquired from SUN dataset (Xiao et al., 2016). Random flips, crops and image colour augmentation are used for training. To get the ground-truth 2D keypoint locations for training the KP-net, we use the 3D coordinates of corner points of the ground-truth shape's bounding box. We transform and project the bounding box coordinates to the image plane using the corresponding object pose, which is used to render the image.

The point cloud AutoEncoder is trained for 400 epochs, and the image regressor is trained for 25 epochs. The KP-net and Sil-net are trained end-to-end for 60 epochs and 25 epochs respectively. Adam optimiser (Kingma et al., 2014) is used to train all these networks.

**Inference details** 6-DoF object pose and the shape latent code are optimised jointly reusing Adam optimiser in Tensorflow (Martın Abadi et al., 2015). It is important to note that we only update the latent code corresponding to inferred object shape and not the decoder network's weights while doing gradient descent. The optimisation is terminated when the rate of change in loss function diminishes, or the maximum 2000 iterations are reached.

## 4.4 Experiments

We provide experimental results of our framework in single-view object reconstruction and object pose estimation. A series of ablation studies are also presented to analyse the effect of our inference procedure and the probabilistic shape prior.

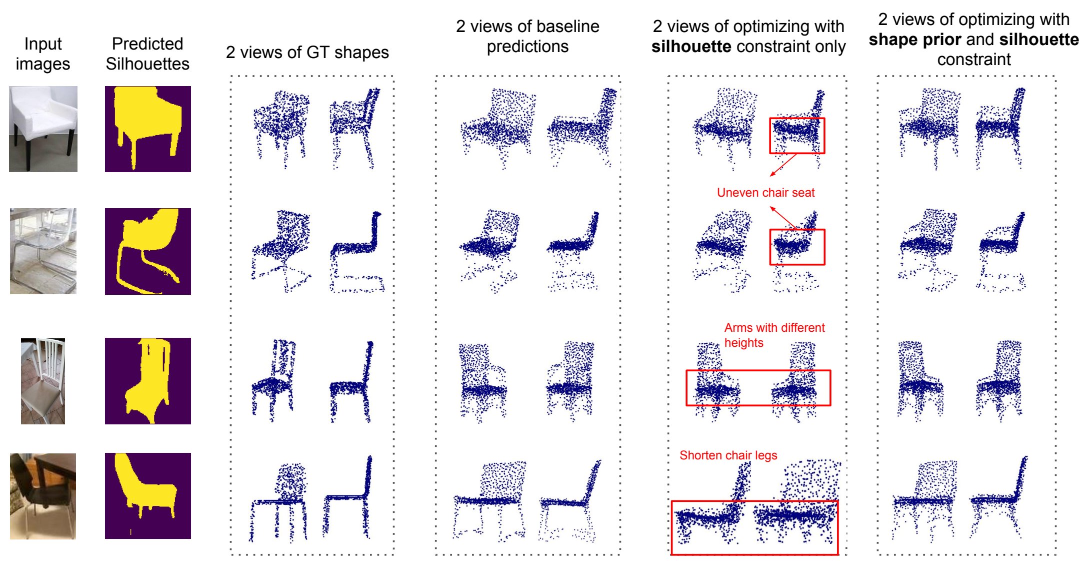

Input images
Predicted Silhouettes
2 views of GT shapes
2 views of baseline predictions
2 views of optimizing with **silhouette** constraint only
2 views of optimizing with **shape prior** and **silhouette** constraint
Uneven chair seat
Arms with different heights
Shorten chair legs

**Figure 4.3:** Qualitative Results on Pix3D. We compare our full model against the baseline (without optimization at inference time), and optimization without the probabilistic shape prior. From left to right: input image, predicted silhouette, ground-truth point cloud, baseline output, optimize without shape prior, full model.

### 4.4.1 Testing Data and Evaluation Protocol

We take advantage of a recently published Pix3D (X. Sun et al., 2018) dataset – a dataset consisting of diverse shapes aligned with corresponding real images. We choose the examples of chair class to demonstrate our method for two reasons. Firstly, chair class is a major component of this dataset and exhibits the most shape diversity. There are 2894 images of chair corresponding 221 unique chair shapes to be used, which outnumbers other object classes greatly. For instance, the table, being the second largest class, only has 738 images corresponding to 63 shapes. Secondly, the authors of Pix3D publish benchmark results for most state-of-the-art methods on the chair class, facilitating a convenient comparison of our work with existing methods.

Other alternatives to Pix3D that have been used popularly in the community are either purely synthetic datasets like ShapeNet (Chang et al., 2015), or PASCAL 3D+ (Xiang et al., 2014) which was created by associating a small number of overlapping 3D CAD models per category for both training and test images of a

| Methods | CD ↓ | EMD ↓ |
|---|---|---|
| 3D-R2N2 | 0.239 | 0.211 |
| PSGN | 0.200 | 0.216 |
| 3D-VAE-GAN | 0.182 | 0.176 |
| DRC | 0.160 | 0.144 |
| MarrNet | 0.144 | 0.136 |
| AtlasNet | 0.125 | 0.128 |
| Chapter 3 | 0.124 | 0.125 |
| Pix3D | 0.119 | 0.118 |
| Ours (baseline) | 0.107 | 0.116 |
| Ours | **0.099** | **0.112** |

**Table 4.1:** 3D shape reconstruction results on Pix3D dataset

particular object category to create pseudo ground truth. Due to this reason, as discussed in Tulsiani et al. (2017a) and Wu et al. (2017a) PASCAL 3D+ is not deemed fit for shape reconstruction evaluation.

We use 3D Chamfer Distance (CD as defined in (4.4)) and Earth Mover Distance (EMD as defined in (4.7)) as evaluation metrics for shape reconstruction.

$$EMD(S_1, S_2) = \frac{1}{\|S_1\|} \min_{\phi : S_1 \to S_2} \sum_{x \in S_1} \|x - \phi(x)\|_2, \tag{4.7}$$

where $\phi : S_1 \to S_2$ is a bijection that matchs a point in $S_1$ to a point in $S_2$ such that the cost (distance) of all matched pairs is minimized. We use the approximation implemented in Pix3D benchmark to calculate the EMD efficiently.

Following the prior art, we evaluate pose using the MedErr: median of geodesic distance between the predicted rotation and ground-truth rotation defined in Equation (4.8) and the $Accu_{\frac{\pi}{6}}$: percentage of geodesic distance smaller than $\frac{\pi}{6}$.

$$\Delta(\boldsymbol{R_p}, \boldsymbol{R_{gt}}) = \frac{\|log(\boldsymbol{R_{gt}^T R_p})\|_F}{\sqrt{2}}, \tag{4.8}$$

### 4.4.2 Shape Reconstruction on Pix3D

We compare the performance of our 3D shape prediction pipeline with prior art including 3D-VAE-GAN (Wu et al., 2016a), 3D-R2N2 (Choy et al., 2016a), PSGN (Fan et al., 2017b), DRC (Tulsiani et al., 2017a), MarrNet (Wu et al., 2017a) and

Pix3D (Wu et al., 2018) which is a variant of MarrNet that decouples shape and pose prediction. The quantitative result is reported in Table 4.1. The performance evaluation for most of these methods listed above is taken from Pix3D paper (X. Sun et al., 2018). "Ours (baseline)" in Table 4.1 refers to the one-shot deep network prediction without any optimization during inference time outlined in Section 4.3.1. It is notable that our baseline already outperforms the prior art. We believe that this performance boost is due to a unique combination of the best practices carefully picked from the prior art to design our one-shot prediction baseline. For instance, our method predicts a canonical point cloud to represent an object shape. Other methods are either opting for in-pose point cloud representation (Fan et al., 2017b), *i.e.*, coupling the shape pose or they work with volumetric or mesh representation for object shapes. In addition, similar to TL-embedding network (Girdhar et al., 2016) and 3D-VAE-GAN (Wu et al., 2016a), we learn a latent space of 3D shape explicitly instead of a direct image to shape mapping. The advantage of building a shape embedding for object shapes becomes apparent when comparing to our work in the previous Chapter. Although we only predict a sparse point cloud in this work, we still outperform Chapter 3 because we train an embedding for object shapes.

More importantly, the most significant point we want to emphasise from the results in Table 4.1 is that our probabilistic shape prior aware optimisation method for shape and pose outperforms this solid baseline (along with other previous methods) by a significant margin. More qualitative examples of our in-pose reconstruction are shown in Figure 4.6 and 4.7 at the end of this Chapter.

| Methods | Shape | | Pose | |
|---|---|---|---|---|
| | CD ↓ | EMD ↓ | MedErr ↓ | $Accu_{\frac{\pi}{6}}$ ↑ |
| baseline | 0.107 | 0.116 | 12.22 | 0.75 |
| $L_{sil}$ only | 0.102 | 0.117 | 11.40 | **0.77** |
| $L_{sil} + L_{shape}$ | **0.099** | **0.112** | **11.00** | **0.77** |
| w/ GT Silhouette | 0.093 | 0.111 | 9.47 | 0.78 |

**Table 4.2:** Quantitative analysis on the effect of silhouette-shape constraint and probabilistic shape prior at inference.

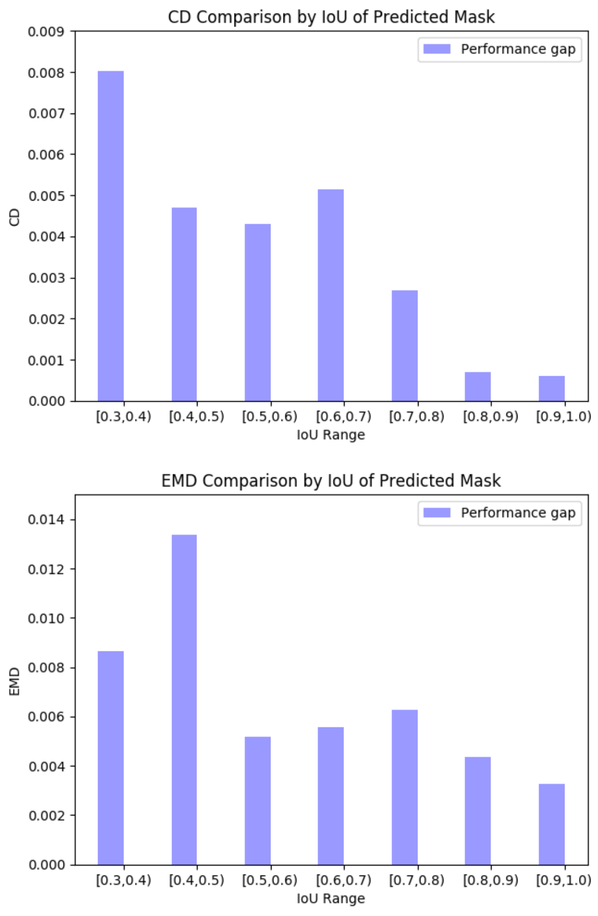


**Figure 4.4:** loss by IoU. We visualize the performance gap on shape reconstruction between optimization with and without shape prior at different IoU levels of the predicted silhouette. It is clear that the shape prior increases the robustness of the optimization to noisy silhouettes.

### 4.4.3 Ablation Study on Shape prior

In this section, we analyse the effect of different components used in our shape and pose inference quantitatively (Table 4.2) and qualitatively (Figure 4.3). In particular, we demonstrate that the proposed probabilistic shape prior plays a crucial part in shape and pose estimation. To that end, we conduct an ablation study on the shape prior constraint $L_{shape}$. We compare performance among our one-shot baseline, optimisation using the silhouette constraint, and optimisation using both silhouette and the shape prior constraint.

Figure 4.3 visualises the input images, predicted silhouettes, and shape reconstruction given by different inference schemes. It is clear that after optimisation using both the shape prior and silhouette constraint can lead to a reconstruction that is closer to the ground-truth shape than that of one-shot prediction. Notably, if the shape prior is discarded, optimising with $L_{sil}$ alone is likely to generate unnatural shapes, such as uneven chair seat, chair arms with different heights, and unrealistic short chair legs, highlighted by red boxes in the figure.

As for the quantitative comparison, it can be seen that minimising silhouette-shape constraint alone only lowers the Chamfer Distance, but worsens when measured by EMD. This may reflect that the estimated reconstructions overfit to the CD metric, which is used as the training loss. When we minimise the objective function, which takes into account the likelihood of the learned latent embedding of shapes, the reconstruction performance improves on both measures significantly. To put this performance boost in perspective, we use the ground-truth silhouette annotations available in Pix3D to optimise the silhouette constraint. By leveraging the shape prior constraint, the optimisation is less affected by imperfect silhouette predictions that are inevitable in practice.

Further, we analyse this performance gap with different accuracy levels of the predicted silhouette in figure 4.4. We plot the performance gap between the silhouette-only optimisation and the shape-prior aware optimisation measured by the median CD and EMD respectively in the top and bottom part of the figure corresponding to different silhouette prediction accuracy quantified in terms of mean IOU. The height of a bar reflects how much worse the silhouette-only optimisation compared to the shape prior aware counterpart given different qualities of the predicted silhouettes (measured by the IoU against the ground-truth silhouette). It is clear that when the predicted silhouettes are inaccurate, the silhouette-only optimisation performs the worst. It demonstrates the evaluation of having the shape prior term in the energy function.

Our contention is that the learned shape prior avoids the regions of the latent space to restrict the inferred shape to be limited to realistic object-like solutions. To further explore this hypothesis, we choose three points from the latent space

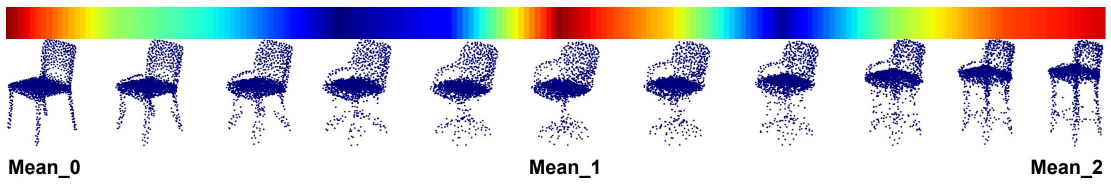


**Figure 4.5:** Exploration on the latent space. We denote the likelihood of a shape explained by the GMM using colors. Red means high likelihood whereas blue represent low likelihood.

| Methods | MedErr ↓ | $Accu_{\frac{\pi}{6}}$ ↑ |
|---|---|---|
| Regression | 21.99 | 0.63 |
| Classification | 15.41 | 0.71 |
| Keypoint | **12.22** | **0.75** |

**Table 4.3:** Quantitative results on object pose estimation.

corresponding to the mean of different Gaussian components and interpolate linearly in the latent space to generate intermediate sample chair shapes. The results are visualized in figure 4.5. It can be seen that the latent space learned using deep AutoEncoder constitutes of highly unlikely points corresponding to very unlikely chair shapes, but all these shapes have high variance – for instance, between the latent code of a four-legged chair (mean_0) and that of a swivel chair (mean_1), the likelihood of an unrealistic shape as a mix of two structures is particularly low.

### 4.4.4 Pose Estimation using Keypoints

We compare our keypoint based approach with direct classification and regression approaches, which directly predict the rotation parameters in a classification or regression manner. To have a fair comparison to these approaches, we use the same training data and similar network architectures except for the last output layer of all these approaches because of different output dimensions. Our keypoint based method outperforms both regression and classification approaches, as shown in Table 4.3.

## 4.5 Conclusion

Inspired by traditional shape prior based reconstruction methods from a single view, we presented a deep learning based framework that takes the silhouette constraint and a probabilistic shape prior explicitly into account. In particular, to enforce the geometry constraint during inference time, we reframed the single view object reconstruction as an optimization problem, where we not only reasoned about the silhouette-shape constraint but also applied a probabilistic shape prior in the latent space of shape. Our proposed method outperformed previous methods on a large-scale real-image dataset.

While this chapter's focus is still reconstructing object shape from a single observation, the inference as optimization scheme provided a better way to aggregate information from multiple observations than the one-shot network prediction. We will introduce how we extend what we have learned in this chapter to a multi-object and multi-view setup in Chapter 5, where we introduce our first object-centric mapping system.

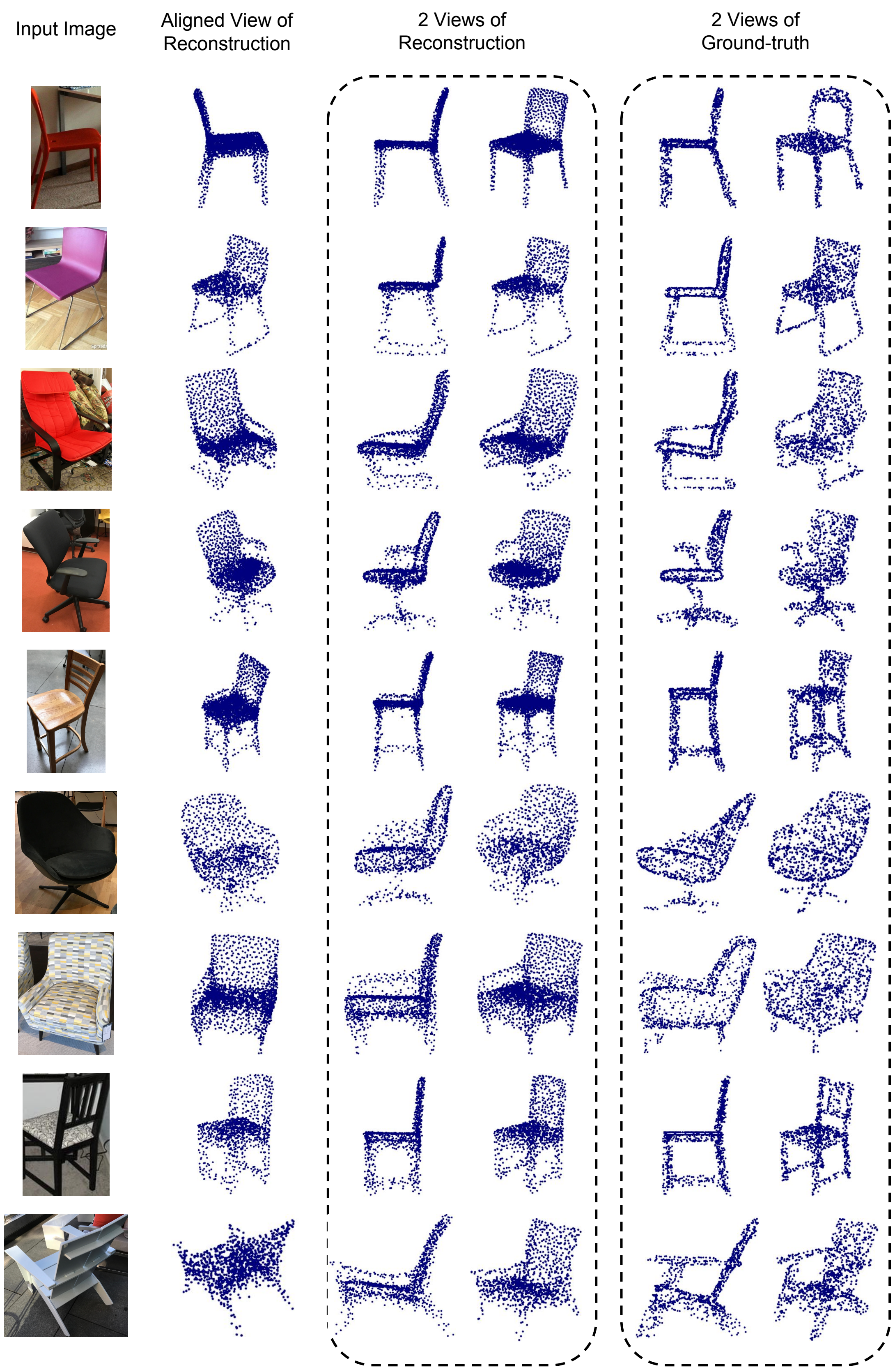


**Figure 4.6:** From left to right: input image, aligned view of reconstruction (the shape is transformed by the predicted pose), 2 views of reconstruction, and 2 views of GT.

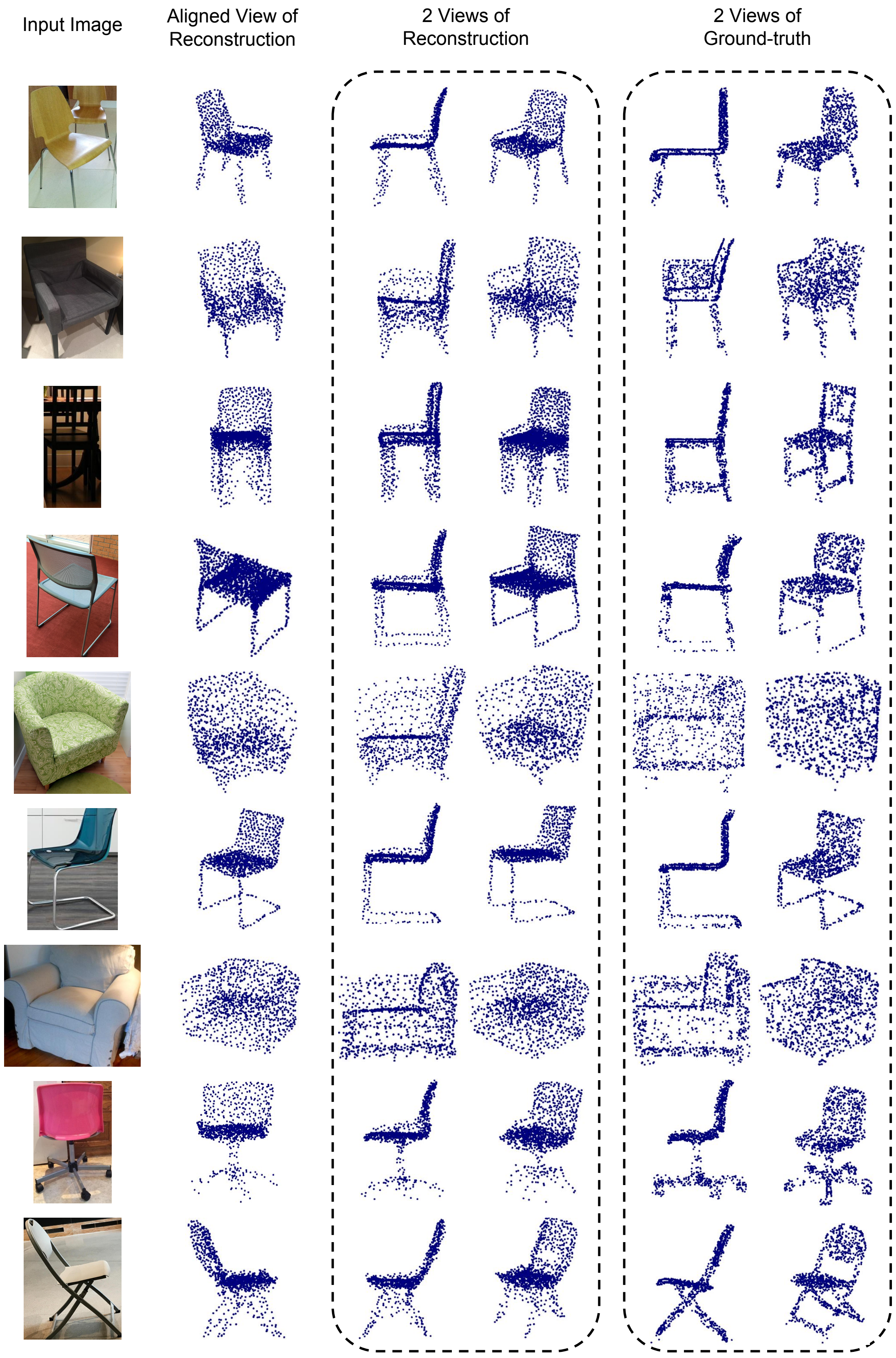


**Figure 4.7:** From left to right: input image, aligned view of reconstruction (the shape is transformed by the predicted pose), 2 views of reconstruction, and 2 views of GT.

# 5
# FroDO: From Detections to 3D Objects

## Contents

*We propose a novel paradigm for object-centric mapping in a monocular modality dubbed FroDO — From Detections to 3D Objects in this chapter. In a coarse to fine fashion, FroDO progressively localizes objects using 3D bounding boxes, reconstructs their coarse 3D shape in the form of a sparse point cloud, and lastly reconstruct high-fidelity object shapes represented by Truncated Signed Distance Functions (TSDF). FroDO first associates 2D detections from images by a line segment clustering algorithm. The associated 2D detections together with camera*

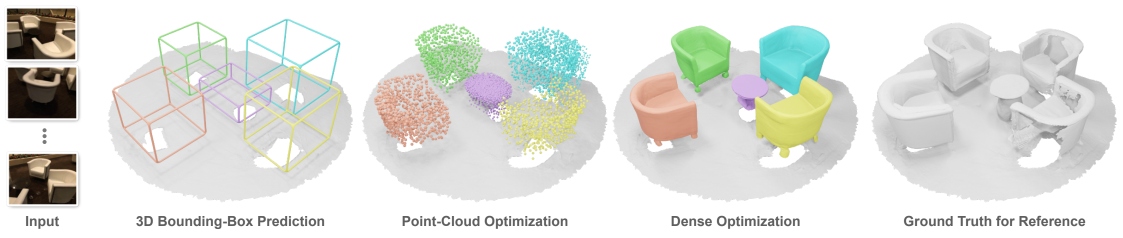


**Figure 5.1:** Given an RGB sequence and extrinsic parameters of each image, FroDO dectects objects and infers their poses and a progressively fine grained and expressive object shape representation. Results on a real-world sequence from ScanNet (Dai et al., 2017a)

*poses are used to fit a 3D ellipsoid for object localization. The key to the coarse-to-fine shape optimization paradigm is a joint shape embedding where we can switch seamlessly between point cloud and SDF. FroDO, as an object-centric mapping approach, shows superior performance over geometric-only reconstruction algorithm by completing missing regions due to either invisibility or low texture. FroDO also outperforms other learning based methods on object reconstruction as a result of the novel shape embedding and the coarse-to-fine optimization. The development presented in this chapter has been published at Computer Vision and Pattern Recognition 2020 (Runz et al., 2020).*

## 5.1 Introduction

From a sparse set of 3D points (Davison et al., 2007) to a dense TSDF (Newcombe et al., 2011a), we have seen a progression towards a denser geometric reconstruction. However, it is increasingly clear that a geometric-only reconstruction regardless of its density is insufficient for advanced robotic tasks where interactions with the environment and humans are expected. An object-centric mapping system offers a natural and compact way to capture geometric and semantic information or even the dynamic elements of an environment. Although object-centric mapping has been tackled extensively, few works have sufficiently presented a system that can work in the wild. For instance, the famous work SLAM++ (Salas-Moreno et al., 2013b) relies on a pre-defined object database. To address this limitation, Dame et al. (2013) built a probabilistic low-dimensional shape embedding via a Gaussian

Process Latent Variable Model (GP-LVM) to represent various object shapes within a category. However, it only works for object categories with small shape variance, such as cars. Furthermore, it can only handle a single object in the scene due to the absence of a data association module. That is to say, associating 2D object detections that belong to the same object instance across different frames in a video.

However, with the development of Chapter 4, we believe it is the right time to revisit shape prior based object-centric mapping. What we proposed in Chapter 4 is a method for single-view object reconstruction combining deep learning and silhouette consistency. As shown in Fig. 4.1, we built a probabilistic low-dimensional shape embedding using GMM and AutoEncoder. Object shape represented by a shape code in this embedding and pose are optimised by maximising the shape likelihood in the embedding and silhouette consistency. We showed in experiments that our approach generalises well to real images. Furthermore, framing reconstruction as an iterative optimisation problem offers the potential of incorporating new observations. To extend Chapter 4 to an object-centric mapping system, the first question to be addressed is data association, which we propose to solve by casting it as a line segment clustering problem. Moreover, unlike the single-view setup, where we can only apply silhouette consistency, we have opportunities to leverage more geometry constraints, such as photometric consistency, in a video-based setup.

Choosing the best shape representation remains an open problem in 3D object reconstruction. Signed Distance Functions (SDF) have emerged as a powerful representation for learning-based reconstruction (Park et al., 2019a; Mescheder et al., 2019a), but they are not as compact or efficient as point clouds. We believe that the best object representation should be subject to task requirements. Close interactions with objects might require a high-fidelity object representation like SDF, but the sparse but efficient point cloud representation is sufficient for short-term tracking and localisation. Therefore, we propose a novel joint embedding for FroDO, in which shape codes can be decoded to both a sparse point cloud and a dense SDF. An example of a single shape code decoded to a sparse point cloud and DeepSDF mesh is shown in Fig. 5.4.

As Fig. 5.2 illustrates, FroDO takes as input an image sequence with each image's intrinsic and extrinsic parameters and proceeds in four distinct steps: *2D detection*, *data association*, *single-view shape code inference*, and *multi-view shape code optimization*. Firstly, per-frame 2D bounding box detections are inferred using an off-the-shelf method (He et al., 2017). Secondly, bounding boxes of multiple frames are associated using the line segment clustering algorithm, and then we fit a 3D ellipsoid using the associated 2D detections. Next, a 64D code is predicted for each detection of an object instance, using the shape encoder network. Per-image shape codes of the same instance are fused into a single code. Finally, each object's shape code and pose are further refined by minimising loss functions based on geometric, photometric and silhouette cues using our joint embedding as a shape prior. Our system's final outputs are dense object meshes placed in the correct position and orientation in the scene.

The contributions of FroDO are as follows:

- We present an object-centric mapping framework — FroDO. It takes as input RGB sequences of real-world multi-object scenes and infers an object-centric map of the environment, leveraging 2D deep learning detection and segmentation, learning-based 3D reconstruction and multi-view optimisation with shape prior.
- We introduce a novel deep joint shape embedding that allows simultaneous decoding to sparse point cloud and continuous SDF representation.
- We propose a coarse-to-fine multi-view optimisation approach that combines photometric and silhouette consistency costs with our deep shape prior.
- FroDO outperforms the state-of-the-art 3D reconstruction methods on real-world datasets — Pix3D (Sun et al., 2018) for a single-object single-view setup and Redwood-OS (Choi et al., 2016) for a single-object multi-view setup. We demonstrate multi-class and multi-object reconstructions on challenging sequences from the ScanNet dataset (Dai et al., 2017a).

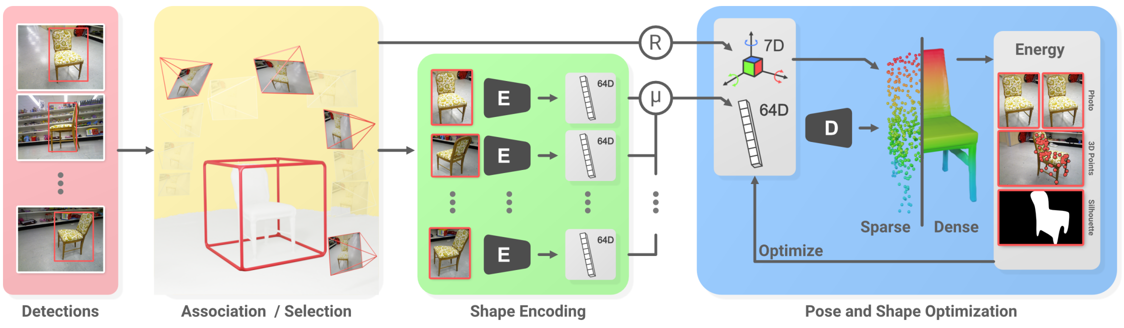


**Figure 5.2:** Given an RGB image sequence along with each image's extrinsic and intrinsic parameters, FroDO detects objects and infers their shape code and per-frame poses in a coarse-to-fine manner. We demonstrate FroDO on challenging sequences from real-world datasets that contain a single object (Redwood-OS) or multiple objects (ScanNet).

## 5.2 Related Work

At its core, our proposed system infers dense object shape reconstructions from RGB frames, so it relates to multiple areas in 3D scene reconstruction and understanding.
**Single-view learning-based shape prediction** In recent years, 3D object shape and pose estimation from images has moved from being purely geometric towards learning techniques, which typically depend on synthetic rendering of ShapeNet (Chang et al., 2015) or realistic 2d-3d datasets like Pix3d (Sun et al., 2018). These approaches can be categorised based on the shape representation, for example, occupancy grids (Choy et al., 2016b; Wu et al., 2016b), point clouds (Fan et al., 2017a), meshes (N. Wang et al., 2018), or implicit functions (Mescheder et al., 2019b). Gkioxari et al. (2019) jointly trained detection and reconstruction by augmenting Mask RCNN with an extra head that outputs volume and mesh. Our single-view reconstruction network leverages a novel joint embedding that simultaneously outputs point cloud and SDF (Fig. 5.3). Our quantitative evaluation shows that our approach provides a better single view reconstruction than competing methods.
**Multi-view category-specific shape estimation** Structure-from-Motion (SfM) and Simultaneous Localisation and Mapping (SLAM) are useful to reconstruct 3D structure from image collections or videos. However, traditional methods are prone to failure when there is a large gap between viewpoints, generally have difficulty with filling featureless areas, and cannot reconstruct occluded surfaces. Deep

learning approaches like 3D-R2N2 (Choy et al., 2016b), LSM (Kar et al., 2017), and Pix2Vox (Xie et al., 2019) have been proposed for 3D object shape reconstruction. These can infer object shape from either single or multiple observations using RNN or voxel-based fusion. However, these pure deep learning fusion techniques trained on synthetic images have difficulty with generalisation to real images, and data association is assumed known.

**3D reconstruction with shape priors** These methods are the most closely related to our approach since they also take as input RGB videos and optimise object shape and pose using 3D or image-based reprojection losses such as photometric and/or silhouette terms while assuming, often category-specific, learnt compact latent shape spaces. Some examples of the low dimensional latent spaces used are PCA (R. Wang et al., 2019a), GPLVM (Victor Adrian Prisacariu et al., 2012; Prisacariu et al., 2011; Dame et al., 2013) or a learnt neural network (Lin et al., 2019a) and Chapter 4.

In a similar spirit to the prior art, we optimise a shape code for each object, using both 2D and 3D alignment losses, but we propose a new shape embedding that jointly encodes point cloud and DeepSDF representations and show that our coarse-to-fine optimisation leads to more accurate results. These latent codes have also been used to infer the overall shapes of the entire scenes (Bloesch et al., 2018b; Sitzmann et al., 2019) without lifting the representation to the level of objects. Concurrent work (S. Liu et al., 2019) proposes to optimise DeepSDF embeddings via sphere tracing, closely related to FroDO's dense optimisation stage.

**Object-centric SLAM** Although our system is not sequential or real-time, it shares common ground with recent object-centric SLAM methods. Visual SLAM has recently evolved from purely geometric mapping (point, surface or volumetric based) to object-level representations which encode the scene as a collection of reconstructed object instances. SLAM++ (Salas-Moreno et al., 2013a) demonstrated one of the first RGB-D object-centric mapping systems where a set of previously known object instances were detected and mapped using an object pose graph. In contrast, others have focused on online object discovery and modeling (Choudhary et al.,

2014) to deal with unknown object instances, dropping the need for known models and pre-trained detectors. Recent RGB-D object-centric SLAM methods leverage the power of state-of-the-art 2D instance semantic segmentation masks (He et al., 2017) to obtain object-level scene graphs and per-object reconstructions (McCormac et al., 2018b) even in the case of dynamic scenes (Runz et al., 2018; Xu et al., 2019). Object-centric SLAM has also been extended to the case of monocular RGB-only (Nicholson et al., 2018; Parkhiya et al., 2018; Hosseinzadeh et al., 2019; Gálvez-López et al., 2015) or visual inertial inputs (Fei et al., 2018). Hosseinzadeh et al. (2019) and Nicholson et al. (2018) fit per-object 3D quadric surfaces while CubeSLAM (S. Yang et al., 2019b) proposes a multi-step object reconstruction pipeline where initial cuboid proposals, generated from single view detections, are further refined through a multi-view bundle-adjustment.

## 5.3 Method

FroDO infers an object-centric map of a scene, in a coarse-to-fine manner, given an RGB image. We assume camera poses and a sparse point cloud have been estimated using standard SLAM or SfM methods such as ORB-SLAM (Mur-Artal et al., 2015). We represent the object-centric map as a set of object poses $\{T^k_{wo}\}$ with associated 3D bounding boxes $\{bb^k_3\}$ and shape codes $\{z^k\}$. $\mathbf{T}_{ba}$ denotes a transformation from coordinate system $a$ to $b$.

The steps in our pipeline are illustrated in Fig. 5.2. First, objects are detected in input images using any off-the-shelf detector (He et al., 2017), correspondences are established between detections of the same object instance in different images (Sec. 5.3.2). Second, associated 2D detections are used to localise an object in 3D via a quadric fitting procedure. The 3D ellipsoid then enables occlusion reasoning for view selection (Sec. 5.3.3). Third, a shape code is predicted for each selected detection of the same object, using a Convolutional Neural Network (Sec. 5.3.4). Fourth, codes are fused into a unique object shape code (Sec. 5.3.5). Finally, object poses and shape codes are incrementally refined by minimising energy terms

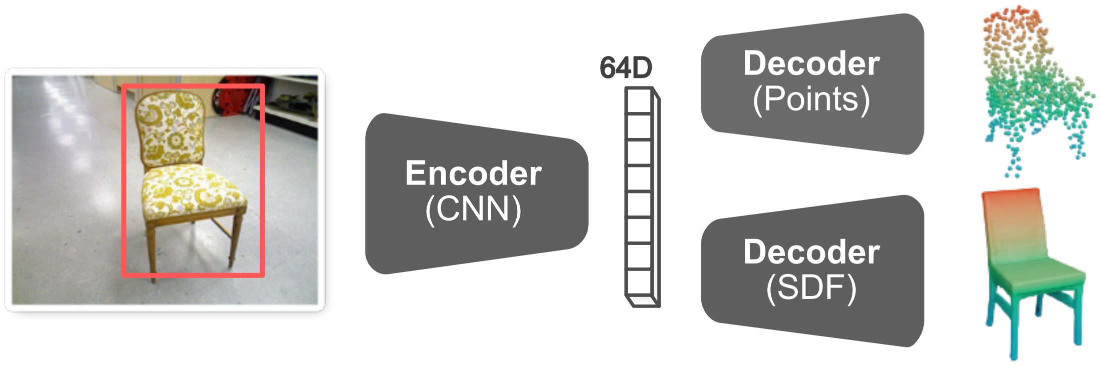


**Figure 5.3:** Our joint shape embedding leverages the advantages of sparse point-based (efficiency) and dense surface (expressiveness) object shape representations.

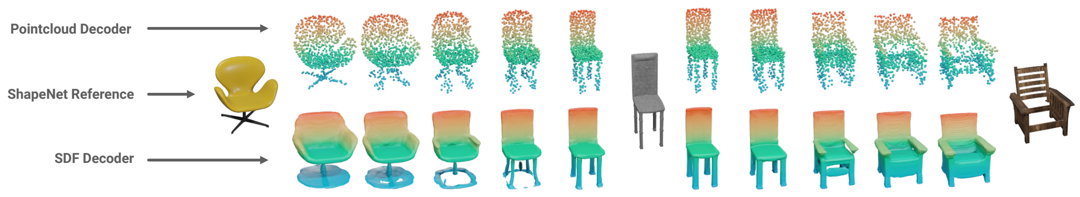


**Figure 5.4:** Joint latent shape space interpolation between 3 ShapeNet instances with ground-truth codes. Point cloud and SDF decodings of intermediate codes are coherent.

based on geometric and multi-view photometric consistency cues using our joint shape embedding as a prior (Sec. 5.3.5).

### 5.3.1 Joint Shape Code Embedding

We propose a joint shape embedding to represent and instantiate complete object shapes in a compact way. This embedding is different from the one in Chapter 4 from several aspects. Firstly, this joint embedding can decode to different object representations, and we strongly believe that this would be compelling to future research when the best shape representation is task-specific. An immediate advantage of this joint embedding is that it outperforms an embedding of a single representation in shape reconstruction shown empirically in the experiments section. We believe this is loss functions for different representation are complementary. Secondly, although it would be trivial to incorporate the Gaussian Mixture Model prior, we do not apply it to this joint embedding. Unlike the single-view setup in Chapter 4, initial tests indicate that object shapes are less likely to degenerate in the multi-view setup that we are interested in this chapter.

We parameterise an object shape as a latent code $\mathbf{z} \in \mathbb{R}^{64}$, which can be jointly decoded by two generative models $\mathbf{X} = G_s(\mathbf{z})$ and $\phi = G_d(\mathbf{z})$ into an explicit sparse 3D point cloud $\mathbf{X}$ and an implicit signed distance function $\phi$. While the point cloud decoder generates 2048 samples on the object surface, the SDF decoder represents the surface densely via its zero-level set. The decoders are trained simultaneously using a supervised reconstruction loss against ground-truth shapes on both representations:

$$L = \lambda_1 D_{\text{C}}(G_s(\mathbf{z}), \mathbf{X}_{gt}) + \lambda_2 L_\phi + \frac{1}{\sigma^2}\|\mathbf{z}\|^2, \tag{5.1}$$

where $D_{\text{C}}$ and $L_\phi$ are the reconstruction loss for point cloud and DeepSDF representation respectively, and computed as below:

$$L_\phi = |clamp(G_d(\mathbf{z}), \delta) - clamp(d_{gt}, \delta)|, \tag{5.2}$$

$$\begin{aligned} D_{\text{C}}(\mathbf{A}, \mathbf{B}) =& \frac{1}{|\mathbf{A}|} \sum_{\mathbf{x} \in \mathbf{A}} \min_{\mathbf{y} \in \mathbf{B}} \|\mathbf{x} - \mathbf{y}\|_2^2 \\ +& \frac{1}{|\mathbf{B}|} \sum_{\mathbf{y} \in \mathbf{B}} \min_{\mathbf{x} \in \mathbf{A}} \|\mathbf{x} - \mathbf{y}\|_2^2 \end{aligned} \tag{5.3}$$

We use 3D models from the CAD model repository ShapeNet (Chang et al., 2015) as ground-truth shapes. The original DeepSDF architecture (Park et al., 2019b) is employed for the SDF decoder, and a variant of PSGN (Fan et al., 2017a) is used as the point cloud decoder. Its architecture is described in detail in section 5.3.6. The idea of simultaneously decoding to both point cloud and mesh from the joint embedding is illustrated in Fig. 5.4.

While inspired by Muralikrishnan et al. (2019) to use multiple shape representations, our embedding offers two advantages. Firstly, the same latent code is used by both decoders, which avoids the need for a latent code consistency loss (Muralikrishnan et al., 2019). Secondly, by employing an AutoDecoder (Park et al., 2019a) as opposed to AutoEncoder, training a shape encoder for each representation is not required.

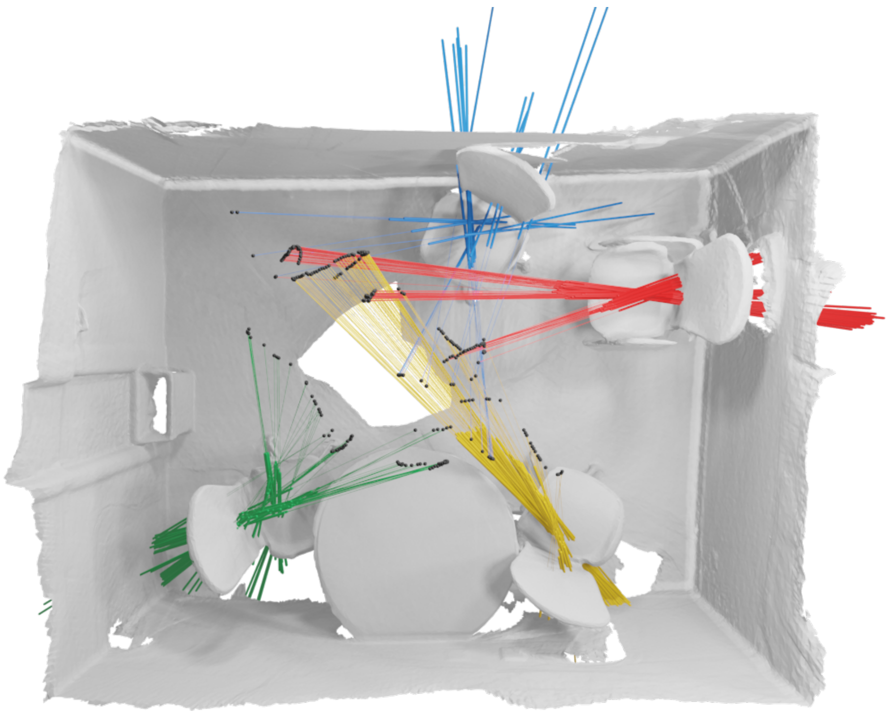


**Figure 5.5:** Data association: 3D line segment clustering to predict 2D detected bounding box correspondences. Colors denote instance IDs.

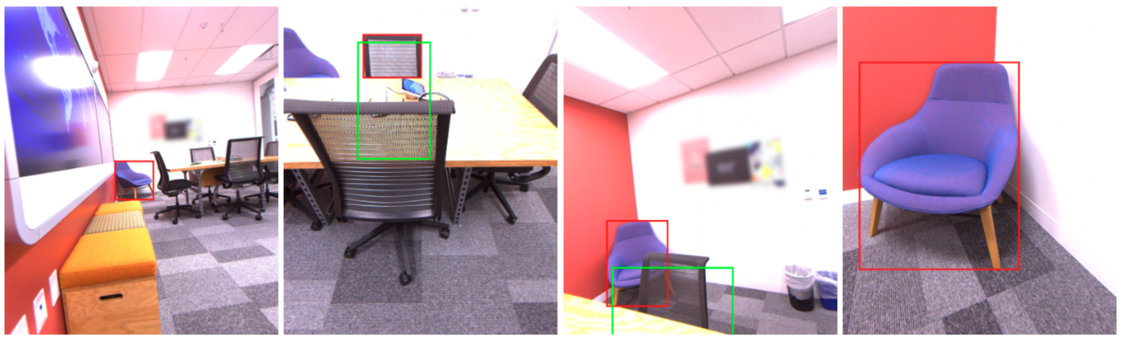


**Figure 5.6:** Examples of 2D detections filtering based on the projection of 3D bounding box. The red and green boxes are 2D detections and projections of 3D bounding boxes respectively. From left to right: reject due to size, reject due to occlusion, reject due to overlap, accept.

### 5.3.2 Object Detection and Data Association

We use a standard instance segmentation network (He et al., 2017) to detect object bounding boxes $bb_i^2$ and object masks $\boldsymbol{M}$ in the input RGB video. We predict correspondences between multiple 2D detections of each 3D object instance to enable multi-view fusion and data aggregation for object shape inference. Since the 3D ray through the centre of a 2D bounding box points in the object centre's direction, the set of rays from all corresponding detections should approximately intersect. Knowledge of the object class sets reasonable bounds on the object scale to further restrict the expected object centre location in 3D to a line segment as indicated by the thicker line segments in Fig. 5.5.

Object instance data association can then be cast as a clustering problem, in which the goal is to identify an unknown number of line segment sets that approximately intersect in a single point in 3D. We adopt an efficient iterative non-parametric clustering approach similar to Dirichlet Process (DP)-means (Kulis et al., 2011) where the observations are line segments, and the cluster centres are 3D points.

DP-mean clustering is similar to K-mean clustering as it also runs in an Expectation-maximisation manner. A key difference is that the total number of cluster is unknown at first. A new cluster is generated when the distance between a line segment and any existing clusters is larger than a cluster penalty threshold, which we set to 0.4 meters. Our clustering algorithm runs as follows: we calculate the distance between this new line segment and existing clusters for each line segment. If the distance is larger than the cluster penalty threshold, this observation becomes a new cluster, and the cluster centre is initialised at the midpoint of the line segment. Algorithm 1 details each step.

### 5.3.3 Quadric Fitting and Detection Filtering

After clustering, each object instance $k$ is associated with a set of 2D image detections $\{I_k\}$. An oriented 3D bounding box $bb_k^3$ is computed from the associated bounding box detections by fitting a dual quadric whose projections are tangent to the 2D bounding boxes as done in Nicholson et al. (2018).

Because the single-view shape inference network (Section 5.3.4) and the following optimisation (Section 5.3.5) assume a complete view of an object (*i.e.* not truncated by the image boundary, or not occluded by another object), we use the estimated ellipsoid to select "good" 2D bounding boxes for the single-view shape inference network by employing a pruning scheme. We use three criteria to reject frames based on (1) bounding box size, (2) occlusions and (3) overlap with other objects.

**Algorithm 1:** DP-mean clustering for 3D line segments

```
Input: $\boldsymbol{r_1}, \boldsymbol{r_2}, ..., \boldsymbol{r_n}$ : $n$ rays
       $\lambda$ :  cluster penalty threshold
Output: $c_1, c_2, ..., c_m$ : $m$ object clusters
1. init. $\boldsymbol{\mu_1} = mid(\boldsymbol{r_1})$, $mid()$ is to compute the middle point of a ray;
2. while not converged do
    for each $\boldsymbol{r_i}$ do
        $d_{ij} = \min_{\mu_1 \ldots k} dist(\boldsymbol{\mu_k}, \boldsymbol{r_i})$;
        if $d_{ij} \geq \lambda$ then
            set k = k + 1;
            $\boldsymbol{\mu_k} = mid(\boldsymbol{r_i})$;
        else
            $z_i = j$;
        end
    end
    $k$ clusters are generated, where $c_k = \{\boldsymbol{r_i} | z_i = k\}$;
    for each $c_i$ do
        $\boldsymbol{\mu_i} = lstsq(c_k)$;
    end
end
```

(1) is implemented via a threshold on the bounding box's width and height, and (2,3) are implemented using a threshold on the intersection over union (IoU) of the bounding boxes. Figure 5.6 illustrates these strategies. The filtered set of image detections $\{I'_k\}$ is used in all the following steps.

### 5.3.4 Single-view Shape Code Inference

As illustrated in Fig. 5.2, a 64D object shape code is predicted for each filtered detection. We train a new encoder network that takes as input a single image patch cropped by the 2D bounding box and regresses to its associated shape code $\mathbf{z_i} \in \mathbb{R}^{64}$ in the joint latent space described in Sec. 5.3.1.

The network is trained in a fully supervised way. However, due to the lack of 3D shape annotations for real-world image datasets, we train the image encoder using synthetic ShapeNet (Chang et al., 2015) renderings. Specifically, we generate training data by rendering ShapeNet CAD models with random viewpoints, materials, environment mapping, and background. We also perturb bounding boxes of rendered objects and feed perturbed crops to the encoder during training. We choose a standard ResNet architecture, modifying its output to the size of the embedding vector. During training, we minimise the Huber loss between predicted and target embeddings, which we know for all CAD models. More training details are covered in Sec. 5.3.6.

### 5.3.5 Multi-view Optimization with Shape Priors

For each object instance $k$ we fuse all single-view shape codes $\{\mathbf{z_i} | i \in I'_k\}$ into a unique code $z^0_k$. We propose two fusion approaches and evaluate them in Table 5.5: *(i)* Average – we average shape codes to form a mean code $z^{\text{mean}}_k$; *(ii)* Majority voting – We find the 4 nearest neighbors of each predicted code $\mathbf{z_i}$ among the models in the training set. The most frequent of these is chosen as $z^{\text{vote}}_k$. Unlike the average code, $z^{\text{vote}}_k$ guarantees valid shapes from the object database.

For each object instance $k$, all images with non-occluded detections are used as input to an energy optimisation approach to estimate object pose $T^k_{wo}$ and shape code $z_k$ in two steps. First, we optimise the energy over a sparse set of surface points, using the point decoder $G_s(\mathbf{z})$ as a shape prior. This step is fast and efficient due to the sparse nature of the representation and the lightweight of the point cloud decoder. Second, we further refine the pose and shape minimising the same energy over dense surface points, using the DeepSDF decoder $G_d(\mathbf{z})$ as the prior. This slower process is more accurate than the sparse point cloud since the loss is evaluated overall surface points and not sparse samples.

**Energy.** Our energy is a combination of losses on the 2D silhouette $E_s$, photometric consistency $E_p$ and geometry $E_g$ with a shape code regularizer $E_r$:

$$E = \lambda_s \cdot E_{\text{s}} + \lambda_p \cdot E_{\text{p}} + \lambda_g \cdot E_{\text{g}} + \lambda_r \cdot E_{\text{r}}\,, \tag{5.4}$$

where $\lambda_{s,p,g,r}$ weigh the contributions of individual terms. The regularisation term $E_{\text{r}} = \frac{1}{\sigma^2}||\mathbf{z}||^2_2$ encourages shape codes to take values in valid regions of the embedding, analogously to the regularizer in Eq. 5.1. Note that the same energy terms are used for sparse and dense optimisation – the main differences being the number of points over which the loss is evaluated, and the decoder $\mathcal{G}(\mathbf{z})$ used as a shape prior.

**Initialization.** The 64D shape code is initialised to the fused shape code (Sec. 5.3.5), while the pose $\mathbf{T}_{wo}$ is initialised from the 3D bounding box $bb_k^3$ (Sec. 5.3.2): translation is set to the vector joining the origin of the world coordinate frame with the 3D bounding box centroid, scale to the 3D bounding box height and rotation is initialised using exhaustive search for the best rotation about the gravity direction – under the assumption that objects are supported by a ground-plane perpendicular to gravity.

**Sparse Optimization.** Throughout the sparse optimisation, the energy $E$ is defined over the sparse set of 2048 surface points $\mathbf{X}$, decoded with the point-based decoder $G_s(\mathbf{z})$. The energy $E$ is minimised using the Adam optimizer (Kingma et al., 2014) with autodiff. We now define the energy terms.

• The photometric loss $E_\text{p}$ encourages the colour of 3D points to be consistent across views. In the sparse case, we evaluate $E_\text{p}$ by projecting points in $\mathbf{X}$ to $N$ nearby frames via known camera poses $\mathbf{T}_{cw}$ and comparing colors in reference $\mathcal{I}^R$ and source $\mathcal{I}_i^S$ images under a Huber norm $\|.\|_h$:

$$\begin{aligned} E_\text{p}(\mathbf{X}, \mathcal{I}^R, \mathcal{I}_1^S, ..., \mathcal{I}_N^S) &= \frac{1}{N \cdot |\mathbf{X}|} \sum_{i=1}^{N} \sum_{\mathbf{x} \in \mathbf{X}} \|r(\mathcal{I}^R, \mathcal{I}_i^S)\|_h \\ r(\mathcal{I}^R, \mathcal{I}^S) &= \mathcal{I}^R(\pi(\mathbf{T}_{cw}^R \mathbf{x})) - \mathcal{I}^S(\pi(\mathbf{T}_{cw}^S \mathbf{x})) \end{aligned} \tag{5.5}$$

where $\pi(\mathbf{x})$ projects 3D point $\mathbf{x}$ into the image.

• The silhouette loss $E_\text{s}$ penalises discrepancies between the 2D silhouette obtained via the projection of the current 3D object shape estimate and the mask predicted by MaskRCNN (He et al., 2017). In practice, we penalize points that project outside the predicted mask using the 2D Chamfer distance:

$$E_s(\mathbf{z}_k, \mathbf{T}_{wo}^k) = D_\text{C}(\mathbf{M}, \pi(\mathbf{T}_{cw} \mathbf{T}_{wo}^k \mathcal{G}(\mathbf{z}))) \tag{5.6}$$

where $\mathbf{M}$ is the set of 2D samples on the predicted mask and $D_C$ is the symmetric Chamfer distance defined in Eq. 5.3.

• The geometric loss $E_\text{g}$ minimizes the 3D Chamfer distance between 3D SLAM (or SfM) points and points on the current object shape estimate:

$$E_{\mathrm{g}}(\mathbf{z_k}, \mathbf{T}^k_{wo}) = D_{\mathrm{C}}(\mathbf{X}_{slam}, \mathbf{T}^k_{wo}\mathcal{G}(\mathbf{z})), \tag{5.7}$$

**Dense Optimization.** The shape code and pose estimated with the sparse optimisation can be further refined with a dense optimisation over all surface points and using the DeepSDF decoder $G_d(\mathbf{z})$. Since $G_d(\mathbf{z})$ uses an implicit representation of the object surface, we compute a proxy mesh at each iteration and formulate the energy over its vertices. This strategy proved faster than sphere tracing (S. Liu et al., 2019), while achieving on-par accuracy, see Table 5.3. We now describe the dense energy terms.

• The photometric and geometric losses $E_{\mathrm{p}}, E_{\mathrm{g}}$ are equivalent to those used in the sparse optimization (see Eq. 5.5, 5.7). However, they are evaluated densely, and the photometric optimisation makes use of a Lucas-Kanade style warp.

• The silhouette loss $E_{\mathrm{s}}$ takes a different form to the sparse case. We follow traditional level set approaches, comparing the projections of object estimates with observed foreground and background probabilities $P_{f,b}$:

$$E_s = \int_\Omega H(\phi)P_f(x) + (1 - H(\phi))\,P_b(x)d\Omega, \tag{5.8}$$

where $\phi$ is a 3D or 2D shape-kernel, and $H$ a mapping to a 2D foreground probability field, resembling an object mask of the current state. Empirically, we found that 3D shape-kernels (Victor A Prisacariu et al., 2012) provide higher quality reconstructions than a 2D formulation (Victor Adrian Prisacariu et al., 2012) because more regions contribute to gradients. While $H$ is a Heaviside function in the presence of 2D level-sets, we interpret signed distance samples of the DeepSDF volume as logits and compute a per-pixel foreground probability by accumulating samples along rays, similar to (Victor Adrian Prisacariu et al., 2012):

$$H = 1 - \exp \prod_{\mathbf{x}\text{ on ray}} \left(1 - \mathrm{sig}(\zeta \cdot \phi(\mathbf{x}))\right)\,, \tag{5.9}$$

where $\zeta$ is a smoothing coefficient, and $1 - \mathrm{sig}(\zeta \cdot \phi(\mathbf{x}))$ the background probability at a sampling location $\mathbf{x}$. A step-size of $\frac{r}{50}$ is chosen, where $r$ is the depth range of the object-space unit-cube.

### 5.3.6 Implementation details

**Decoder of the joint embedding.** Following Park et al. (2019a), we use AutoDecoder (not AutoEncoder) to train our embedding and decoder network. The AutoDecoder comprises a set of embedding in $\mathcal{R}^{N\times 64}$, where $N$ is the number of training CAD models, and 64 is the embedding dimension, and a decoder network. Note that both the embedding and network weights are randomly initialised and optimised during training. The decoder is split into two independent branches for point cloud and DeepSDF respectively. The point cloud branch comprises four fully connected layers, each with 512, 1024, 2048, and $2048 \times 3$ as output dimensions. The DeepSDF branch comprises eight fully connected layers, each with 256 as output dimensions. The final SDF value is obtained with hyperbolic tangent non-linear activation. We use ReLU as a non-linear activation function following each fully connected layer except the last one. The joint AutoDecoder is trained for 2000 epochs with a batch size of 64. There are 16384 SDF samples for each shape. The learning rate for network parameters and latent vectors are $5 \cdot 10^{-3}$, and $10^{-3}$ respectively, decayed by 0.5 every 500 epochs.

**Encoder Network.** After we train the decoder network, we obtain a set of embeddings $\mathbf{z} \in \mathcal{R}^{64}$ for the corresponding CAD models. In the second stage, we train an encoder network that maps an image to the latent code. We tailor ResNet50 to output a vector of dimension 64 and initialise the network with pretrained models. We train the network for 50 epochs to minimise a Huber loss with a polynomial decaying learning rate of $10^{-3}$. The network for the embedding is trained in a way similar to Y. Li et al. (2015). However, our deep learning based shape embedding is very different from the non-parametric embedding used in Y. Li et al. (2015).

Off-the-shelf MaskRCNN (He et al., 2017) is used to predict object detections and masks. We run Orb-SLAM (Mur-Artal et al., 2015) to estimate trajectories and keypoints for experiments on Redwood-OS but use the provided camera poses and no keypoints on ScanNet.

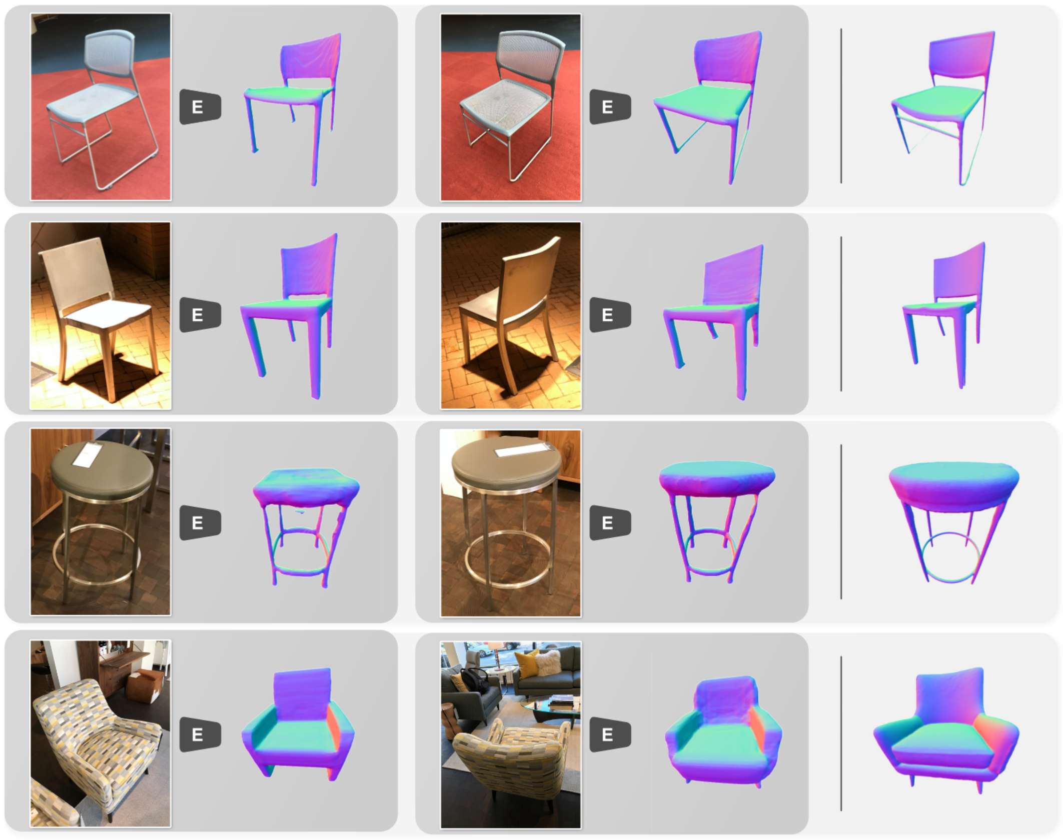


**Figure 5.7:** Examples of single view reconstruction on Pix3D dataset (Sun et al., 2018). Ground truth on the right for reference.

## 5.4 Experiments

Our focus is to evaluate the performance of FroDO on real-world datasets wherever possible. We evaluate *quantitatively* in two scenarios: *(i)* single-view, single object on Pix3D (Sun et al., 2018); and *(ii)* multi-view, single object on the Redwood-OS (Choi et al., 2016) dataset. In addition, we evaluate our full approach *qualitatively* on challenging sequences from the real-world ScanNet dataset (Dai et al., 2017a) that contain multiple object instances in different object categories.

### 5.4.1 Single-View Object Reconstruction

First, we evaluate the performance of our single-view shape code prediction network (Sec. 5.3.4) on the real-world dataset Pix3D (Sun et al., 2018). Table 5.1 shows a comparison with competing approaches on the *chair* category. The evaluation protocol described in (Sun et al., 2018) was used to compare IoU, Earth Mover Distance

| | IoU ↑ | EMD ↓ | CD ↓ |
|---|---|---|---|
| 3D-R2N2 (Choy et al., 2016b) | 0.136 | 0.211 | 0.239 |
| PSGN (Fan et al., 2017a) | N/A | 0.216 | 0.200 |
| 3D-VAE-GAN (Wu et al., 2016b) | 0.171 | 0.176 | 0.182 |
| DRC (Tulsiani et al., 2017b) | 0.265 | 0.144 | 0.160 |
| MarrNet (Wu et al., 2017b) | 0.231 | 0.136 | 0.144 |
| AtlasNet (Groueix et al., 2018) | N/A | 0.128 | 0.125 |
| Sun et al. (2018) | 0.282 | 0.118 | 0.119 |
| Ours (DeepSDF Embedding) | **0.302** | **0.112** | **0.103** |
| Ours (Joint Embedding) | **0.325** | **0.104** | **0.099** |

**Table 5.1:** Results on Pix3D (Sun et al., 2018). Our method gives the highest Intersection over Union and lowest Earth Mover's and Chamfer Distances.

| Optim. Method | Energy Terms | CD (cm.) |
|---|---|---|
| Sparse | $E_s + E_r$ | 8.97 |
| Sparse | $E_s + E_p + E_g + E_r$ | 8.59 |
| Sparse + Dense | $E_s + E_r$ | 7.41 |
| Sparse + Dense | $E_s + E_p + E_g + E_r$ | 7.38 |

**Table 5.2:** Ablation study of estimates after sparse and dense optimization stages on the Redwood-OS dataset. We compare the effect of different energy terms in Eq. (5.4).

(EMD) and Chamfer Distance (CD) errors (results of competing methods are from (Sun et al., 2018)). Our proposed encoder network outperforms related works in all metrics. Table 5.1 also shows an improvement in performance when our new joint shape embedding is used (Ours Joint Embedding) instead of DeepSDF (Park et al., 2019b) (Ours DeepSDF Embedding). Figure 5.7 shows example reconstructions.

### 5.4.2 Multi-View Single Object Reconstruction

We quantitatively evaluate our complete multi-view pipeline on the *chair* category of the real-world Redwood-OS dataset (Choi et al., 2016). We perform two experiments: an ablation study to motivate the choice of terms in the energy function (Table 5.2) and a comparison with related methods (Table 5.5). Table 5.3 includes a comparison of our dense photometric optimization with the two closest related approaches (Lin et al., 2019a; S. Liu et al., 2019) on a commonly-used synthetic dataset (Lin et al., 2019a).

| | PMO (o) | PMO (r) | DIST (r) | Ours (r) |
|---|---|---|---|---|
| Cars | 0.661 | 1.187 | 0.919 | 1.202 |
| Planes | 1.129 | 6.124 | 1.595 | 1.382 |

**Table 5.3:** Non-symmetric Chamfer distance (completion) on first 50 instances of the synthetic PMO (Lin et al., 2019a) test set. While (o) indicates the original PMO method with its own initialization, (r) indicates random initialization.

| Method | sec. /iteration | # of iteration |
|---|---|---|
| PMO Lin et al. (2019a) | 12.59 | 100 |
| Optim. Dense | 4.96 | 100 |
| Optim. Sparse | 0.07 | 200 |

**Table 5.4:** Speed comparison between PMO (Lin et al., 2019a) and our method on the same sequence of 60 frames.

**Ablation study.** Table 5.2 shows the result of an ablation study on different energy terms in our sparse and dense optimizations (Eq. 5.4). The combination of geometric and photometric cues with a regulariser on the latent space achieves the best result. We also compare the proposed fusion techniques (average and majority voting) in Table 5.5. "Average" outperforms "majority voting" slightly and it is also more efficient.

**Joint embedding gain** The joint embedding achieves better performance in single-view reconstruction than the DeepSDF-only embedding as shown in the last two rows in Table 5.1. Moreover, an important motivation for using a joint embedding is that the efficiency of a sparse optimisation can be leveraged, while exploiting richer information when subsequently applying fewer dense iterations. A comparison of optimization run-times shows that the pointcloud-based representation is approximately two orders of magnitude faster than the mesh optimization used by PMO (Lin et al., 2019a) and DeepSDF (Park et al., 2019b) as shown in Table 5.4.

**Synthetic dataset.** Table 5.3 shows a direct comparison of the performance on the synthetic PMO test set (Lin et al., 2019a) of our dense optimization when only the photometric loss $E_p$ is used in our energy (for fair comparison), with the two closest related methods: PMO (Lin et al., 2019a) and DIST (S. Liu et al., 2019). Notably, both DIST and our approach achieve comparable results to PMO from only random initialisations. When PMO is also initialised randomly, the results degrade substantially.

**Redwood-OS dataset.** Table 5.5 shows a comparison with Pix2Vox (Xie et al., 2019), a purely deep learning approach, and with PMO (Lin et al., 2019a). For reference, we also compare with COLMAP (Schönberger et al., 2016a; Schönberger et al., 2016b), a traditional SfM approach that generates geometric-only reconstructions. Since COLMAP reconstructs the full scene without segmenting objects, we only select points within the ground-truth 3D bounding box for evaluation. We report errors using: Chamfer distance (CD), accuracy (ACC (5cm)), completion (COMP (5cm)) and F1 score – all four commonly used when evaluating on Redwood-OS. Chamfer distance (CD) measures the symmetric error, while shape accuracy captures the 3D error as the distance between predicted points to their closest point in the ground truth shape and vice-versa in the case of shape completion. Both shape accuracy and completion are measured in percentage of points with an error below 5cm. Following (Lin et al., 2019a), we use an average of 35 input frames sampled from the RGB sequences, though for completeness we show results with 350 views. Fig. 5.8 shows some qualitative results and more examples can be found in Figure 5.10 at the end of this Chapter.

We outperform Xie et al. (2019) by a significant margin which could point to the lack of generalisation of purely learning-based approaches. We also outperform PMO (Lin et al., 2019a), a shape prior based optimisation approach like ours, but lacks our joint embedding and the coarse-to-fine shape optimisation. Although COLMAP achieves high accuracy regardless of the number of viewpoints, it fails to reconstruct full 3D shapes when the number of input images or the baseline of viewpoints is limited as it cannot leverage shape priors. Although, as expected, the performance of COLMAP increases drastically with the number of input images, it requires hundreds of views to perform comparably to our approach. A distinction to COLMAP is that we are reconstructing at the object level, while COLMAP's reconstructions do not represent the environment semantically.

| Method | Few observations (average 35 views) | | | | Over-complete observations (average 350 views) | | | |
|---|---|---|---|---|---|---|---|---|
| | CD (cm) | ACC (5cm) | COMP (5cm) | F1 score | CD (cm) | ACC (5cm) | COMP (5cm) | F1 score |
| COLMAP | 10.58 | **84.16** | 54.28 | 65.99 | **6.05** | **91.41** | **94.59** | **92.97** |
| Pix2Vox | 12.12 | 55.27 | 64.74 | 59.63 | 11.87 | 55.88 | 66.09 | 60.56 |
| PMO | 12.13 | 53.08 | 69.42 | 60.16 | 11.93 | 54.80 | 69.54 | 61.30 |
| **FroDO** Code Fusion (Vote) | 12.19 | 60.74 | 60.55 | 60.64 | 11.97 | 61.37 | 58.20 | 59.74 |
| **FroDO** Code Fusion (Avg.) | 10.74 | 61.31 | 72.11 | 66.27 | 10.57 | 61.06 | 72.14 | 66.14 |
| **FroDO** Optim. Sparse | 8.69 | 70.58 | 79.10 | 74.60 | 8.59 | 71.69 | 81.63 | 76.34 |
| **FroDO** Optim. Dense | **7.38** | 73.70 | **80.85** | **76.64** | 7.37 | 74.78 | 81.08 | 77.32 |

**Table 5.5:** Quantitative evaluation on 86 sequences of Redwood-OS. We compare state of the art competitors Pix2Vox (Xie et al., 2019) and PMO (Lin et al., 2019a) with the results at different stages of our multi-view pipeline (code fusion → sparse optimization → dense optimization).

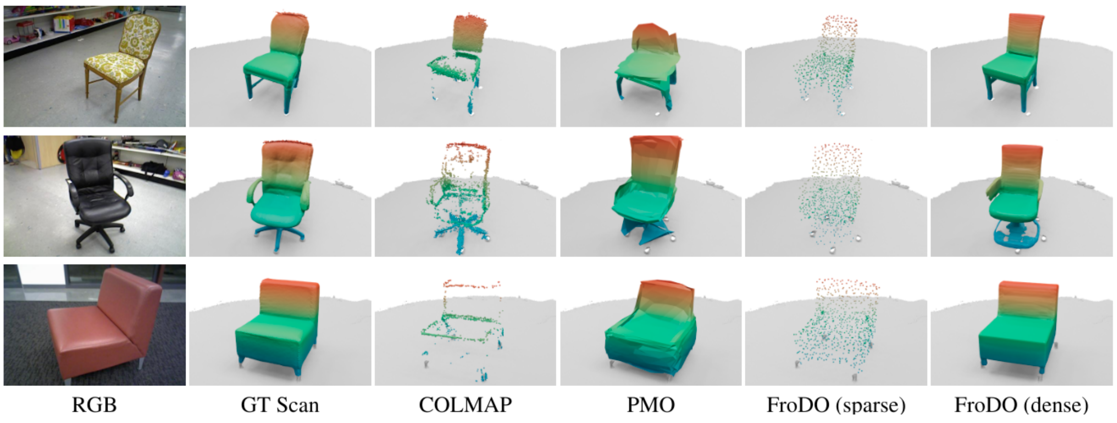


**Figure 5.8:** Example 3D reconstructions achieved with different approaches on three sample sequences from Redwood-OS. In all cases 35 input views were used.

### 5.4.3 Multi-object Reconstruction

We demonstrate qualitative results of our full approach on the ScanNet dataset (Dai et al., 2017a) on challenging real-world scenes with multiple object instances of two object categories (table and chair) in Fig. 5.1 and Fig. 5.9. The association of object detections to 3D object instances becomes an additional challenge when dealing with multi-object scenarios. Our results show that our line segment clustering approach successfully associates detected bounding boxes across frames, and our coarse-to-fine optimisation scheme provides high-quality object poses and reconstructions.

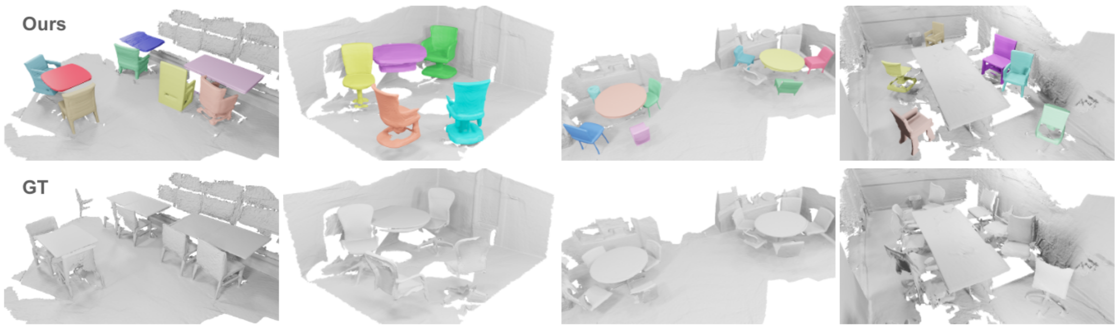


**Figure 5.9:** Qualitative results on four ScanNet RGB input sequences. We reconstruct multiple instances of the chair and table classes. While outputs are satisfactory for the first three scenes, the last one highlights failures due to heavy occlusions and partial observations. Top row: Object instances reconstructed by FroDO are shown in colour while grey shows the ground-truth background (not reconstructed by our method) for reference. Bottom row: full ground-truth scan for comparison.

## 5.5 Conclusion

In this chapter, we introduced FroDO, a novel object-centric mapping framework that takes as input monocular RGB images and their extrinsic parameters and infers the location, pose and accurate shape of objects in a scene. Key to FroDO is using a novel deep learnt shape encoding throughout the different shape estimation steps. We demonstrated FroDO on challenging sequences from real-world datasets in single-view, multi-view and multi-object settings.

Currently, FroDO is not an online system because the line segment clustering for data association takes as input object detections from an entire sequence. As shown in the rightmost column in Fig. 5.9, while the detection pruning using the 3D ellipsoid can help to remove truncated detections, objects that are occluded by other objects are difficult to be reconstructed. Furthermore, FroDO is only well suited to a static environment.

Ideally, an object-centric mapping should be able to handle object motion potentially caused by humane interactions. The next chapter addresses the limitations above by introducing an online object-centric mapping framework that jointly detects, tracks, and reconstructs objects in a scene.

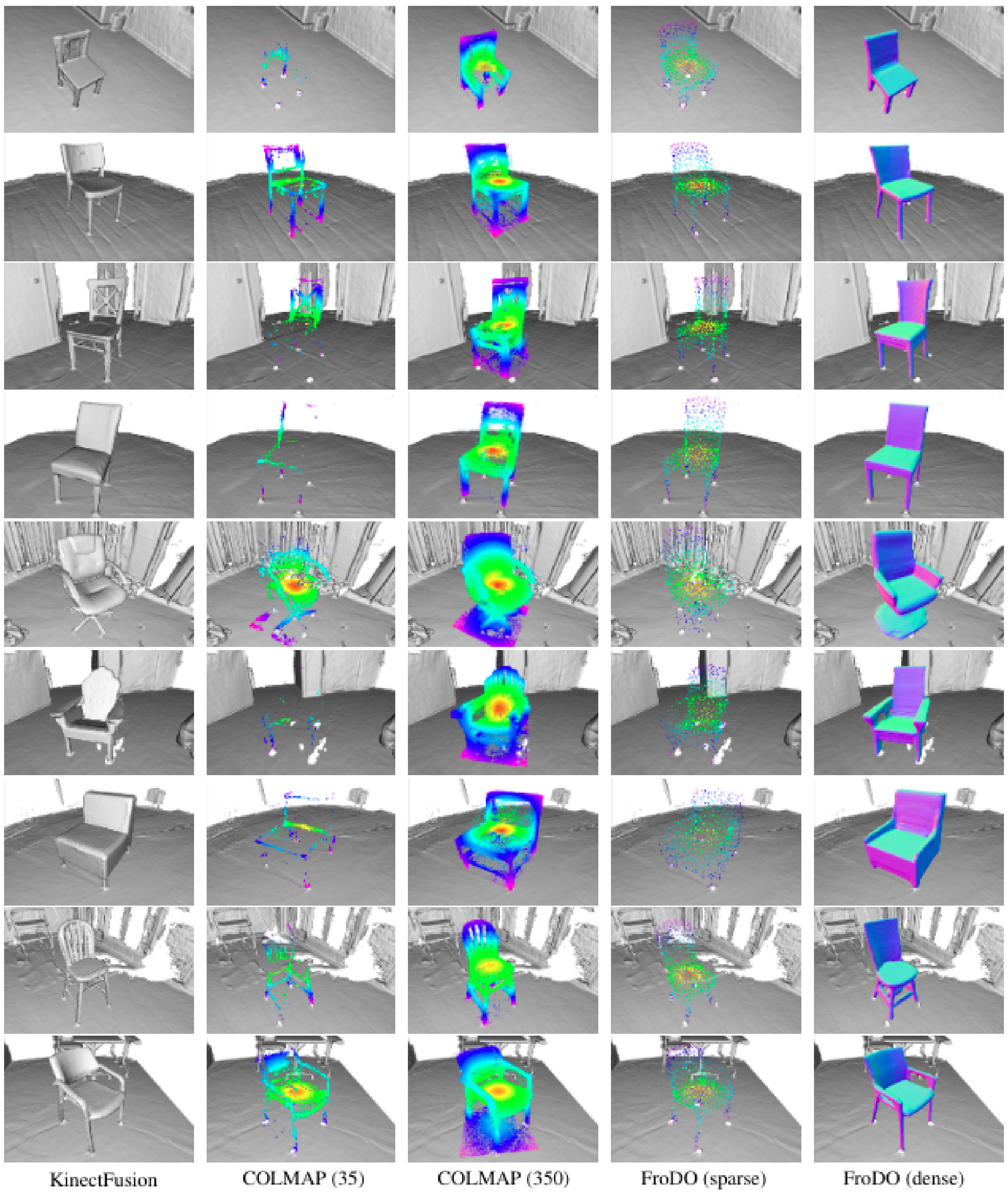


**Figure 5.10:** Comparison of different approaches to object shape reconstruction on some examples from Redwood-OS dataset.

# 6

# MO-LTR: Multiple Object Localization, Tracking, and Reconstruction

## Contents

*In Chapter 5 we introduced FroDO, a framework to reconstruct a scene at the level of objects given a monocular video in a coarse-to-fine manner. While clustering 2D detections using unprojected 3D line segments can localize objects in 3D, it is an*

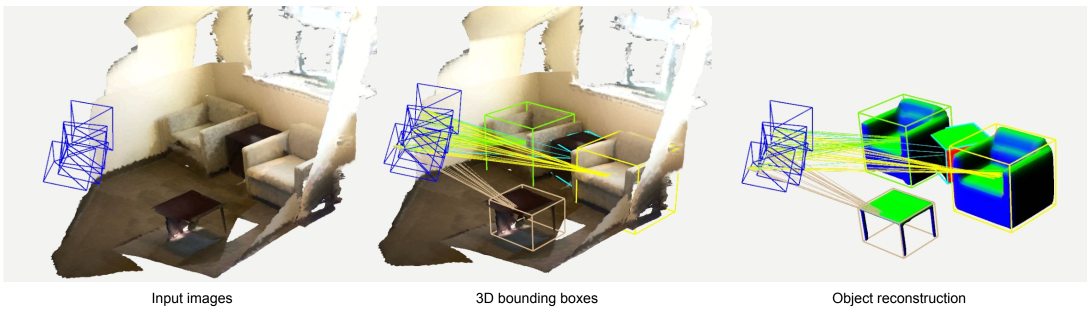


**Figure 6.1:** A subset of input RGB images are represented by blue frustums at the left image. Detection and tracking are shown on the middle image, in which the colored rays indicate the associated detections to different object instances. The object-centric reconstruction from MO-LTR is shown on the right image. Note that the scene mesh is **not** used by MO-LTR, but shown for visualization purpose only.

*off-line operation and assumes a static environment. We tackle these limitations in this chapter by proposing MO-LTR— Multiple Object Localization, Tracking, and Reconstruction given a monocular video. MO-LTR, shifting away from 2D detections and line segment clustering based data association, adopts monocular 3D detection to localize objects in the scene from a single view. Although single-view 3D detections are noisy, they provide a strong cue for our on-line data association module in 3D. Objects are localized using the aggregated 3D detections, and object motion is tracked by a multiple model Bayesian filter. Like FroDO, object shape is represented by a shape code in a learned shape embedding, which is progressively refined by the associated detections. MO-LTR not only addresses the fundamental limitations of FroDO but also shows superior performance over FroDO in static environments. This work is accepted in IEEE Robotics and Automation Letters.*

## 6.1 Introduction

In this chapter, we are concerned with the problem of *online* detection, tracking, and reconstruction of *potentially dynamic* objects from *monocular* videos. Although elements of this problem have been tackled extensively, few works have adequately addressed all three – online, dynamic, and monocular – simultaneously. Our system MO-LTR, judiciously combines a number of traditional and deep learning techniques to address the problem. Given a new RGB frame, a monocular 3D detector is used to localise objects presented in this view, and each detected object is mapped to a learned shape embedding by our shape encoder. Meanwhile, the state, which includes kinematics and motion status (*i.e.* dynamic/static), of each existing object in the map is tracked via a multiple model Bayesian filter. Matchable objects (more details to be covered in 6.3.4), selected based on the motion state, are used to associate to the newly detected objects, after which, associated detections are merged to the map, the filters are updated, and object shapes are incrementally refined by shape code fusion.

While approaches like Fusion++ (McCormac et al., 2018a) and FroDO (Runz et al., 2020) assume a static environment, we argue that it is necessary to handle dynamic objects, because an agent (either a robot or a user in an AR/VR headset) is very likely to move objects when it interacts with the environment. Given the diverse motion patterns (*e.g.* static, dynamic, and switch between these two), we employ a multiple model Bayesian filter to track object kinematics and motion status.

Modelling the motion status explicitly gives us further advantage to manage the termination of object trajectory during tracking. In contrast to common practice in most Multiple Object Tracking (MOT) approaches (Weng et al., 2020; Shenoi et al., 2020), which use a predefined fixed time threshold to terminate object trajectories if not observed, we maintain static objects in the map even if they have not been observed for a long time. The intuition behind is that static objects would persist whether they are observed or not while dynamic objects are more likely to move out from the environment.

MO-LTR also stands out from previous object-centric mapping approaches in terms of input modality. Unlike most existing systems that require a depth sensor (McCormac et al., 2018a; Xu et al., 2019; Runz et al., 2018), MO-LTR takes just monocular RGB images as input. Although works like MOTSFusion (Luiten et al., 2020) also tried to replace a depth sensor with a monocular depth estimation network, the disadvantages of our approach are twofold: 1) depth prediction on object boundaries is extremely noisy, and fusion using the noisy prediction accumulates error whereas the object surface of our reconstruction is smooth and clean by leveraging shape prior knowledge; 2) The depth map based reconstruction is not as complete as our shape prior based reconstruction because fusing multiple depth maps can only recover the visible surfaces. We show quantitatively in experiments that the object reconstruction produced by the depth network is less accurate than our shape prior based reconstruction.

To summarise, the main contributions of this chapter are:

- We present MO-LTR, a unified framework for object-centric mapping, which can localise, track, and reconstruct multiple objects in an online fashion given monocular RGB videos.
- We demonstrate that the combination of monocular 3D detection, multiple model Bayesian filter and deep learned shape prior leads to robust multiple object tracking and reconstruction.
- We evaluate the proposed system extensively showing more accurate reconstruction and robust tracking than previous approaches on both indoor and outdoor datasets.

## 6.2 Related Work

In recent years, we have seen impressive progress in semantic reconstruction. Early works (Stückler et al., 2014; Hermans et al., 2014; Pham et al., 2015) use graphical models to assign semantic labels to a geometric reconstruction. SemanticFusion (McCormac et al., 2017) employs a deep network to predict pixel-wise semantic labels given RGB frames, which are then fused into a semantic mapping by leveraging the geometric reconstruction from an RGBD SLAM. Although these approaches enrich the geometric reconstruction by attaching semantic labels, they are not object-centric as they cannot separate objects of the same class.

Pioneering works on the object-centric mapping are based on template matching and thus is limited to a set of *a-priori* known objects. Gálvez-López et al. (2016)

propose a monocular-based SLAM that matches detections against objects in a database using bags of binary words. SLAM++ (Salas-Moreno et al., 2013b), an RGBD SLAM, uses point pair features to detect and align CAD models into the map. To remove the object template database, a number of approaches turn to deformable template (Bao et al., 2013a; Parkhiya et al., 2018).

Learning a shape prior that takes advantage of object shape regularity is another research trend for object shape reconstruction. Intra-class full 3D shape variance is captured in a learnt latent space. Object shape is optimised in this latent space given image or depth evidence, and thus full 3D objects can be reconstructed even if only partial observations are available. Shape latent space is often learnt via (Kernal) PCA (Dambreville et al., 2008b; R. Wang et al., 2019b) or GP-LVM (Prisacariu et al., 2012; Dame et al., 2013).

Motivated by the success of deep learning in scene recognition, deep networks are used as function approximators that map an image (Fan et al., 2017b; Tulsiani et al., 2017a) or images (Choy et al., 2016a; Xie et al., 2019) to a 3D object shape. Instead of a direct mapping, another line of works (Wu et al., 2016a; R. Zhu et al., 2018; Lin et al., 2019b) including ours presented in Chapter 4 apply deep networks as powerful dimension compression tools to learn a shape embedding. Object shapes can be optimised in the embedding given visual observations. However, these methods are often constrained to single-object scenes where all observations can be assigned to the same object. When there are multiple objects in a scene (*e.g.*, a dining room with a table surrounded by multiple chairs), data association that assigns observations to different objects is essential to apply those methods.

Although Chapter 5 and this chapter share common ground in using shape prior for object reconstruction, the line segment clustering algorithm in Chapter 5 assumes a static environment whereas we can work with both dynamic and static objects in this chapter. Additionally, MO-LTR is an on-line approach, whereas FroDO is off-line.

There are several RGBD based approaches that leverage modern instance segmentation networks to fuse depth maps of each object instance separately to achieve object-centric mapping. Assuming a static environment, Fusion++ (McCormac et al., 2018a) generates a TSDF reconstruction for each object detected given a RGBD image sequence. MID-Fusion (Xu et al., 2019) takes a step forward by tracking each object's pose to handle dynamic objects. Co-Fusion (Rünz et al., 2017) and MaskFusion (Runz et al., 2018) using surfels to represent object shapes can also handle dynamic objects by tracking the object motion using Iterative Closest Point (ICP). Recently, Sucar et al. (2020), following shape prior reconstruction, propose to optimise object shapes in a learnt embedding given multiple depth observations. In contrast to methods aforementioned focusing on the indoor environment, DynSLAM (Bârsan et al., 2018) is a stereo-based system for dynamic object tracking reconstruction in the outdoor environment. The distinctive advantage of MO-LTR over these methods is our simple sensory input being a monocular RGB camera. While Luiten et al. (2020) explores the possibility to replace the depth sensor with a monocular depth estimation network, fusing multiple noisy depth prediction is error-prone. An additional benefit of our reconstruction using shape prior is the completeness of the reconstruction.

## 6.3 Method

We first give an overview of the system, and then describe the details of the system components in the subsequent sections, as indicated below.

### 6.3.1 System Overview

Given a new RGB frame, MO-LTR first employs a monocular 3D detector to predict a 9-DoF object pose, object class label, and 2D bounding box (6.3.2). For each detected object, an image patch cropped by the 2D bounding box of an object is mapped to a shape code in a learnt shape embedding (6.3.3). State (*i.e.* pose and motion status) of each existing object in the map is modelled by a multiple model Bayesian filter. Prior to data association, we use the filter to predict object location and decide whether an object is matchable using the predicted motion status. The new detections are associated with the matchable objects based on

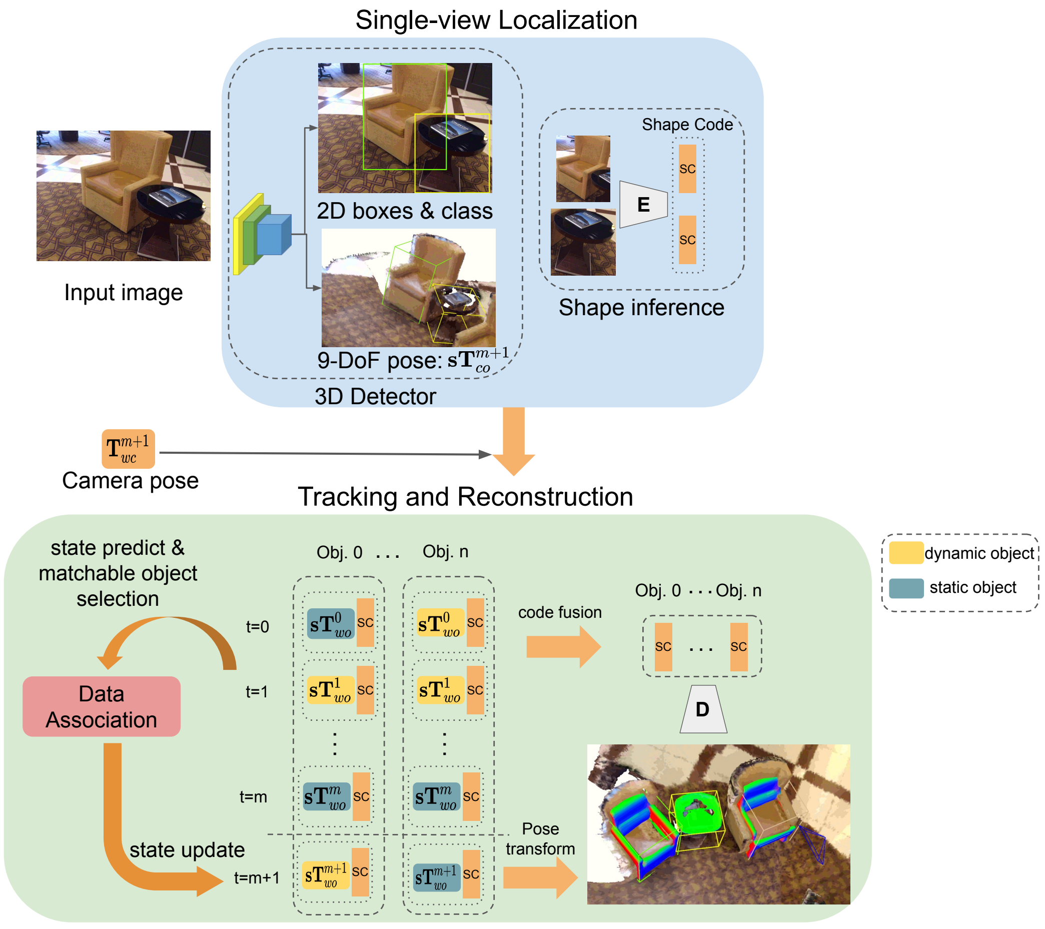


**Figure 6.2:** MO-LTR pipeline. Given a new RGB frame, we predict 6-DoF object pose with respect to the camera $\mathbf{T}_{co}^{m+1}$ and object scale $\mathbf{s}$ (*i.e.* 3D dimension) for each object of interest, which is visualised as an oriented 3D bounding box. We also predict object class and 2D bounding box for each object. An image patch cropped by the 2D bounding box is mapped to a single-view shape code via the shape encoder. The state of objects in the map is tracked by a multiple model Bayesian filter. The motion status is indicated by different background colours of object poses. After filter prediction, matchable objects are used to associate to the new set of detections. A matched detection is attached to the object track, and the shape is progressively reconstructed by decoding the fused shape codes.

a simple but practical pairwise cost (*i.e.* 3D Generalized IoU (Rezatofighi et al., 2019)) as the matching cost. We solve the linear assignment problem using the Munkres algorithm (Munkres, 1957) to decide whether a detection merges to an object track or instantiates a new object in the map. Filters are updated using the associated detections (6.3.4). To reconstruct an object shape, multiple single-view shape codes are fused into a single one by taking the mean, which is decoded by the shape decode to a TSDF. The object shape is transformed to the world coordinate using the updated object pose (6.3.5). Lastly, we introduce an approach to solve the scale ambiguity in monocular SLAM using our object shape reconstruction (6.3.6). Fig. 6.2 illustrates the pipeline of our system.

### 6.3.2 3D Localisation

First, MO-LTR detects objects of interest given an RGB image. We apply a monocular 3D detector that takes a single RGB image as input and outputs both 2D attributes (*i.e.* object class and 2D bounding box) and 3D attributes (*i.e.* object translation $\mathbf{t}_{co}$ and viewpoint $\mathbf{R}_{co}$ with respect to the camera, and object 3D dimension $(s_x, s_y, s_z)$). Technically, the detector is trained to predict an offset $(\Delta x, \Delta y)$ between the center of the 2D bounding box $(x_{2d}, y_{2d})$ and the projection center of the 3D shape $(x_{3d}, y_{3d})$ on the image plane. We also predict the object depth value $z$. Assuming we know the camera intrinsic parameters $f_x$, $f_y$, $c_x$, $c_y$, the object's 3D center $\mathbf{t}_{co}$ in camera coordinate frame is recovered as follows:

$$\mathbf{t}_{co} = (\frac{x_{2d} + \Delta x - c_x}{f_x} z, \frac{y_{2d} + \Delta y - c_y}{f_y} z, z) \tag{6.1}$$

To handle the multi-modal nature of symmetric objects, we reformulate object viewpoint prediction as a classification problem, where azimuth $\mathbf{R}_{azi}$ and elevation $\mathbf{R}_{ele}$ are discretised into 36 and 10 bins respectively. The rotation matrix is $\mathbf{R}_{co} = \mathbf{R}_{ele}\mathbf{R}_{azi}$. The transformation matrix $\mathbf{T}_{co} \in \mathrm{SE}(3)$ from the canonical object space to camera coordinate frame is:

$$\mathbf{T}_{co} = \begin{bmatrix} \mathbf{R}_{co} & \mathbf{t}_{co} \\ \mathbf{0} & 1 \end{bmatrix} \tag{6.2}$$

Together with the scale parameters, each detected object is localized as an oriented 3D bounding in the camera coordinate frame.

### 6.3.3 Shape Embedding and Inference

As a shape prior based reconstruction, we are interested in reconstructing the complete object shape even if only a partial observation is available. The formulation of our shape embedding and inference follows Chapter 5 closely. We use compact k-dimension shape codes $\mathbf{l} \in \mathbb{R}^k$ embedded in a learnt latent space to parameterise normalised object shapes in a canonical pose throughout our system. This latent representation effectively allows us to leverage the learnt latent space as a shape prior. A TSDF, where the zero-crossing level set is the object surface, can be decoded from the latent code via a DeepSDF decoder $G(\mathbf{l})$ (Park et al., 2019a).

After each object has been detected, we estimate a single-view shape code by mapping its cropped 2D bounding box to the shape embedding using the shape encoder. Note that at this point we do not reconstruct the shape by decoding the single-view shape code; instead, the shape is decoded later, once shape codes have been fused over time, as described in 6.3.5.

### 6.3.4 Tracking

Because single-view detections are mostly noisy, a common approach in Multiple Object Tracking is to apply a Bayesian filter on the object motion to smooth the tracking trajectory (Weng et al., 2020; Shenoi et al., 2020), and to provide motion predictions for better data association. To deal with both dynamic and static objects, because there is no one-size-fit-all motion model for use within a Bayesian filter, we employ the well-known Interacting Multiple Model (IMM) filter (Blom et al., 1988). The state vector of the IMM filter comprises $\mu = (x, \pi)$ two parts: $x$ for object's kinematics, and $\pi$ for mode probability (*i.e.* how likely an object has the motion status of dynamic/static).

Given a new RGB frame, we first use the IMM filter to predict each existing object's state (*i.e.* kinematics and motion status) from the previous frame to the current frame. Because there are $N$ ($N = 2$ in our case) parallel filters in an IMM filter, the prediction involves three steps:

1) computing the mixed initial conditions:

$$\bar{x}^i_{t-1} = \sum_j \hat{x}^j_{t-1} \mu^{j|i}_{t-1}, \tag{6.3}$$

where $\bar{x}^i_{t-1}$ is the mixed initial state of the $i^{th}$ filter, $\hat{x}^j_{t-1}$ is the updated state of the $j^{th}$ filter at the $t-1$ frame, and $\mu^{j|i}_{t-1}$ is the mixing probability of the $j^{th}$ filter in the $i^{th}$ filter computed as below:

$$\mu^{j|i}_{t-1} = \pi_{ji}\mu^j_{t-1}/\tilde{\mu}^i_t, \tag{6.4}$$

$$\tilde{\mu}^i_t = \sum_j \pi_{ji}\mu^j_{t-1}, \tag{6.5}$$

where $\pi_{ji}$ is the transition probability from the $j^{th}$ filter to the $i^{th}$ filter, $\tilde{\mu}^i_t$ is the predicted mode probability, and $\mu^j_{t-1}$ is the $j^{th}$ filter's updated mode probability at $t-1$ frame.

2) computing each filter's prediction using the mixed initial condition:

$$\tilde{x}^i_t = \mathbf{F}^i\bar{x}^i_{t-1}, \tag{6.6}$$

where $\tilde{x}^i_t$ is the predicted state of the $i^{th}$ filter, and $\mathbf{F}^i$ is the state transition matrix.

3) computing the mixed prediction using the weighted sum of each filter:

$$\tilde{x}_t = \sum_i \tilde{x}^i_t\tilde{\mu}^i_t, \tag{6.7}$$

where $\tilde{x}_t$ is the predicted state of the IMM filter, and $\pi^i$ is the mode probability of the $i^{th}$ filter.

After objects' states are predicted to the current frame, the next step is to associate the new detections to the existing objects. We do not match all existing objects though, an object is matchable if its probability in the static mode is larger than that of the dynamic mode, or it was visible at least once in the previous $t$ frames. We construct a $M \times N$ cost matrix between $M$ new detections in the current frame and $N$ matchable objects in the map, where each element represents the cost for associating the $m^{th}$ detected objects to the $n^{th}$ objects in the map, and the cost is measured by the negative of pairwise 3D generalised IoU (Rezatofighi et al., 2019). To calculate the IoU, the detections are transformed to world coordinate where the existing objects are. The optimal matching is found by the Munkres algorithm (Munkres, 1957).

Each IMM filter is updated using the associated detections as follows: 1) update of each Kalman filter:

$$r_t^i = z_t - \mathbf{H}^i \tilde{x}_t^i, \tag{6.8}$$

$$\hat{x}_t^i = \tilde{x}_t^i + \mathbf{K}_t^i r_t^i, \tag{6.9}$$

where $\hat{x}_t^i$ is the updated state of the $i^{th}$ Kalman filter, $\mathbf{K}_t^i$ is the Kalman gain, $r_t^i$ is the measurement residual, and $z_t$ is the measurement given by the monocular 3D detector; 2) update mode probability for the next frame:

$$L_t^i = \mathcal{N}(r_t^i | 0, \mathbf{S}_t^i), \tag{6.10}$$

$$\hat{\mu}_t^i = \frac{\tilde{\mu}_t^i L_t^i}{\sum_j \tilde{\mu}_t^j L_t^j}, \tag{6.11}$$

where $L_t^i$ is the model likelihood in the $i^{th}$ filter assuming Gaussian noise, $\mathbf{S}_t^i$ is residual covariance, and $\hat{\mu}_t^i$ is the $i^{th}$ filter's updated mode probability; 3) update the mixed IMM state:

$$x_t = \sum_i \hat{x}_k^i \hat{\mu}_k^i, \tag{6.12}$$

where $x_t$ is the the IMM fse, there are two dynamic models of interest: a constant velocity model for the dynamic state and a stationary model for stationary state, each of which is represented by a Kalman Filter.

Similar to any MOT approaches that have to handle trajectory birth and death, we deal with trajectory birth via the standard method, which is instantiating a tentative trajectory from an unassociated detection. A tentative trajectory is turned to a confirmed trajectory only if it is observed $n$ consecutive times. We treat trajectory termination differently. The death of an object trajectory is not controlled by a predefined fixed time threshold. Instead, an object trajectory is terminated only if it is a dynamic object and it is not observed in the last $n$ frames (*i.e.* static objects remain in the map even if not observed). Objects with a live trajectory are considered matchable objects in the data association step.

If an object instance is moved during an unobserved period, it is considered a new object instance when re-observed. The old object instance (if the old position is visible later) is pruned using negative information. Technically, if the largest 2D IoU between the projection of a 3D object shape and all detected 2D bounding boxes of an image is less than 0.5, then this object is considered not visible and pruned for this time step.

### 6.3.5 Reconstruction

After object tracking, the last step of MO-LTR at each RGB frame is to reconstruct each object's dense shape in the map. In Chapter 5, we investigated two fusion techniques for shape codes, averaging and majority voting, and empirical experiments (Table 5.5) suggests that averaging outperforms majority voting. Therefore, we fuse all single-view shape codes up to the current frame by averaging them into a single code $\mathbf{l}_f = \frac{1}{N}\sum_i^N \mathbf{l}_i$. A TSDF that represents an object shape in the canonical object coordinate is decoded from the shape decoder $\mathbf{X}^o = G(\mathbf{l}_f)$ given the fused shape code. An object shape mesh is extracted from the TSDF by the Marching Cube algorithm (Lorensen et al., 1987). The mesh is then transformed to the world coordinate using updated pose from object tracking:

$$\mathbf{X}^w = \mathbf{T}_{wo}\mathbf{S}\mathbf{X}^o \tag{6.13}$$

$$\mathbf{S} = \begin{bmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & s_z \end{bmatrix} \tag{6.14}$$

where $\mathbf{T}_{wo}$ is the rigid transformation from object coordinate to world coordinate and $\mathbf{S}$ is the scale matrix.

The shape and pose can be further optimised using visual cues, such as silhouette and photometric consistency, as shown in Chapter 5.

### 6.3.6 Scale Recovery of Camera Trajectory

It is commonly known that the camera trajectory and reconstruction given by a monocular SLAM is only up to a free scale parameter. That is to say, the camera trajectory and the reconstruction is consistent with each other but not in metric units, and a scale parameter is needed to scale the trajectory and reconstruction up/down to the metric units. This scale ambiguity caused by the unknown absolute baseline between two images is fundamental in a monocular geometric-only SLAM. However, we here show an approach to recover the scale parameter by leveraging our reconstructed object shapes and scale, which further demonstrates the benefit of object-centric mapping.

Given an RGB image, our 3D detector can localise objects in the metric scale. Together with the object meshes, we are able to render a depth map $\mathcal{D}_{obj}$ at this viewpoint. We can also project the sparse feature points $\mathbf{P}$ of the monocular SLAM to the same image plane.

$$\begin{bmatrix}\mathbf{i} & \mathbf{j} & \mathbf{z}\end{bmatrix}^T = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \mathbf{P}, \tag{6.15}$$

where $f_x$, $f_y$, $c_x$, $c_y$ are camera intrinsic parameters, and $[\mathbf{i}, \mathbf{j}]$ are the pixel locations.

The key idea is that if the SLAM reconstruction is indeed in the metric scale, then the depth values of the feature points should be consistent with the rendered depth map from the object shape reconstructions in the metric scale. Therefore, the scale factor $\alpha$ can be obtained by minimizing the discrepancy between the rendered depth map from shape reconstructions and the depth values of the sparse feature points:

$$L(\alpha) = (\mathbb{1}(\mathcal{D}_{obj}[\mathbf{i}, \mathbf{j}] - \alpha \mathbf{z})), \tag{6.16}$$

where $\mathbb{1}$ is an indicator, whose value is 1 if the pixel location $[i, j]$ has a valid depth value from rendered depth map. A pixel does not have a valid depth value if object meshes do not project onto it. We render a single depth map from all visible objects to take care of object occlusions. Because there is only a single scale parameter of the entire sequence, we optimise the $\alpha$ by minizing the loss over all $N$ frames in the sequence:

$$\alpha = \arg\min_{\alpha} \sum_{i}^{N} L(\alpha), \tag{6.17}$$

### 6.3.7 Implementation Details

We dedicate this section to describe the implementation details of each component in the pipeline.

Our detector is built on top of a 2D detector DETR (Carion et al., 2020). DETR takes an image and a set of object queries, each of which is represented by a vector, and predicts each query's 2D bounding box, and object class. To predict the additional 3D attributes – 3D dimension, translate, and rotation, we extend the original DETR by adding an independent prediction feed-forward network for each additional attribute. For indoor scenes, we train the 3D detector on the ScanNet (Dai et al., 2017a) images. Because ScanNet annotations do not provide object dimensions and orientation that we need for training our detector, we use the CAD model annotations provided by Scan2CAD (Avetisyan et al., 2019). We finetune the detector from the official release on the official ScanNet train/val split for 10 epochs. For outdoor scenes, we use an off-the-shelf 3D monocular detector from X. Zhou et al. (2020).

We use $k = 64$ dimensions for our shape embedding. The architecture of our shape decoder is identical to the original DeepSDF (Park et al., 2019a), and we closely follow DeepSDF in the training procedure. The only difference is that we train separate embeddings for indoor classes(*e.g.* chair, table and display), and outdoor classes (*e.g.* car). The architecture of our shape encoder is modified from the ResNet18 by changing the original output dimension to our embedding dimension. It is trained on synthetic images rendered from the ShapeNet (Chang et al., 2015) CAD models with random backgrounds.

We formulate the state vector of the IMM filter as a 7-dimensional vector $\mu = (\mathbf{x}, \pi)$, where $\mathbf{x} = [c_x, c_y, c_z, v_x, v_y, v_z]$ is a 6-dimensional vector that represents the centre and velocity of an object and $\pi$ is the model selection variable. Object centre observation is from the monocular 3D detector, and the measurement covariance is set to $0.01\mathbf{I} \in \mathbb{R}^{3\times3}$ and $0.25\mathbf{I} \in \mathbb{R}^{3\times3}$ (based on empirical results shown in Table 6.2) for indoor and outdoor environment respectively. We use zero velocity and the first observation to initialise the state mean, and covariance is initialised using identity matrix $\mathbf{I} \in \mathbb{R}^{6\times6}$. We use a constant velocity model (with acceleration as process noise) and a zero velocity model (also known as random walk) for dynamic and static motion model respectively. The transition probability matrix is set to$[[0.6, 0.4]; [0.4, 0.6]]$. An object is classified as static if $p(\pi = static) > p(\pi = dynamic)$. Note that at present, we do not incorporate object rotation in the filter state. One complexity of doing so is that the noise of rotation observation predicted by the deep network is non-Gaussian due to object symmetry. For instance, given a car's side-view image, the network prediction has two modes being the left and right side of the car. We attempt to capture this categorical distribution using a Particle Filter for object rotation, but it is discarded later in the development due to the concern about speed.

The data association gating threshold measured by the 3D Generalised IoU is set to 1.75 for outdoor scenes and 0.25 for indoor scenes. We need a higher threshold in the outdoor environment to accommodate that outdoor objects move faster.

We use ground-truth camera poses in our experiments on KITTI and ScanNet for a fair comparison, but the proposed system could work with any off-the-shelf SLAM systems to obtain camera poses. We also demonstrate that our system can work with estimated camera trajectory from DF-VO (Zhan et al., 2020) on KITTI dataset.

## 6.4 Experiments

### 6.4.1 Datasets

We quantitatively evaluate MO-LTR on KITTI (Geiger et al., 2012) and ScanNet (Dai et al., 2017a). KITTI is a popular dataset used for object tracking benchmark in outdoor scenes. It consists of image sequences captured by a camera mounted on a moving vehicle in different road conditions. In contrast, sequences in ScanNet are captured in various indoor scenes (*e.g.*, offices, living rooms, or conference rooms) using a handheld device. However, because the annotated bounding boxes in ScanNet are subject to occlusion or reconstruction failure that leads to incomplete bounding boxes, following FroDO, we instead use the annotations from Scan2CAD (Avetisyan et al., 2019) to obtain amodal 3D bounding boxes for evaluation. Since objects in ScanNet are static, we demonstrate indoor dynamic objects using self-recorded videos qualitatively.

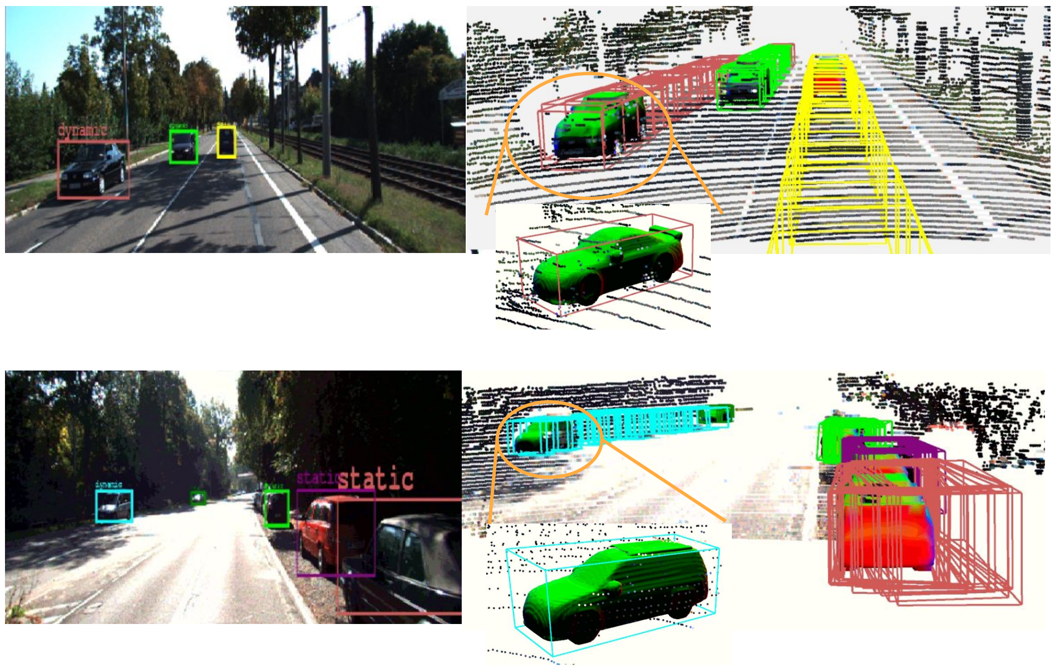


**Figure 6.3:** Object tracking on KITTI dataset. The tracking is consistent, and we correctly label dynamic/static objects using the mode selection variable in the IMM filter. The reconstruction of vehicles are also highlighted. Lidar points are used for visualization purpose.

### 6.4.2 Localization

We compare MO-LTR with FroDO (Runz et al., 2020) on object localisation in 3D to demonstrate the effect of our monocular 3D detector and data association. Table 6.1 presents the comparison with FroDO on three common object categories (*i.e.*, chair, table and display) in indoor scenes. The evaluation metric is the widely adopted mean Average Precision (mAP) in object detection, and the Intersection over Union threshold is set to 0.5.

We outperform FroDO on both chair and display class and have similar performance on table class. We believe that the improvement is due to the differences in our detection and data association approach. FroDO used 2D detections and a ray clustering approach for data association. The 3D bounding boxes are obtained by triangulating associated 2D detections. The ray clustering based data association suffers from local minima and leads to incorrect matching if objects are close to each other. MO-LTR circumvents this problem by using monocular 3D detection,

| mAP @ IoU=0.5 | Chair ↑ | Table ↑ | Display ↑ |
|---|---|---|---|
| FroDO (Chapter 5) | 0.32 | 0.06 | 0.04 |
| Ours | **0.39** | 0.06 | **0.10** |

**Table 6.1:** 3D detection comparison

and thus the following data association works in the 3D space directly. Qualitative results of MO-LTR on ScanNet is shown in Fig. 6.4. It can be seen that the proposed method is more effective on chairs than tables (consistent with the results shown in Table 6.1), large dining table or conference table in particular. This is because the 3D detector tends to perform worse on truncated views (that cannot observe the full extent of an object) of objects. If an image only captures a corner of a table, it is hard to determine how far this table would extend beyond the image frame. Hence, the predictions on dimension and object translation are more uncertain.

### 6.4.3 Tracking

In the conventional KITTI benchmark evaluation, results of 3D MOT are evaluated following the 2D MOT evaluation, where 3D tracking results are projected to the image plane for evaluation. Therefore, it fails to reflect errors on depth direction (*i.e.* an object located at any point along with the projection ray results in the same error). We instead use the 3D MOT evaluation recently proposed in AB3DMOT Weng et al., 2020. The evaluation metrics are Multiple Object Tracking Accuracy (MOTA), Multiple Object Tracking Precision (MOTP) and ID Switches (IDS).

We use CenterTrack (X. Zhou et al., 2020), a monocular 3D multiple object tracking framework, as a baseline. While we share the same monocular 3D detector, the main difference is in the motion tracking and data association method. CenterTrack predicts the 2D object motion on the image plane using a deep network, and associates detections between adjacent frames using the IoU between 2D bounding boxes as a matching cost. Matches are found by a greedy search. We model each object motion using an independent IMM filter in the 3D space and choose Munkres algorithm (Munkres, 1957) over a greedy search.

| Methods | MOTA ↑ | MOTP ↑ | IDS ↓ |
|---|---|---|---|
| CenterTrack (X. Zhou et al., 2020) | 0.34 | 0.53 | 68 |
| Ours (No filter) | 0.37 | 0.53 | 12 |
| Ours ($\sigma$=0.1) | 0.39 | 0.53 | 7 |
| Ours ($\sigma$=1) | 0.36 | 0.51 | 4 |
| Ours ($\sigma$=0.25) | **0.39** | 0.53 | **0** |
| CenterTrack (X. Zhou et al., 2020)(w/VO) | 0.31 | 0.51 | 68 |
| Ours (w/VO) | 0.36 | 0.52 | 0 |

**Table 6.2:** 3D Object Tracking comparison on KITTI (Geiger et al., 2012), $\sigma$ is multiplier on the diagonal covariance matrix.

| Methods | Error metrics | | | | accuracy | | |
|---|---|---|---|---|---|---|---|
| | RMSE ↓ | log RMSE ↓ | Abs Rel ↓ | Seq Rel ↓ | $\delta < 1.25$ ↑ | $\delta < 1.25^2$ ↑ | $\delta < 1.25^3$ ↑ |
| MOTSFusion (Mono.) | 5.33 | 0.26 | 0.17 | 1.71 | 0.76 | 0.91 | 0.94 |
| Ours | **2.09** | **0.13** | **0.12** | **0.50** | **0.88** | **0.97** | **0.99** |

**Table 6.3:** Quantitative depth comparison to MOTSFusion

The quantitative result is shown in Table 6.2. We reduce ID-switches significantly. We believe the reason for the improvement is twofold: 1) We perform object tracking and data association in the 3D space so objects in similar depth direction but with different values can be separate easily; 2) we solve the linear assignment problem in data association instead of a greedy search. The object ID consistency is crucial for object-centric mapping as duplicate object instances degenerate the reconstruction quality. We visualise our tracking result and the motion state estimation by the IMM filter in Fig. 6.3.

To verify that our approach can also work with estimated camera poses from a SLAM or VO system, we also run evaluation with estimated camera poses from the state-of-the-art Visual Odometry system – DF-VO (Zhan et al., 2020). The performance of both CenterTrack and our method drops slightly due to the camera tracking error, but note that ours still outperform CenterTrack.

### 6.4.4 Reconstruction

We compare MO-LTR against the monocular MOTSFusion, where they use a monocular depth estimation network followed by an instance segmentation network for object reconstruction to recover the *visible* surfaces of objects. Because

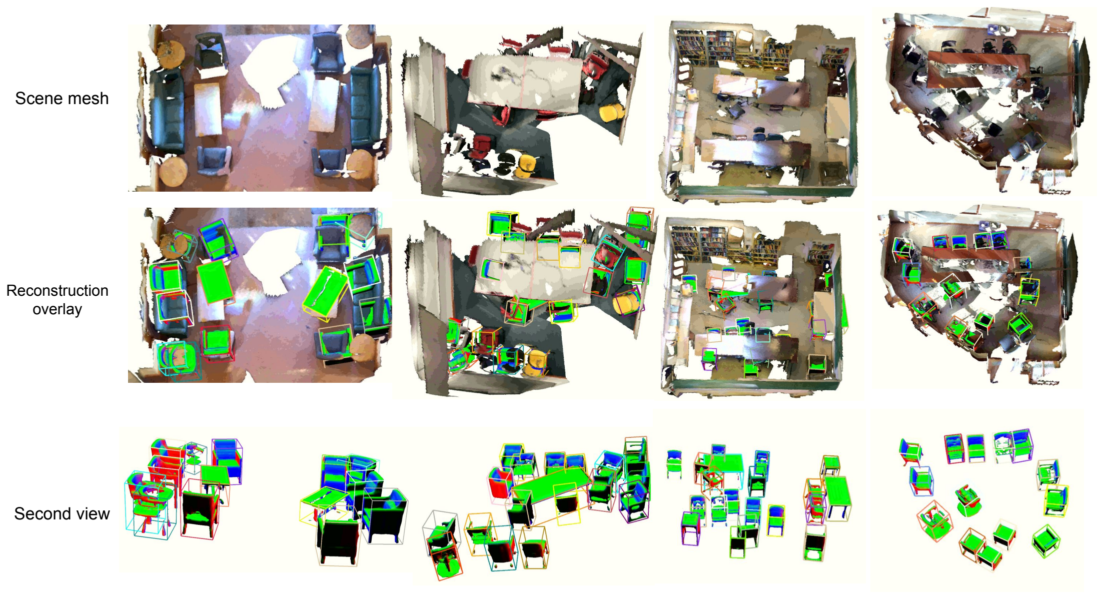


**Figure 6.4:** Localization and reconstruction on ScanNet sequences. Top row: Ground-truth scan mesh for reference, middle row: objects overlay on the ground-truth mesh to show localization quality. Bottom row: object shape reconstruction. Scan mesh is for visualization purpose only. The input to MO-LTR is camera poses and RGB images only.

MOTSFusion does not reconstruct full 3D shape, our method would be favoured if we were to evaluate reconstruction in 3D. Instead for a fairer comparison, we render our full 3D shape onto the image plane as a depth map and compare the reconstruction quality against MOTSFusion via depth map evaluation.

Our shape prior driven approach outperforms MOTSFusion by a large margin, as shown in Table 6.3. MOTSFusion particularly suffers from RMSE, indicating it is affected by the blurry object edge from the depth prediction and instance segmentation. To better contrast both methods, we visualise the comparison in Fig. 6.5. Even when MOTSFusion can reconstruct the surface accurately, our shape prior-based reconstruction is still advantageous as we can reconstruct an object's full 3D shape.

## 6.4.5 Indoor Tracking and Reconstruction

We run MO-LTR on a self-recorded video[1] with the focus on demonstrating tracking objects whose motion status switches between dynamic and static throughout the

[1]https://youtu.be/oTG3nF3aUF8

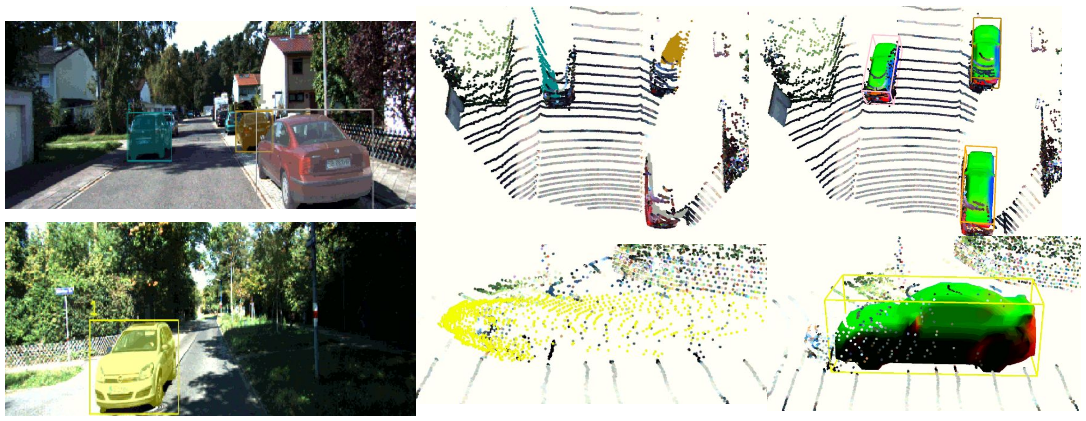


**Figure 6.5:** Reconstruction comparison to monocular MOTSFusion on KITTI. Left column: Current frame, middle column: reconstruction by MOTSFusion, right column: our reconstruction. Note that colored lidar points are used for visualization only, not part of the processing.

video. MO-LTR is able to classify whether an object is static or dynamic accurately at each time step, which indicates that our IMM filter captures the model switching behaviour. Another highlight of this video is that a chair is tracked successfully even though it was entirely occluded by another object for a period of time. That is because the chair in the red bounding box is classified as static correctly, and we use the motion status to control object trajectory termination. The occluded chair is at risk of being discarded if we follow common MOT practice of a predefined fixed time threshold to terminate unobserved objects.

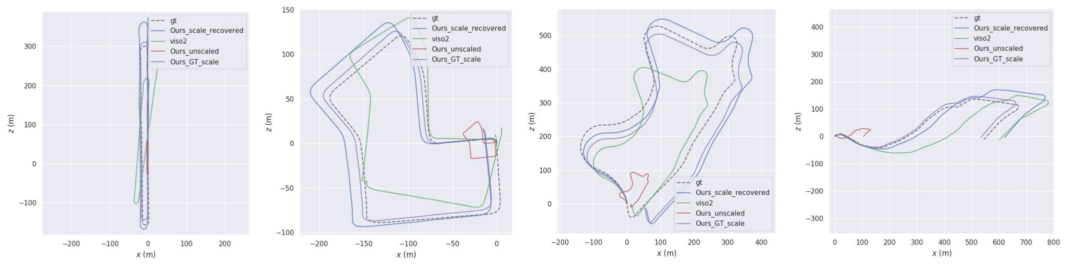


**Figure 6.6:** Camera trajectories comparison between ours, ours with ground-truth scale, ours without scale recovery, and the monocular VISO2 on KITTI sequences 06, 07, 09, 10 from left to right.

| | monocular VISO2 | | | Ours | | | Ours (GT scale) | | |
|---|---|---|---|---|---|---|---|---|---|
| Seqs. | ATE | RPE (m) | RPE (°) | ATE | RPE (m) | RPE (°) | ATE | RPE (m) | RPE (°) |
| 06 | 40.7 | 0.56 | 0.15 | 23.1 | 0.21 | 0.03 | 4.8 | 0.05 | 0.03 |
| 07 | 18.3 | 0.32 | 0.35 | 7.8 | 0.08 | 0.03 | 3.7 | 0.04 | 0.03 |
| 09 | 52.6 | 0.34 | 0.16 | 25.1 | 0.15 | 0.04 | 15.3 | 0.14 | 0.04 |
| 10 | 57.2 | 1.12 | 0.33 | 34.9 | 0.15 | 0.05 | 6.7 | 0.08 | 0.05 |

**Table 6.4:** Quantitative odometry evaluation on KITTI odometry sequences

| Seqs. | 06 | 07 | 09 | 10 | Avg. |
|---|---|---|---|---|---|
| Relative error | 14.1% | 7.0% | 8.1% | 13.9% | 10.7% |

**Table 6.5:** relative error on scale recovry

### 6.4.6 Scale Recovery

The input to our scale recovery algorithm is camera trajectories estimated by a monocular SLAM (Mur-Artal et al., 2017) (shown as red colour in Fig. 6.6). The goal is to bring these unscaled camera trajectories to metric scale. We first measure the performance using the relative error between our estimation and the ground-truth scale parameter obtained by running the SIM(3) alignment (*i.e.* pose and scale) to the ground-truth camera trajectories. The result is reported in Table 6.5. To better showcase the effect of the scale on odometry evaluation, we also compare our method to the monocular VISO2 (Geiger et al., 2011), which assumes a fixed camera height over the ground plane to estimate the scale factor. Comparing trajectories are aligned to the ground-truth trajectories using SE(3) optimisation, such that the scale error is reflected on the metrics. We also report the errors of our camera trajectories aligned the ground-truth scale, which is the upper bound of our scale recovery method. The trajectory comparison is shown in Table 6.4 and visualised in Fig. 6.6. It is clear that the scale of camera trajectories after applying our scale estimation is closer to the ground-truth than VISO2.

### 6.4.7 Runtime analysis

All experiments are run on an Intel Core i7 desktop with 16 GB RAM and an Nvidia GeForce GTX 1070 GPU. The runtime cost for each component is shown in

| stage | Detection | shape encode | association | shape decode |
|---|---|---|---|---|
| time (ms) | 111/frame | 4/det. obj. | 3/frame | 35/obj. |

**Table 6.6:** Runtime analysis breakdown for each system component. det. obj. refers to detected object

Table 6.6. In practice, we can run between 2 - 4 Hz in a scene with five objects.

## 6.5 Conclusion

In this chapter, we presented MO-LTR, a framework for multi-object localization, tracking and reconstruction given monocular image sequences. We leveraged the deep shape prior for complete and accurate shape reconstruction and the IMM filter to jointly track an object's motion and discriminate motion status. We evaluated MO-LTR extensively on both indoor and outdoor scenes under both static and dynamic environments. While we have shown that the data association using 3D GIoU works in practice, an interesting future direction is to develop a learning-based approach for data association. This could furthermore pave the way for an end-to-end learnable system. We observed that the filter over-smooths object motion due to a predefined high measurement noise to handle the noisy single-view detections. It is worth exploring probabilistic object detection to improve the tracking performance. Another immediate step to extend MO-LTR is to integrate it into a full SLAM framework such that the object prior knowledge could be leveraged in SLAM for a scale-consistent camera trajectory.

# 7 Conclusion

**Contents**

## 7.1 Thesis summary

This thesis has explored methods of combining deep learning and traditional geometry to achieve object-centric mapping. We started by addressing the problem of object shape reconstruction given a single image. Although deep networks have shown expressive results on this topic, traditional representations used for shapes, such as voxels and point clouds, do not scale well. Thus, reconstructions generated by a deep network are generally limited to a low resolution due to the memory limitation of GPUs. To generate a dense point cloud, Lin et al. (2018b) proposed a multi-view representation that predicted multiple depth maps from fixed viewpoints, which can then fused into a dense point cloud using the known camera intrinsic and extrinsic parameters. However, this representation leads to the undesirable sparsity of the fused point cloud because a predicted mask filters out background depth pixels that cannot be unprojected to the object surface. To address this problem, we proposed a novel representation in Chapter 3. Instead of predicting a binary mask to filter out background pixels, we predict a deformation field to alter pixel locations on the depth map image plane before unprojecting them to a 3D point using the predicted depth values. Moreover, the other disadvantage of using a predicted mask to filter out background pixels is that a threshold must be set to binarise the mask. We showed in our experiments that there is no universally fit threshold across different object instances. As a result, our efficient representation can reconstruct an object shape using a denser and more accurate point cloud.

While we tried to integrate our deep learning based single-view object reconstruction network into an object SLAM system (Hosseinzadeh et al., 2019), we identified several undesirable properties of the pure deep learning based shape inference approach. Firstly, there is a domain gap between synthetic images used for training and the real images when testing. Secondly, the prediction cannot be updated when information that may help improve the reconstruction (*e.g.* a new observation, object silhouette) are available. Chapter 4 introduced a new approach for single-view object reconstruction to address the limitations above. The core of this approach is the probabilistic deep shape prior. We first map object shapes into a low-dimensional embedding using an AutoEncoder. We then fit a Gaussian Mixture Model in this embedding to obtain a probabilistic shape prior. The single-view object reconstruction is cast as an optimisation problem, in which the energy function consists of a silhouette consistency term and a shape prior term, and the variables to be optimised are the object shape and object pose. We demonstrated that our approach outperforms the prior art on a benchmark dataset for single-view object reconstruction. The shape prior term induced by the GMM is crucial to generate plausible object shapes when the predicted silhouette from a semantic segmentation network is not accurate.

Largely built upon the techniques developed in Chapter 3 and 4, we introduced our object-centric mapping system – FroDO – in Chapter 5. Taking as input an RGB image sequence and their poses, FroDO can represent a scene by a set of object reconstructions and their poses. We proposed to use a ray clustering algorithm to address the data association problem. Object shape and pose are jointly optimised given the associated observations using a set of geometry constraints. Following Chapter 4, each object shape is encoded in a low-dimensional shape embedding, However, to make the best of different shape representations, we proposed a joint embedding for point cloud (fast to inference, but sparse) and DeepSDF (slow, but accurate and dense). FroDO is the first object-centric mapping system that combines deep-learnt shape prior and traditional geometry using RGB-only image sequences.

FroDO focused on static scenes, but object motions are inevitable in many applications, especially in those that involve human interactions. To this end, we introduced MO-LTR in Chapter 6 as an effort to achieve object-centric mapping in dynamic environments. While we relied on multiple associated 2D bounding boxes to localise an object in 3D in FroDO, we proposed a single-view 3D object detector in this chapter. Although the 3D predictions are noisy, they provide a strong cue for data association. We employed an Interactive Multiple Models(IMM) filter to smooth the noisy predictions and track an object's motion. We showed that MO-LTR could handle object motions in both indoor and outdoor environments.

## 7.2 Future work

The works presented in this thesis demonstrate the possibility of building an object-centric mapping system that combines both traditional geometry and modern deep learning. However, there is still a long way for object-centric mapping to reach a maturity level for successful commercial applications. Some exciting research directions are discussed in this section.

**End-to-end object detection and tracking** This is a promising research direction that we have not had a chance to explore in this thesis. Detection and tracking have never been addressed entirely in a unified deep learning framework. However, researchers are moving toward this direction. For instance, Sarlin et al. (2020) trained a graph neural network to match feature points between two images. Similar ideas can be applied to transform heuristic-driven data association used in the thesis to a data-driven approach. Brasó et al. (2020) proposed to train a message passing graph neural network to solve the Multiple Object Tracking (MOT) problem. However, this approach still needs to take input from an external detector.

An end-to-end detection and tracking network is likely to offer improvements as follows: Firstly, unlike estimating objects' 3D location from a single image in MO-LTR, an end-to-end framework that takes input as an image sequence can leverage multi-view information for 3D object detection. Secondly, tracking will be more robust to detector errors by training jointly. A tracker taking input from an external detector cannot rectify detection errors in existing detection-based tracking frameworks. In contrast, training detection and tracking in a unified framework may alleviate the problem of error propagation. Lastly, while we only used geometry for data association in Chapter 5 and 6, an end-to-end network could extract object feature for matching.

**Semi-supervised learning on object detection and tracking** Training an end-to-end network for 3D object detection and tracking in a fully supervised manner comes with the cost of much expensive data annotation. Annotating each objects' location in 3D at every frame for millions of images is unimaginably expensive using the current annotation tools. A strong motivation for developing an object-centric mapping system in this thesis is for advanced robotic and AR/VR applications. When AR/VR headsets gain popularity in the mainstream, there would be many images and videos recorded/uploaded everyday. Exploring semi-supervised learning methods to train a network using a small subset of labelled data is more feasible than annotating all these images and videos. Generative Pretrained Transformer-3 (GPT3) (Brown et al., 2020) showed that a language model could be trained unsupervised. They demonstrated impressive results on various Natural Language Processing (NLP) tasks by finetuning this pretrained model. Does that mean we can also train an "pretrained image model" using similar techniques when we have enough images? We then can use the subset of labelled data to finetune the "pretrained image model" for any specific image-related tasks.

**Deformable Objects** This thesis has focused on rigid objects. We captured category-level shape variances using a low-dimensional shape embedding.

Deformable objects are also crucial in object-centric mapping. An example of deformable objects is human body shapes. Kocabas et al. (2020) showed impressive results on human body shape reconstruction in a video. However, this work is limited to a single person and does not handle occlusion well. Further work on data association and multi-person occlusion would be the key to apply this method to videos in the wild. Besides human body shapes, furniture could also become deformable due to the interaction with human beings. For instance, a human can adjust the height of an office chair; he/she can open a drawer of a cabinet. Object part recognition (Qi et al., 2017) and affordance reasoning (Do et al., 2018) could offer some opportunities to deal with furniture deformation.

**Unifying object-centric mapping and scene layout understanding** Our work on object-centric mapping focus on objects solely and handle each object independently. However, many other elements could help a system to perceive the environment better. We have exploited shape prior knowledge in this thesis inspired by human beings' ability to complete an object shape given limited observations using experience. There might be scene prior knowledge at a room level that captures room structures, room types, room configuration, and object relationships. There is some preliminary research on this direction (Purkait et al., 2020), but they are limited to simple scene configurations. A top-down approach utilising scene prior and shape prior jointly could enforce more constraints to robustify the mapping. Furthermore, a mapping system that reasons from detailed object shapes to high-level scene configurations would facilitate research in many directions, such as visual navigation and visual question answering.

# Bibliography

Davison, Andrew J, Ian D Reid, Nicholas D Molton, and Olivier Stasse (2007). “MonoSLAM: Real-time single camera SLAM”. In: *IEEE transactions on pattern analysis and machine intelligence* 29.6, pp. 1052–1067.

Klein, Georg and David Murray (2007). “Parallel tracking and mapping for small AR workspaces”. In: *2007 6th IEEE and ACM international symposium on mixed and augmented reality*. IEEE, pp. 225–234.

Newcombe, Richard A, Shahram Izadi, Otmar Hilliges, David Molyneaux, David Kim, Andrew J Davison, Pushmeet Kohi, Jamie Shotton, Steve Hodges, and Andrew Fitzgibbon (2011a). “KinectFusion: Real-time dense surface mapping and tracking”. In: *2011 10th IEEE International Symposium on Mixed and Augmented Reality*. IEEE, pp. 127–136.

Whelan, Thomas, Stefan Leutenegger, R Salas-Moreno, Ben Glocker, and Andrew Davison (2015). “ElasticFusion: Dense SLAM without a pose graph”. In: Robotics: Science and Systems.

Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E Hinton (2017). “Imagenet classification with deep convolutional neural networks”. In: *Communications of the ACM* 60.6, pp. 84–90.

Simonyan, Karen and Andrew Zisserman (2014). “Very deep convolutional networks for large-scale image recognition”. In: *arXiv preprint arXiv:1409.1556*.

He, Kaiming, Xiangyu Zhang, Shaoqing Ren, and Jian Sun (2016). “Deep residual learning for image recognition”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778.

Girshick, Ross, Jeff Donahue, Trevor Darrell, and Jitendra Malik (2015). “Region-based convolutional networks for accurate object detection and segmentation”. In: *IEEE transactions on pattern analysis and machine intelligence* 38.1, pp. 142–158.

Girshick, Ross (2015). “Fast r-cnn”. In: *Proceedings of the IEEE international conference on computer vision*, pp. 1440–1448.

Ren, Shaoqing, Kaiming He, Ross Girshick, and Jian Sun (2016). “Faster r-cnn: Towards real-time object detection with region proposal networks”. In: *IEEE transactions on pattern analysis and machine intelligence* 39.6, pp. 1137–1149.

He, Kaiming, Georgia Gkioxari, Piotr Dollár, and Ross Girshick (2017). “Mask r-cnn”. In: *Proceedings of the IEEE international conference on computer vision*, pp. 2961–2969.

Carion, Nicolas, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko (2020). “End-to-End Object Detection with Transformers”. In: *arXiv preprint arXiv:2005.12872*.

Vaswani, Ashish, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin (2017). "Attention is all you need". In: *Advances in neural information processing systems*, pp. 5998–6008.

Long, Jonathan, Evan Shelhamer, and Trevor Darrell (2015). "Fully convolutional networks for semantic segmentation". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 3431–3440.

Goodfellow, Ian, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio (2014). "Generative adversarial nets". In: *Advances in neural information processing systems*, pp. 2672–2680.

Zhu, Junyan, Taesung Park, Phillip Isola, and Alexei Efros (2017). "Unpaired image-to-image translation using cycle-consistent adversarial networks". In: *Proceedings of the IEEE international conference on computer vision*, pp. 2223–2232.

Qi, Charles R, Hao Su, Kaichun Mo, and Leonidas J Guibas (2017). "Pointnet: Deep learning on point sets for 3d classification and segmentation". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 652–660.

Riegler, Gernot, Ali Osman Ulusoy, and Andreas Geiger (2017). "Octnet: Learning deep 3d representations at high resolutions". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3577–3586.

Dai, Angela and Matthias Nießner (2018). "3dmv: Joint 3d-multi-view prediction for 3d semantic scene segmentation". In: *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 452–468.

Qi, Charles R, Or Litany, Kaiming He, and Leonidas J Guibas (2019). "Deep hough voting for 3d object detection in point clouds". In: *Proceedings of the IEEE International Conference on Computer Vision*, pp. 9277–9286.

Hou, Ji, Angela Dai, and Matthias Nießner (2019). "3d-sis: 3d semantic instance segmentation of rgb-d scans". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4421–4430.

Nie, Yinyu, Ji Hou, Xiaoguang Han, and Matthias Nießner (2020). "RfD-Net: Point Scene Understanding by Semantic Instance Reconstruction". In: *arXiv preprint arXiv:2011.14744*.

Dai, Angela, Angel X Chang, Manolis Savva, Maciej Halber, Thomas Funkhouser, and Matthias Nießner (2017a). "Scannet: Richly-annotated 3d reconstructions of indoor scenes". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 5828–5839.

Lowe, David G (2004). "Distinctive image features from scale-invariant keypoints". In: *International journal of computer vision* 60.2, pp. 91–110.

Bay, Herbert, Andreas Ess, Tinne Tuytelaars, and Luc Van Gool (2008). "Speeded-up robust features (SURF)". In: *Computer vision and image understanding* 110.3, pp. 346–359.

Rublee, Ethan, Vincent Rabaud, Kurt Konolige, and Gary Bradski (2011). "ORB: An efficient alternative to SIFT or SURF". In: *2011 International conference on computer vision*. Ieee, pp. 2564–2571.

DeTone, Daniel, Tomasz Malisiewicz, and Andrew Rabinovich (2018). "Superpoint: Self-supervised interest point detection and description". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pp. 224–236.

Sarlin, Paul-Edouard, Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich (2020). "Superglue: Learning feature matching with graph neural networks". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 4938–4947.

Eigen, David, Christian Puhrsch, and Rob Fergus (2014). "Depth map prediction from a single image using a multi-scale deep network". In: *Advances in neural information processing systems* 27, pp. 2366–2374.

Wang, Xiaolong, David Fouhey, and Abhinav Gupta (2015). "Designing deep networks for surface normal estimation". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 539–547.

Fischer, Philipp, Alexey Dosovitskiy, Eddy Ilg, Philip Häusser, Caner Hazırbaş, Vladimir Golkov, Patrick Van der Smagt, Daniel Cremers, and Thomas Brox (2015). "Flownet: Learning optical flow with convolutional networks". In: *arXiv preprint arXiv:1504.06852*.

Kendall, Alex, Hayk Martirosyan, Saumitro Dasgupta, Peter Henry, Ryan Kennedy, Abraham Bachrach, and Adam Bry (2017). "End-to-end learning of geometry and context for deep stereo regression". In: *Proceedings of the IEEE International Conference on Computer Vision*, pp. 66–75.

Garg, Ravi, Vijay Kumar Bg, Gustavo Carneiro, and Ian Reid (2016). "Unsupervised cnn for single view depth estimation: Geometry to the rescue". In: *European conference on computer vision*. Springer, pp. 740–756.

Zhou, Matthew Brown, Noah Snavely, and David G Lowe (2017). "Unsupervised learning of depth and ego-motion from video". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1851–1858.

Eigen, David and Rob Fergus (2015). "Predicting depth, surface normals and semantic labels with a common multi-scale convolutional architecture". In: *Proceedings of the IEEE international conference on computer vision*, pp. 2650–2658.

Tsai, Andy, Anthony Yezzi, William Wells, Clare Tempany, Dewey Tucker, Ayres Fan, W Eric Grimson, and Alan Willsky (2003). “A shape-based approach to the segmentation of medical imagery using level sets”. In: *IEEE transactions on medical imaging* 22.2, pp. 137–154.

Dambreville, Samuel, Yogesh Rathi, and Allen Tannenbaum (2008a). “A framework for image segmentation using shape models and kernel space shape priors”. In: *IEEE transactions on pattern analysis and machine intelligence* 30.8, pp. 1385–1399.

Prisacariu, Aleksandr Segal, and Ian Reid (2012). “Simultaneous monocular 2D segmentation, 3D pose recovery and 3D reconstruction”. In: *Asian conference on computer vision*. Springer, pp. 593–606.

Fan, Haoqiang, Hao Su, and Leonidas J Guibas (2017a). “A point set generation network for 3d object reconstruction from a single image”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 605–613.

Choy, Christopher B, Danfei Xu, JunYoung Gwak, Kevin Chen, and Silvio Savarese (2016a). “3d-r2n2: A unified approach for single and multi-view 3d object reconstruction”. In: *European Conference on Computer Vision*. Springer, pp. 628–644.

Girdhar, Rohit, David F Fouhey, Mikel Rodriguez, and Abhinav Gupta (2016). “Learning a predictable and generative vector representation for objects”. In: *European Conference on Computer Vision*. Springer, pp. 484–499.

Chang, Angel X, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, et al. (2015). “Shapenet: An information-rich 3d model repository”. In: *arXiv preprint arXiv:1512.03012*.

Tulsiani, Shubham, Tinghui Zhou, Alexei A Efros, and Jitendra Malik (2017a). “Multi-view supervision for single-view reconstruction via differentiable ray consistency”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2626–2634.

Yan, Xinchen, Jimei Yang, Ersin Yumer, Yijie Guo, and Honglak Lee (2016). “Perspective transformer nets: Learning single-view 3d object reconstruction without 3d supervision”. In: *Advances in neural information processing systems*, pp. 1696–1704.

Kanazawa, Angjoo, Shubham Tulsiani, Alexei A Efros, and Jitendra Malik (2018). “Learning category-specific mesh reconstruction from image collections”. In: *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 371–386.

Dai, Angela, Charles Ruizhongtai Qi, and Matthias Nießner (2017b). “Shape completion using 3d-encoder-predictor cnns and shape synthesis”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 5868–5877.

Yang, Bo, Stefano Rosa, Andrew Markham, Niki Trigoni, and Hongkai Wen (2018). “Dense 3D object reconstruction from a single depth view”. In: *IEEE transactions on pattern analysis and machine intelligence* 41.12, pp. 2820–2834.

Pujari, Arun K (1993). “Volume intersection for shape from silhouettes”. In: *Sadhana* 18.2, pp. 325–336.

Seitz, Steven M and Charles R Dyer (1999). “Photorealistic scene reconstruction by voxel coloring”. In: *International Journal of Computer Vision* 35.2, pp. 151–173.

Kutulakos, Kiriakos N and Steven M Seitz (2000). “A theory of shape by space carving”. In: *International journal of computer vision* 38.3, pp. 199–218.

Zhang, Ruo, Ping-Sing Tsai, James Edwin Cryer, and Mubarak Shah (1999). “Shape-from-shading: a survey”. In: *IEEE transactions on pattern analysis and machine intelligence* 21.8, pp. 690–706.

Sridhar, Srinath, Davis Rempe, Julien Valentin, Bouaziz Sofien, and Leonidas J Guibas (2019). “Multiview aggregation for learning category-specific shape reconstruction”. In: *Advances in Neural Information Processing Systems*, pp. 2351–2362.

Wiles, Olivia and Andrew Zisserman (2017). “Silnet: Single-and multi-view reconstruction by learning from silhouettes”. In: *arXiv preprint arXiv:1711.07888.*

Kar, Abhishek, Christian Häne, and Jitendra Malik (2017). “Learning a multi-view stereo machine”. In: *Advances in neural information processing systems*, pp. 365–376.

Tatarchenko, Maxim, Alexey Dosovitskiy, and Thomas Brox (2017). “Octree Generating Networks: Efficient Convolutional Architectures for High-resolution 3D Outputs”. In: *arXiv preprint arXiv:1703.09438.*

Johnston, Adrian, Ravi Garg, Gustavo Carneiro, Ian Reid, and Anton van den Hengel (2017). “Scaling CNNs for High Resolution Volumetric Reconstruction From a Single Image”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 939–948.

Wang, Nanyang, Yinda Zhang, Zhuwen Li, Yanwei Fu, Wei Liu, and Yu-Gang Jiang (2018). “Pixel2mesh: Generating 3d mesh models from single rgb images”. In: *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 52–67.

Henderson, Paul and Vittorio Ferrari (2018). “Learning to generate and reconstruct 3d meshes with only 2d supervision”. In: *arXiv preprint arXiv:1807.09259.*

Pontes, Jhony K, Chen Kong, Sridha Sridharan, Simon Lucey, Anders Eriksson, and Clinton Fookes (2018). “Image2mesh: A learning framework for single image 3d reconstruction”. In: *Asian Conference on Computer Vision.* Springer, pp. 365–381.

Gkioxari, Georgia, Jitendra Malik, and Justin Johnson (2019). “Mesh r-cnn”. In: *Proceedings of the IEEE International Conference on Computer Vision*, pp. 9785–9795.

Lin, Chen-Hsuan, Chen Kong, and Simon Lucey (2018a). “Learning Efficient Point Cloud Generation for Dense 3D Object Reconstruction”. In: *AAAI Conference on Artificial Intelligence (AAAI).*

Park, Jeong Joon, Peter Florence, Julian Straub, Richard Newcombe, and Steven Lovegrove (2019a). “DeepSDF: Learning continuous signed distance functions for shape representation”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 165–174.

Mescheder, Lars, Michael Oechsle, Michael Niemeyer, Sebastian Nowozin, and Andreas Geiger (2019a). “Occupancy networks: Learning 3d reconstruction in function space”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4460–4470.

Liu, Shaohui, Yinda Zhang, Songyou Peng, Boxin Shi, Marc Pollefeys, and Zhaopeng Cui (2019). “DIST: Rendering Deep Implicit Signed Distance Function with Differentiable Sphere Tracing”. In: *arXiv preprint arXiv:1911.13225.*

Triggs, Bill, Philip F McLauchlan, Richard I Hartley, and Andrew W Fitzgibbon (1999). “Bundle adjustment—a modern synthesis”. In: *International workshop on vision algorithms.* Springer, pp. 298–372.

Mur-Artal, Raul and Juan D Tardós (2017). “Orb-slam2: An open-source slam system for monocular, stereo, and rgb-d cameras”. In: *IEEE Transactions on Robotics* 33.5, pp. 1255–1262.

Newcombe, Richard A, Steven J Lovegrove, and Andrew J Davison (2011b). “DTAM: Dense tracking and mapping in real-time”. In: *2011 international conference on computer vision.* IEEE, pp. 2320–2327.

Lovegrove, Steven (2012). “Parametric dense visual SLAM”. In:

Hosseinzadeh, Mehdi, Yasir Latif, and Ian Reid (2017). “Sparse Point-plane Slam”. In:

Yang, Shichao, Yu Song, Michael Kaess, and Sebastian Scherer (2016). “Pop-up slam: Semantic monocular plane slam for low-texture environments”. In: *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS).* IEEE, pp. 1222–1229.

Henry, Peter, Michael Krainin, Evan Herbst, Xiaofeng Ren, and Dieter Fox (2014). “RGB-D mapping: Using depth cameras for dense 3D modeling of indoor environments”. In: *Experimental robotics.* Springer, pp. 477–491.

Newcombe, Richard A, Dieter Fox, and Steven M Seitz (2015). “Dynamicfusion: Reconstruction and tracking of non-rigid scenes in real-time”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 343–352.

Bloesch, Michael, Jan Czarnowski, Ronald Clark, Stefan Leutenegger, and Andrew J Davison (2018a). “CodeSLAM-Learning a Compact, Optimisable Representation for Dense Visual SLAM”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition.*

Cadena, Cesar, Luca Carlone, Henry Carrillo, Yasir Latif, Davide Scaramuzza, José Neira, Ian Reid, and John J Leonard (2016). “Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age”. In: *IEEE Transactions on robotics* 32.6, pp. 1309–1332.

Kahler, Olaf and Ian Reid (2013). “Efficient 3d scene labeling using fields of trees”. In: *Proceedings of the IEEE International Conference on Computer Vision*, pp. 3064–3071.

Nowozin, Sebastian, Carsten Rother, Shai Bagon, Toby Sharp, Bangpeng Yao, and Pushmeet Kohli (2011). “Decision tree fields”. In: *2011 International Conference on Computer Vision.* IEEE, pp. 1668–1675.

Jancsary, Jeremy, Sebastian Nowozin, Toby Sharp, and Carsten Rother (2012). “Regression Tree Fields—An efficient, non-parametric approach to image labeling problems”. In: *2012 IEEE Conference on Computer Vision and Pattern Recognition.* IEEE, pp. 2376–2383.

Pham, Trung T, Ian Reid, Yasir Latif, and Stephen Gould (2015). “Hierarchical higher-order regression forest fields: An application to 3d indoor scene labelling”. In: *Proceedings of the IEEE international conference on computer vision*, pp. 2246–2254.

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

Valentin, Julien, Vibhav Vineet, Ming-Ming Cheng, David Kim, Jamie Shotton, Pushmeet Kohli, Matthias Nießner, Antonio Criminisi, Shahram Izadi, and Philip Torr (2015). “Semanticpaint: Interactive 3d labeling and learning at your fingertips”. In: *ACM Transactions on Graphics (TOG)* 34.5, pp. 1–17.

McCormac, John, Ankur Handa, Andrew Davison, and Stefan Leutenegger (2017). “Semanticfusion: Dense 3d semantic mapping with convolutional neural networks”. In: *2017 IEEE International Conference on Robotics and automation (ICRA).* IEEE, pp. 4628–4635.

Tateno, Keisuke, Federico Tombari, Iro Laina, and Nassir Navab (2017). “Cnn-slam: Real-time dense monocular slam with learned depth prediction”. In: *Proceedings* *of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 6243–6252.

Zhi, Shuaifeng, Michael Bloesch, Stefan Leutenegger, and Andrew J Davison (2019). “Scenecode: Monocular dense semantic reconstruction using learned encoded scene representations”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 11776–11785.

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

Dame, Amaury, Victor A Prisacariu, Carl Y Ren, and Ian Reid (2013). “Dense reconstruction using 3D object shape priors”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1288–1295.

Bao, Yingze, Manmohan Chandraker, Yuanqing Lin, and Silvio Savarese (2013a). “Dense object reconstruction with semantic priors”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1264–1271.

Rubino, Cosimo, Marco Crocco, and Alessio Del Bue (2017). “3d object localisation from multi-view image detections”. In: *IEEE transactions on pattern analysis and machine intelligence* 40.6, pp. 1281–1294.

Nicholson, Lachlan, Michael Milford, and Niko Sünderhauf (2018). “Quadricslam: Dual quadrics from object detections as landmarks in object-oriented slam”. In: *IEEE Robotics and Automation Letters* 4.1, pp. 1–8.

Hosseinzadeh, Mehdi, Kejie Li, Yasir Latif, and Ian Reid (2019). “Real-time monocular object-model aware sparse SLAM”. In: *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, pp. 7123–7129.

Yang, Shichao and Sebastian Scherer (2019a). “Cubeslam: Monocular 3-d object slam”. In: *IEEE Transactions on Robotics* 35.4, pp. 925–938.

McCormac, John, Ronald Clark, Michael Bloesch, Andrew Davison, and Stefan Leutenegger (2018a). “Fusion++: Volumetric object-level slam”. In: *2018 international conference on 3D vision (3DV)*. IEEE, pp. 32–41.

Xu, Binbin, Wenbin Li, Dimos Tzoumanikas, Michael Bloesch, Andrew Davison, and Stefan Leutenegger (2019). “Mid-fusion: Octree-based object-level multi-instance dynamic SLAM”. In: *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, pp. 5231–5237.

Runz, Martin, Maud Buffier, and Lourdes Agapito (2018). “Maskfusion: Real-time recognition, tracking and reconstruction of multiple moving objects”. In: *2018 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*. IEEE, pp. 10–20.

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

Sucar, Edgar, Kentaro Wada, and Andrew Davison (2020). "Neural Object Descriptors for Multi-View Shape Reconstruction". In: *arXiv preprint arXiv:2004.04485*.

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

Li, Kejie, Trung Pham, Huangying Zhan, and Ian Reid (2018). "Efficient Dense Point Cloud Object Reconstruction Using Deformation Vector Fields". In: *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 497–513.

Braunstein, Myron L, Jeffrey C Liter, and James S Tittle (1993). "Recovering three-dimensional shape from perspective translations and orthographic rotations." In: *Journal of Experimental Psychology: Human Perception and Performance* 19.3, p. 598.

Aloimonos, John (1988). "Shape from texture". In: *Biological cybernetics* 58.5, pp. 345–360.

Prados, Emmanuel and Olivier Faugeras (2006). "Shape from shading". In: *Handbook of mathematical models in computer vision*. Springer, pp. 375–388.

Saxena, Ashutosh, Min Sun, and Andrew Y Ng (2008). "Make3D: Depth Perception from a Single Still Image." In: *AAAI*, pp. 1571–1576.

Liu, Miaomiao, Mathieu Salzmann, and Xuming He (2014). "Discrete-continuous depth estimation from a single image". In: *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on*. IEEE, pp. 716–723.

Liu, Fayao, Chunhua Shen, and Guosheng Lin (2015). "Deep convolutional neural fields for depth estimation from a single image". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 5162–5170.

Godard, Clément, Oisin Mac Aodha, and Gabriel J Brostow (2017). "Unsupervised monocular depth estimation with left-right consistency". In: *CVPR*. Vol. 2. 6, p. 7.

Zhan, Huangying, Ravi Garg, Chamara Saroj Weerasekera, Kejie Li, Harsh Agarwal, and Ian Reid (2018). "Unsupervised Learning of Monocular Depth Estimation and Visual Odometry with Deep Feature Reconstruction". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 340–349.

Sinha, Ayan, Asim Unmesh, Qixing Huang, and Karthik Ramani (2017). "SurfNet: Generating 3D shape surfaces using deep residual networks". In: *Proc. CVPR*.

Kong, Chen, Chen-Hsuan Lin, and Simon Lucey (2017). “Using locally corresponding cad models for dense 3d reconstructions from a single image”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. Vol. 2.

Fan, Haoqiang, Hao Su, and Leonidas Guibas (2016). “A point set generation network for 3d object reconstruction from a single image”. In: *arXiv preprint arXiv:1612.00603*.

Tatarchenko, Maxim, Alexey Dosovitskiy, and Thomas Brox (2016). “Multi-view 3d models from single images with a convolutional network”. In: *European Conference on Computer Vision*. Springer, pp. 322–337.

Martin, Worthy N and Jagdishkumar Keshoram Aggarwal (1983). “Volumetric descriptions of objects from multiple views”. In: *IEEE transactions on pattern analysis and machine intelligence* 2, pp. 150–158.

Kar, Abhishek, Shubham Tulsiani, Joao Carreira, and Jitendra Malik (2015). “Category-specific object reconstruction from a single image”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1966–1974.

Huang, Qixing, Hai Wang, and Vladlen Koltun (2015). “Single-view reconstruction via joint analysis of image and shape collections”. In: *ACM Transactions on Graphics (TOG)* 34.4, p. 87.

Kurenkov, Andrey, Jingwei Ji, Animesh Garg, Viraj Mehta, JunYoung Gwak, Christopher Choy, and Silvio Savarese (2017). “DeformNet: Free-Form Deformation Network for 3D Shape Reconstruction from a Single Image”. In: *arXiv preprint arXiv:1708.04672*.

Häne, Christian, Shubham Tulsiani, and Jitendra Malik (2017). “Hierarchical surface prediction for 3d object reconstruction”. In: *arXiv preprint arXiv:1704.00710*.

Mescheder, Lars, Michael Oechsle, Michael Niemeyer, Sebastian Nowozin, and Andreas Geiger (2019b). “Occupancy Networks: Learning 3D Reconstruction in Function Space”. In: *Proceedings IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*.

Borgefors, Gunilla (1986). “Distance transformations in digital images”. In: *Computer vision, graphics, and image processing* 34.3, pp. 344–371.

Jaderberg, Max, Karen Simonyan, Andrew Zisserman, et al. (2015). “Spatial transformer networks”. In: *Advances in neural information processing systems*, pp. 2017–2025.

Martın Abadi et al. (2015). *TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems*. Software available from tensorflow.org. URL: https://www.tensorflow.org/.

Kingma, Diederik P and Jimmy Ba (2014). “Adam: A method for stochastic optimization”. In: *arXiv preprint arXiv:1412.6980*.

Sun and *et al* (2018). “Pix3D: Dataset and Methods for Single-Image 3D Shape Modeling”. In: *CVPR*.

Xiao, Jianxiong, Krista A Ehinger, James Hays, Antonio Torralba, and Aude Oliva (2016). “Sun database: Exploring a large collection of scene categories”. In: *International Journal of Computer Vision* 119.1, pp. 3–22.

Li, Kejie, Ravi Garg, Ming Cai, and Ian Reid (2019). “Single-view Object Shape Reconstruction Using Deep Shape Prior and Silhouette”. In: *30th British Machine Vision Conference 2019, Cardiff, UK*. BMVA Press, p. 163.

Fan, Haoqiang, Hao Su, and Leonidas J Guibas (2017b). “A Point Set Generation Network for 3D Object Reconstruction from a Single Image.” In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. Vol. 2. 4, p. 6.

Prisacariu and Ian Reid (2011). “Shared shape spaces”. In: *Proceedings of the 2011 International Conference on Computer Vision*. IEEE Computer Society, pp. 2587–2594.

Bao, Yingze, M. Chandraker, Y. Lin, and S. Savarese (June 2013b). “Dense Object Reconstruction with Semantic Priors”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1264–1271.

Engelmann, Francis, Jörg Stückler, and Bastian Leibe (2016). “Joint object pose estimation and shape reconstruction in urban street scenes using 3D shape priors”. In: *German Conference on Pattern Recognition*. Springer, pp. 219–230.

Hinton, Geoffrey E and Ruslan R Salakhutdinov (2006). “Reducing the dimensionality of data with neural networks”. In: *science* 313.5786, pp. 504–507.

Vincent, Pascal, Hugo Larochelle, Yoshua Bengio, and Pierre-Antoine Manzagol (2008). “Extracting and composing robust features with denoising autoencoders”. In: *Proceedings of the 25th international conference on Machine learning*. ACM, pp. 1096–1103.

Kingma, Diederik P and Max Welling (2013). “Auto-encoding variational bayes”. In: *arXiv preprint arXiv:1312.6114*.

Wu, Jiajun, Chengkai Zhang, Tianfan Xue, Bill Freeman, and Josh Tenenbaum (2016a). “Learning a probabilistic latent space of object shapes via 3d generative-adversarial modeling”. In: *Advances in Neural Information Processing Systems*, pp. 82–90.

Gwak, JunYoung, Christopher B Choy, Manmohan Chandraker, Animesh Garg, and Silvio Savarese (2017). *Weakly supervised 3D Reconstruction with Adversarial Constraint*.

Wu, Jiajun, Yifan Wang, Tianfan Xue, Xingyuan Sun, Bill Freeman, and Josh Tenenbaum (2017a). "Marrnet: 3d shape reconstruction via 2.5 d sketches". In: *Advances in neural information processing systems*, pp. 540–550.

Wu, Jiajun, Tianfan Xue, Joseph J Lim, Yuandong Tian, Joshua B Tenenbaum, Antonio Torralba, and William T Freeman (2018). "3D Interpreter Networks for Viewer-Centered Wireframe Modeling". In: *International Journal of Computer Vision*, pp. 1–18.

Zhu, Rui, Chaoyang Wang, Chen-Hsuan Lin, Ziyan Wang, and Simon Lucey (2018). "Object-centric photometric bundle adjustment with deep shape prior". In: *IEEE Winter Conference on Applications of Computer Vision (WACV)*. IEEE, pp. 894–902.

Achlioptas, Panos, Olga Diamanti, Ioannis Mitliagkas, and Leonidas Guibas (2018). "Learning Representations and Generative Models for 3D Point Clouds". In:

Moré, Jorge J (1978). "The Levenberg-Marquardt algorithm: implementation and theory". In: *Numerical analysis*. Springer, pp. 105–116.

Chen, Liang-Chieh, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille (2018). "Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs". In: *IEEE transactions on pattern analysis and machine intelligence* 40.4, pp. 834–848.

Sun, Xingyuan, Jiajun Wu, Xiuming Zhang, Zhoutong Zhang, Chengkai Zhang, Tianfan Xue, Joshua B Tenenbaum, and William T Freeman (2018). "Pix3D: Dataset and Methods for Single-Image 3D Shape Modeling". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2974–2983.

Xiang, Yu, Roozbeh Mottaghi, and Silvio Savarese (2014). "Beyond pascal: A benchmark for 3d object detection in the wild". In: *Applications of Computer Vision (WACV), 2014 IEEE Winter Conference on*. IEEE, pp. 75–82.

Runz, Martin, Kejie Li, Meng Tang, Lingni Ma, Chen Kong, Tanner Schmidt, Ian Reid, Lourdes Agapito, Julian Straub, Steven Lovegrove, et al. (2020). "FroDO: From Detections to 3D Objects". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 14720–14729.

Choi, Sungjoon, Qian-Yi Zhou, Stephen Miller, and Vladlen Koltun (2016). "A Large Dataset of Object Scans". In: *arXiv:1602.02481*.

Choy, Christopher B, Danfei Xu, JunYoung Gwak, Kevin Chen, and Silvio Savarese (2016b). "3d-r2n2: A unified approach for single and multi-view 3d object reconstruction". In: *European conference on computer vision*. Springer, pp. 628–644.

Wu, Jiajun, Chengkai Zhang, Tianfan Xue, Bill Freeman, and Josh Tenenbaum (2016b). "Learning a probabilistic latent space of object shapes via 3d generative-adversarial modeling”. In: *Advances in neural information processing systems*, pp. 82–90.

Xie, Haozhe, Hongxun Yao, Xiaoshuai Sun, Shangchen Zhou, Shengping Zhang, and Xiaojun Tong (2019). “Pix2Vox: Context-aware 3D Reconstruction from Single and Multi-view Images”. In: *arXiv preprint arXiv:1901.11153*.

Wang, Rui, Nan Yang, Joerg Stueckler, and Daniel Cremers (2019a). “DirectShape: Photometric Alignment of Shape Priors for Visual Vehicle Pose and Shape Estimation”. In: *arXiv preprint arXiv:1904.10097*.

Prisacariu, Victor Adrian, Aleksandr V Segal, and Ian Reid (2012). “Simultaneous monocular 2d segmentation, 3d pose recovery and 3d reconstruction”. In: *Asian conference on computer vision*. Springer, pp. 593–606.

Lin, Chen-Hsuan, Oliver Wang, Bryan C Russell, Eli Shechtman, Vladimir G Kim, Matthew Fisher, and Simon Lucey (2019a). “Photometric Mesh Optimization for Video-Aligned 3D Object Reconstruction”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 969–978.

Bloesch, Michael, Jan Czarnowski, Ronald Clark, Stefan Leutenegger, and Andrew J Davison (2018b). “CodeSLAM—learning a compact, optimisable representation for dense visual SLAM”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2560–2568.

Sitzmann, Vincent, Michael Zollhöfer, and Gordon Wetzstein (2019). “Scene Representation Networks: Continuous 3D-Structure-Aware Neural Scene Representations”. In: *arXiv preprint arXiv:1906.01618*.

Choudhary, Siddharth, Alexander J. B. Trevor, Henrik I. Christensen, and Frank Dellaert (2014). “SLAM with object discovery, modeling and mapping”. In: *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 1018–1025.

McCormac, John, Ronald Clark, Michael Bloesch, Andrew Davison, and Stefan Leutenegger (2018b). “Fusion++: Volumetric object-level slam”. In: *2018 International Conference on 3D Vision (3DV)*. IEEE, pp. 32–41.

Parkhiya, Parv, Rishabh Khawad, J Krishna Murthy, Brojeshwar Bhowmick, and K Madhava Krishna (2018). “Constructing Category-Specific Models for Monocular Object-SLAM”. In: *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, pp. 1–9.

Gálvez-López, Dorian, Marta Salas, Juan D. Tardós, and J. M. M. Montiel (2015). “Real-time Monocular Object SLAM”. In: *Robotics Auton. Syst.* 75, pp. 435–449.

Fei, Xiaohan and Stefano Soatto (2018). “Visual-inertial object detection and mapping”. In: *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 301–317.

Yang, Shichao and Sebastian A. Scherer (2019b). “CubeSLAM: Monocular 3-D Object SLAM”. In: *IEEE Transactions on Robotics* 35, pp. 925–938.

Mur-Artal, Raul, Jose Maria Martinez Montiel, and Juan D Tardos (2015). “ORB-SLAM: a versatile and accurate monocular SLAM system”. In: *TRO* 31.5, pp. 1147–1163.

Park, Jeong Joon, Peter Florence, Julian Straub, Richard Newcombe, and Steven Lovegrove (2019b). “Deepsdf: Learning continuous signed distance functions for shape representation”. In: *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR).*

Muralikrishnan, Sanjeev, Vladimir G. Kim, Matthew Fisher, and Siddhartha Chaudhuri (June 2019). “Shape Unicode: A Unified Shape Representation”. In: *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR).*

Kulis, Brian and Michael I Jordan (2011). “Revisiting k-means: New algorithms via Bayesian nonparametrics”. In: *arXiv preprint arXiv:1111.0352.*

Prisacariu, Victor A and Ian D Reid (2012). “PWP3D: Real-time segmentation and tracking of 3D objects”. In: *International journal of computer vision* 98.3, pp. 335–354.

Li, Yangyan, Hao Su, Charles Ruizhongtai Qi, Noa Fish, Daniel Cohen-Or, and Leonidas J Guibas (2015). “Joint embeddings of shapes and images via cnn image purification”. In: *ACM transactions on graphics (TOG)* 34.6, p. 234.

Tulsiani, Shubham, Tinghui Zhou, Alexei A Efros, and Jitendra Malik (2017b). “Multi-view supervision for single-view reconstruction via differentiable ray consistency”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2626–2634.

Wu, Jiajun, Yifan Wang, Tianfan Xue, Xingyuan Sun, Bill Freeman, and Josh Tenenbaum (2017b). “Marrnet: 3d shape reconstruction via 2.5 d sketches”. In: *Advances in neural information processing systems*, pp. 540–550.

Groueix, Thibault, Matthew Fisher, Vladimir G. Kim, Bryan Russell, and Mathieu Aubry (2018). “AtlasNet: A Papier-Mâché Approach to Learning 3D Surface Generation”. In: *Proceedings IEEE Conf. on Computer Vision and Pattern Recognition (CVPR).*

Schönberger, Johannes Lutz and Jan-Michael Frahm (2016a). “Structure-from-Motion Revisited”. In: *Conference on Computer Vision and Pattern Recognition (CVPR).*

Schönberger, Johannes Lutz, Enliang Zheng, Marc Pollefeys, and Jan-Michael Frahm (2016b). “Pixelwise View Selection for Unstructured Multi-View Stereo”. In: *European Conference on Computer Vision (ECCV).*

Weng, Xinshuo, Jianren Wang, David Held, and Kris Kitani (2020). “AB3DMOT: A Baseline for 3D Multi-Object Tracking and New Evaluation Metrics”. In: *arXiv preprint arXiv:2008.08063.*

Shenoi, Abhijeet, Mihir Patel, JunYoung Gwak, Patrick Goebel, Amir Sadeghian, Hamid Rezatofighi, Roberto Martin-Martin, and Silvio Savarese (2020). “JR-MOT: A Real-Time 3D Multi-Object Tracker and a New Large-Scale Dataset”. In: *arXiv preprint arXiv:2002.08397.*

Luiten, Jonathon, Tobias Fischer, and Bastian Leibe (2020). “Track to reconstruct and reconstruct to track”. In: *IEEE Robotics and Automation Letters* 5.2, pp. 1803–1810.

Stückler, Jörg and Sven Behnke (2014). “Multi-resolution surfel maps for efficient dense 3D modeling and tracking”. In: *Journal of Visual Communication and Image Representation* 25.1, pp. 137–147.

Gálvez-López, Dorian, Marta Salas, Juan D Tardós, and JMM Montiel (2016). “Real-time monocular object slam”. In: *Robotics and Autonomous Systems* 75, pp. 435–449.

Dambreville, Samuel, Romeil Sandhu, Anthony Yezzi, and Allen Tannenbaum (2008b). “Robust 3d pose estimation and efficient 2d region-based segmentation from a 3d shape prior”. In: *European Conference on Computer Vision.* Springer, pp. 169–182.

Wang, Rui, Nan Yang, Joerg Stueckler, and Daniel Cremers (2019b). “Directshape: Photometric alignment of shape priors for visual vehicle pose and shape estimation”. In: *arxiv*, arXiv–1904.

Lin, Chen-Hsuan, Oliver Wang, Bryan C Russell, Eli Shechtman, Vladimir G Kim, Matthew Fisher, and Simon Lucey (2019b). “Photometric mesh optimization for video-aligned 3d object reconstruction”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 969–978.

Rünz, Martin and Lourdes Agapito (2017). “Co-fusion: Real-time segmentation, tracking and fusion of multiple objects”. In: *2017 IEEE International Conference on Robotics and Automation (ICRA).* IEEE, pp. 4471–4478.

Bârsan, Ioan Andrei, Peidong Liu, Marc Pollefeys, and Andreas Geiger (2018). “Robust dense mapping for large-scale dynamic environments”. In: *2018 IEEE International Conference on Robotics and Automation (ICRA).* IEEE, pp. 7510–7517.

Rezatofighi, Hamid, Nathan Tsoi, JunYoung Gwak, Amir Sadeghian, Ian Reid, and Silvio Savarese (2019). “Generalized intersection over union: A metric and a loss for bounding box regression”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 658–666.

Munkres, James (1957). “Algorithms for the assignment and transportation problems”. In: *Journal of the society for industrial and applied mathematics* 5.1, pp. 32–38.

Blom, Henk AP and Yaakov Bar-Shalom (1988). “The interacting multiple model algorithm for systems with Markovian switching coefficients”. In: *IEEE transactions on Automatic Control* 33.8, pp. 780–783.

Lorensen, William E and Harvey E Cline (1987). “Marching cubes: A high resolution 3D surface construction algorithm”. In: *ACM siggraph computer graphics* 21.4, pp. 163–169.

Avetisyan, Armen, Manuel Dahnert, Angela Dai, Manolis Savva, Angel X. Chang, and Matthias Niessner (June 2019). “Scan2CAD: Learning CAD Model Alignment in RGB-D Scans”. In: *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR).*

Zhou, Xingyi, Vladlen Koltun, and Philipp Krähenbühl (2020). “Tracking Objects as Points”. In: *arXiv preprint arXiv:2004.01177.*

Zhan, Huangying, Chamara Saroj Weerasekera, Jia-Wang Bian, and Ian Reid (2020). “Visual odometry revisited: What should be learnt?” In: *2020 IEEE International Conference on Robotics and Automation (ICRA).* IEEE, pp. 4203–4210.

Geiger, Andreas, Philip Lenz, and Raquel Urtasun (2012). “Are we ready for Autonomous Driving? The KITTI Vision Benchmark Suite”. In: *Conference on Computer Vision and Pattern Recognition (CVPR).*

Geiger, Andreas, Julius Ziegler, and Christoph Stiller (June 2011). “StereoScan: Dense 3d Reconstruction in Real-time”. In: *IEEE Intelligent Vehicles Symposium.* Baden-Baden, Germany.

Lin, Chen-Hsuan, Chen Kong, and Simon Lucey (2018b). “Learning Efficient Point Cloud Generation for Dense 3D Object Reconstruction”. In: *AAAI Conference on Artificial Intelligence.*

Brasó, Guillem and Laura Leal-Taixé (2020). “Learning a neural solver for multiple object tracking”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 6247–6257.

Brown, Tom B, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. (2020). “Language models are few-shot learners”. In: *arXiv preprint arXiv:2005.14165.*

Kocabas, Muhammed, Nikos Athanasiou, and Michael J Black (2020). “Vibe: Video inference for human body pose and shape estimation”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 5253–5263.

Do, Thanh-Toan, Anh Nguyen, and Ian Reid (2018). “Affordancenet: An end-to-end deep learning approach for object affordance detection”. In: *2018 IEEE international conference on robotics and automation (ICRA)*. IEEE, pp. 5882–5889.

Purkait, Pulak, Christopher Zach, and Ian Reid (2020). “SG-VAE: Scene Grammar Variational Autoencoder to generate new indoor scenes”. In: *European Conference on Computer Vision*. Springer, pp. 155–171.