Title
Inertial-aided Visual Perception of Geometry and Semantics

Author
Fei, Xiaohan

Publication Date
2019

Peer reviewed|Thesis/dissertation

UNIVERSITY OF CALIFORNIA

Los Angeles

Inertial-aided Visual Perception of Geometry and Semantics

A dissertation submitted in partial satisfaction

of the requirements for the degree

Doctor of Philosophy in Computer Science

by

Xiaohan Fei

2019

ABSTRACT OF THE DISSERTATION

Inertial-aided Visual Perception of Geometry and Semantics

by

Xiaohan Fei
Doctor of Philosophy in Computer Science
University of California, Los Angeles, 2019
Stefano Soatto, Chair

We describe components of a visual perception system to understand the geometry and semantics of the three-dimensional scene by utilizing monocular cameras and inertial measurement units (IMUs). The use of the two sensor modalities is motivated by the wide availability of the camera-IMU sensor packages present in mobile devices from phones to cars, and their complementary sensing capabilities: IMUs can track the motion of the sensor platform over a short period of time accurately, and provide a *scaled* and *gravity-aligned* global reference frame, while cameras can capture rich photometric signatures of the scene, and provide relative motion constraints between images up to scale.

We first show that visual 3D reconstruction can be improved by leveraging the global orientation frame – easily inferred from inertials. In the gravity-aligned global orientation frame, a shape prior can be imposed in depth prediction from a single image, where the normal vectors to surfaces of objects of certain classes tend to align with gravity or orthogonal to it. Adding such a prior to baseline methods for monocular depth prediction yields improvements beyond the state-of-the-art and illustrates the power of utilizing inertials in 3D reconstruction.

The global reference provided by inertials is not only gravity-aligned but also scaled, which is exploited in depth completion: We describe a method to infer dense metric depth from camera motion and sparse depth as estimated using a visual-inertial odometry system. Unlike other scenarios using point clouds from lidar or structured light sensors, we have

few hundreds to few thousand points, insufficient to inform the topology of the scene. Our method first constructs a piecewise planar scaffolding of the scene, and then uses it to infer dense depth using the image along with the sparse points. We use a predictive cross-modal criterion, akin to self-supervision, measuring photometric consistency across time, forward-backward pose consistency, and geometric compatibility with the sparse point cloud. We also launch the first visual-inertial + depth dataset (dubbed "VOID"), which we hope will foster additional exploration into combining the complementary strengths of visual and inertial sensors. To compare our method to prior work, we adopt the unsupervised KITTI depth completion benchmark, and show state-of-the-art performance on it.

In addition to dense geometry, the camera-IMU sensor package can also be used to recover the semantics of the scene. We present two methods to augment a point cloud map with class-labeled objects represented in the form of either scaled and oriented bounding boxes or CAD models. The tradeoff of the two shape representation resides in their generality and capability to model detailed structures. While being more generic, 3D bounding boxes fail to model the details of the objects, whereas CAD models preserve the finest shape details but require more computation and are limited to previously seen objects. Nevertheless, both methods populate an unknown environment with 3D objects placed in a Euclidean reference frame inferred causally and on-line using monocular video along with inertial sensors. Besides, both methods include bottom-up and top-down components, whereby deep networks trained for detection provide likelihood scores for object hypotheses provided by a nonlinear filter, whose state serves as memory. We test our methods on KITTI and SceneNN datasets, and also introduce the VISMA dataset, which contains ground truth pose, point-cloud map, and object models, along with time-stamped inertial measurements.

To reduce the drift of the visual-inertial SLAM system – a building block of all the visual perception systems we have built, we introduce an efficient loop closure detection approach based on the idea of hierarchical pooling of image descriptors. We also open-sourced a full-fledged SLAM system equipped with mapping and loop closure capabilities. The code is publicly available at https://github.com/ucla-vision/xivo.

The dissertation of Xiaohan Fei is approved.

Ying Nian Wu

Yizhou Sun

Quanquan Gu

Stefano Soatto, Committee Chair

University of California, Los Angeles

2019

To my parents.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGMENTS

I am grateful for my advisor Stefano Soatto who not only intrigued my interests in geometric computer vision, probabilistic inference, robotics, but also scientific and engineering subjects beyond computer vision. I remember so many invaluable discussions, lessons, and advice from my advisor during the past five years – I took the advice to heart and will consult them from time to time in my whole career. One of the more important lessons I've learned from Stefano is always to approach a problem from first principles, and seek a thorough understanding of the problem. This is sometimes a hard practice, yet it is probably *the only way* to address computer vision or any other scientific problem.

Many thanks go to the members of my committee, Prof. Ying Nian Wu, Prof. Yizhou Sun, and Prof. Quanquan Gu for their support and suggestions.

I am very fortunate to have worked with many exceptional individuals at UCLA Vision Lab. I want to give special thanks to Konstantine Tsotsos and Jingming Dong, who are both my mentors and collaborators, and Alex Wong, with whom I explored the potential of deep learning in multi-sensor settings.

Konstantine Tsotsos mentored me when I started graduate school at UCLA. It is Konstantine who led me to the world of visual-inertial odometry, and showed me how to write great C++ code. We had great times working together on various projects, including my very first live demo at SCR (Southern California Robotics Symposium), 2016, my first ECCV paper on efficient loop closure detection, and later on the semantic mapping live demo at CVPR 2016, Las Vegas.

Jingming Dong mentored me in summer 2013 for my undergraduate research internship at UCLA Vision Lab, which started my journey of computer vision. Back then, I worked on a real-time experimental platform to facilitate the design of multi-view feature descriptors. After I joined the lab, we also worked together on the visual-inertial object detection and mapping project, which was one of the most precious experiences I've ever had. Together, we had a great real-time live demo on semantic mapping with Konstantine Tsotsos and Nikolaos Karianakis at CVPR 2016, Las Vegas and later a paper accepted by CVPR 2017.

I also want to thank my great collaborator Alex Wong. It is Alex who initiated my interests in learning-based depth inference, which, combined with my passion in visual-inertial sensor fusion, led to several papers on learning-based sensor fusion, one of which won the *best paper award in robot vision* at ICRA 2019, Montreal, Canada. When I couldn't make the trip to Montreal, Alex took the challenge to give the talk at ICRA and finally won the award. Without him, this would not happen.

In addition to my collaborators, I also want to thank my colleagues at UCLA Vision Lab, who support me a lot during my journey. I want to first thank Nikolaos Kariannakis for his advice leading me to the field of deep learning, working together on the semantic mapping project, and all the encouragement and help over the course. Vasiliy Karasev and Brian Taylor proofread many of my conference submissions and constantly gave me life and work advice. Joshua Hernandez helped a lot when I interned at Meta. I want to thank Safa Cicek for enriching my knowledge about planning, semi-supervised learning, and many other interesting topics he is working on. I want to thank Yanchao Yang, Virginia Estellers, Pratik Chaudhari, Shay Deutsch, Georgios Georgiadis, Alessandro Achille, Simon Korman, and Alhussein Fawzi for many valuable discussions and lessons on variational optimization, deep learning, graph-based algorithms, etc. I want to thank Peng Zhao, Tong He, Albert Zhao, Stephanie Tsuei, Xinzhu Bei, and Alexandre Tiard for their efforts on helping some of my research projects. I also want to thank our visitors: Weizhe (Jason) Liu, Kareem Ahmed, Isaac Deutsch, and Matteo Terzi for discussions, and Antonio Loquercio for proofreading my ICRA submission.

Finally, I want to acknowledge the unconditional support of my parents, to whom the thesis is dedicated.

VITA

2010–2014 B.Eng. in Information Science and Electronic Engineering, Zhejiang University

2014–present Ph.D. student, Computer Science Department, UCLA

PUBLICATIONS

1. A. Wong*, **X. Fei***, and S. Soatto. *VOICED: Depth Completion from Inertial Odometry and Vision*. UCLA Technical Report #190001.

2. **X. Fei**, A. Wong, and S. Soatto. *Geo-Supervised Visual Depth Prediction*. In Proceedings of International Conference on Robotics and Automation (ICRA), May 2019. (**Best Paper Award in Robot Vision**) Also in Robotics and Automation Letters.

3. **X. Fei** and S. Soatto. *Visual-Inertial Object Detection and Mapping*. In Proceedings of the European Conference on Computer Vision (ECCV), September 2018.

4. J. Dong*, **X. Fei***, and S. Soatto. *Visual-Inertial Scene Representation for 3D Object Detection*. In Proceedings of Computer Vision and Pattern Recognition (CVPR), July 2017.

5. **X. Fei**, K. Tsotsos, S. Soatto. *A Simple Hierarchical Pooling Data Structure for Loop Closure*. In Proceedings of the European Conference on Computer Vision (ECCV), October 2016.

* Equal contributors.

CHAPTER 1

Introduction

> *What does it mean, to see? The plain man's answer would be, to know what is where by looking. In other words, vision is the process of discovering from images what is present in the world, and where it is.*
>
> – David Marr, *Vision: A Computational Investigation into the Human Representation and Processing of Visual Information*, 1982

We as human beings heavily rely on visual processing to survive – in fact, nearly two-thirds of the human brain is involved in processing visual information. As such, it is not surprising that researchers work very hard to understand how intelligence works by studying vision, and vice versa to build visual perception systems to mimic intelligence. In this thesis, we will focus on the construction of several artificial visual perception systems my colleagues and I have built over the past few years.

The various systems we built are designed to produce a wide spectrum of representations suitable for different tasks. The visual-inertial odometry and loop closure systems we built (Chapter 6) produce 6 degrees-of-freedom (6 DoF) poses of the sensor platform and a sparse point-cloud reconstruction of the environment aimed to address the localization problem, or in other words, to know where we are. The depth prediction and completion systems (Chapters 2 and 3) produce range maps as the scene representation to enable autonomous navigation and exploration. The object detection and mapping systems (Chapters 4 and 5) generate object-level maps (or semantic maps) of the surrounding environment for augmented reality (AR), virtual reality (VR), and robotic manipulation tasks.

While these systems produce all kinds of different representations, they share several

things in common: 1) the systems are built in such a way to complete the perception-control loop with the ultimate goal of achieving full autonomy bore in mind, and as such 2) we focus on producing representations in three-dimensional (3D) spaces, in which most intelligent agents reside 3) *without using range sensors* (*e.g.,* lidar, radar, and RGB-D). Instead, we attempted to do so with only monocular cameras and inertial measurement units (IMUs) – a minimal setup to achieve metric space visual perception. It is well known that a moving monocular camera can produce ego-motion estimation and 3D reconstruction up to an unknown scale at best, the IMU, on the other hand, measures the linear acceleration and rotational velocity of the sensor platform and thus renders the metric scale observable. Besides, monocular cameras and IMUs are very low-cost, and quite ubiquitious – available on almost all smartphones, tablets, modern cars, drones, and robots, etc.

From the perspective of using multiple sensor-modalities, the systems present in this thesis can be seen as sensor-fusion systems in general. The fusion of visual and inertial information is not a new topic. In fact, there is a vast amount of visual-inertial sensor fusion for localization ([MR07, Jon09, TCS15] and references therein) – also known as visual-inertial odometry (VIO). However, the systems present in this work are not restricted to the localization problem – we focus on a much wider spectrum of applications including 1) localization of the sensor platform, 2) dense depth inference, and 3) detection and mapping objects in the scene. In all these applications, the scaled and gravity-aligned global reference frame inferred from the two sensors is leveraged, where dense metric depth of the scene is inferred, and class-labeled objects are identified and placed correctly in the scene to enable high-level tasks further.

We adopt the hypothesis-testing inference framework in most of the systems we built (e.g., SLAM, semantic mapping), which produces a posterior distribution rather than a point estimate of the quantity of interests. For instance, in the semantic mapping system, we invent the semantic filter – a variant of the bootstrap particle filter– to estimate the posterior of the semantic label and the pose of objects in the scene – totaling a five-dimensional random variable. Strong assumptions and simplifications are made to make the inference computationally tractable. However, it is not always easy to make such assumptions/simplifications.

On the other hand, deep neural networks as generic modeling tools perform quite well in lots of application domains, though most neural networks only produce a point estimate – leaving out the characterization of the uncertainty. To this end, we leverage the great representation power of deep neural networks and attempt to obtain prediction/point estimate of the dense scene geometry, which can be used at least as a prior or intermediate representation for other tasks.

1.1 Organization of the thesis

In Chapter 2, we proposed a system to exploit gravity, which can be accurately and robustly inferred from inertial measurements[1], as a prior in learning-based reconstruction. To reconstruct the three-dimensional scene from a set of two-dimensional images is an ill-posed problem, especially for texture-less image regions where reliable correspondences across multiple images cannot be easily established. The conventional way to handle this is to introduce some priors, or regularizers, to the modeling process, such as piece-wise smoothness, which has been commonly used in the past several decades. Recent works in self-supervised depth prediction networks adopt the same strategy where: A piece-wise smoothness term (regularizer) and a photometric error term (data term) are minimized during training. To improve the quality of the reconstruction, we introduced two gravity-induced regularizers: One to penalize the deviation of the surface normal from the direction of gravity for horizontal planes such as roads, sidewalks, and countertops, etc.; the other to penalize the non-orthogonality of surface normals to the direction of gravity for vertical planes such as walls, billboards, and buildings, etc. The application of the regularizer is conditioned on the semantic meaning of the image regions, in other words, the regularizers are selectively applied. We experimented our proposed regularizers on both indoor and outdoor datasets, and observe systematic performance improvement over a wide spectrum of top-performing baseline models.

In Chapter 3, we extend the idea of using inertial measurements in image-based recon-

[1]For a platform standing still, it is fairly easy to obtain the direction of gravity which is the dominant component of the reading of the accelerator. For moving platforms equipped with both cameras and IMUs, the direction of gravity is usually inferred as part of the state of the VIO system.

struction even further. While in Chapter 2, inertials are only used to infer the direction of gravity which further induces the category-specific regularizers, in Chapter 3, inertials are tightly coupled with monocular videos to infer camera motion and sparse depth – both in metric scale. The sparse depth estimates are then fused with the raw images to produce dense depth maps. This procedure of fusing sparse depths with raw images to produce dense depths is known as *depth completion* and is essentially a sensor fusion problem. In contrast to the traditional sensor fusion framework where filtering techniques are deployed, we develop a novel learning pipeline in this chapter as the "fuser". New techniques inspired by geometric intuitions are introduced to reduce the model complexity while acheving state-of-the-art performance.

Chapters 4 and 5 are dedicated to object-level mapping. In Chapter 4, we develop a system to detect objects in 3D. Different from most existing object detection systems which detect objects using a single image and output a set of 2D bounding boxes on the image plane, our system is able to reason both intrinsic (identity and scale) and extrinsic (6 DoF pose) attributes of objects in Euclidean space. We argue that our system is more useful for robotic tasks such as manipulation, since to do so one needs to know the location, orientation and spatial extent of the objects in the scene as only knowing 2D bounding boxes of objects on the image plane is insufficient. The idea behind this system is: (a) leverage on state-of-the-art visual-inertial navigation and CNN-based object detection algorithms; (b) formulate the problem of object reasoning in a hypothesis-testing framework; (c) use CNN in both bottom-up data-driven proposal generation and top-down hypothesis validation procedures to make inference efficient. Both quantitative and qualitative evaluations are provided.

Chapter 5 presents a different implementation of the object-level mapping system, where CAD models instead of scaled and oriented 3D bounding boxes are used to model the shape of the objects–trading off computational complexity with reconstruction details. With the new modeling assumptions, a bootstrap particle filter is introduced as the new inference machinery to fuse a 1) object likelihood term provided by an object detection network and a 2) edge likelihood term which is inspired by classic model-based tracking literature. The benefits include: 1) better localization of the objects, 2) more accurate shape modeling,

and 3) fine-grained occlusion inference. However, the modeling power comes at the price of generality: Only previously seen objects of which CAD models are available can be inferred by the system.

In Chapter 6, we tackle the problem of loop closure detection in vision-based navigation. Without loop closure, a vision-based navigation system suffers from drifting – as a probabilistic graph has been built incrementally over the course, there are no close-loop contraints to relate the current state of the agent to its memory (the map has been built in the past). The focus of Chapter 6 is to develop a hierarchical data structure to detect loop closure constraints efficiently with minimum or zero loss of performance. Comprehensive evaluation and extension to the more general image retrieval tasks are also provided.

Chapter 7 discusses the limitations of the systems present in the thesis and some possible improvements. Also, some interesting topics related to the thesis are briefly covered as potential pointers to my future research and development in the domain of robot vision, machine learning, and more specifically, multi-sensor fusion.

CHAPTER 2

Geo-Supervised Visual Depth Prediction

2.1 Introduction

The visual world is heavily affected by gravity, including the shape of many artifacts such as buildings and roads, and even natural objects such as trees. Gravity provides a globally consistent orientation reference that can be reliably measured with low-cost inertial sensors present in mobile devices from phones to cars. We call a machine learning system able to exploit global orientation, *geo-supervised*. Gravity can be easily inferred from inertial sensors without the need for dead-reckoning, and the effect of biases is negligible in the context of our application.

To measure the influence of gravity as a supervisory signal, we choose the extreme example of predicting depth from a single image. This is, literally, an impossible task in the sense that there are infinitely many three-dimensional (3D) scenes that can generate the same image. So, any process that yields a point estimate has to rely heavily on priors. We call the resulting point estimate a *hypothesis*, or *prediction*, and use public benchmark datasets to quantitatively evaluate the improvement brought about by exploiting gravity. Of course, only certain objects have a shape that is influenced by gravity. Therefore, our prior has to be applied *selectively*, in a manner that is informed by the semantics of the scene.

Our approach to geo-supervised Visual Depth Prediction is based on training a system end-to-end to produce a map from a single image and an estimate of the orientation of gravity in the (calibrated) camera frame to an inverse depth (disparity) map. In one mode of operation, the training set uses calibrated and rectified stereo pairs, together with a semantic segmentation module, to evaluate a loss function differentially on the images where

geo-referenced objects are present. In a second mode, we use monocular videos instead and minimize the reprojection (prediction) error. Optionally, we can leverage modern visual-inertial odometry (VIO) and mapping systems that are becoming ubiquitous from hand-held devices to cars.

The key to our approach is a prior, or regularizer, that selectively biases certain regions of the image that correspond to geo-referenced classes such as roads, buildings, vehicles, and trees. Specifically, points in space that lie on the surface of such objects should have normals that either align with, or are orthogonal to, gravity. This is in addition to standard regularizers used for depth prediction, such as left-right consistency and piecewise smoothness.

While at training time a semantic segmentation map is needed to apply our prior selectively, it is never passed as input to the network. Therefore, at test time it is not needed, and an image is simply mapped to the disparity.

The ultimate test for a prior is whether it helps improve end-performance. To test our prior, we first incorporated it into two top-performing methods, one binocular (Sect. 2.3.3.1) and one monocular (Sect. 2.3.3.3), in the KITTI benchmark [GLS13], and showed consistent performance improvement in all metrics. To further challenge our prior, we took two other baselines which were not the top performers. We then added our prior and tested the results against the top performers in the latest benchmark. We also performed generalizability tests (Sect. 2.5.5), ablation studies (Sect. 2.5.4) and demonstrated our approach with VIO on hand-held devices (Sect. 2.5.6).

2.2 Related work

Early learning-based depth prediction approaches [SCN06, SSN09, KWI13, KLK12] predict depth using local image patches and then refine it using Markov random fields (MRFs). Recent works [EPF14, LRB16] leverage deep networks to directly learn a representation for depth prediction where the networks are typically based on the multi-scale fully convolutional encoder-decoder structure. These methods are fully supervised and do not generalize well outside the datasets on which they are trained. Latest self-supervised methods [GBC16,

GMB17, ZBS17] have shown better performance on benchmarks with better generalization.

There is a large body of work [MWA18, YS18, WBZ18, ZGW18] on self-supervised monocular depth prediction following Godard *et al.* [GMB17] and Zhou *et al.* [ZBS17], which simply use the reprojection error as a learning criterion, as has been customary in 3D reconstruction for decades. Generic priors such as piecewise smoothness and left-right consistency are also encoded into the network as additional loss terms. Our work is in-line with these self-supervised approaches, but we also exploit class-specific regularizers beyond the generic ones.

In terms of exploiting the relation of different geometric quantities in an end-to-end learning framework, closely related works include [WSR16, QLL18, LYC18], where surface normals are explicitly computed by using either a network [WSR16] or some heuristics [QLL18]. While the former is computation intensive, the latter relies on heuristics and thus is sub-optimal. In contrast, by using losses proposed in this paper, we directly regularize depth via the depth-gravity relation without a separate surface normal predictor. Besides, both [WSR16] and [LYC18] are supervised, while ours is self-supervised with the photometric loss and guided by global orientation and the semantics of the scene.

Earlier work on semantic segmentation [SJC08] relied on local features, and have been improved by incorporating global context using various structured prediction techniques [KK11, RKT09]. Starting from the work of Long *et al.* [LSD15], fully convolutional encoder-decoder networks have been a staple in semantic segmentation. Although we do not address semantic segmentation, we leverage per-pixel semantic labeling enabled by existing systems to aid depth prediction in the form of providing class-specific priors and an attention mechanism to selectively apply such priors, which is different from joint segmentation and depth prediction approaches [JGK17].

The idea of using class-specific priors to facilitate reconstruction is not new [HZC13, KLD14]. In [HZC13], class-specific shape priors in the form of spatially-varying anisotropic smoothness terms are used in an energy minimization framework to reconstruct small objects. Though promising, this system does not scale well. An efficient inference framework [KK11]

has been used with a CRF model over a voxel-grid to achieve real-time performance by [KLD14]. While all these methods explore class-specific priors in various ways, none has used them in an end-to-end learning framework. Also, all the methods above take range images as inputs, which are then fused with semantics during optimization, while ours exploits semantics at an earlier stage – when generating such range images which themselves can serve as priors for dense reconstruction and other inference tasks.

2.3 Methodology

In this section, we introduce our loss functions as regularizers added to existing models at training time, in addition to data terms (photometric loss) and generic regularizers (smoothness loss). We dub our loss semantically informed geometric loss (SIGL) because geometric constraints are selectively applied to certain image regions, where a semantic segmentation module informs the selection. Fig. 2.1 illustrates part of our training diagram. In Sect. 2.3.3, we review baseline models used in our experiments and show that the application of our losses on top of them improves performance (Sect. 2.5).

2.3.1 Semantically informed geometric loss

During training, we assume to be given a partition of the image plane into semantic classes $c \in C$ that have a consistent geometric correlate. For instance, a pixel with image coordinates $(x, y) \in \mathbb{R}^2$ and class $c(x, y) =$ "road" is often associated to a normal plane oriented along the vertical direction (direction of gravity), whereas $c =$"building" has a normal vector orthogonal to it. We also assume we are given the calibration matrix K of the camera capturing the images, so the pixel coordinates (x, y) on the image plane back-project to points in space via

$$\mathbf{X} = \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = K^{-1} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} Z(x, y) \tag{2.1}$$

where $Z(x, y)$ is the depth Z of the point along the projection ray determined by (x, y).

Any subset $\Omega \subset \mathbb{R}^2$ of the image plane that is the image of a spatial plane with normal vector $N \in \mathbb{R}^3$, at distance $\|N\|$ from the center of projection, satisfies a constraint of the form $\mathbf{X}_i^T N = 1$ for all i, assuming the plane does not go through the optical center. Stacking all the points into a matrix $\bar{\mathbf{X}} \doteq [\mathbf{X}_1, \mathbf{X}_2 \cdots \mathbf{X}_M]^\top$, we have $\bar{\mathbf{X}} N = \mathbf{1}$, where $\mathbf{1}$ is a vector of M ones, and $M = |\Omega|$ is the cardinality of the set Ω. If the direction, but not the norm, of the vector N is known, a scale-invariant constraint can be easily obtained by removing the mean of the points, so that (details in Sect. 2.3.2)

$$(\mathbf{I} - \frac{1}{M}\mathbf{1}\mathbf{1}^\top)\bar{\mathbf{X}} N = 0. \tag{2.2}$$

The scale-invariant constraint above can be used to define a loss to penalize deviation from planarity:

$$L_{HP}(\Omega_{HP}) = \frac{1}{|\Omega_{HP}|}\|(\mathbf{I} - \frac{1}{|\Omega_{HP}|}\mathbf{1}\mathbf{1}^\top)\bar{\mathbf{X}}\gamma\|_2^2 \tag{2.3}$$

where N in Eq. (2.2) is replaced by the normalized gravity γ due to the homogeneity of constraint (2.2), and the squared norm is taken assuming the network predicts per-pixel depth $Z(x, y)$ up to additive zero-mean Gaussian noise. $\Omega_{HP} \subset \mathbb{R}^2$ is a subset of the image plane whose associated semantic classes have horizontal surfaces, such as "road", "sidewalk", "parking lot", etc. We call this loss "horizontal plane" loss, where the direction of gravity γ can be reliably and globally estimated.

Similarly, a "vertical plane" loss can be constructed to penalize deviation from a vertical plane whose normal N has *both unknown direction and norm* but lives in the null space of γ, *i.e.*, $N \in \mathcal{N}(\gamma)$. Thus, the vertical plane loss reads

$$L_{VP}(\Omega_{VP}) = \min_{\substack{N \in \mathcal{N}(\gamma) \\ \|N\|=1}} \frac{1}{|\Omega_{VP}|}\|(\mathbf{I} - \frac{1}{|\Omega_{VP}|}\mathbf{1}\mathbf{1}^\top)\bar{\mathbf{X}} N)\|_2^2 \tag{2.4}$$

where the constraint $\|N\| = 1$ avoids trivial solutions $N = 0$ again due to the homogeneity of the objective; Ω_{VP} is a subset of the image plane whose associated semantic classes have vertical surfaces, such as "building", "fence", "billboard", etc. The constrained minimization problem in the vertical plane loss L_{VP} is due to the unknown direction of the surface normals and introduces some difficulties in training. We discuss approximations in Sect. 2.3.2.

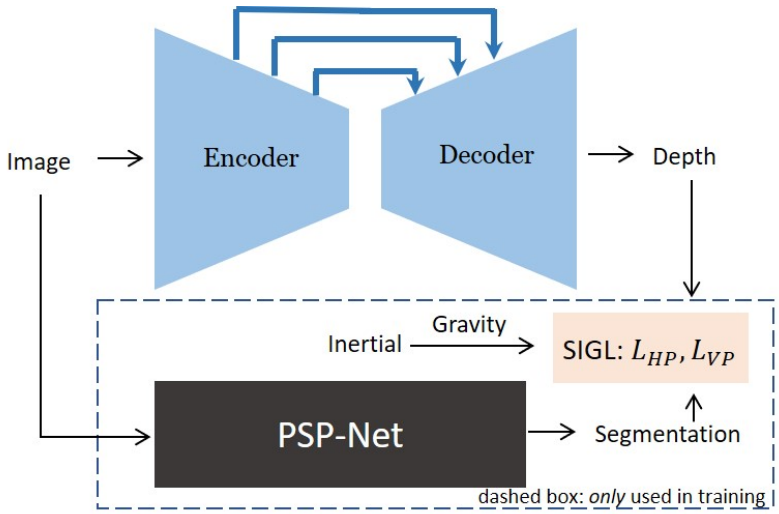


Figure 2.1: *Illustration of geo-supervised visual depth prediction.* Our visual depth predictor is an encoder-decoder convolutional neural network with skip connections. At inference time, the network takes an RGB image as the only input and outputs an inverse depth map. At training time, gravity extracted from inertial measurements biases the depth prediction *selectively*, which is informed by semantic segmentation produced by PSPNet. The other identical stream of the network and the photometric losses used for training are omitted in this figure.

2.3.2 Explanation of the objectives

Our idea is essentially to use priors about surface normals to regularize depth prediction. An intuitive way to achieve this is to compute the surface normals from the depth values first and then impose regularity, which will eventually bias the depth predictor via backpropagation. However, such a method involves normal estimation from depth, which can be problematic, especially with a simplistic but noisy normal estimator [QLL18].[1] On the other hand, one could train a deep network to compute surface normals [WSR16], which is costly. Therefore, *we do not compute surface normals but directly regularize the depth values* via the scale-invariant constraint Eq. (2.2) which is a function of depth and the direction of gravity.

In what follows, we give an explanation of L_{HP} Eq. (2.3) from a statistical perspective.

[1]For instance, one can compute the point-wise surface normal as the cross product of two vectors tangent to the surface, where the tangent vectors are approximated by connecting the underlying point to its nearest neighbors on the surface.

Let $M = |\Omega_{HP}|$ to avoid notation clutter and expand Eq. (2.3)

$$(\mathbf{I} - \frac{1}{M}\mathbf{1}\mathbf{1}^\top)\bar{\mathbf{X}}\gamma \tag{2.5}$$

$$= \begin{bmatrix} 1 - \frac{1}{M} & \cdots & -\frac{1}{M} \\ \vdots & \ddots & \vdots \\ -\frac{1}{M} & \cdots & 1 - \frac{1}{M} \end{bmatrix} \begin{bmatrix} \mathbf{X}_1^\top \gamma \\ \mathbf{X}_2^\top \gamma \\ \cdots \\ \mathbf{X}_M^\top \gamma \end{bmatrix} = \begin{bmatrix} \vdots \\ \left(\mathbf{X}_i - \frac{1}{M}\sum_{j=1}^{M} \mathbf{X}_j\right)^\top \gamma \\ \vdots \end{bmatrix}. \tag{2.6}$$

Let $\mu = \frac{1}{M}\sum_{j=1}^{M} \mathbf{X}_j$ be the sample mean of the 3D coordinates and the horizontal plane loss L_{HP} reads

$$L_{HP}(\Omega_{HP}) = \frac{1}{M}\sum_{i=1}^{M} \left((\mathbf{X}_i - \mu)^\top \gamma\right)^2 \tag{2.7}$$

which is the sample variance of the 3D coordinates projected to the direction of gravity γ (coinciding with the surface normal for horizontal planes). To minimize L_{HP} is to minimize the variance of the 3D coordinates along the surface normal.

Similarly, to minimize L_{VP} Eq. (2.4) is to minimize the variance of the 3D coordinates along some direction perpendicular to gravity. However, if the direction is unknown, one needs to jointly solve the direction while minimizing L_{VP}, which explains the constrained quadratic problem in L_{VP}. Though this can be solved via eigendecomposition, the gradients of the solver – needed in backpropagation – are non-trivial to compute. In fact, representing an optimization procedure as a layer of a neural network is an open research problem [AK17]. To alleviate both numerical and implementation difficulties, we uniformly sample unit vectors from the null space of gravity and compute the minimum of the objective over the samples as an approximation to the loss. Empirically, we found using eight directions sampled every 45 degrees from 0 to 360 generally performs well.

2.3.3 View synthesis as supervision and baselines

To showcase the ability to improve upon existing self-supervised monocular depth prediction networks, we add our losses to two publicly available models – Godard [GMB17] (`LR-Consistency`) and Yin [YS18] (`GeoNet`) – as baselines and perform both quantitative

and qualitative comparisons. We additionally apply our losses to Zhan [ZGW18] (`Stereo-Temporal`) and Wang [WBZ18] (`DDVO`), the state-of-the-art methods in their respective training setting, stereo pairs/videos, and monocular videos. `LR-Consistency` is trained with rectified stereo image pairs, `GeoNet` and `DDVO` use monocular videos while `Stereo-Temporal` uses stereo videos. At test time, all training settings result in a system that takes a single image as input and predicts an inverse depth map as output. We show that by applying our losses to the baselines `LR-Consistency` and `GeoNet`, we achieve better performance than the state-of-the-art methods `Stereo-Temporal` and `DDVO`. Furthermore, we produce new state-of-the-art results by applying our losses to `Stereo-Temporal` and `DDVO`.

2.3.3.1 Training with stereo pairs

At training time, our first baseline model (`LR-Consistency`) takes a single left image as its input and predicts two disparity maps $D^L, D^R : \mathbb{R}^2 \supset \Omega \to \mathbb{R}_+$ for both left and right cameras. The network follows the fully convolutional encoder-decoder structure with skip connections. The total loss consists of three terms: Appearance loss, smoothness of disparity and left-right consistency, each of which is evaluated on both the left and the right streams across multiple scale levels. Here we address the view synthesis loss, which serves as the data term and is part of the appearance loss:

$$L_{\text{vs}}^L = \frac{1}{|\Omega|} \sum_{(x,y)\in\Omega} \|I^L(x,y) - I^R(x + D^L(x,y), y)\|_1. \tag{2.8}$$

The view synthesis loss is essentially the photometric difference of the left image $I^L(x,y)$ and the right image warped to the left view $I^R(x + D^L(x,y), y)$ according to the left disparity prediction $D^L(x,y)$. The right view synthesis loss is constructed in the same way. Though only one disparity map is needed at inference time, it has been shown that predicting both left and right disparity maps and including the left-right consistency loss Eq. (2.9) are in general beneficial [GMB17].

$$L_{\text{lr}}^L = \frac{1}{|\Omega|} \sum_{(x,y)\in\Omega} \|D^L(x,y) - D^R(x + D^L(x,y), y)\|_1 \tag{2.9}$$

2.3.3.2 Training with stereo videos

In our second baseline `Stereo-Temporal`, stereo videos are used to train a monocular depth predictor, where two frames of a stereo pair and another frame one time step ahead are involved in constructing a stereo-temporal version of the photometric loss: For the stereo pair, Eq. (2.8) is applied while for the temporal pair, Eq. (2.10) (detailed below) is applied.

2.3.3.3 Training with monocular videos

To train our third and fourth baseline models (`GeoNet` and `DDVO`), a single reference frame I_t is fed into the depth network and frames $I_{t'}, t' \in W_t$ in a temporal window centered at t are used to construct the view synthesis loss, also known as reprojection error:

$$L_{\text{vs}} = \frac{1}{|W_t||\Omega|} \sum_{t' \in W_t} \sum_{(x,y) \in \Omega} \| I_t(x,y) - I_{t'}\big(\pi(\hat{g}_{t't}\mathbf{X})\big) \|_1 \tag{2.10}$$

which is the difference between the reference frame I_t and neighboring frames $I_{t'}$ warped to it. $\mathbf{X}$ is the back-projected point defined in Eq. (2.1), π is a central (perspective) projection, and $\hat{g}_{t't}$ is the relative camera pose up to an unknown scale predicted by an auxiliary pose network which takes both I_t and $I_{t'}$ as its input. Note that the pose and depth networks are coupled via the view synthesis loss at training time; at test time, the depth network alone is needed to perform depth prediction with a single image as its input. Interestingly, in Sect. 2.5.6 we found that replacing the pose network with pose estimation from VIO produces better results compared to the multi-task learning diagram where pose and depth networks are trained simultaneously, which sheds light on the use of classic SLAM/Odometry systems in developing better learning algorithms.

A detailed discussion about other losses serving as regularization terms is beyond the scope of this paper and can be found in [GMB17, ZBS17, YS18, WBZ18].

2.4 Implementation Details

2.4.1 Semantic segmentation

At training time, we use PSPNet [ZSQ17] pre-trained on the CityScapes dataset [COR16] provided by the authors to obtain per-pixel labeling. For every pixel $(x, y) \in \mathbb{R}^2$, a probability distribution over 19 classes is predicted by PSPNet, of which the most likely class $c(x, y) \in C$ determines the orientation of the surface where the back-projected point $\mathbf{X}$ sits. We group the 19 classes into 7 categories[2] according to the CityScapes benchmark and test our losses on all of them. Empirically, we found that it is most beneficial to apply our losses to the "flat", "vehicle" and "construction" categories and therefore all the comparisons on KITTI against baseline methods are made with these categories regularized. The influence of other categories is studied in Sect. 2.5.4.

2.4.2 Gravity

For imagery captured by a static platform equipped with an inertial measurement unit (IMU), one can use the gravity $\gamma_b \in \mathbb{R}^3$ measured in the body frame (coinciding with the IMU frame) and simply apply the body-to-camera rotation $R_{cb} \in \mathrm{SO}(3)$ to obtain the gravity in the camera frame $\gamma = R_{cb}\gamma_b$ which is then used in Eq. (2.3) and (2.4). For moving platforms, one resorts to robust VIO, which is well studied [MR07, TCS15]. In Sect. 2.5.6, we demonstrated our approach on a visual-inertial odometry dataset, where both camera pose and gravity are estimated online by VIO.

For our experiments on the KITTI dataset, thanks to the GPS/IMU sensor package which provides linear acceleration of the sensor platform measured both in the body frame ($\alpha_b \in \mathbb{R}^3$) and the spatial frame ($\alpha_s \in \mathbb{R}^3$), we are able to compute the spatial-to-body rotation $R_{bs} \in \mathrm{SO}(3)$ and then bring the gravity $\gamma_s = [0, 0, 9.8]^\top$ from the spatial frame to the camera frame $\gamma = R_{cb}R_{bs}\gamma_s$. In all settings, R_{cb} (the rotational part of the body-to-

[2] "flat": road, sidewalk; "human": rider, person; "vehicle": car, truck, bus, train, motorcycle, bicycle; "construction": building, wall, fences; "object": pole, traffic light, traffic sign; "nature": vegetation, terrain; "sky": sky.

camera transformation) is obtained via offline calibration procedures.

2.4.3 Training details

A GTX 1080 Ti GPU and Adam [KB14] optimizer are used in our experiments. Depending on different model variants and input image sizes, the training time varies from 8 hours to 16 hours. For `LR-Consistency` and `GeoNet` which were initially implemented in TensorFlow, we implemented our losses also in TensorFlow and applied them to the existing code bases. Code of `Stereo-Temporal` is available online, but in Caffe, thus we migrated their model to TensorFlow and applied our losses. We also implemented our losses in PyTorch, which were then applied to `DDVO` of which the PyTorch version was made available by the author. Our code is available at https://github.com/feixh/GeoSup.

2.5 Experiments

To enable quantitative evaluation, we exploit the KITTI benchmark, and test our approach against the state-of-the-art as described in detail below (Sect. 2.3.3.1 and Sect. 2.3.3.3). We also carried out ablation studies (Sect. 2.5.4) and tested the generalizability of our approach (Sect. 2.5.5). In addition to KITTI, which features planar motion in driving scenarios, we have conducted experiments on VISMA dataset [FS18] – an indoor visual-inertial odometry dataset captured under non-trivial ego-motion (Sect. 2.5.6).

2.5.1 KITTI Eigen split

We compare our approach with recent state-of-the-art methods on the monocular depth prediction task using the KITTI Eigen split [EPF14] in two training domains: stereo pairs/videos and monocular videos (Sect. 2.3.3). The Eigen split test set contains 697 test images selected from 29 of 61 scenes provided by the raw KITTI dataset. Of the remaining 32 scenes containing 23,488 stereo pairs, 22,600 pairs are used for training, and the rest is used for validation per the training split proposed by [GBC16]. To generate ground truth depth maps for val-

idation and evaluation, we take the Velodyne data points associated with each image and project them from the Velodyne frame to the left RGB camera frame. Each resulting ground truth depth map covers approximately 5% of the corresponding image and may be erroneous. To handle this, first, we use the cropping scheme proposed by [GBC16], which masks out the potentially erroneous extremities from the left, right and top areas of the ground truth depth map. Then we evaluate depth prediction only at pixels where ground truth depth is available. For visualization, we linearly interpolate each sparse depth map to cover the entire image (Fig. 2.2).

We additionally provide quantitative evaluations of variants of the models pre-trained on CityScapes and fine-tuned on KITTI. CityScapes dataset contains 22,973 training stereo pairs captured in various cities across Germany with a similar modality as KITTI. We cropped each input image to keep only the top 80% of the image, removing the reflective hood.

The error and accuracy metrics, which are initially proposed by [EPF14] and adopted by others, are used (Table 2.1). Also as a convention in the literature, performances evaluated with depth prediction capped at 50 and 80 meters are reported as suggested by [GMB17]. The choice of 80 meters is two-fold: 1) maximum depth present in the KITTI dataset is on the order of 80 meters and 2) non-thresholded measures can be sensitive to the significant errors in depth caused by prediction errors at small disparity values. For the same reason, depth prediction is capped at 70 meters in the Make3D experiment. Prediction capped at 50 meters is also evaluated since depth at the closer range is more applicable to real-world scenarios.

2.5.2 Training with stereo pairs

The first baseline we adopt is Godard [GMB17] (with `VGG` [SZ14] as feature extractor), to which SIGL is imposed at training time along with the view synthesis loss Eq. (2.8) and other generic regularizers used in [GMB17]. The model is trained from scratch with stereo pairs following the Eigen split and compared to both supervised [EPF14, LSL16] and self-

Metric	Definition
AbsRel	$\frac{1}{\lvert\Omega\rvert}\sum_{(x,y)\in\Omega}\frac{\lvert Z(x,y)-Z^{\mathrm{gt}}(x,y)\rvert}{Z^{\mathrm{gt}}(x,y)}$
SqRel	$\frac{1}{\lvert\Omega\rvert}\sum_{(x,y)\in\Omega}\frac{\lvert Z(x,y)-Z^{\mathrm{gt}}(x,y)\rvert^2}{Z^{\mathrm{gt}}(x,y)}$
RMSE	$\sqrt{\frac{1}{\lvert\Omega\rvert}\sum_{(x,y)\in\Omega}\lvert Z(x,y)-Z^{\mathrm{gt}}(x,y)\rvert^2}$
RMSE log	$\sqrt{\frac{1}{\lvert\Omega\rvert}\sum_{(x,y)\in\Omega}\lvert\log Z(x,y)-\log Z^{\mathrm{gt}}(x,y)\rvert^2}$
$\log_{10}$	$\frac{1}{\lvert\Omega\rvert}\sum_{(x,y)\in\Omega}\lvert\log Z(x,y)-\log Z^{\mathrm{gt}}(x,y)\rvert$
Accuracy	% of $Z(x,y)$ s.t. $\delta \doteq \max\left(\frac{Z(x,y)}{Z^{\mathrm{gt}}(x,y)}, \frac{Z^{\mathrm{gt}}(x,y)}{Z(x,y)}\right) <$ threshold

Table 2.1: *Error and Accuracy Metrics.* $Z(x,y)$ is the predicted depth at $(x,y)\in\Omega$ and $Z^{\mathrm{gt}}(z,y)$ is the corresponding ground truth. Three different thresholds ($1.25, 1.25^2$, and 1.25^3) are used in the accuracy metric as a convention in the literature.

supervised methods [GMB17, ZGW18]. In addition, we apply our losses to variants of the baseline (with `ResNet` [HZR16] as feature extractor; w/ & w/o post-processing) and evaluate different training schemes (w/ & w/o pre-training on CityScapes). Quantitative comparisons can be found in Table 2.2, where the results with SIGL added as an additional regularizer follow the results of the baseline models and variants. In the column marked "Data", `K` refers to Eigen split benchmark on the KITTI dataset, and `CS` refers to the CityScapes dataset. Methods marked with `CS+K` are pre-trained on CityScapes and then fine-tuned on KITTI Eigen split. `pp` denotes post-processing. Cap Xm means depth predictions are capped at X meters. Results of Zhan [ZGW18] `Stereo-Temporal` are taken from their paper. The rest of the results are taken from [GMB17] unless otherwise stated.

We want to remind the reader that the first baseline model atop which we built ours is `VGG` [GMB17] which initially performed worse than `Stereo-Temporal` [ZGW18] by a large margin, but by applying our losses to the baseline at training time we managed to boost its performance and make it perform even better than `Stereo-Temporal` at test time. Note that `Stereo-Temporal` also exploits temporal information in addition to stereo pairs for training while our first baseline built atop Godard does not.

As a second baseline, we apply our losses additionally to `Stereo-Temporal` to further push the state-of-the-art. Table 2.2 shows that our losses improve `Stereo-Temporal` across

Method	Data	Error metric				Accuracy ($\delta <$)		
		AbsRel	SqRel	RMSE	RMSElog	1.25	1.25^2	1.25^3
Depth: cap 80m								
TrainSetMean*	K	0.361	4.826	8.102	0.377	0.638	0.804	0.894
Eigen [EPF14] `Coarse`*	K	0.214	1.605	6.563	0.292	0.673	0.884	0.957
Eigen [EPF14] `Fine`*	K	0.203	1.548	6.307	0.282	0.702	0.890	0.958
Liu [LSL16]*	K	0.201	1.584	6.471	0.273	0.680	0.898	0.967
Godard [GMB17] `VGG`	K	0.148	1.344	5.927	0.247	0.803	0.922	0.964
+SIGL	K	**0.139**	**1.211**	**5.702**	**0.239**	**0.816**	**0.928**	**0.966**
Zhan [ZGW18] `Stereo-Temporal`	K	0.144	1.391	5.869	0.241	0.803	0.928	0.969
+SIGL	K	**0.137**	**1.061**	**5.692**	**0.239**	**0.805**	0.928	0.969
Godard [GMB17] `VGG pp`	CS+K	0.124	1.076	5.311	0.219	0.847	0.942	0.973
+SIGL	CS+K	**0.114**	**0.885**	**4.877**	**0.203**	**0.858**	**0.950**	**0.978**
Godard [GMB17] `ResNet pp`	CS+K	0.114	0.898	4.935	0.206	0.861	0.949	0.976
+SIGL	CS+K	**0.112**	**0.836**	**4.892**	**0.204**	**0.862**	**0.950**	**0.977**
Depth: cap 50m								
Garg [GBC16]	K	0.169	1.080	5.104	0.273	0.740	0.904	0.962
Godard [GMB17] `VGG`	K	0.140	0.976	4.471	0.232	0.818	0.931	0.969
+SIGL	K	**0.132**	**0.891**	**4.312**	**0.225**	**0.831**	**0.936**	**0.970**
Zhan [ZGW18] `Stereo-Temporal`	K	0.135	0.905	4.366	0.225	0.818	0.937	0.973
+SIGL	K	**0.131**	**0.829**	**4.217**	**0.224**	**0.824**	0.937	0.973
Godard [GMB17] `VGG pp`	CS+K	0.112	0.680	3.810	0.198	0.866	0.953	0.979
+SIGL	CS+K	**0.108**	**0.658**	**3.728**	**0.192**	**0.870**	**0.955**	**0.981**
Godard [GMB17] `ResNet pp`	CS+K	0.108	0.657	3.729	0.194	0.873	0.954	0.979
+SIGL	CS+K	**0.106**	**0.615**	**3.697**	**0.192**	**0.874**	**0.956**	**0.980**

Table 2.2: *Training with stereo pairs on KITTI.* Methods marked with * are supervised by ground-truth depth, and +SIGL indicates that SIGL is imposed to the preceeding method.

all error metrics with the accuracy metrics $\delta < 1.25^2$ and $\delta < 1.25^3$ being comparable. Another variant of Zhan's model pre-trains on NYU-V2 [SHK12] in a fully supervised fashion and is therefore not pertinent to this comparison. Fig. 2.2 shows a head-to-head qualitative comparison of ours and the baseline models.

2.5.3 Training with monocular videos

To demonstrate the effectiveness of our loss in the second training setting (monocular videos), we impose SIGL to our third (Yin [YS18]) and fourth (Wang [WBZ18]) baseline. Using the KITTI Eigen split, we follow the training and validation 3-frame sequence selection

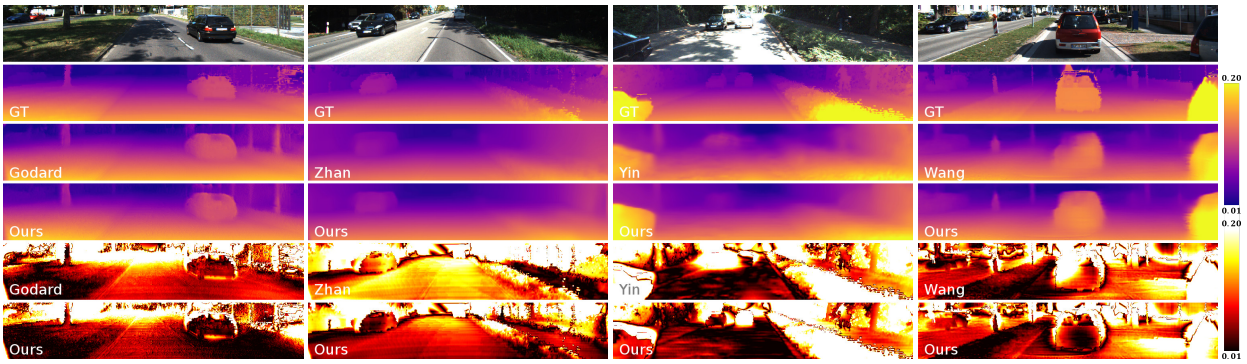


Figure 2.2: *Qualitative results on KITTI Eigen split.* (best viewed at 5× with color) Top to bottom, each column shows an input RGB image, the corresponding ground truth inverse depth map, the predictions of baseline models trained without and with our priors, AbsRel error maps of baseline models trained without and with our priors. All the models are trained on KITTI Eigen split. For the purpose of visualization, ground truth is interpolated and all the images are cropped according to [GBC16]. For the error map, darker means smaller error. Typical image regions where we do better (darker in the error map) include cars, roads and walls.

proposed by [ZBS17] where the first and third frames are treated as the source views and the central (second) frame is treated as the reference as in Eq. (2.10). Of the 44,540 total sequences, 40,109 are used for training and 4,431 for validation. We evaluate our system on the aforementioned 697 test images [EPF14]. The same training and evaluation scheme are also applied to other top-performing methods [ZBS17, MWA18] in addition to the selected baselines.

Table 2.3 shows detailed comparisons against state-of-the-art methods self-supervised using monocular video sequences. We compare against best-performing model variants of Wang [WBZ18] (`PoseCNN` & `PoseCNN+DDVO`) and Yin [YS18] (`ResNet`) with and without pre-training on CityScapes. By adding our losses to existing models, we observe systematic performance improvement across all metrics. Though initially performing worse than Wang [WBZ18] `PoseCNN+DDVO`, Yin [YS18] `ResNet` with the proposed losses even outperforms the original `PoseCNN+DDVO`. Moreover, we achieve new state-of-the-art by adding our losses to `PoseCNN+DDVO` trained on both CityScapes and KITTI. Fig. 2.2 illustrates representative image regions where we do better.

Method	Data	Error metric				Accuracy ($\delta <$)		
		AbsRel	SqRel	RMSE	RMSElog	1.25	1.25^2	1.25^3
Depth: cap 80m								
Zhou [ZBS17]	K	0.208	1.768	6.856	0.283	0.678	0.885	0.957
Mahjourian [MWA18]	K	0.163	1.240	6.220	0.250	0.762	0.916	0.968
Yin [YS18] ResNet	K	0.155	1.296	5.857	0.233	0.793	0.931	0.973
+SIGL	K	**0.142**	**1.124**	**5.611**	**0.223**	**0.813**	**0.938**	**0.975**
Wang [WBZ18] PoseCNN	K	0.155	1.193	**5.613**	0.229	0.797	0.935	0.975
+SIGL	K	**0.147**	**1.076**	5.640	**0.227**	**0.801**	0.935	0.975
Wang [WBZ18] PoseCNN+DDVO	K	0.151	1.257	5.583	0.228	**0.810**	0.936	0.974
+SIGL	K	**0.146**	**1.068**	**5.538**	**0.224**	0.809	**0.938**	**0.975**
Zhou [ZBS17]	CS+K	0.198	1.836	6.565	0.275	0.718	0.901	0.960
Mahjourian [MWA18]	CS+K	0.159	1.231	5.912	0.243	0.784	0.923	0.970
Yin [YS18] ResNet	CS+K	0.153	1.328	5.737	0.232	0.802	0.934	0.972
+SIGL	CS+K	**0.147**	**1.076**	**5.468**	**0.222**	**0.806**	**0.938**	**0.976**
Wang [WBZ18] PoseCNN+DDVO	CS+K	0.148	1.187	5.496	0.226	0.812	0.938	0.975
+SIGL	CS+K	**0.142**	**1.094**	**5.409**	**0.219**	**0.821**	**0.941**	**0.976**
Depth: cap 50m								
Zhou [ZBS17]	K	0.201	1.391	5.181	0.264	0.696	0.900	0.966
Mahjourian [MWA18]	K	0.155	0.927	4.549	0.231	0.781	0.931	0.975
Yin [YS18] ResNet	K	0.147	0.936	4.348	0.218	0.810	0.941	0.977
+SIGL	K	**0.135**	**0.834**	**4.193**	**0.208**	**0.831**	**0.948**	**0.979**
Wang [WBZ18] PoseCNN†	K	0.149	0.920	4.303	0.216	0.813	0.943	0.979
+SIGL	K	**0.140**	**0.816**	**4.234**	**0.212**	**0.818**	**0.945**	**0.980**
Wang [WBZ18] PoseCNN+DDVO†	K	0.144	0.935	4.234	0.214	**0.827**	0.945	0.977
+SIGL	K	**0.139**	**0.808**	**4.180**	**0.209**	0.826	**0.948**	**0.980**
Zhou [ZBS17]	CS+K	0.190	1.436	4.975	0.258	0.735	0.915	0.968
Mahjourian [MWA18]	CS+K	0.151	0.949	4.383	0.227	0.802	0.935	0.974
Yin [YS18] ResNet*	CS+K	/	/	/	/	/	/	/
+SIGL	CS+K	**0.141**	**0.837**	**4.160**	**0.209**	**0.823**	**0.947**	**0.980**
Wang [WBZ18] PoseCNN+DDVO†	CS+K	0.142	0.901	4.202	0.213	0.827	0.946	0.978
+SIGL	CS+K	**0.135**	**0.832**	**4.119**	**0.206**	**0.836**	**0.949**	**0.980**

Table 2.3: *Training with monocular videos on KITTI.* Results of methods marked with * are inavailable, † indicates that the results are obtained by evaluating the prediction provided by the author of each corresponding method, and +SIGL indicates that SIGL is imposed to the preceeding method.

2.5.4 Ablation study

To study the contribution of each semantic category to the performance improvement, we performed an ablation study: We apply our losses to different semantic categories, one at a time, train the network until convergence, and show how the quality of depth prediction varies (Table 2.4). In Table 2.4, Godard *et al.* [GMB17] is the baseline model where only the most generic regularizers, *e.g.,* smoothness, and consistency, are used. The second column indicates the semantic category of which the depth prediction is regularized using our losses in addition to the generic regularizers. For the meaning of the semantic categories, see Sect. 2.4.1.

It turns out that the "flat" category contributes most to the performance gain over the baseline model, which is expected because most of the KITTI images contain a large portion of roads and sidewalks. We also observed that regularization of the "construction" and "vehicle" category provides reasonable improvement while the "nature" category (trees and hedges) helps a little. Applying our priors to the "human", "sky" and "object" categories does not consistently improve over the baseline, for the following reasons: "sky" does not have well-defined surface normals; "human" has deformable surfaces of which normals can point arbitrarily; "object" category consists of thin structures which project to few pixels rendering it hard to apply segmentation and our losses. The best is achieved when we apply our losses to "vehicle", "construction" and "flat" categories, denoted by `V+C+F` in Table 2.4.

2.5.5 Generalize to other datasets: Make3D

To showcase the generalizability of our approach, we follow the convention of [GMB17, ZBS17, YS18, WBZ18]: Our model trained *only* on KITTI Eigen split is directly tested on Make3D [SSN09]. Make3D contains 534 images with 2272×1707 resolution, of which 134 are used for testing.[3] Low resolution ground truth depths are given as 305×55 range maps and must be resized and interpolated for evaluation. We follow [GMB17] and [ZBS17] in

[3]Ideally we want to test on the whole Make3D dataset since we do not train on Make3D, but other methods to which we compare train on it. For a fair comparison, we only use the 134 images for testing.

Method	Category	Error metric				Accuracy ($\delta <$)		
		AbsRel	SqRel	RMSE	RMSElog	1.25	1.25^2	1.25^3
Godard [GMB17]	/	0.148	1.344	5.927	0.247	0.803	0.922	0.964
Ours	Human	0.152	1.394	5.945	0.251	0.801	0.921	0.963
Ours	Sky	0.148	1.368	5.864	0.245	0.807	0.923	0.964
Ours	Object	0.146	1.335	5.986	0.249	0.800	0.920	0.963
Ours	Nature	0.146	1.292	5.826	0.247	0.804	0.923	0.964
Ours	Vehicle	0.143	1.304	5.797	0.241	0.814	0.927	0.966
Ours	Construction	0.142	1.252	5.729	0.240	0.810	0.928	0.967
Ours	Flat	0.141	1.270	5.779	0.239	0.814	0.927	0.966
Ours	V+C+F	0.139	1.211	5.702	0.239	0.816	0.928	0.966

Table 2.4: *Ablation study on KITTI.*

applying a central cropping to generate a 852×1707 crop centered on the image. We use the standard $C1$ evaluation metrics for Make3D and measure our performance on depths less than 70 meters. Table 2.5 shows a quantitative comparison to the competitors, both supervised and self-supervised, with two different training settings. Note that the results of [KLK12, LSL16, LRB16] are directly taken from [GMB17]. Since the exact cropping scheme used in [GMB17] is not available, we re-implemented it closely following the description in [GMB17]. We trained our model on KITTI Eigen split and compared against models of [GMB17, ZBS17, YS18, WBZ18] also trained on Eigen split (as provided by the authors) for a fair comparison.

A careful inspection of the baseline models (Godard [GMB17] in stereo and Yin [YS18] in monocular supervision) versus ours reveals that the application of our losses does not hurt the generalizability of the baselines. Fig. 2.3 shows some qualitative results on Make3D. Though our model registers some failure cases in texture-less regions, a rough scene layout is present in the prediction. Regarding that the model is only trained on KITTI, of which the data modality is very different from that of Make3D, the prediction is sensible. But after all, a single image only affords to hypothesize depth, so we expect that any method using

Method	Supervision	AbsRel	SqRel	RMSE	$\log_{10}$
TrainSetMean	Depth	0.893	15.517	11.542	0.223
Karsch [KLK12]	Depth	0.417	4.894	8.172	0.144
Liu [LSL16]	Depth	0.462	6.625	9.972	0.161
Laina [LRB16]	Depth	**0.198**	**1.665**	**5.461**	**0.082**
Godard [GMB17] `VGG`	Stereo	0.468	9.236	12.525	0.165
Ours	Stereo	**0.458**	**8.681**	**12.335**	**0.164**
Zhou [ZBS17]	Mono	0.407	5.367	11.011	0.167
Yin [YS18]`ResNet`	Mono	0.376	4.645	10.350	0.152
Wang [WBZ18]`PoseCNN+DDVO`	Mono	0.387	4.720	**8.09**	0.204
Ours	Mono	**0.356**	**4.517**	10.047	**0.144**

Table 2.5: *Generalizability test on Make3D.*

such predictions would have mechanisms to handle model deficiencies.

2.5.6 Evaluation on indoor datasets

To the best of our knowledge, none of the top-performing methods in self-supervised depth prediction have shown experimental results beyond planar motion, *i.e.,* driving scenarios such as KITTI and CityScapes, probably due to two reasons: Lack of rectified stereo pairs for training [GMB17, ZGW18] and difficulty to learn complex ego-motion along with depth prediction from video sequences [ZBS17, YS18, WBZ18].

However, with two modifications to the `GeoNet` model of Yin [YS18] – a multi-task learning approach where ego-motion and depth prediction are jointly learned, we managed to train our model and outperform `GeoNet` on publicly available VISMA [FS18] dataset which features monocular videos of indoor scenes captured by a hand-held visual-inertial sensor platform under challenging motion. As a first modification, we replace the pose network in `GeoNet` with pose estimation from a VIO system [TCS15], which makes the network easier to train (we call this model `OursVIO`). Second, to further improve the quality of predicted depth maps, we impose our gravity-induced regularization terms to `OursVIO`, where gravity

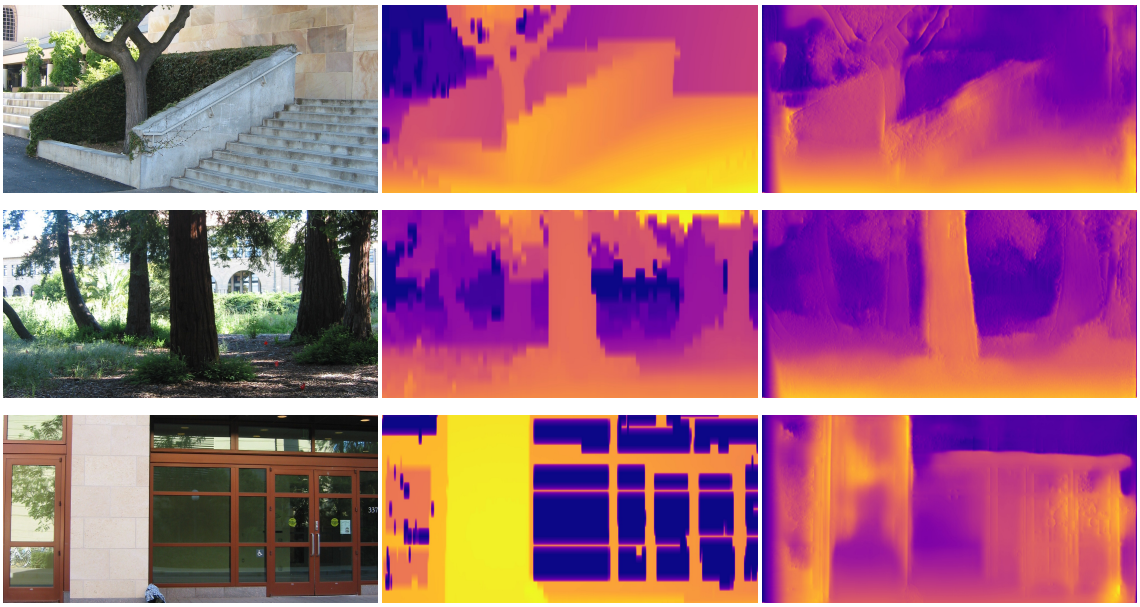


Figure 2.3: *Qualitative results on Make3D.* Left to right, each row shows an input RGB image, the corresponding ground truth disparity map and our prediction. Our model is *only* trained on KITTI and directly applied to Make3D.

is also estimated online by VIO. Our second model is named `OursVIO++`.

VISMA dataset contains time-stamped monocular videos (30 Hz) from a PointGrey camera and inertial measurements (100 Hz) from an Xsens unit, which are used in both VIO and network training. RGB-D reconstructions (dense point clouds) of the same scenes from a Kinect are also available, along with the spatial alignment $g_{\text{VIO}\leftarrow\text{RGBD}} \in \text{SE}(3)$ from RGB-D to VIO provided by the author. To get ground truth depth for cross-modality validation, we apply $g_{\text{VIO}\leftarrow\text{RGBD}}$ to the dense point clouds which are then projected to the PointGrey video frames. PSPNet trained on ADE20K [ZZP17] produces segmentation masks for training.[4] Of the $10K$ frames in VISMA, we remove static ones and construct 3-frame sequences (triplet) which are five frames apart in the original video to ensure sufficient parallax, resulting in $8,511$ triplets in total. We randomly sample 100 triplets for validation and use the rest for training. Fig. 2.4 and Table 2.6 show comparisons of `GeoNet`, `OursVIO` and `OursVIO++`, all trained from scratch on VISMA until validation error stops decreasing. Both `OursVIO` and

[4]Among the 91 categories in ADE20K which PSPNet is trained on, we select "floor", "ceiling", "wall", "window", "door", "building", "chair", "cabinet", "desk", "table" to apply our losses.

Method	Error metric				Accuracy ($\delta <$)		
	AbsRel	SqRel	RMSE	RMSElog	1.25	1.25^2	1.25^3
`GeoNet`	0.204	0.157	0.518	0.250	0.702	0.914	0.975
`OursVIO`	0.154	0.111	0.446	0.211	0.796	0.940	0.983
`OursVIO++`	**0.149**	**0.105**	**0.421**	**0.202**	**0.820**	**0.947**	0.983

Table 2.6: *Quantitative results on VISMA validation.*

`OursVIO++` improve over the baseline model by a large margin. Moreover, `OursVIO++` trained with our gravity-induced losses has the capability to further refine results of `OursVIO` trained without our losses.

2.6 Discussion

Gravity informs the shape of objects populating the scene, which is a powerful prior to visual scene analysis. We have presented a simple illustration of this power by adding a prior to standard monocular depth prediction methods that biases the normals of surfaces of known classes to align to gravity or its complement. Far more can be done: While in this work we use known biases in the shape of certain object classes, such as the fact that roads tend to be perpendicular to gravity, in the future we could learn such biases directly.

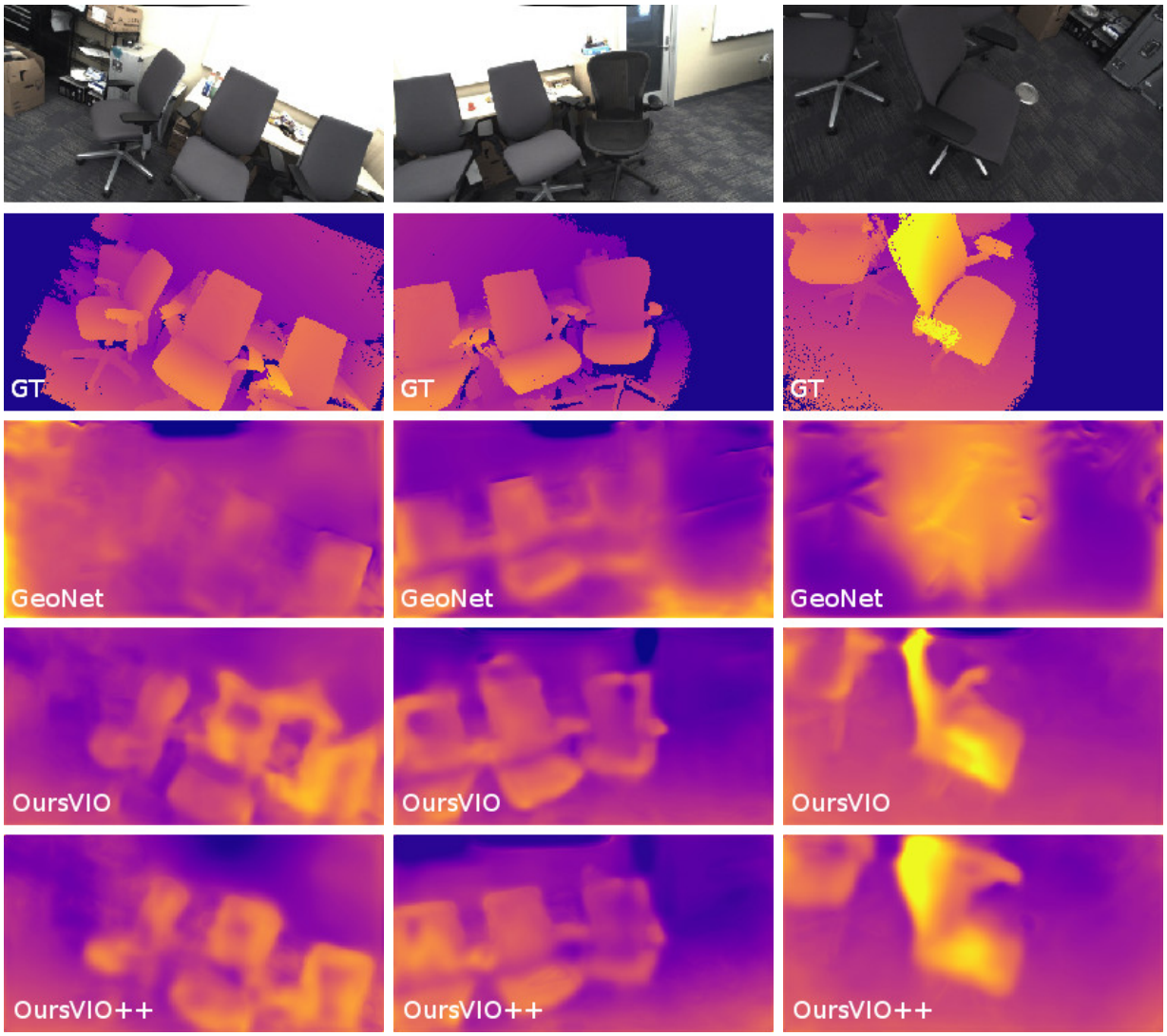


Figure 2.4: *Qualitative comparison on VISMA validation.* Top to bottom, each column shows an input RGB image, the corresponding ground truth inverse depth map, results of `GeoNet` (baseline), `OursVIO`, and `OursVIO++`. Both `OursVIO` and `OursVIO++` show largely improved results over the baseline, especially for images captured at extreme viewpoint (large in-plane rotation and top-down view). `OursVIO++` (with gravity-induced priors) further improves over `OursVIO` (without priors) at planar regions, *e.g.,* the chair backs, where holes have been filled.

CHAPTER 3

Depth Completion from Visual-Inertial Odometry

3.1 Introduction

A sequence of images is a rich source of information about both the three-dimensional (3D) shape of the environment and the motion of the sensor within. Motion can be inferred at most up to a scale and a global Euclidean reference frame, provided sufficient parallax and a number of visually discriminative Lambertian regions that are stationary in the environment, and are visible from the camera. The position of such regions in the scene defines the Euclidean reference frame, with respect to which motion is estimated. Scale as well as two directions of orientation can be further identified by fusion with inertial measurements (accelerometers and gyroscopes) and, if available, a magnetometer can fix the last (Gauge) degree of freedom. Because the regions defining the reference frame have to be visually distinctive ("features"), they are typically *sparse*. In theory, three points are sufficient to define a Euclidean Gauge if visible at all times. In practice, because of occlusions, any Structure From Motion (SFM) or simultaneous localization and mapping (SLAM) system maintains an estimate of the location of a sparse set of features, or "sparse point cloud," typically in the hundreds to thousands. These are sufficient to support a point-estimate of motion, but a rather poor representation of shape as they do not reveal the topology of the scene: The empty space between points could be empty, or occupied by a solid with a smooth surface radiance (appearance). Attempts to *densify* the sparse point cloud, by interpolation or regularization with generic priors such as smoothness, piecewise planarity and the like, typically fail since SFM yields far too sparse a reconstruction to inform topology. This is where the image comes back in.

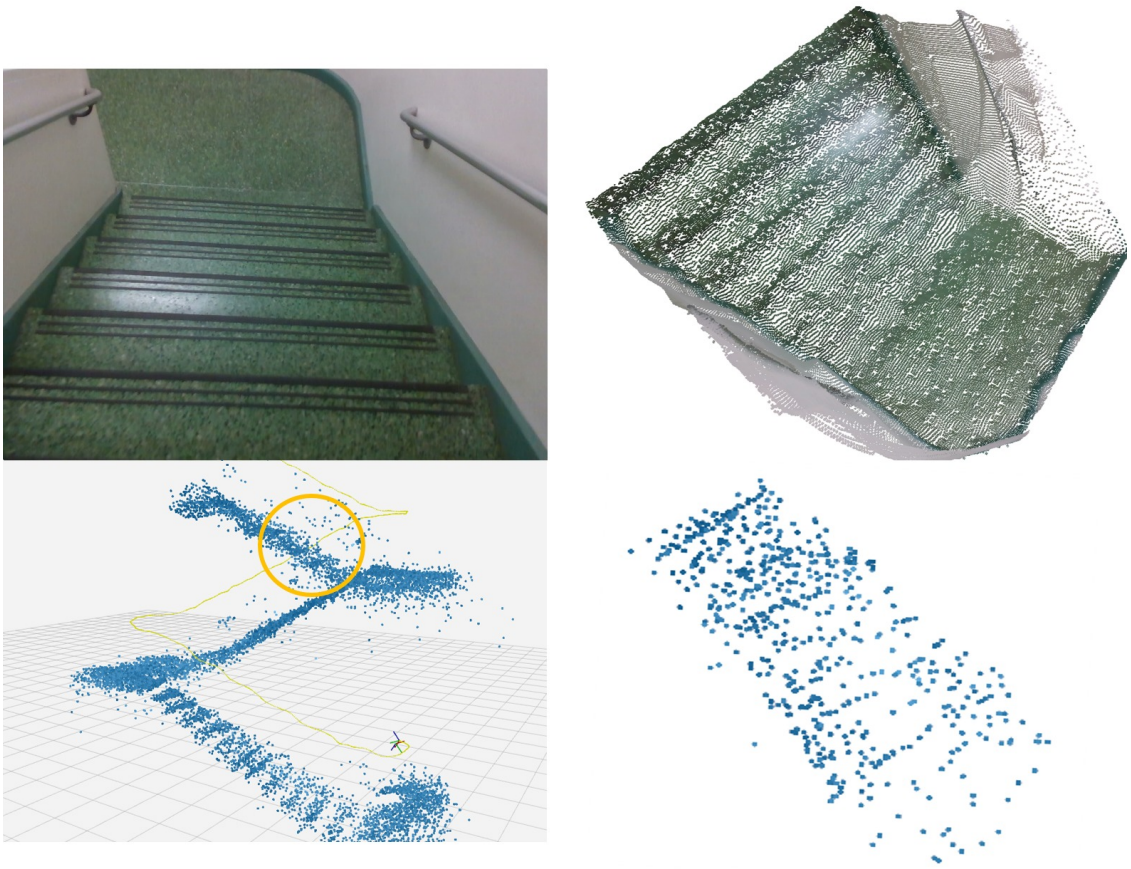


Figure 3.1: *Depth completion with Visual-Inertial Odometry (VIO) on the proposed VOID dataset* (best viewed in color at 5×). Bottom left: sparse reconstruction (blue) and camera trajectory (yellow) from VIO. The highlighted region is densified and zoomed in on the top right. Top left shows an image of the same region which is taken as input, and fused with the sparse depth image by our method. On the bottom right is the same view showing only the sparse points, insufficient to determine scene geometry and topology.

Inferring shape is ill-posed, even if the point cloud was generated with a lidar or structured light sensor. Filling the gaps relies on assumptions about the environment. Rather than designing ad-hoc priors, we wish to use the image to inform and restrict the set of possible scenes that are compatible with the given sparse points. Some methods use "ground truth" dense depth to learn a map from images to (point-estimates of) depth [SSN09]. Since an image is compatible with infinitely many shapes, a point estimate makes little sense in our context; others have used the image to compute a *prior* on dense depth [YWS19].

Summary of contributions

We use a predictive cross-modal criterion to score dense depth from images and sparse depth. This kind of approach is sometimes referred to as "self-supervised." Specifically, our method (i) exploits a set of constraints from temporal consistency (a.k.a. photometric consistency across temporally adjacent frames) to pose (forward-backward) consistency in a combination that has not been previously explored.

The challenge in using sparse depth as a supervisory (feedback) signal is precisely that it is sparse. Information at the points does not propagate to fill the domain where depth is defined. Some computational mechanism to "diffuse the information" from the sparse points to their neighbors is needed. Our approach proposes (ii) a simple method akin to using a piecewise planar *"scaffolding"* of the scene, sufficient to transfer the supervisory signal from sparse points to their neighbors. This yields a two-stage approach, where the sparse points are first processed to design the scaffolding ("meshing and interpolation") and then "refined" using the images as well as priors from the constraints just described.

One additional contribution of our approach is (iii) to launch the first visual-inertial + depth dataset. Since inertial sensors are now ubiquitous and typically co-located with cameras in many mobile devices from phones to cars, we hope this dataset will foster additional exploration into combining the complementary strengths of visual and inertial sensors.

To evaluate our method, since no other visual-inertial + depth benchmark is available, and to facilitate comparison with similar methods, we adopt the KITTI benchmark, where

a Velodyne (lidar) sensor provides sparse points with scale, unlike monocular SFM, but like visual-inertial odometry (VIO). Although the biases in lidar are different from VIO, this can be considered a baseline. Note that we only use the monocular stream of KITTI (not stereo) for fair comparison.

Among more fine-grained modeling choices and innovations, we use (iv) various photometric measures including L_1 distance and SSIM, and represent motion using exponential coordinates. The result is a single network that is simpler than competing methods, yet achieves state-of-the-art performance in the "unsupervised" KITTI benchmark (a misnomer). The supervision in the KITTI benchmark is really fusion from separate sensory channels, combined with ad-hoc interpolation and extrapolation. It is unclear whether the benefit from having such data is outweighed by the biases it induces on the estimate, and in any case such supervision does not scale, so we forgo (pseudo) ground truth annotations altogether.

3.2 Related Work

Supervised Depth Completion minimizes the discrepancy between ground truth depth and depth predicted from an RGB image and sparse depth measurements. Methods focus on network topology [MCK19, USS17, YWS19], optimization [CWL18, DVP18, ZF18], and modeling [EFK18, HFY18]. To handle sparse depth, [MCK19] employed early fusion, where the image and sparse depth are convolved separately and the results concatenated as the input to a ResNet encoder. [JCW18] proposed late fusion via a U-net containing two NASNet encoders for image and sparse depth and jointly learned depth and semantic segmentation, whereas [YWS19] used ResNet encoders for late fusion. [EFK18] proposed a normalized convolutional layer to propagate sparse depth and used a binary validity map as a confidence measure. [HFY18] proposed an upsampling layer and joint concatenation and convolution to deal with sparse inputs. All these methods require per-pixel ground-truth annotation. What is called "ground truth" in the benchmarks is actually the result of data processing and aggregation of 11 consecutive frames. We skip such supervision and just infer dense

depth by learning the cross-modal fusion from the virtually infinite volume of un-annotated data.

Unsupervised Depth Completion include [MCK19, SNC19, YWS19] who predict depth by minimizing the discrepancy between prediction and sparse depth input as well as the photometric error between the input image and its reconstruction from other viewpoints available only during training. [MCK19] used Perspective-n-Point (PnP) [LMF09] and Random Sample Consensus (RANSAC) [FB81] to align monocular image sequences for their photometric term with a second-order smoothness prior. Yet, [MCK19] does not generalize well to indoor scenes that contains many textureless regions (e.g. walls), where PnP with RANSAC may fail. [SNC19] used a local smoothness term, but instead minimized the photometric error between rectified stereo-pairs where pose is known. [YWS19] also leveraged stereo pairs and a more sophisticated photometric loss (SSIM [WBS04]), and replaced the generic smoothness term with a conditional prior to measure compatibility between the prediction and a learned depth model obtained by training a separate network on ground-truth depth. This method can be considered semi-unsupervised, and requires ground truth for training the prior. Using a network trained on a specific domain (e.g. outdoors) as a prior for an unsupervised method will not generalize when given extra data on a different domain (e.g. indoors). In contrast, our method is *fully unsupervised* and do not use any auxiliary ground-truth supervision. Moreover, we show that our method outperforms [MCK19, YWS19] on the KITTI depth completion benchmark [USS17] while using many fewer parameters.

Rotation Parameterization To construct the photometric consistency loss during training, an auxiliary pose network is needed if no camera poses are available. While the translational part of the relative pose can be modeled as $T \in \mathbb{R}^3$, the rotational part belongs to the special orthogonal group $R \in \mathrm{SO}(3) \doteq \{R \in \mathbb{R}^{3\times3} | R^\top R = I, \det(R) = +1\}$ [MSK12], which is represented by a 3×3 matrix. [KGC15] uses quaternions, which require an *additional* norm constraint; this is a soft constraint imposed in the loss function, and thus is not guaranteed. [FWS19, YS18, ZBS17] use Euler angles which requires the composition of several matrices that may result in the rotation matrix to no longer be orthogonal. We use the exponential map on SO(3) to map the output of the pose network to a rotation matrix. Though theo-

retically similar, we empirically found that the exponential map is more beneficial than the Euler angles in Sect. 3.8.

Our contributions are a simple, yet effective two-stage approach resulting in a large reduction in network parameters while achieving state-of-the-art performance on the unsupervised KITTI depth completion benchmark; using exponential parameterization of rotation for our pose network; a pose consistency term that enforces forward and backward motion to be the inverse of each other, and finally a new depth completion benchmark for visual-inertial odometry systems with indoor and outdoor scenes and challenging motion.

3.3 Methodology

We reconstruct a 3D scene given an RGB image $I_t : \mathbb{R}^2 \supset \Omega \mapsto \mathbb{R}^3_+$ and the associated set of sparse depth measurements $z_s : \Omega \supset \Omega_s \mapsto \mathbb{R}_+$.

We begin by assuming that world surfaces are graphs of smooth functions (charts) locally supported on a piecewise planar domain (scaffolding). We construct the scaffolding from the sparse point cloud ("interpolated depth" in Fig. 3.2) to obtain z_i, then learn a completion model refining z_i by leveraging the monocular sequences (I_{t-1}, I_t, I_{t+1}), of frames before and after the given time t, and the sparse depth z_s. We compose a surrogate loss $\mathcal{L}$ (3.3) for driving the training process, using an encoder-decoder architecture $f_\theta(\cdot)$ parameterized by weights θ, where the input is an image with its scaffolding (I_t, z_i), and the output is the dense depth $\hat{z} = f_\theta(I_t, z_i)$.

3.3.1 A Two-Stage Approach

As each sparse depth measurement can be viewed as a Dirac delta, [EFK18, HFY18, USS17] focused on propagating sparse depth through the network – a conventional convolution over the sparse depth input will give mostly zero activations. We, instead, circumvent this problem using our scaffolding.

However, the topology of the scene is not informed by the sparse depth input. We start

with a Delaunay triangulation [BDD96], resulting in a triangular mesh in Barycentric coordinates. We then approximate each surface using linear interpolation within the Barycentric coordinates. Our approach, therefore, is a two-stage pipeline, where we first generate a coarse approximation of the scene and then we feed the resulting depth image along with the associated RGB image to our network for refinement (Fig. 3.2). Our network achieves state-of-the-art performance on the unsupervised KITTI depth completion benchmark with half as many parameters as the prior art.

3.3.2 The Exponential Map

To construct our objective function (3.3), we leverage a pose network [KGC15] to regress the relative camera poses $g = (R, T) \in \mathrm{SE}(3) \doteq \{(R, T) | R \in \mathrm{SO}(3), T \in \mathbb{R}^3\}$. There exists a logarithmic map: $\log : \mathrm{SO}(3) \mapsto so(3)$, where $so(3)$ is the tangent space of $\mathrm{SO}(3)$, and an exponential map: $\exp : so(3) \mapsto \mathrm{SO}(3)$ – allowing us to map back and forth between $\mathrm{SO}(3)$ and $so(3)$. We use the logarithmic map to construct the pose consistency loss (3.9), and the exponential to map the output of the pose network $\omega \doteq [\omega_1, \omega_2, \omega_3]^\top \in \mathbb{R}^3$ as coordinates in $so(3)$ to a rotation matrix:

$$R(\omega) = \exp(\hat{\omega}) \doteq I + \hat{\omega} \sin \|\omega\|_2 + \hat{\omega}^2 (1 - \cos \|\omega\|_2) \tag{3.1}$$

where the hat operator $\hat{\cdot}$ maps $\omega \in \mathbb{R}^3$ to a skew-symmetric matrix [MSK12]

$$\hat{\omega} \doteq \begin{bmatrix} 0 & -\omega_3 & \omega_2 \\ \omega_3 & 0 & -\omega_1 \\ -\omega_2 & \omega_1 & 0 \end{bmatrix} \in \mathbb{R}^{3\times 3}. \tag{3.2}$$

With no explicit supervision, the training of our pose network is driven by a surrogate loss (3.4).

3.4 Network Architecture

We propose two encoder-decoder architectures with skip connections following the late fusion paradigm [JCW18, YWS19]. Each encoder has an image branch and a depth branch – the

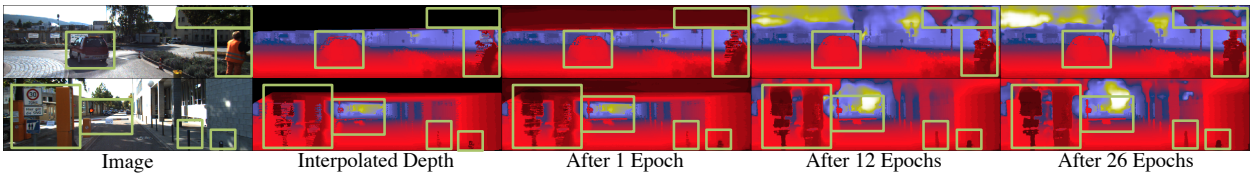


Figure 3.2: *Learning to refine* (best viewed at 5× with color). Our network learns to refine the input interpolated depth. Green rectangles highlight the regions for comparison throughout the course of training. The network first learns to copy the input and later learns to fuse information from RGB image to refine the interpolated depth (see row 1 pedestrian and row 2 street signs).

image branch contains 75% of the total features in the encoder and the depth branch 25%. The latent representation of the branches are concatenated and fed to the decoder. We propose a VGG11 encoder (≈ 5.7M parameters) containing 8 convolution layers for each branch for our best performing model and a VGG8 encoder (≈ 2.4M parameters) containing only 5 convolution layers for each branch for our light-weight model. Both VGG11 and VGG8 encoders are coupled to a generic decoder with ≈ 4 million parameters – giving us a total of ≈ 9.7M and ≈ 6.4M parameters, respectively. This is in contrast to other unsupervised methods [MCK19] (who follows early fusion and concatenates features from the two branches after the first convolution) and [YWS19] (late fusion) – both of whom use ResNet34 encoders with ≈ 23.8M and ≈ 14.8M parameters, respectively. Both [MCK19, YWS19] employ the same decoder with ≈ 4M parameters – totaling to ≈ 27.8M and ≈ 18.8M parameters, respectively.

Compared to [MCK19] and [YWS19], our VGG11 model has a 76.1% and 61.5% reduction in the encoder parameters and 65.1% and 48.4% overall, respectively. Our VGG8 model has a 89.9% and 83.9% reduction in the encoder and 80% and 66% overall compared to that of [MCK19] and [YWS19], respectively. Despite having fewer parameters, our method outperforms that of [MCK19, YWS19]. Moreover, we note that the performance of our VGG8 model is still comparable to that of VGG11 and still surpasses [MCK19] and [YWS19]. More details on our network architectures can be found in Supp Mat.

3.5 Loss Function

Our loss function is a linear combination of four terms that constrain (i) the photometric consistency between the observed image and its reconstructions from the monocular sequence, (ii) the predicted depth to be similar to that of the associated available sparse depth, (iii) the product of the predicted forward and backward relative poses to be the identity, and (iv) the prediction to adhere to local smoothness.

$$\mathcal{L} = w_{ph}L_{ph} + w_{sz}L_{sz} + w_{pc}L_{pc} + w_{sm}L_{sm} \tag{3.3}$$

where L_{ph} denotes photometric consistency, L_{sz} sparse depth consistency, L_{pc} pose consistency, and L_{sm} local smoothness. Each loss term L is described in the next subsections and the associated weight w in Sect. 3.7.

3.5.1 Photometric Consistency

We enforce temporal consistency by minimizing the color and structural discrepancy between each observed image I_t and its reconstruction $\hat{I}_\tau$ from temporally adjacent images I_τ, where $\tau \in T \doteq \{t-1, t+1\}$:

$$\hat{I}_\tau(x) = I_\tau\big(\pi g_{\tau t}\mathbf{K}^{-1}\bar{x}z(x)\big) \tag{3.4}$$

where $\bar{x} = [x^T\ 1]^T$ are the homogeneous coordinates of $x \in \Omega$, $g_{\tau t} \in \mathrm{SE}(3)$ is the relative pose of the camera from time t to τ, $\mathbf{K}$ denotes the camera intrinsics, and π refers to the perspective projection.

Our photometric consistency term is a two-part loss corresponding to color and structural consistency between the observed image I_t and its reconstructions $\hat{I}_\tau$.

Color Consistency measures the average per pixel reprojection residual with an $L1$ penalty:

$$l_{co} = \frac{1}{|\Omega|}\sum_{\tau\in T}\sum_{x\in\Omega}|I_t(x) - \hat{I}_\tau(x)| \tag{3.5}$$

Structural Consistency uses `SSIM`, a perceptual metric that is invariant to local illumination changes. We apply `SSIM` to 3×3 image patches centered at location x for an image

I_t and its reconstruction $\hat{I}_\tau$. As a high SSIM score means I_t and $\hat{I}_\tau$ are similar, we subtract the score from 1 to denote a distance metric:

$$l_{st} = \frac{1}{|\Omega|} \sum_{\tau \in T} \sum_{x \in \Omega} 1 - \texttt{SSIM}(I_t(x), \hat{I}_\tau(x)) \tag{3.6}$$

Our photometric consistency loss can therefore be written as the linear combination of the color and structural consistency terms weighted by w_{co} and w_{st} (Sect. 3.7), respectively:

$$L_{ph} = w_{co} l_{co} + w_{st} l_{st} \tag{3.7}$$

3.5.2 Sparse Depth Consistency

Our sparse depth consistency term provides our predictions with metric scale by encouraging the predictions $\hat{z}$ to be similar to that of the available sparse depth z_s. Our sparse depth consistency loss is the $L1$-norm of the difference between the predicted depth $\hat{z}$ and the sparse depth z_s averaged over Ω_s (the support of the sparse depth)

$$L_{sz} = \frac{1}{|\Omega_s|} \sum_{x \in \Omega_s} |\hat{z}(x) - z_s(x)|. \tag{3.8}$$

3.5.3 Pose Consistency

A pose network takes an ordered pair of images (I_t, I_τ) and outputs the relative pose $g_{\tau t} \in$ SE(3) (forward pose). When a temporally swapped pair (I_τ, I_t) is fed to the network, the network is expected to output $g_{t\tau}$ (backward pose) – the inverse of $g_{\tau t}$, *i.e.*, $g_{\tau t} \cdot g_{t\tau} = e \in$ SE(3). The forward-backward pose consistency thus penalizes the deviation of the composed pose from the identity:

$$L_{pc} = \| \log(g_{\tau t} \cdot g_{t\tau}) \|_2^2 \tag{3.9}$$

where $\log : \text{SE}(3) \mapsto se(3)$ is the logarithmic map.

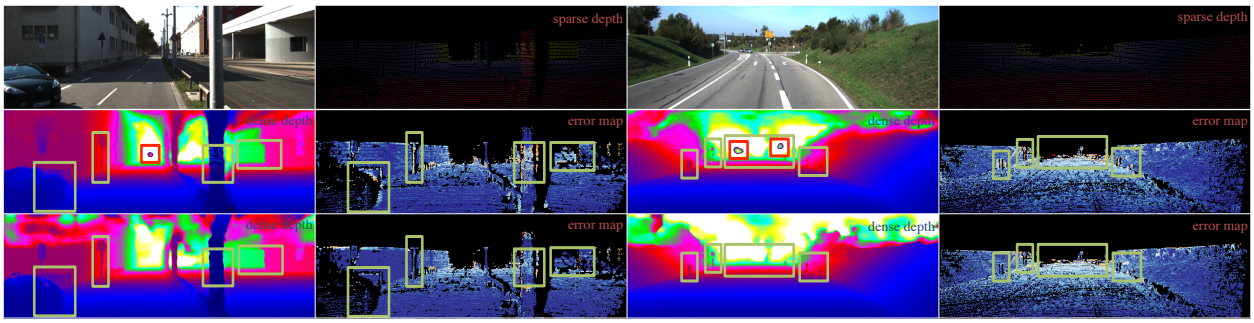


Figure 3.3: *Qualitative evaluation on KITTI benchmark.* Row 1: input image and sparse depth. Row 2: results of [MCK19] taken from the KITTI online test results. Row 3: our results on the KITTI online test server. Warmer colors in the error map denote higher error. Green rectangles highlight regions for detail comparison. Our method performs better in general, particularly on thin structures and far regions. Also, the results of [MCK19] exhibit artifacts resembling scanlines of the Velodyne and "circles" for far away regions (highlighted in red).

3.5.4 Local Smoothness

We impose a smoothness loss on the predicted depth $\hat{z}$ by applying an $L1$ penalty to the gradients in both the x and y directions of the predicted depth $\hat{z}$:

$$L_{sm} = \frac{1}{|\Omega|} \sum_{x \in \Omega} \lambda_X(x) |\partial_X \hat{z}(x)| + \lambda_Y(x) |\partial_Y \hat{z}(x)| \tag{3.10}$$

where $\lambda_X = e^{-|\partial_X I_t(x)|}$ and $\lambda_Y = e^{-|\partial_Y I_t(x)|}$ are the edge-awareness weights to allow for discontinuities in regions corresponding to object boundaries.

3.6 Datasets

3.6.1 KITTI Benchmark

We evaluate our approach on the KITTI depth completion benchmark [USS17]. The dataset provides $\approx 80,000$ raw image frames and associated sparse depth maps. The sparse depth maps are the raw output from the Velodyne lidar sensor, each with a density of $\approx 5\%$. The ground-truth depth map is created by accumulating the neighbouring 11 raw lidar scans, with dense depth corresponding to the bottom 30% of the images. We use the officially

selected 1,000 samples for validation and we apply our method to 1,000 testing samples, with which we submit to the official KITTI website for evaluation. The results are reported in Table 3.2.

3.6.2 VOID Benchmark

While KITTI provides a standard benchmark for evaluating depth completion in the driving scenario, there exists no standard depth completion benchmark for the indoor scenario. [MCK19, YWS19] used NYUv2 [SHK12] – an RGB-D dataset – to develop and evaluate their models on indoor scenes. Yet, each perform a different evaluation protocol with different sparse depth samples – varying densities of depth values were randomly sampled from the depth frame, preventing direct comparisons between methods. Though this is reasonable as a proof of concept, it is not realistic in the sense that no sensor measures depth at random locations.

The VOID dataset. We propose a new publicly available dataset for a real world use case of depth completion by bootstrapping sparse reconstruction in *metric* space from a SLAM system. While it is well known that metric scale is not observable in the purely image-based SLAM and SFM setting, it has been resolved by the recent advances in VIO [JS11, MR07], where real-time pose and structure estimation can be realized in a gravity-aligned and scaled reference frame using a inertial measurement unit (IMU). To this end, we leverage an Extended Kalman Filter (EKF) based VIO system, atop which we construct our dataset and develop our depth completion model. While there are some visual-inertial datasets (e.g. TUM-VI [SGD18] and PennCOSYVIO [PSD17]), they do not have per-frame dense depth measurements for cross-modal validation, and are also relatively small – rendering them unsuitable for training deep learning models.

Our dataset is dubbed "Visual Odometry with Inertial and Depth" or "VOID" for short and is comprised of RGB video streams and inertial measurements for *metric* reconstruction along with per-frame dense depth for cross-modal validation.

Data acquisition. Our data was collected using the latest Intel RealSense D435i camera [1], which was configured to produce synchronized accelerometer and gyroscope measurements at 400 Hz, along with synchronized VGA-size (640×480) RGB and depth streams at 30 Hz. The depth frames are acquired using active stereo and is aligned to the RGB frame using the sensor factory calibration. All the measurements are time-stamped.

The SLAM system we use is based on [JS11] – an EKF-based VIO model. While the VIO recursively estimates a joint posterior of the state of the sensor platform (e.g. pose, velocity, sensor biases, and camera-to-IMU alignment) and a small set of reliable feature points, the 3D structure it estimates is extremely sparse – typically $20 \sim 30$ feature points (in-state features). To facilitate 3D reconstruction, we track a moderate amount of out-of-state features in addition to the in-state ones, and estimate the depth of the feature points using auxiliary filters [MSK12].

To give some flavor of the VOID dataset, Fig. 3.4 shows a set of images (top inset) sampled from video sequences in VOID, and output of our visual-inertial odometry (VIO) system (bottom), where the blue pointcloud is the sparse reconstruction of the underlying scene and the yellow trace is the estimated camera trajectory.

The benchmark. We evaluate our method on the VOID depth completion benchmark, which contains 56 sequences in total, both indoor and outdoor with challenging motion. Typical scenes include classrooms, offices, stairwells, laboratories, and gardens. Of the 56 sequences, 48 sequences ($\sim 40K$ frames) are designated for training and 8 sequences for testing, from which we sampled 800 frames to construct the testing set. Our depth completion benchmark provides sparse depth images at 3 density levels. We configured our SLAM system to track and estimate depth of 1500, 500 and 150 feature points, corresponding to $0.5\%, 0.15\%$ and 0.05% density of VGA size, which are then used in the depth completion task.

[1] https://realsense.intel.com/depth-camera/

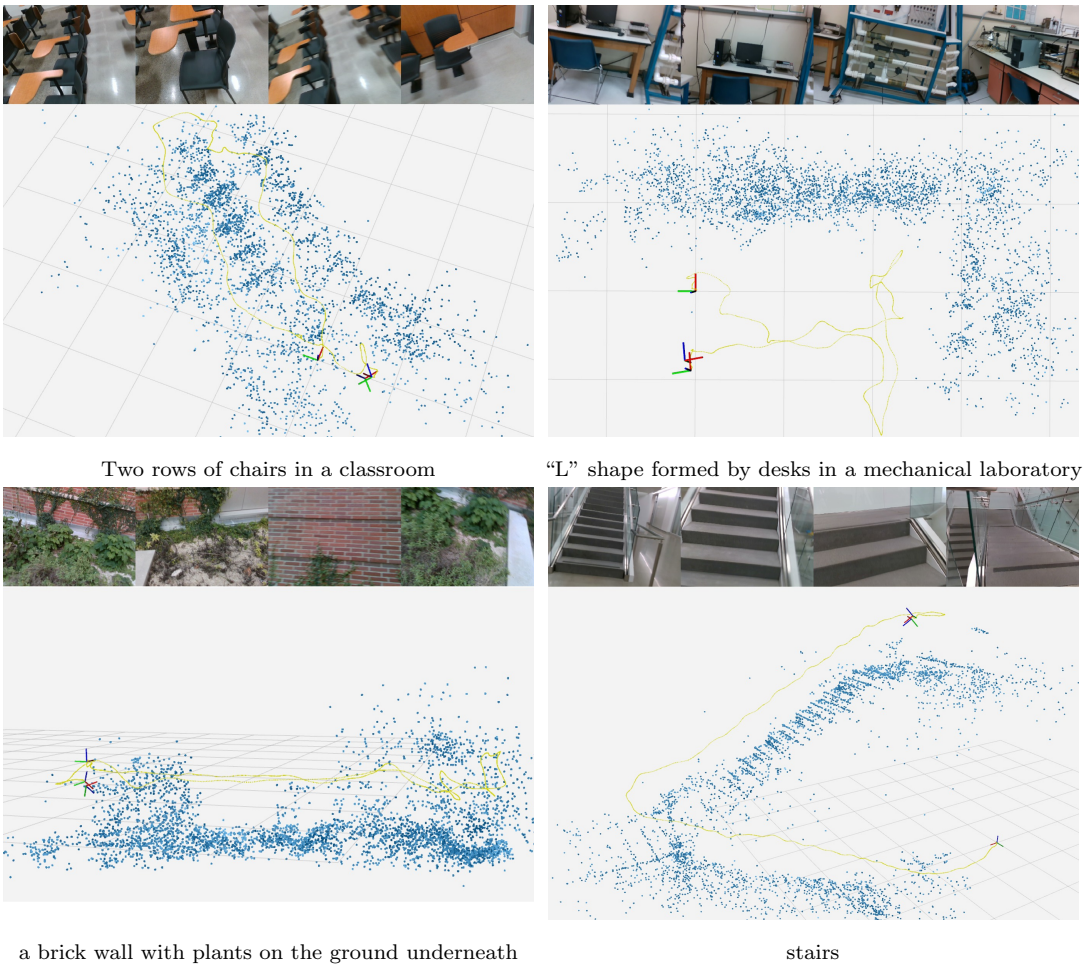


Figure 3.4: *Sample sequences in VOID dataset* (best viewed in color at 5×). In each panel, the top inset shows 4 sample images of a video sequence in our VOID dataset; the bottom shows the sparse pointcloud reconstruction (blue) and camera trajectory (yellow) from our VIO.

3.7 Implementation Details

Our approach was implemented using TensorFlow [ABC16]. With a Nvidia GTX 1080Ti, training takes ≈ 90 hours for our VGG11 model and ≈ 70 hours for our VGG8 model on KITTI depth completion benchmark (Sect. 3.6.1) for 30 epochs; whereas training takes ≈ 10 hours and ≈ 7 hours on the VOID benchmark (Sect. 3.6.2) for 10 epochs. Inference takes ≈ 22 ms per image. We used Adam [KB14] with $\beta_1 = 0.9$ and $\beta_2 = 0.999$ to optimize our network end-to-end with a base learning rates of 1.2×10^{-4} for KITTI and 1×10^{-4} for VOID. We decrease the learning rate by half after 18 epochs for KITTI and 6 epochs for

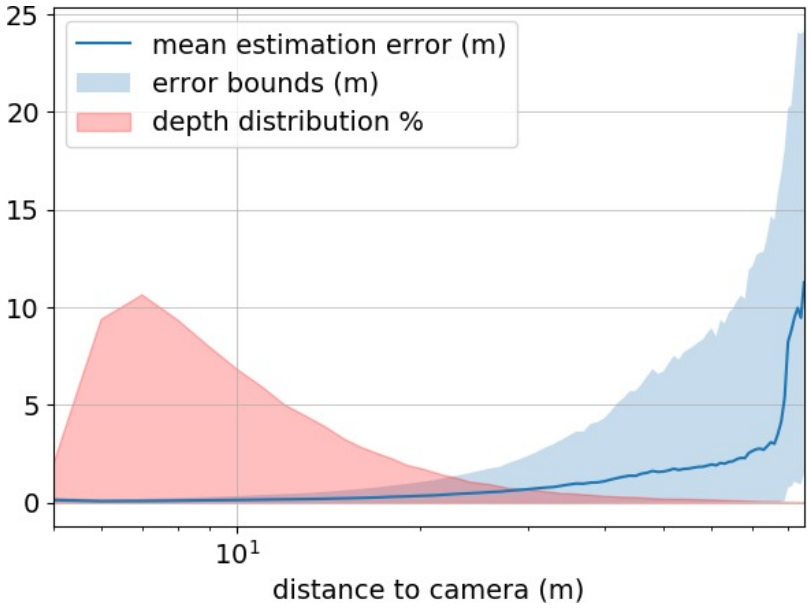


Figure 3.5: *Error characteristics of our model on KITTI.* The abscissa shows the distance of sparse data points measured by Velodyne, of which the percentage of all the data points is shown in red; the blue curve shows the mean absolute error of the estimated depth at the given distance, of which the 5-*th* and 95-*th* percentile enclose the light blue region.

VOID, and again after 24 epochs and 8 epochs, respectively. We train our network with a batch size of 8 using a 768×320 resolution for KITTI and 640×480 for VOID. We are able to achieve our results on the KITTI benchmark using the following set of weights for each term in our loss function: $w_{ph} = 1.00$, $w_{co} = 0.20$, $w_{st} = 0.40$, $w_{sz} = 0.20$, $w_{pc} = 0.10$ and $w_{sm} = 0.01$. For the VOID benchmark, we increased w_sz to 1.00 and w_{sm} to 0.10. We do not use any data augmentation.

3.8 Experiments

3.8.1 KITTI Depth Completion Benchmark

We compare the performance of our approach with recent unsupervised depth completion methods on the official KITTI depth completion benchmark in Table 3.2 using error metrics in Table 3.1 and show quantitative results in Fig. 3.3. The results of the methods listed are taken directly from their papers. We note that [YWS19] only reported their result in

Metric	units	Definition
MAE	*mm*	$\frac{1}{\lvert\Omega\rvert}\sum_{x\in\Omega}\lvert\hat{z}(x)-z_{gt}(x)\rvert$
RMSE	*mm*	$\left(\frac{1}{\lvert\Omega\rvert}\sum_{x\in\Omega}\lvert\hat{z}(x)-z_{gt}(x)\rvert^2\right)^{1/2}$
iMAE	*1/km*	$\frac{1}{\lvert\Omega\rvert}\sum_{x\in\Omega}\lvert 1/\hat{z}(x)-1/z_{gt}(x)\rvert$
iRMSE	*1/km*	$\left(\frac{1}{\lvert\Omega\rvert}\sum_{x\in\Omega}\lvert 1/\hat{z}(x)-1/z_{gt}(x)\rvert^2\right)^{1/2}$

Table 3.1: *Error Metrics* for evaluating KITTI and VOID depth completion benchmarks, where z_{gt} is the ground truth.

their paper and do have have an entry for KITTI depth completion benchmark for their unsupervised model. Hence, we compare qualitatively with the prior art [MCK19]. Our VGG11 model outperforms the state-of-the-art [YWS19] on every metric by as much as 12.8% on MAE, 7.4% on RMSE, 9.1% on iMAE while using 48.4% fewer parameters. Our light-weight VGG8 model also outperforms [YWS19] on MAE by 11.3%, RMSE by 7.8% and iMAE by 3% while having 66% fewer parameters; [YWS19] beat our light-weight model by 2.2% on iRMSE. We note that [YWS19] trains a separate network using ground-truth depth and uses it as supervision to train their model for depth completion. Moreover, [YWS19] exploits rectified stereo-imagery where the pose of the cameras is known; whereas, we learn our pose by jointly training the pose network with our depth predictor. In comparison to [MCK19] (who also leverages monocular videos), our VGG11 model outperforms them by 14.5% on MAE, 10% on RMSE, 23.6% on iMAE, and 12.5% on iRMSE while using 65.1% fewer parameters. Our VGG8 model outperforms [MCK19] by 13.1% on MAE, 10.4% on RMSE, 18.5% on iMAE, and 10.1% on iRMSE while using 80% fewer parameters. We also note that the qualitative results of [MCK19] contains artifacts such as apparent scanlines of the Velodyne and "circles" for far regions. As an introspective exercise, we plot the mean error of our model at varying distances on the KITTI validation set (Fig. 3.5) and overlay it with the ground truth depth distribution to show that our model performs very well in distances that matter in real-life scenarios. Our performance begins to degrade at distances larger than 80 meters; this is due to the lack of sparse measurements and insufficient parallax – problems that plague methods relying on multi-view supervision.

Method	MAE	RMSE	iMAE	iRMSE
Schneider et al. [SSP16]	605.47	2312.57	2.05	7.38
Ma et al. [MCK19]	350.32	1299.85	1.57	4.07
Yang et al. [YWS19]	343.46	1263.19	1.32	3.58
Ours VGG8	304.57	**1164.58**	1.28	3.66
Ours VGG11	**299.41**	1169.97	**1.20**	**3.56**

Table 3.2: *KITTI depth completion benchmark.* We compare our model to unsupervised methods on the KITTI depth completion benchmark [USS17]. Our VGG11 model outperforms state-of-the-art [YWS19] across all metrics while using 48.4% less parameters. Our light-weight (VGG8) model achieves similar performance and in fact marginally outperforms our VGG11 model despite having 34% fewer parameters than our VGG11 model. Moreover, our VGG8 model outperforms [MCK19] and across all metrics and [YWS19] on MAE, RMSE, and iMAE despite having 80% and 66% fewer parameters, respectively.

3.8.2 KITTI Depth Completion Ablation Study

We analyze the effect brought by each of our contributions through a quantitative evaluation on the KITTI depth completion validation set (Table 3.3). We see that our two baseline models (row 1 and 2), Scaffolding and vanilla model trained without interpolation, perform poorly in comparison to the models that are trained with interpolated depth as input – showcasing the effectiveness of our refinement approach. Although the loss functions are identical, we see that exponential parameterization consistently improves over Euler angles across all metrics. While other works [FWS19, WMZ18, YS18] train their pose network using the photometric error as a surrogate loss with no additional constraint, we show that it is in fact beneficial to impose our pose consistency constraint (Sect. 3.9). By constraining the forward and backward poses to be inverse of each other, we are able to obtain a more accurate pose resulting in better depth prediction. Our experiments verify this claim as we see an improvement in MAE by 2.3%, RMSE by 1.3%, iMAE by 5.5%, and iRMSE by 3.9% in Table 3.3. We note that the improvement does not seem significant on KITTI as the motion

Model	Encoder	Rot.	MAE	RMSE	iMAE	iRMSE
Scaffolding	-	-	443.57	1990.68	1.72	6.43
$L_{ph} + L_{sz} + L_{sm}$*	VGG11	Euler	347.14	1330.88	1.46	4.22
$L_{ph} + L_{sz} + L_{sm}$	VGG11	Euler	327.84	1262.46	1.31	3.87
$L_{ph} + L_{sz} + L_{sm}$	VGG11	Exp.	312.10	1255.21	1.28	3.86
$L_{ph} + L_{sz} + L_{pc} + L_{sm}$	VGG11	Exp.	**305.06**	1239.06	**1.21**	**3.71**
$L_{ph} + L_{sz} + L_{pc} + L_{sm}$	VGG8	Exp.	308.81	**1230.85**	1.29	3.84

Table 3.3: *KITTI depth completion ablation study.* We compare variants of our model on the KITTI depth completion validation set. Each model is denoted by its loss function. The results of Scaffolding Only is produced using linear interpolation over a triangular mesh; we assign average depth to regions with missing interpolated depth. It is clear that scaffolding alone (row 1) and our baseline model trained *without* interpolated depth (row 2, indicated by *) do poorly compared to our models that combine both (rows 3-6). Our full model using VGG11 produces the best overall results and achieves state-of-the-art on the test set Table 3.2. We note that our light-weight VGG8 model achieve similar performance and even marginally beating our VGG11 model on the RMSE metric despite having 34% fewer parameters.

is mostly planar; however, when predicting non-trivial 6 DoF motion (Sect. 3.8.4), we see a significant boost when employing this term. Our model trained with the full loss function produces the best results (bolded in Table 3.2) and is the state-of-the-art for unsupervised KITTI depth completion benchmark. We further propose a light-weight (VGG8) model that only contains $\approx$ 6.4M parameters. Although our light-weight model has 3.3M fewer (34% reduction) parameters than our VGG11 model, we note that the performance does not degrade by much – our VGG8 model only trails the VGG11 model by 1.2% in MAE, 6.6% in iMAE, 3.5% in iRMSE, and even marginally beating our VGG11 model on RMSE by 0.7% on the KITTI validation set.

3.8.3 VOID Depth Completion Benchmark

We evaluate our method on the VOID depth completion benchmark for all three density levels (Table 3.4) using error metrics in Table 3.1. As the photometric loss is largely dependent on obtaining the correct pose, we additionally propose a hybrid model, where the relative camera poses from our visual-inertial SLAM system are used to construct the photometric loss to show a upper bound on performance.

In contrast to the KITTI depth completion benchmark, which provides $\approx 5\%$ sparse depth over the image domain concentrated on the bottom third of the image, the VOID benchmark only provides $\approx 0.5\%$, $\approx 0.15\%$ and $\approx 0.05\%$ densities in sparse depth (10, 33, and 100 times less than KITTI). Yet, our method is still able to produce reasonable results for indoor scenes with a MAE of ≈ 8.5 centimeters on 0.5% density and ≈ 17.9 centimeters when given only 0.05%. As most scenes contain textureless regions, sparse depth supervision becomes important as photometric reconstruction is unreliable. Hence, we see a degrade in performance as the density decreases. Yet, we degrade gracefully: as the density decreases by 100X, our error only doubles. Also, we observe systematic performance improvement in all the evaluation metrics (Table 3.4) when replacing the pose network with SLAM pose. This can be largely attributed to the necessity for the correct pose to minimize photometric error during training. Our pose network may not be able to consistently predict the correct pose due to the challenging motion of the dataset. Fig. 3.6 and 3.7 show some sample RGB images with the densified depth images back-projected to 3D, colored, and viewed from a different vantage point.

3.8.4 VOID Depth Completion Ablation Study

To better understand the effect of rotation parameterization and our pose consistency loss (3.9) on the depth completion task, we compare variants of our model and again replace the pose network with SLAM pose to show an upper-bound on performance. Although exponential outperforms Euler parameterization, we note that their results are in fact 29.2 and 32.9% worse than using SLAM pose on MAE, 18.2 and 30.1% worse on RMSE, 33% and 34%

Density	Pose From	MAE	RMSE	iMAE	iRMSE
$\sim 0.5\%$	PoseNet	85.05	169.79	48.92	104.02
	SLAM	73.14	146.40	42.55	93.16
$\sim 0.15\%$	PoseNet	124.11	217.43	66.95	121.23
	SLAM	118.01	195.32	59.29	101.72
$\sim 0.05\%$	PoseNet	179.66	281.09	95.27	151.66
	SLAM	174.04	253.14	87.39	126.30

Table 3.4: *Depth completion on VOID with sparse input of varying density.* The VOID dataset contains VGA size images (480×640) of both indoor and outdoor scenes with challenging motion. For "Pose From", SLAM refers to relative poses estimated by a SLAM system, and PoseNet refers to relative poses predicted by a pose network.

worse on iMAE, and 29% and 34.7% worse on iRMSE, respectively. However, we observe a performance boost when applying our pose consistency term and our model improves over exponential without pose consistency by 17.7% on MAE, 5.2% on RMSE, 23.4% on iMAE, and 20.6% on iRMSE. Moreover, it only trails the one trained with SLAM pose by 14% on MAE, 13.8% on RMSE, 13% on iMAE, and 10.4% on iRMSE. This trend still holds when density decreases (Table 3.4). This suggests that despite the additional constraint, the pose network still have some difficulties predicting the pose due to the challenging motion. This finding, along with results from Table 3.4, sheds light to the usage of classic SLAM systems in the era of deep learning, which also urges us to develop and test pose networks on the VOID dataset which features non-trivial 6 DoF motion – much more challenging than the mostly-planar motion found in the KITTI dataset.

3.9 Pose Ablation Study

In this section, we directly evaluate the pose network on the KITTI odometry dataset in Table 3.6. We show qualitative results on the trajectory obtained by chaining pairwise

Figure 3.6: *Qualitative evaluation on VOID benchmark.* Top: Input RGB images. Bottom: Densified depth images back-projected to 3D, colored, and viewed from a different vantage point.

camera poses estimated by each pose network in Fig. 3.8 and provide an analysis of the results in Sect. 3.9.2.

3.9.1 Pose Evaluation Metrics

To evaluate the performance of the pose network and its variants, we adopt two most widely used metrics in evaluating simultaneous localization and mapping (SLAM) systems: absolute trajectory error (ATE) and relative pose error (RPE) [SEE12] along with two novel metrics tailored to the evaluation of pose networks.

Given a list of estimated camera poses $\hat{g}^T \doteq \{\hat{g}_1, \hat{g}_2, \cdots, \hat{g}_T\}$, where $\hat{g}_t \in \mathrm{SE}(3)$, relative to a fixed world frame, and the list of corresponding ground truth poses $g^T \doteq \{g_1, g_2, \cdots, g_T\}$,

Figure 3.7: *More Qualitative results on VOID dataset.* In each panel, the left shows a sample RGB image fed to our depth completion network as input; the right shows the completed depth map back-projected to 3D, colored, and viewed from a different vantage point. Our method recovers the scene structure with details at various ranges in both indoor and outdoor settings.

where $g_t \in \mathrm{SE}(3)$, ATE reads

$$\mathrm{ATE}(\hat{g}^T, g^T) = \sqrt{\frac{1}{T}\sum_{t=1}^{T} \|\mathrm{trans}(g_t^{-1}\hat{g}_t)\|_2^2} \tag{3.11}$$

where the function trans : $\mathrm{SE}(3) \mapsto \mathbb{R}^3$ extracts the translational part of a rigid body transformation. ATE is essentially the root mean square error (RMSE) of the translational part of the estimated pose over all time indices. [ZBS17] proposed a "5-frame" version of ATE (ATE-5F) – the root mean square of ATE of a 5-frame sliding window over all time indices, which we also incorporate.

While ATE measures the overall estimation accuracy of the whole trajectory – suitable for evaluating full-fledged SLAM systems where a loop closure module presents, it does not faithfully reflect the accuracy of our pose network since 1) our pose network is designed to

Method	MAE	RMSE	iMAE	iRMSE
PoseNet + Eul.	108.97	212.16	64.54	142.64
PoseNet + Exp.	103.31	179.05	63.88	131.06
PoseNet + Exp. + L_{pc}	85.05	169.79	48.92	104.02
SLAM Pose	**73.14**	**146.40**	**42.55**	**93.16**

Table 3.5: *VOID depth completion benchmark and ablation study.* We compare the variants of our pose network. SLAM Pose replaces the output of pose network with SLAM estimated pose to gauge an upper bound in performance. When using our pose consistency term with exponential parameterization, our method approaches the performance of our method when using SLAM pose.

estimate pairwise poses, and 2) thus by simply chaining the pose estimates overtime, the pose errors at earlier time instants are more pronounced. Therefore, we also adopt RPE to measure the estimation accuracy locally:

$$\mathrm{RPE}(\hat{g}^T, g^T; \Delta) = \sqrt{\frac{1}{T-\Delta}\sum_{t=1}^{T-\Delta} \|\mathrm{trans}\big((g_t^{-1} g_{t+\Delta})^{-1}(\hat{g}_t^{-1}\hat{g}_{t+\Delta})\big)\|_2^2} \tag{3.12}$$

which is essentially the end-point relative pose error of a sliding window averaged over time. By measuring the end-point relative pose $\hat{g}_{t\tau} \doteq \hat{g}_t^{-1}\hat{g}_{t+\Delta}$, where $\tau \doteq t + \Delta$, over a sliding window $[t, t+\Delta]$, we are able to focus more on the relative pose estimator (the pose network) itself rather than the overall localization accuracy. In our evaluation, we choose a sliding window of size 1, i.e., $\Delta = 1$. However, RPE is affected only by the accuracy of the translational part of the estimated pose, as we expand the relative pose error:

$$g_{t\tau}^{-1}\hat{g}_{t\tau} = (R_{t\tau}, T_{t\tau})^{-1} \cdot (\hat{R}_{t\tau}, \hat{T}_{t\tau}) \tag{3.13}$$

$$= (R_{t\tau}^\top \hat{R}_{t\tau}, -R_{t\tau}^\top \hat{T}_{t\tau} + T_{t\tau}) \tag{3.14}$$

leading to $\mathrm{trans}(g_{t\tau}^{-1}\hat{g}_{t\tau}) = -R_{t\tau}\hat{T}_{t\tau} + T_{t\tau}$, where the rotational part $\hat{R}_{t\tau}$ of the estimated pose disappears! Therefore, to better evaluate the rotation estimation, and, more importantly, to study the effect of different rotation parameterization and the pose consistency term, we

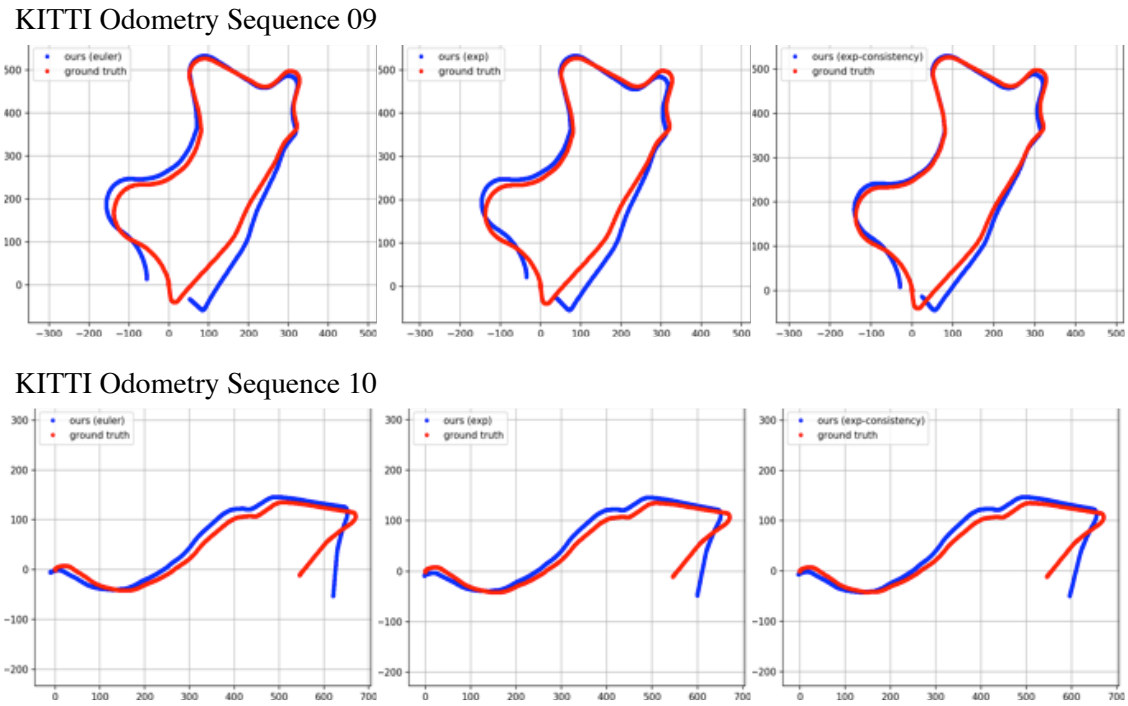


Figure 3.8: *Qualitative Pose Ablation Study KITTI Odometry Sequence 09 and 10.* We perform an ablation study on our pose representation by jointly training our depth completion network and pose network on KITTI depth completion dataset and testing only the pose network on KITTI Odometry sequence 09 and 10. We obtain the camera trajectories by chaining the pairwise camera poses estimated by our pose network. We observe that the trajectory of our method using exponential parameterization trained with pose consistency (Sect. 3.5.3) is most closely aligned with the ground-truth trajectory.

propose the relative rotation error (RRE) metric:

$$\mathrm{RRE}(\hat{g}^T, g^T; \Delta) = \sqrt{\frac{1}{T-\Delta} \sum_{t=1}^{T-\Delta} \| \log\left(\mathrm{rot}(g_{t\tau}^{-1} \hat{g}_{t\tau})\right) \|_2^2} \tag{3.15}$$

where $\mathrm{rot} : \mathrm{SE}(3) \mapsto \mathrm{SO}(3)$ extracts the rotational part of a rigid body transformation, and $\log : \mathrm{SO}(3) \mapsto \mathbb{R}^3$ is the logarithmic map for rotations.

3.9.2 Ablation Study on KITTI Odometry

We perform an ablation study on the effects of our pose parameterizations and our pose consistency in Table 3.6 and provide qualitative results showing the trajectory predicted by our pose network in Fig. 3.8. We jointly trained our depth completion network and our pose network on the KITTI depth completion dataset and evaluate the pose network on sequence 09 and 10 of the KITTI Odometry dataset.

For sequence 09, our pose network using exponential parameterization performs comparably to Euler angles on the ATE-5F and RPE metrics while outperforming Euler by $\approx 20\%$ on ATE and $\approx 3.4\%$ on RRE. This result suggests that while within a small window Euler and exponential perform comparably on translation, exponential is a better pose parameterization and globally more correct. We additionally see that exponential outperforms Euler angles on all metrics in sequence 10.

Our best results are achieved using exponential parameterization with our pose consistency term (Sect. 3.5.3): on sequence 09, it outperformed Euler and exponential without pose consistency by $\approx 47.1\%$ and $\approx 28.9\%$ on ATE, $\approx 12.1\%$ and $\approx 13\%$ on RPE, $\approx 10.8\%$ and $\approx 7.6\%$ on RRE, respectively, and both by $\approx 12.1\%$ on ATE-5F. On sequence 10, it outperformed Euler and exponential by $\approx 24\%$ and $\approx 2.3\%$ on ATE, $\approx 13.8\%$ and $\approx 11\%$ on RPE, and $\approx 13.1\%$ and $\approx 3.1\%$ on RRE, respectively. It also beat Euler by $\approx 12\%$ on RPE and is comparable to exponential on the metric.

3.10 Discussion

In this work, we introduced a two-stage approach that achieves state-of-the-art performance on the KITTI depth completion benchmark. By learning a model to refine the scaffolding built from sparse points, we show that we can bypass the sparse input problem that previous works have tried to solve by using sparsity-invariant operations. We additionally explored rotation parameterization and proposed a pose consistency constraint that enforced forward-backward motion consistency. This consistency term contributed to our performance on both

Pose	ATE (m)	ATE-5F (m)	RPE (m)	RRE ($^\circ$)
Sequence 09				
Euler	34.38	0.091	0.107	0.176
Exp.	27.57	0.091	0.108	0.170
Exp. w/ Consistency	**18.18**	**0.080**	**0.094**	**0.157**
Sequence 10				
Euler	32.37	0.067	0.094	0.251
Exp.	25.18	**0.059**	0.091	0.225
Exp. w/ Consistency	**24.60**	**0.059**	**0.081**	**0.218**

Table 3.6: *Quantitative Pose Ablation Study KITTI Odometry Sequence 09 and 10.* We perform an ablation study on our pose representation by jointly training our depth completion network and pose network on KITTI depth completion dataset and testing only the pose network on KITTI Odometry sequence 09 and 10. We evaluate the performance of each pose network using metrics described in Sect. 3.9.1. While performance of exponential parameterization and Euler angles are similar on ATE-5F, and RPE, exponential outperforms Euler angles in ATE and RRE on both sequences. Our model using exponential with pose consistency performs the best.

the KITTI and our newly proposed VOID dataset benchmarks. We showed that our pose consistency term improves the predicted pose on both datasets and also improves our results on the depth completion task. However, we note that the performance of our model using a pose network still trails the model trained with SLAM pose on the VOID dataset. This can be attributed to the challenging motion on VOID as opposed to the planar motion on KITTI.

While deep networks have attracted a lot of attention as a general framework to solve an array of problems, we must note that pose may be difficult to learn on datasets with non-trivial 6 DoF motion – which the SLAM community has studied for decades. We hope that VOID will serve as a platform to develop models that can handle challenging motion and further foster fusion of multi-sensory data. Furthermore, we show that deep learning can be

applied to predict the dense reconstruction from extremely sparse point clouds (e.g. features tracked by SLAM). We also show that we can improve the performance of our model by directly using pose from a SLAM system instead of pose network. These findings motivate a possible marriage between SLAM and deep learning that can benefit one another.

CHAPTER 4

Visual-Inertial Scene Representation for 3D Object Detection

4.1 Introduction

We deem an "object detector" to be a system that takes as input *images* and produces as output decisions as to the presence of *objects in the scene.* We design one based on the following premises: (a) Objects exist in the scene, not in the image; (b) they persist, so confidence on their presence should grow as more evidence is accrued from multiple (test) images; (c) once seen, the system should be aware of their presence even when temporarily not visible; (d) such awareness should allow it to predict when they will return into view, based on scene geometry and topology; (e) objects have characteristic shape and *size* in 3D, and vestibular (inertial) sensors provide a global scale and orientation reference that the system should leverage on.

Detecting objects from images is not the same as detecting images of objects (Fig. 4.4). Objects do not flicker in-and-out of existence, and do not disappear when not seen, like peekaboo (Fig. 4.5). What we call "object detectors" traditionally refers to algorithms that process a single image and return a decision as to the presence of objects of a certain class in said image, missing several critical elements (a)-(e) above. Nevertheless, such algorithms can be modified to produce *not* decisions, but *evidence* (likelihood) for the presence of objects, which can be processed over time and integrated against the geometric and topological structure of the *scene*, to yield an object detector that has the desired characteristics. The scene context encompasses both the identity and co-occurrence of objects (semantics) but also to their spatial arrangement in three-dimensional (3D) space (syntax).

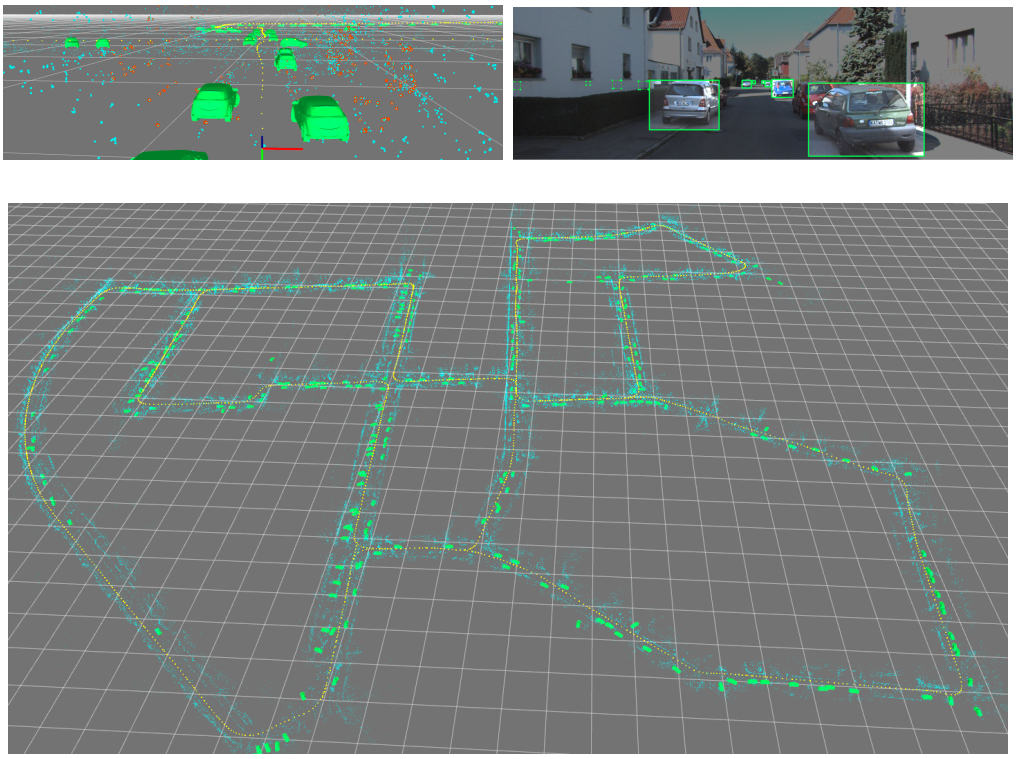


Figure 4.1: *Illustration of our system to detect objects-in-scenes.* Top left: state of the system with reconstructed scene representation (cyan), currently tracked points (red), viewer trajectory from a previous loop (yellow) and current pose (reference frame). All cars detected are shown as point-estimates (the best-aligned generic CAD model) in green, including those previously-seen ones on side streets (far left). Top right: visualization of the implicit measurement process: Objects in the state are projected onto the current image based on the mean vehicle pose estimate (green boxes) and their likelihood score is computed (visualized as contrast: sharp regions have high likelihood, dim regions low). Cars in different streets, known to not be visible, are visualized as dashed boxes and their score not computed. Bottom: Top view of the state from the entire KITTI-00 sequence (best viewed at 5×).

4.1.1 Summary of Contributions and Limitations

To design an object detector based on the premises above, we (a) formalize an explicit model of the posterior probability of object attributes, both semantic (identity) and syntactic (pose), natively in the 3D scene (Sect. 4.3), which (b) maintains and updates such a posterior,

processing each image causally over time (Sect. 4.3.2); (c) the posterior distribution is a form of short-term memory (representation), which we use to (d) predict visibility and occlusion relations (Sect. 4.5.3). We exploit the availability of cheap inertial sensors in almost every mobile computing platform, from phones to drones, to (e) impose class-specific priors on the size of objects (Sect. 4.5.2).

The key insight from the formalization (a) is that an optimal (minimal sufficient invariant [SC16]) representation for objects in the scene (4.1) naturally factors into two components: One geometric – which can be computed recursively by any localization (SLAM) system (4.3) – and the other a likelihood term, which can be computed instantaneously by a discriminatively-trained convolutional neural network (CNN, (4.4)) operating on a single image. Some consequences of this insight are discussed in Sect. 4.6. In practice, this means that we can implement our system using some off-the-shelf components, fine-tuning a pre-trained CNN, and at least for some rudimentary modeling assumptions, our system operates in real-time, generating object-scene representations at 10-30 frames-per second. In Sect. 4.5 we report the results of a representative sample of qualitative and quantitative tests.

There are several novel elements to our system: To the best of our knowledge, we are the first to exploit inertial sensors to provide both scale discrimination and global orientation reference for visual recognition (Fig. 4.4). Most (image)-object detectors assume images are gravity-aligned, which is a safe bet for photographic images, but not so for robots or drones. We are also the first to integrate CNN-based detectors in a recursive Bayesian inference scheme, and to implement the overall system to run in real-time, as we have demonstrated publicly in [rev16].

While our formalization of the problem of object detection is general, our real-time implementation has several limitations. First, it only returns a joint geometric and semantic description for *static objects*. Moving objects are detected in the image, but their geometry – shape and pose, estimating which would require sophisticated class-specific deformation priors – is not inferred. Second, it models objects' shape as a parallelepiped, or bounding box in 3D. While this is a step forward from bounding boxes in the image, it is still a rudimentary model of objects, based on which visibility computation is rather crude. We have performed

several tests with dense reconstruction [GBS15], as well as with CAD models [ISS16], but matching and visibility computation based on those is not yet at the level of accuracy (dense reconstruction) or efficiency (CAD matching) to enable real-time computation. Nevertheless, we use CAD models for some categories, such as cars, chairs, and TV monitors, in Sect. 4.5. The third limitation is that a full joint syntactic-semantic prior is not enforced. While ideally we would like to predict not only what objects are likely to become visible based on context, but also *where* they will appear relative to each other, this is still computationally prohibitive at scale.

In Sect. 4.3 we start by defining an object representation as a sufficient invariant for detection, and show that the main factor can be updated recursively as an integral, where the measure represents the syntactic context, and can be computed by any SLAM system, and the other factor can be computed by (the pre-softmax activations of) a CNN. While the update is straightforward and *top-down* (the system state generate predictions for image-projections, whose likelihood is scored by a CNN), initialization requires defining a prior on object identity and pose. For this we use the same CNN in a *bottom-up* mode, where putative detection (high-likelihood regions) are used to initialize object hypotheses (or, rather, regions with no putative detections are assumed free of objects), and several heuristics are put in place for genetic phenomena (birth, death and merging of objects, Sect. 4.4).

4.2 Related Work

This work, by its nature, relates to a vast body of literature on "scene understanding" in Computer Vision, Robotics [LBR12, PL15] and AI [KAJ11] dating back decades [Wal81]. Most recently, with the advent of cheap consumer range sensors, there has been a wealth of activity in this area [LFU13, TTD12, WLS14, CK13, SK13, DTL15, GAM13, KMF13, SNS13, HFL14, BS15, GAG15, HZC13, SGS13, VML15, LBR11, KMT16, SX15, RS15]. The use of RGB-D cameras unfortunately restricts the domain of applicability mostly indoors and at close range whereas we target mobility applications where the camera, which typically has an inertial sensor strapped on it, but not (yet) a range sensor, can be used both indoor

and outdoors. We expect that, on indoor sequences, our method would underperform a structured light or other RGB-D source, but this is subject of future investigation.

There is also work that focuses on scene understanding from visual sensors, specifically video [KLD14, AYB15, LSH16, SHK12, BGC15, YFU12], although none integrates with inertial sensors, despite a resurgent interest in sensor fusion [ZCV15]. Additional related work includes [HZC13, CLC08, BSF08, SHP15].

To the best of our knowledge, no work has been published to leverage inertial sensing for object detection. This is critical to provide a scale estimate in a monocular setting, and validate object hypotheses in a Bayesian setting, so that, for instance, a model car in our system is *not* classified as a car (Fig. 4.4), despite image-based evidence of the contrary.

Semantic scene understanding *from a single image* is also an active area of research ([FHG15] and references therein). We are instead interested in agents embedded in physical space, for which the restriction to a single image is limiting. There is also a vast literature on scene segmentation, including dedicated workshops and special issues ([HHX15] and references therein), mostly using range (RGB-D) sensors, although also from video. One popular pipeline for dense semantic segmentation is adopted by [HFL14, MHD17, VML15, KLD14, ABS16]: Depth maps obtained either from RGB-D or stereo are fused; 2D semantic labeling is transferred to 3D and smoothed with a fully-connected CRF [Kol11]. Also related methods on joint semantic segmentation and reconstruction are [SHL16, OBG16, BVR15].

There is also work on 3D recognition [KKS13, STL14, MLC14], but again with no inertial measurements and no motion. Some focus on real-time operation [CFN14], but most operate off-line [ZSS15, CRU16]. None of the datasets commonly used in these works [COR16, XOT13] provide an inertial reference, except for KITTI. In terms of 3D object detection on KITTI, some authors focus on image-based detection [GDD14, Gir15, RHG15, RDG16, LAE16] and then place objects into the scene [XCL15, XCL16], while others focus on 3D object proposal generation and verification using a network [CKZ16, CKZ15]. [XCL15] trains a 3D Voxel Pattern (3DVP) based detector to infer object attributes and demonstrates the ability to accurately localize cars in 3D on KITTI. Their subsequent work [XCL16]

trains a CNN to classify 3DVPs. Different representations of object proposals are also exploited, such as 3D cuboids [FDU12] and deformable 3D wireframes [ZSS15]. Various priors are also considered: [WFU15] exploits geo-tagged images; geometric priors of objects are incorporated into various optimization frameworks to estimate object attributes [ZZD15, CRU16]. While most of these algorithms report very good performance on detection ($\sim 90\%$ mAP), none reports scores for the semantic-syntactic state of objects in 3D, except for [XCL15, XCL16] and [CKZ15, CKZ16]. Since the latter are dominated by the former, we take [XCL16] as a paragon for comparison in Sect. 4.5.

The aforementioned 3D object recognition methods are based on 2D detection without temporal consistency. Therefore, the comparison is somewhat unfair as single-image based detectors cannot reliably detect objects in space, which is our main motivation for the proposed approach. For details on comparison methodology, see Sect. 4.5 and Supplementary Material (Sup. Mat.).

Recent work in data association [LZD16] aims to directly infer the association map, which is computationally prohibitive for the scale needed in our real-time system. We therefore resort to heuristics, described in Sect. 4.4. More specifically to our implementation, we leverage existing visual-inertial filters [HKB13, LM14, TCS15] and single image-trained CNNs [GDD14, RDG16, XCL16].

4.3 Methods

4.3.1 Representations

A scene ξ is populated by a number of objects $z_j \in \{z_1, \dots, z_N\}$, each with geometric (pose, shape)[1] and semantic (label) attributes $z_j = \{s_j, l_j\}$. Measurements (*e.g.,* images) up to the current time t, $y^t \doteq \{y_1, \dots, y_t\}$ are captured from a sensor at pose g_t. A *semantic* representation of the scene is the joint posterior $p(\xi, z^j | y^t)$ for up to the j-th objects seen

[1]Object pose is its position and orientation in world frame. With inertials, pose can be reduced to position and rotation along gravity. Sensor pose is full 6 degree-of-freedom position and orientation.

up to time t, where sensor pose g_t and other nuisances are marginalized. The joint posterior can be decomposed as $p(\xi, z^j|y^t) = p(\xi|z^j)p(z^j|y^t)$ with the first factor ideally updated asynchronously each time a new object z_{j+1} becomes manifest starting from a prior $p(\xi)$ and the second factor updated each time a new measurement y_{t+1} becomes available starting from $t = 0$ and given $p(z)$.

A representation of the scene in support of *(geometric) localization tasks* is the posterior $p(g_t, x|y^t)$ over sensor pose g_t (which, of course, is not a nuisance for this task) and a sparse attributed[2] point cloud $x = [x_1, \dots, x_{N_x}]$, given all measurements (visual I^t and inertial u^t) up to the current time. Conditioning the semantics on the geometry we can write the second factor above as

$$p(z^j|y^t) = \int p(z^j|g_t, x, y^t) dP(g_t, x|y^t) \tag{4.1}$$

where the integrand can be updated as more data y_{t+1} becomes available as $p(z^j|g_{t+1}, x, y^{t+1})$, which is proportional to (Chapman-Kolmogorov)

$$p(y_{t+1}|z^j, g_{t+1}, x) \int p(g_{t+1}|g_t, u_t) dP(z^j|g_t, x, y^t). \tag{4.2}$$

4.3.2 Approximations

The measure in (4.1) can be approximated in wide-sense using an Extended Kalman Filter (EKF), as customary in simultaneous localization and mapping (SLAM): $p(g_t, x|y^t) \simeq \mathcal{N}(\hat{g}_{t|t}, \hat{x}_{t|t}; P_{t|t})$. (4.1) is a diffusion around the mean/mode $\hat{g}_{t|t}, \hat{x}_{t|t}$; if the covariance $P_{t|t}$ is small, it can be further approximated: Given

$$\hat{g}_{t|t}, \hat{x}_{t|t} = \arg\max_{g_t, x} p_{\text{SLAM}}(g_t, x|y^t), \tag{4.3}$$

$\hat{p}_{g,x}(z^j|y^t) \doteq p(z^j|g_t = \hat{g}_{t|t}, x = \hat{x}_{t|t}, y^t) \simeq p(z^j|y^t)$. Otherwise the marginalization in (4.1) can be performed using samples from the SLAM system. Either way, omitting the subscripts, we have

$$\hat{p}(z|y^{t+1}) \propto \underbrace{p(y_{t+1}|z, \hat{g}_{t|t}u_t, \hat{x}_{t|t})}_{\text{CNN}} \underbrace{\hat{p}(z|y^t)}_{\text{BF}} \tag{4.4}$$

[2]Attributes include sparse geometry (position in the inertial frame) and local photometry (feature descriptor, sufficient for local correspondence).

where the likelihood term is approximated by a convolutional neural network (CNN) as shown in Sect. 4.3.3 and the posterior is updated by a Bayesian filter (BF) which is a bank of EKFs (Sect. 4.3.4). That only leaves the first factor $p(\xi|z^j)$ in the posterior, which encodes context. While one could approximate it with a recurrent network, that would be beyond our scope here; we even forgo using the co-occurrence prior, which amounts to a matrix multiplication that rebalances the classes following [CLT10], since for the limited number of classes and context priors we experimented with, it makes little difference.

Approximating the likelihood in (4.4) appears daunting because of the purported need to generate future data y_{t+1} (the color of each pixel) from a given object class, shape and pose, and to normalize with respect to all possible images of the object. Fortunately, the latter is not needed since the product on the right-hand side of (4.4) needs to be normalized anyway, which can be done easily in a particle/mixture-based representation of the posterior by dividing by the sum of the weights of the components. Generating actual images is similarly not needed. What is needed is a mechanism that, for a given image y_{t+1}, allows quantifying the likelihood that an object of *any* class with *any* shape being present in *any* portion of the image where it projects to from the vantage point g_t. In Sect. 4.3.3 we will show how a discriminatively-trained CNN can be leveraged to this end.

4.3.3 Measurement Process

At each instant t, an image I_t is processed by "probing functions" ϕ, which can be designed or trained to be invariant to nuisance variability. The SLAM system processes all past image measurements I^t and current inertial measurements u_t, which collectively we refer to as $y_t = \{\phi_\kappa(I_t), u_t\}$, where $\phi_\kappa(I_t)$ is a collection of sparse contrast-invariant feature descriptors computed from the image for N_i visible regions of the scene, and produces a joint posterior distribution of poses g_t and a sparse geometric representation of the scene $x = [x_1, \ldots, x_{N_i(t)}]$, assumed uni-modal and approximated by a Gaussian:

$$p_{\text{SLAM}}(g_t, x|y^t) \simeq \mathcal{N}(\hat{g}_{t|t}, \hat{x}_{t|t}; P_{\{g,x\}\,t|t}) \tag{4.5}$$

where $x \in \cup_j s_j$, *i.e.,* the scene is assumed to be composed by the union of objects, including the default class "background" l_0. This localization pipeline is borrowed from [TCS15, MMT15], and is agnostic of the organization of the scene into objects and their identity. It also restricts x to a subset of the scene that is rigid, co-visible for a sufficiently long interval of time, and located on surfaces that, locally, exhibit Lambertian reflection.

To compute the marginal likelihood for each class $l_k \in \{l_0, \dots, l_K\}$, we leverage on a CNN trained discriminatively to classify a given image region b_j into one of $K+1$ classes, including the background class. The architecture has a soft-max layer preceded by $K+1$ nodes, one per class, and is trained using the cross-entropy loss, providing a normalized score $\phi_{\text{CNN}}(l|I_{t_{|b_j}})_{[k]}$ for each class and image bounding box b_j. We discard the soft-max layer, and forgo class-normalization. The activations at the $K+1$ nodes in the penultimate layer of the resulting network provide a mechanism for, given an image I_t, quantifying the likelihood of each object class l_k being present at each bounding box b_j, which we interpret the (marginal) likelihoods for (at least an instance of) each class being present at the given bounding box:

$$\phi_{\text{CNN}}(l|I_{t_{|b_j}})_{[k]} \simeq p(I_t|l_k, b_j). \tag{4.6}$$

This process induces a likelihood on object classes being present in the *visible portion of the scene* regions of s_j and corresponding vantage points g_t, via $b_j = \pi(g_t s_j)$ where π is the projection. Since inertials u_t are directly measured, up to a Gaussian noise, we have:

$$p(y_t|z^j, g_t, \hat{x}) \simeq \phi_{\text{CNN}}(l|I_{t_{|\pi(g_t s_j)}})_{[k]} \mathcal{N}(\bar{u}; Q) \tag{4.7}$$

where $\bar{u}$ are the inertial biases and Q the noise covariance; here the object attributes z^j are the labels $l_j = l_k$ and geometry s_j. Thus, given an image I_t, for each possible object pose and shape s_j and vantage point g_t, we can test the presence of at least one instance of each class l_k within. Note that the visibility function is implicit in the map π. If an object is not visible, its likelihood given the image I_t is constant/uniform. Note that this depends on the global layout of the scene, since the map π must take into account occlusions, so objects cannot be considered independently.

4.3.4 Dependencies and Co-visibility

Computing the likelihood of an object being present in the scene requires ascertaining whether it is visible in the image, which in turn depends on all other objects, so the scene has to be modeled holistically rather than as an independent collection of objects. In addition, the presence of certain objects, and their configuration, affects the probability that other objects that are not visible be present.[3]

To capture these dependencies, we note that the geometric representation $p(g_t, x|y^t)$ can be used to provide a joint distribution on the position of all objects and cameras $p(g^t, x|y^t)$, which yields *co-visibility* information, specifically the probability of each point in x being visible by any camera in g^t. It is, however, of no use in determining visibility of objects, since it contains no topological information: We do not know if the space between two points is empty, or occupied by an object void of salient photometric features.[4] To enable visibility computation, we can use the point cloud together with the images to compute the *dense shape* of objects in a maximum-likelihood sense: $\hat{s}_j = \arg\max p(s_j|g^t, x, y^t)$ using generic regularizers. This can be done but not at the level of accuracy and efficiency needed for live operation. An alternative is to approximate the shape of objects with a parametric family, for instance cuboids or ellipsoids, and compute visibility accordingly, also leveraging the co-visibility graph computed as a corollary from the SLAM system and priors on the size and aspect ratios of objects. To this end, we approximate

$$\hat{p}_{g,x}(z^j|y^t) \doteq p(z^j|y^t, g_t, x) \simeq \prod_j p(z_j|y^t, g_t, x, z^{-j}) \tag{4.8}$$

where z^{-j} indicates all objects but z_j. Each factor $p(s_j, l_j|y^t, g_t, x, z^{-j})$ is then expanded as the product

[3]For instance, seeing a keyboard and a monitor on a desk affects the probability that there is a mouse in the scene, even if we cannot see it at present. Their relative pose also informs the vantage point that would most reduce the uncertainty on the presence of the mouse.

[4]In an active perception setting, given structured illuminator, one could take a control action to reduce uncertainty and resolve the conundrum.

$$\underbrace{p(s_j|l_j, y^t, g_t, x, s^{-j})}_{\text{EKF}} \underbrace{P(l_j|y^t, g_t, x, l^{-j})}_{\text{PMF}} \tag{4.9}$$

where PMF indicates a probability mass filter; this effectively yields a bank of class-conditional EKFs. These provide samples from $\hat{p}(z|y^t)$ in the right-hand side of (4.4), that are scored with the CNN to update the posterior.

4.4 Implementation Details

We have implemented two renditions of the above program: One operating in real-time and demonstrated live at [rev16] in June 2016. The other operating off-line and used for the experiments reported in Sect. 4.5.

In both cases, we have taken some shortcuts to improve the efficiency of the approximation of the likelihood function implemented by a CNN. Also, the semantic filter needs initialization and data association, which requires some heuristics to be computationally viable. We describe such heuristics in order.

Visual Odometry and Baseline 2D CNN We use robust SLAM implemented from [MMT15, TCS15] to acquire sparse point clouds and camera pose x, g_t at each t. This occurs in $10-20$ms per VGA frame. For our real-time system, we use YOLO [RDG16] as a baseline method to compute object likelihoods in $150-200$ms, whereas in the off-line system we use SubCNN [XCL16]. In either case, the result is, for each given window, a positive score for each class k, read out from the penultimate layer (before softmax). These are used both to compute the likelihood, and to generate proposals for initialization as discussed later.

Filter Organization Each object is represented by a PMF filter over class labels and K class-conditional EKFs, one for each class (4.9). Thus each object is represented by a mixture of K EKFs, some of which pruned as we describe later. Each maintains a posterior estimate of position, scale and orientation relative to gravity. The state predicts the projection of (each of the K instances) of each object onto the image plane, where the CNN evaluates the

likelihood. For some object classes, we use a shape prior, enforced as a pseudo-measurement with uncertainty manually tuned to expected class-variability. For instance, people are parallelepipeds of $1m^3$ expected volume with an anisotropic covariance along coordinate axes in the range of few decimeters, whereas couches have significantly more uncertainty.

Data Association To avoid running the baseline CNN multiple times on overlapping regions (each object is represented by multiple, often very similar, regions, one per each current class hypothesis), we do not query the CNN sequentially for each prediction. Instead, we run the CNN once, with lax threshold so as to obtain a large number of (low-confidence) regions. While this is efficient, it does create a data association problem, as we must attribute (possibly multiple) image regions to each (of multiple) object hypotheses, each of which has multiple possible class labels. Data association is a classically hard problem, which has received renewed attention lately [AZD14], but remains challenging at scale. We adopt simple heuristics instead: first we generate predictions from the filter; then occluded objects are excluded from likelihood evaluation. For all others, we generate four-tuple coordinates of the bounding box, as a 4-dimensional Gaussian given the projection of the current state. This is a sloppy prediction, for the image of a parallelepiped is in general not an axis-aligned rectangle on the image. Nevertheless, we use this for scoring by using the likelihood produced by the CNN for each predicted class. A (class-dependent) threshold is used to decide if the bounding box should be used to update the object. Bounding boxes with lower likelihood are given small weights in the filter update. This requires accurate initialization, which we will describe below. The silver lining is that inter-frame motion is usually small, so data association proceeds smoothly, unless multiple instances of the same object class are present nearby and partially occlude each other.

Initialization Putative 2D CNN detections not associated to any object are used as (bottom-up) proposals for initialization. The new object is positioned at the weighted centroid of the sparse points whose projections lie within the detection region. The weight at center is the largest and decreases exponentially outwards. Orientation is initialized as the

"azimuth" from SubCNN, rotated according to camera pose and gravity. Given the position and orientation, scale is optimized by minimizing the reprojection error.

Merge Objects are assumed to be simply-connected and compact, so two objects cannot occupy the same space. Yet, their projected bounding boxes can overlap. If multiple instances from the same object are detected, initialized and propagated, they will eventually merge when their overlap in space is sufficiently large. Only objects from the same class are allowed to merge as different classes may appear co-located and intersecting in their sloppy parallelepipedal shape model, *e.g.*,, a chair under a table.

Termination Each object maintains a probability over K classes, each associated with a class-conditional filter. If one of the class becomes dominant (maximum probability above a threshold), all other filters will be eliminated to save computation cost. Most objects converge to one or two classes (*e.g.*,, chair, couch) within few iterations. Objects that disappear from view are retained in the state (short-term memory), and if not seen for a sufficiently long time, they are stored in long-term memory ("semantic map") for when they will be seen again.

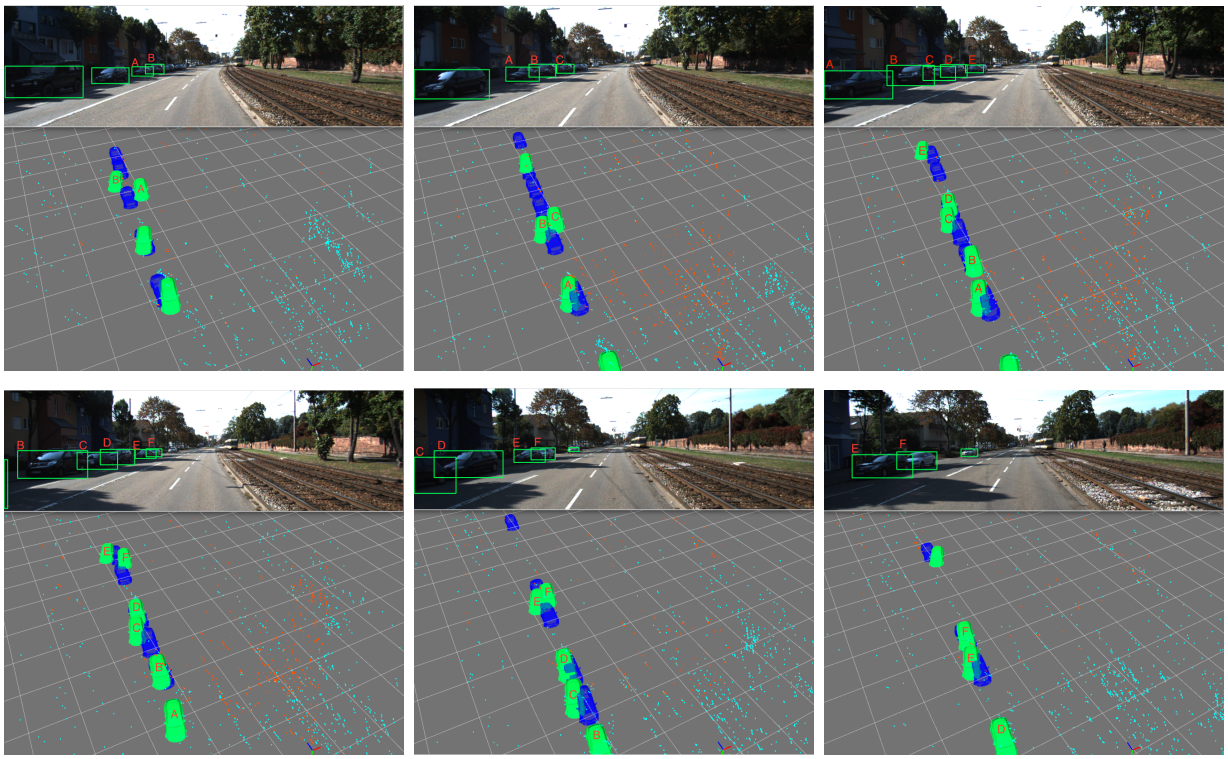


Figure 4.2: *Evolution of the state (Green) against ground-truth annotation (Blue) (best viewed at 5×, left to right, top to bottom).* When first seen (top left) cars 'A' and 'B' are estimated to be side-by-side; after a few frames, however, 'A' and 'B' fall into place, but a new car 'C' appears to flank 'B'. As time goes by, 'C' too falls into place, as new cars appear, 'D', 'E', 'F.' The error in pose (position and orientation) relative to ground truth can be appreciated qualitatively. Quantitative results are shown in Table 4.1.

4.5 Experiments

4.5.1 Quantitative Results

As explained in Sec. 4.2, we choose SubCNN [XCL16] as the paragon, even though it is based on a single image, because it is the top performer for 3D recognition in KITTI among non-anonymous and reproducible ones, in particular it dominates [CKZ16]. Being single-image based, SubCNN returns different results in each frame, therefore naturally at a disadvantage. To make the comparison fair, one would have to average or integrate detections for

each object across all frames when it is visible. However, SubCNN does not provide data association, making direct comparison challenging. To make comparison as fair as possible, without developing an alternate aggregation method for SubCNN, we compare it to our algorithm on a frame-by-frame basis. Specifically, for each frame, we transfer the ground truth to the camera frame, and remove occluded objects. Then we can compare detections from SubCNN to our point estimate (conditional mean) computed causally by the filter at the current time. We call this method `Ours-INST`. On the other hand, we can benefit from aggregating temporal information for as long as possible, so we also report results based on the point-estimate of the filter state at the last time instant when each object is seen. The estimate is then mapped back to the current frame, which we call `Ours-FNL`. To the best of our knowledge, there are no known methods for 3D recognition that causally update posterior estimates of object identity/presence and geometric attributes, and even naive temporal averaging of a method like [XCL16] is not straightforward because of the absence of data association across different frames. This is precisely what motivates us.

4.5.1.1 Dataset

There are many datasets for image-based object detection [EVW10, RDS15] which provide 2D ground truth. There are also 3D object detection datasets [XOT13], most using extra sensor data, *e.g.,*, depth from a structured-light sensor. None provide inertial measurements, except KITTI [GLS13], whose object detection benchmark contains 7181 images, from which we exclude 3682 frames used for SubCNN training [XCL16], leaving us a validation set of 3799 frames. We then find 10 videos which cover most of the validation set, where 598 objects are observed 24590 times at 4895 instants, which is the same order of magnitude of the 2D validation set.

4.5.1.2 Evaluation Metrics

KITTI provides ground-truth object *tracklets* we use to define true positives, miss detections and false alarms. A *true positive* is the nearest detection of a ground truth object within

a specified error threshold in both position and orientation (Table 4.1). A *miss* occurs if there is no detection within the threshold. A *false alarm* occurs when an object is detected despite no true object being within the threshold in distance and orientation. *Precision* is the fraction of true positives over all detections, and *Recall* is the percentage of detected instances among all true objects.

4.5.1.3 Benchmark Comparison

Table 4.1 shows result on the KITTI dataset, averaged over all sequences. On average, `Ours-INST` already outperforms `SubCNN` even if our initialization can be rather inaccurate. `Ours-FNL` further improves the results by a large margin. Fig. 4.2 shows how our method refines the state over time. Visual comparison is shown in Fig. 4.3 for ground truth (Blue), `Ours-FNL` (Green) and `SubCNN` (Yellow).

	Position error	$< 0.5m$			$< 1m$			$< 2m$		
Orientation error	method	#TP	Precision	Recall	#TP	Precision	Recall	#TP	Precision	Recall
$< 30°$	`Ours-FNL`	**190**	**0.14**	**0.10**	**451**	**0.33**	**0.23**	**728**	**0.54**	**0.37**
	`Ours-INST`	165	0.12	0.08	344	0.25	0.18	546	0.40	0.28
	`SubCNN`	113	0.09	0.06	289	0.22	0.15	537	0.42	0.27
$< 45°$	`Ours-FNL`	**197**	**0.15**	**0.10**	**465**	**0.34**	**0.24**	**758**	**0.56**	**0.39**
	`Ours-INST`	172	0.13	0.09	360	0.26	0.18	576	0.42	0.29
	`SubCNN`	118	0.09	0.06	300	0.23	0.15	561	0.43	0.29
–	`Ours-FNL`	**210**	**0.16**	**0.11**	**531**	**0.39**	**0.27**	**876**	**0.65**	**0.45**
	`Ours-INST`	182	0.13	0.09	403	0.29	0.21	671	0.49	0.34
	`SubCNN`	136	0.10	0.07	350	0.27	0.18	671	0.52	0.34

Table 4.1: *Quantitative evaluation on KITTI and comparison with SubCNN [XCL16].* The number of true positives having positional error (row), and angular error (column) less than a threshold is shown, along with Precision and Recall. Scores are aggregated across all 4895 ground-truth labeled frames in the dataset, with 598 annotated objects. The last 3 rows discard orientation error.

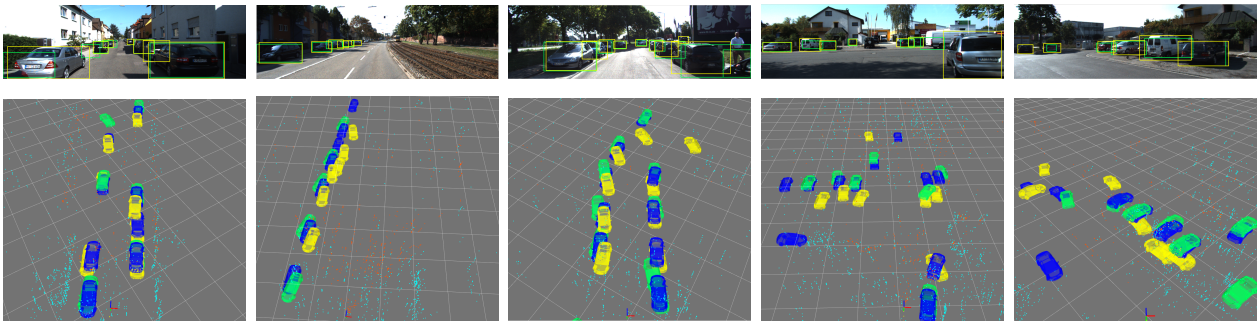


Figure 4.3: *Qualitative comparison with SubCNN.* Top: Images with back-projected objects from our method (Green), the same with SubCNN (Yellow). Bottom: top-view of the corresponding portion of the scene. Ground truth is shown in Blue.

4.5.2 Class-specific Priors

Objects have characteristic scales, which are lost in perspective projection but inferable with an inertial sensor. We impose a class-dependent prior on size and shape (*e.g.*,, volume, aspect ratios). In Fig. 4.4, a toy car is detected as a car by an image-based detector (Yellow), but rejected by our system as inconsistent with the scale prior (Green). Fig. 4.4b shows two background cars in the far field, whose images are smaller than the toy car, yet they are detected correctly, whereas the toy car is rejected.

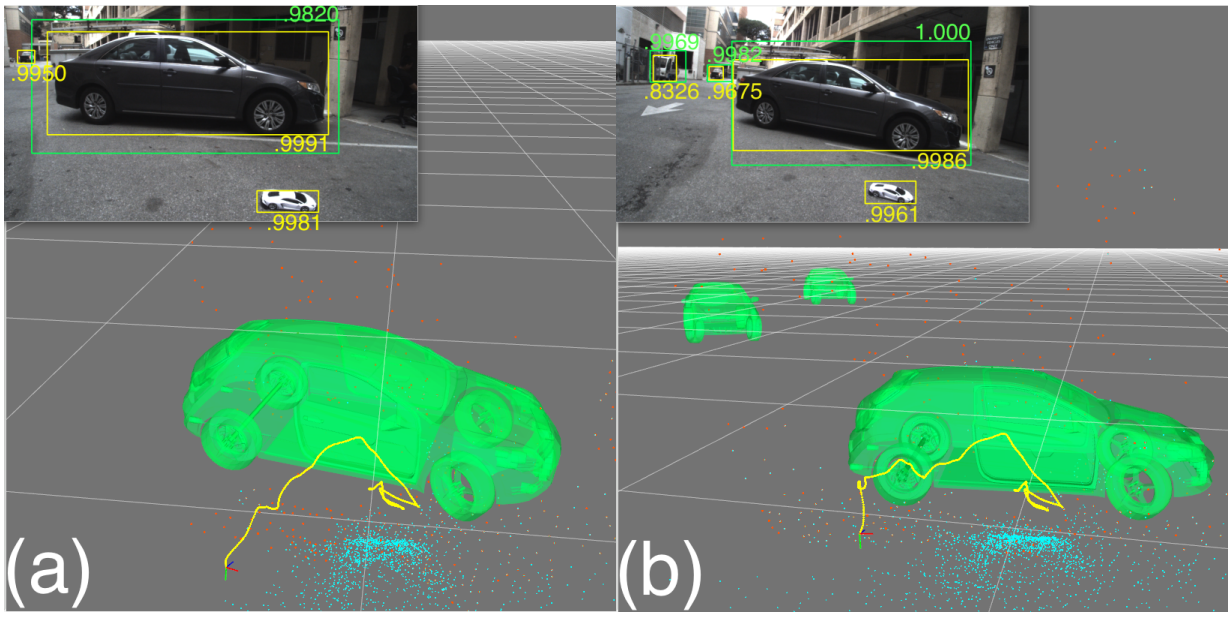


Figure 4.4: *Class-specific scale prior.* (a): A real car is detected by our system, unlike the toy car, despite both scoring high likelihood and therefore being detected by an image-based system (Yellow). As time goes by, the confidence on the real car increases (best viewed at 5×) (b). See `Video11` in the Sup. Mat.

4.5.3 Occlusion and Memory

Our system represents objects in the state even while they are not visible, or detected by an image-based detector. This allows predicting the re-appearance of objects in future frames, and to resume update if new evidences appear. Fig. 4.5 shows a chair first detected and then occluded by a monitor, later reappearing. The system predicts the chair to be completely occluded, and therefore does not use the image to update the chair, but resumes doing so when it reappears, by which time it is known to be the *same* chair that was previously seen (re-detection). In Sect. 4.5.4, we show the same phenomenon in a large-scale driving sequence.

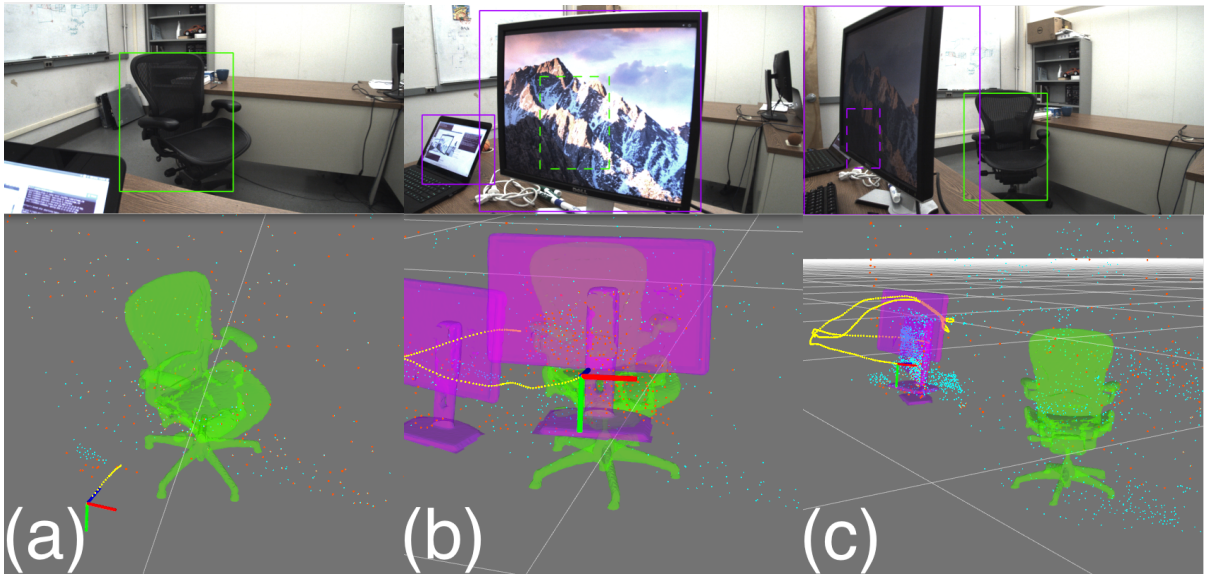


Figure 4.5: *Occlusion management and short-term memory.* (a): A chair is detected and later becomes occluded by the monitor (b). Its projection onto the image is shown in dashed lines, indicating occlusion. The model allows prediction of dis-occlusion (c) which allows resuming update when the chair comes back into view. See `Video12` in Sup. Mat.

4.5.4 Large-scale Driving Sequences

Fig. 4.1 and `Video13` in Sup. Mat. show our results on a 3.7km-long sequence from KITTI. It contains hundreds of cars along the route. Once recognized as a car, we replace the bounding box with a CAD model of similar car, aligned with the pose estimate from the filter, in a manner similar to [SNS13], that however uses RGB-D data. In this sequence, we can also see cars on different streets "through walls" if they have been previously detected, which can help navigation.

4.5.5 Indoor Sequences

We have tested our system live in a public demo [rev16], operating in real time in cluttered environments with people, chairs, tables, monitors and the like. Representative examples are shown for simpler scenes, for illustrative purposes, in Fig. 4.6, where again CAD models

of objects are rendered once detected, a' la [SNS13]. Our system does not produce exact orientation estimates, as seen in Fig. 4.6, so there is plenty of room for improvement.

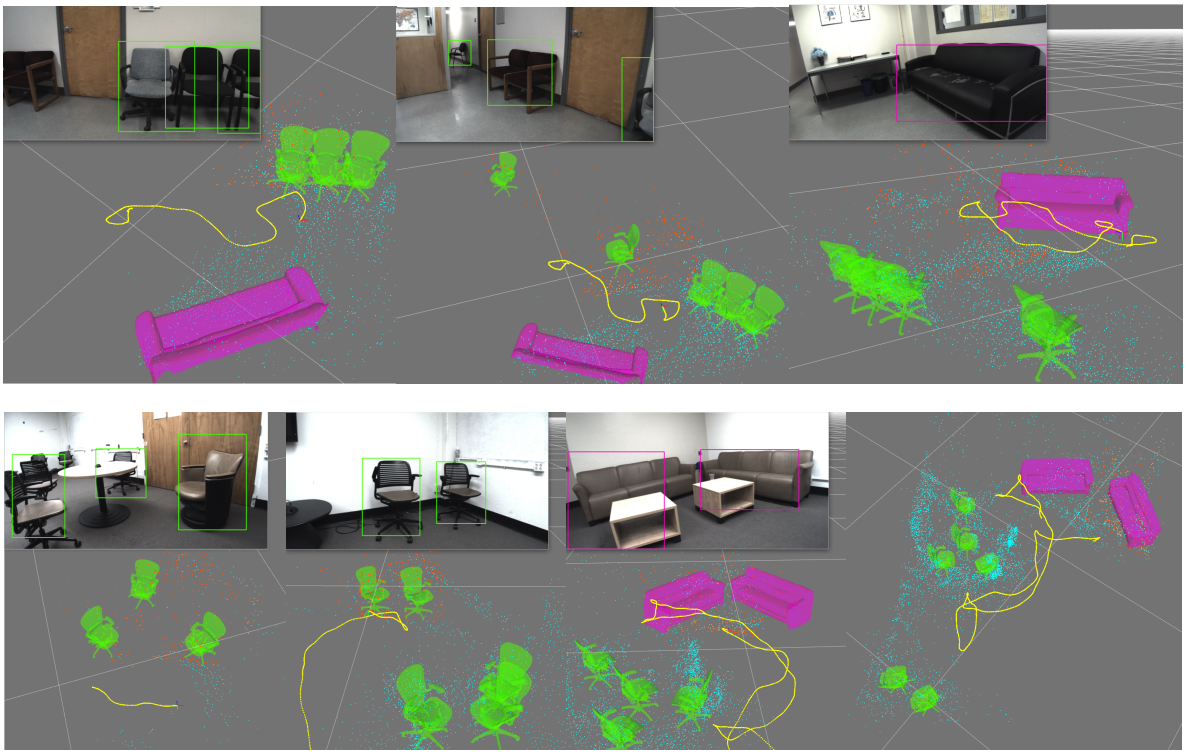


Figure 4.6: *Indoor sequences.* Top: An office area (`Video14` in Sup. Mat.). Bottom: A Lounge area (`Video15` in Sup. Mat.).

4.6 Discussion

Inertial sensors are in every modern phone, tablet, car, even many toys, all devices embedded in physical space and occasionally in need to interact with it. It makes sense to exploit inertials, along with visual sensors, to help detecting objects that exist in 3D physical space, and have characteristic shape and size, in addition to appearance. We have recorded tremendous progress in object detection in recent years, if by object one means a group of pixels in an image. Here we leverage such progress to design a detector that follows the prescriptions (a)-(e) indicated in the introduction.

We start by defining a representation as a minimal sufficient invariant statistic of object attributes, in line with [SC16]. We then marginalize on camera Euclidean pose – which allows us to enforce priors on the class-specific scale of objects – and update the measure by a Bayesian filter, where a CNN is in charge of computing the likelihood function.

We note that a minimal sufficient invariant for localization is an attributed point cloud, and therefore there is no need to deploy the machineries of Deep Learning to determine camera pose (Deep Learning could still be used to infer the attributes at points, which are used for correspondence). Instead, we use an off-the-shelf Extended Kalman Filter, conditioned on which the update for object attributes can be performed by a Mixture-of-Kalman filter.

The result is a system whereby objects do not flicker in-and-out of existence, our confidence in their presence grows with accrued evidence, we know of their presence even if temporarily occluded, we can predict when they will be seen, and we can enforce known scale priors to reject spurious hypotheses from the bottom-up proposal mechanism.

We have made stringent and admittedly restrictive assumptions in order to keep our model viable for real-time inference. One could certainly relax some of these assumptions and obtain more general models, but forgo the ability to operate in real time.

The main limitation of our system is its restriction to static objects. While in theory the framework is general, the geometry of moving and deforming objects is not represented, and therefore their attributes remain limited to what can be inferred in the image. Also, our representation of objects' shape is rather rudimentary, and as a result visibility computation rather fragile. These are all areas prime for further future development.

CHAPTER 5

Visual-Inertial Object Detection and Mapping

5.1 Introduction

We aim to detect, recognize, and localize objects in the three-dimensional (3D) scene. We assume that previous views of the object are sufficient to construct a dense model of its shape, in the form of a closed and water-tight surface, and its appearance (a texture map). So, as soon as an object is detected from a monocular image, and localized in the scene, the corresponding region of space can be mapped with the object model, including the portion not visible in the current image (Fig. 5.4 and 5.5).

While single monocular images provide evidence of objects in the scene – in the form of a likelihood score for their presence, shape and pose – they should not be used to make a decision. Instead, evidence should be accumulated over time, and the likelihood at each instant combined into a posterior estimate of object pose and identity. This is often referred to as "semantic mapping," an early instance of which using depth sensors (RGB-D images) was given in [SNS13]. Our method aims at the same goal, but using a monocular camera and inertial sensors, rather than a range sensor.

Inertial sensors are increasingly often present in sensor suites with monocular cameras, from cars to phones, tablets, and drones. They complement vision naturally, in an information-rich yet cheap sensor package. Unlike RGB-D, they can operate outdoor; unlike stereo, they are effective at far range; unlike lidar, they are cheap, light, and provide richer photometric signatures. Inertial sensors provide a globally consistent orientation reference (gravity) and scale up to some drift. This allows reducing pose space to four dimensions instead of six. We leverage recent developments in visual-inertial sensor fusion, and its use

for semantic mapping, an early instance of which was given in [DFS17], where objects were represented by bounding boxes in 3D. Our method extends that work to richer object models, that allow computing fine-grained visibility and estimating accurate pose.

Contributions We focus on applications to (indoor and outdoor) navigation, where many objects of interest are rigid and static: parked cars, buildings, furniture. Our contribution is a method and system that produces camera poses and a point-cloud map of the environment, populated with 3D shape and appearance models of objects recognized. It is semantic in the sense that we have identities for each object instance recognized. Also, all geometric and topological relations (proximity, visibility) are captured by this map.

We achieve this by employing some tools from the literature, namely visual-inertial fusion, and crafting a novel likelihood model for objects and their pose, leveraging recent developments in deep learning-based object detection. The system updates its state (memory) causally and incrementally, processing only the current image rather than storing batches.

Another contribution is the introduction of a dataset for testing visual-inertial based semantic mapping and 3D object detection. Using inertials is delicate as accurate time-stamp, calibration and bias estimates are needed. To this date, we are not aware of any dataset for object detection that comes with inertials.

We do not address intra-class variability. Having said that, the method is somewhat robust to modest changes in the model. For instance, if we have a model Aeron chair (Fig. 5.2) with arm rests, we can still detect and localize an Aeron chair without them, or with them raised or lowered.

Organization In Sect. 5.2, we describe our method, which includes top-down (filter) and bottom-up (likelihood/proposals) components. In particular, Sect. 5.2.4 describes the novel likelihood model we introduce, using a detection and edge scoring network. Sect. 5.3 describes our implementation, which is tested in Sect. 5.4, where the VISMA dataset is described. We discuss features and limitations of our method in Sect. 5.5, in relation to prior related work.

5.1.1 Relation to the Prior Art

Many efforts have been made to incorporate semantics into SLAM, and vice versa. Early attempts [CKM10, CGR11] rely on feature matching to register 3D objects to point clouds, which are sensitive to illumination and viewpoint changes, and most importantly, cannot handle texture-less objects. These issues are resolved by considering both semantic and geometric cues in our method (Fig. 5.4 and 5.5). In [KLD14], voxel-wise semantic labeling is achieved by fusing sparse reconstruction and pixel-wise semantic segmentation with a CRF model over voxel grids. The same scheme has been adopted by [HFL14, VML15, MHD17] which explore different sensors to get better reconstruction. Although these methods produce visually pleasing semantic labeling at the level of voxels, object-level semantic understanding is missing without additional steps to group together the potentially over-segmented voxels. Our method treats objects in the scene as first-class citizens and places objects in the scene directly and immediately without post-processing. The works that are closest to ours are RGB-D based SLAM++ [SNS13] and visual-inertial based [DFS17] and [BAD17], where the former models objects as generic parallelepipeds and the latter focuses on the data association problem and only estimates translation of objects, while ours estimates precise object shape and 6DoF pose.

This work is related to visual-inerital sensor fusion [MR07] and vision-only monocular SLAM [KM07] in a broader sense. While classic SLAM outputs a descriptor-attached point cloud for localization, ours also populates objects in the scene to enable augmented reality (AR) and robotic tasks.

This work, by its nature, also relates to recent advances in object detection, either in two stages [Gir15, RHG15, HGD17], which consist of a proposal generation and a regression/classification step, or in a single shot [LAE16, RDG16], where pre-defined anchors are used. Though single-shot methods are in general faster than two-stage methods, the clear separation of the architecture in the latter suits our hypothesis testing framework better (Fig. 5.1a). Image-based object detectors have encouraged numerous applications, however they are insufficient to fully describe the 3D attributes of objects. Efforts in making 2D

detectors capable of 6DoF pose estimation include [XMS14, XKC16], which are single image based and do not appreciate a globally consistent spatial reference frame, in which evidence can be accumulated over time as we did in our system.

The idea of using edge as a likelihood to estimate object pose dates back to the RAPiD algorithm [DC02] followed by [KM06, CC12]. [LF05] is a recent survey on model-based tracking, which is a special and simplistic case of our system: In model-based tracking, the 3D model being tracked is selected and its pose initialized manually while in our setting, such quantities are found by the algorithm. Another line of work [PR12, TSS17] on model-based tracking relies on level-set and appearance modeling, which we do not adopt because appearance is subject to illumination and viewpoint changes while edges are geometric and more robust.

5.2 Methodology

To facilitate semantic analysis in 3D, we seek to reconstruct a model of the scene sufficient to provide a Euclidean reference where to place object models. This cannot be done with a single monocular camera. Rather than using lidar (expensive, bulky), structured light (fails outdoors), or stereo (ineffective at large distances), we exploit inertial sensors frequently co-located with cameras in many modern sensor platforms, including phones and tablets, but also cars and drones. Inertial sensors provide a global and persistent orientation reference from gravity, and an estimate of scale, sufficient for us to reduce Euclidean motion to a four-dimensional group. In the next section we describe our visual-inertial simultaneous localization and mapping (SLAM) system.

5.2.1 Gravity-referenced and scaled mapping

We wish to estimate $p(Z_t, X_t|y^t)$ the joint posterior of the state of the sensor platform X_t and objects in the scene $Z_t \doteq \{z\}_t^N$ given data $y^t = \{y_0, y_1, \cdots, y_t\}$ that consists of visual (image I_t) and inertial (linear acceleration α_t and rotational velocity ω_t) measurements, *i.e.*,

$y_t \doteq \{I_t, \alpha_t, \omega_t\}$. The posterior can be factorized as

$$p(Z_t, X_t|y^t) \propto p(Z_t|X_t, y^t)p(X_t|y^t) \tag{5.1}$$

where $p(X_t|y^t)$ is typically approximated as a Gaussian distribution whose density is estimated recursively with an EKF [Jaz70] in the visual-inertial sensor fusion literature [MR07, TCS15]. Upon convergence where the density $p(X_t|y^t)$ concentrates at the mode $\hat{X}_t$, the joint posterior can be further approximated using a point estimate of $\hat{X}_t$.

Visual-inertial SLAM has been used for object detection by [DFS17], whose notation we follow here. The state of a visual-inertial sensor platform is represented as

$$X_t \doteq [\Omega_{sb}^\top, T_{sb}^\top, \Omega_{bc}^\top, T_{bc}^\top, v^\top, \alpha_{\text{bias}}^\top, \omega_{\text{bias}}^\top, \gamma^\top, \tau]^\top$$

where $g_{sb}(t) \doteq (\Omega_{sb}, T_{sb}) \in \text{SE}(3)$ is the transformation of the body frame to the spatial frame, $g_{bc}(t) \doteq (\Omega_{bc}, T_{bc}) \in \text{SE}(3)$ is the camera-to-body alignment, $v \in \mathbb{R}^3$ is linear velocity, $\alpha_{\text{bias}}, \omega_{\text{bias}} \in \mathbb{R}^3$ are accelerometer and gyroscope biases respectively, $\gamma \in \mathbb{R}^3$ is the direction of gravity and $\tau \in \mathbb{R}$ is the temporal offset between visual and inertial measurements. The transformation from camera frame to spatial frame is denoted by $g_{sc} \doteq g_{sb}g_{bc}$. The implementation details of the visual-inertial SLAM system adopted are in Sect. 5.3. Next, we focus on objects.

5.2.2 Semantic Mapping

For each object $z_t \in Z_t$ in the scene, we simultaneously estimate its pose $g \in \text{SE}(3)$ and identify shape $\mathcal{S} \subset \mathbb{R}^3$ over time. We construct beforehand a database of 3D models, which covers objects of interest in the scene. Thus the task of estimating shape of objects is converted to the task of determining shape label $k \in \{1, 2, \cdots, K\}$ of objects, which is a discrete random variable. Once the shape label k is estimated, its shape $\mathcal{S}(k)$ can be simply read off from the database. Furthermore, given an accurate estimate of gravity direction γ from visual-inertial SLAM, the 6DoF (degrees of freedom) object pose can be reduced to a four-dimensional group element $g \doteq (t, \theta)$: Translation $t \in \mathbb{R}^3$ and rotation around gravity (azimuth) $\theta \in [0, 2\pi)$.

We formulate the semantic mapping problem as estimating the posterior $p(z_t = \{k, g\}_t | \hat{X}_t, I^t)$ conditioned on mode $\hat{X}_t$, which can be computed in a hypothesis testing framework, of which the hypothesis space is the Cartesian product of shape label and pose $\{k\} \times \{g\}$. To facilitate computation and avoid cluttered notations, we drop $\hat{X}_t$ behind the condition bar and introduce an auxiliary discrete random variable: Category $c \in \{1, 2, \cdots, C\}$.

$$p(\{k, g\}_t | I^t) = \sum_{c_t} p(\{k, g, c\}_t | I^t) \tag{5.2}$$

$$\propto \sum_{c_t} p(I_t | \{k, g, c\}_t) \int p(\{k, g, c\}_t | \{k, g, c\}_{t-1}) dP(\{k, g, c\}_{t-1} | I^{t-1}) \tag{5.3}$$

where marginalization is performed over all possible categories. By noticing that category c_t is a deterministic function of shape label k_t, *i.e.*, $p(c_t|k_t) = \delta(c_t - c(k_t))$, the posterior $p(\{k, g\}_t | I^t)$ can be further simplified as follows:

$$\sum_{c_t} \delta(c_t - c(k_t)) p(I_t | \{k, g, c\}_t) \int p(\{k, g\}_t | \{k, g\}_{t-1}) dP(\{k, g\}_{t-1} | I^{t-1}) \tag{5.4}$$

where the first term in the summation is the likelihood (Sect. 5.2.4) and the second term can be approximated by numerical integration of weighted particles (Sect. 5.3.5).

5.2.3 Parameterization and Dynamics

Each object is parametrized locally and attached to a reference camera frame at time t_r with pose $g_{sc}(t_r)$ and the translational part of object pose is parameterized by a bearing vector $[x_c, y_c]^\top \in \mathbb{R}^2$ in camera coordinates and a log depth $\rho_c \in \mathbb{R}$ where $z_c = \exp(\rho_c) \in \mathbb{R}_+$. Log depth is adopted because of the positivity and cheirality it guarantees. Inverse depth [CDM08], though often used by the SLAM community, has singularities and is not used in our system. The object centroid is then $T_{co} = \exp(\rho_c) \cdot [x_c, y_c, 1]^\top$ in the reference camera frame and $T_{io} = g_{sc}(t_r) T_{co}$ in the spatial frame. For azimuth θ, we parameterize it in the spatial frame and obtain the rotation matrix via Rodrigues' formula:

$$\mathbf{R}_{so}(\theta) = \mathbf{I} + \sin\theta \widehat{\gamma} + (1 - \cos\theta) \widehat{\gamma}^2 \tag{5.5}$$

where γ is the direction of gravity and the hat operator $\widehat{\cdot}$ constructs a skew-symmetric matrix from a vector. Therefore the object pose in the spatial frame is $g_{so} = [\mathbf{R}_{so} | T_{so}] \in \mathrm{SE}(3)$.

Although the pose parameters are unknown constants instead of time varying quantities, we treat them as stochastic processes with trivial dynamics as a common practice: $[\dot{x}_c, \dot{y}_c, \dot{\rho}_c, \dot{\theta}]^\top = [n_x, n_y, n_\rho, n_\theta]^\top$ where n_x, n_y, n_ρ and n_θ are zero-mean Gaussian noises with small variance.

5.2.4 Measurement Process

In this section, we present our approximation to the log-likelihood $L(\{k, g, c\}_t | I_t) \doteq \log p(I_t | \{k, g, c\}_t)$ of the posterior (5.4). Given the prior distribution $p(\{k, g\}_{t-1} | I^{t-1})$, a hypothesis set $\{k, g\}_t$ can be constructed by a diffusion process around the prior $\{k, g\}_{t-1}$. To validate the hypothesis set, we use a log-likelihood function which consists of two terms:

$$L(\{k, g, c\}_t | I_t) = \alpha \cdot \Phi_{\text{CNN}}(\{k, g, c\}_t | I_t) + \beta \cdot \Phi_{\text{edge}}(\{k, g\}_t | I_t) \tag{5.6}$$

where α and β are tuning parameters. The first term in the log-likelihood is a convolutional neural network which measures the likelihood of an image region is to contain a certain object. The second term scores the likelihood of an edge in the image. We describe them in order.

5.2.4.1 CNN as Likelihood Mechanism

Given a hypothesis $\{k, g\}_t$ in the reference frame, we first bring it to the current camera frame by applying a relative transformation and then project it to the current image plane via a rendering process. A minimal enclosing bounding box of the projection is found and then fed into an object detection network. The score of the hypothesis is simply read off from the network output (Fig. 5.1a).

$$\Phi_{\text{CNN}}(k, g, c; I) = \text{Score}\Big(I_{|b=\pi\left(g_{sc}^{-1}(t) g_{so}(t_r) \mathcal{S}(k)\right)}, c\Big) \tag{5.7}$$

where $\pi(\cdot)$ denotes the process to render the contour map of the object of which the minimal enclosing bounding box b is found; $g_{so}(t_r)$ is the transformation to bring the object from local reference frame at time t_r to the spatial frame and $g_{cs} = g_{sc}^{-1}(t)$ is the transformation to bring the object from the spatial frame to current camera frame.

Either a classification network or a detection network can be used as our scoring mechanism. However, due to the size of the hypothesis set at each time instant, which is then mapped to bounding boxes sitting on the same support, it is more efficient to use a detection network where the convolutional features are shared by object proposals via ROI pooling: Once predicted, all the box coordinates are fed to the second stage of Faster R-CNN as object proposals in a single shot, where only one forward pass is carried out.

5.2.4.2 Edge likelihood

An object detection network is trained to be invariant to viewpoint change and intra-class variabilities, which makes it ill-suited for pose estimation and shape identification. To that end, we train a network to measure the likelihood of edge correspondence:

$$\Phi_{\text{edge}}(k, g; I) = h\Big(\pi\big(g_{sc}^{-1}(t) g_{so}(t_r) \mathcal{S}(k)\big), \text{EdgeNet}(I)\Big) \tag{5.8}$$

where $h(\cdot, \cdot)$ is some proximity function which measures the proximity of edge map constructed from pose and shape hypothesis via rendering (first argument of h) and edge map extracted from the image (second argument of h).

A popular choice for proximity function h is one-dimensional search [BI97, DC02, KM06], which we adopt (see Sup. Mat. for details). Such a method is geometric and more robust than appearance based methods which are photometric and subject to illumination change. However, due to its nature of locality, this method is also sensitive to background clutter and can be distracted by texture-rich image regions. Fortunately, these weaknesses are easily compensated by Φ_{CNN} which has a large receptive field and is trained on semantics. Also, instead of using Canny [Can87] or other non-learning-based edge features, we design an edge detection network (Sect. 5.3.2) on semantically relevant training sets. Fig. 5.5 shows examples illustrating background distraction.

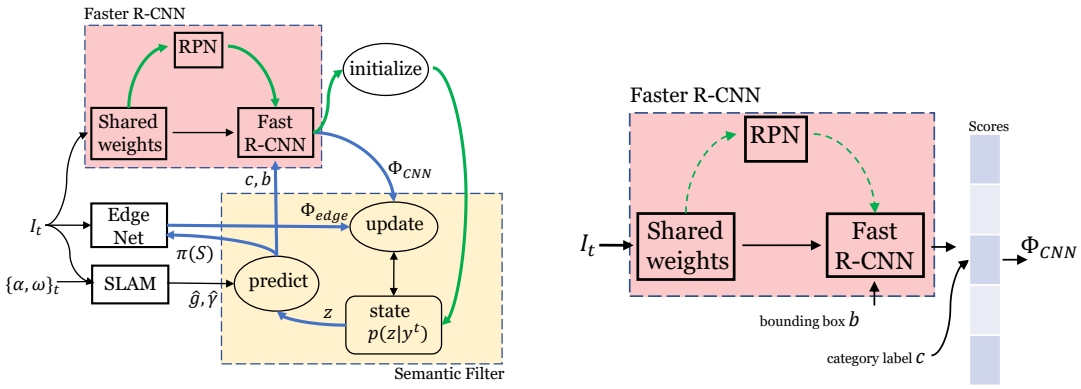
5.3 Implementation Details

5.3.1 System Overview

An overview of the system is illustrated in the system flowchart (Fig. 5.1a). We perform Bayesian inference by interleaving bottom-up (the green pathway) and top-down (the blue pathway) processing over time, which both rely on CNNs. Faster R-CNN as a bottom-up proposal generation mechanism takes input image I_t and generates proposals for initialization of new objects. In the top-down hypothesis validation process, both geometric (edge net, takes object contour $\pi(S)$ and outputs likelihood Φ_{edge}) and semantic (Fast R-CNN, takes predicted bounding box b and class label c and outputs likelihood Φ_{CNN}) cues are used. Faster R-CNN consists of a region proposal network (RPN) and a Fast R-CNN, which share weights at early convolutional layers. RPN is only activated in the bottom-up phase to feed Fast R-CNN object proposals of which bounding box coordinates are regressed and class label is predicted. During top-down phase, proposals needed by Fast R-CNN are generated by first sampling from the prior distribution $p(z|y^{t-1})$ followed by a diffusion and then mapping each sample to a bounding box b and a class label c. Fig. 5.1b illustrates the scoring process. The semantic filter (yellow box) is a variant of bootstrap algorithm [GSS93] and recursively estimates the posterior $p(z|y^t)$ as a set of weighted particles. Point estimates of gravity γ and camera pose g are from the SLAM module.

5.3.2 SLAM and Network Modules

We implement the system in C++ and OpenGL Shading Language (GLSL, for rendering) and follow a modular design principle: Each major module runs in its own process and communicates via a publish/subscribe message transport system, which enables expandability and possible parallelism in the future. The visual-inertial SLAM is based on [TCS15] which produces gravity-referenced and scaled camera pose estimates needed by the semantic mapping module. An off-the-shelf Faster R-CNN implementation [GRG18] with weights pre-trained on Microsoft COCO is turned into a service running constantly in the back-



(a) System flowchart

(b) Scoring process

Figure 5.1: **Left** *System flowchart.* Green pathway: Faster R-CNN as a bottom-up proposal generation mechanism. Blue pathway: Top-down hypothesis validation process. Pink box: Faster R-CNN. Yellow box: Semantic filter. **Right** *CNN as scoring mechanism.* Dashed pathway (proposal generation) is inactive during hypothesis testing. See system overview of Sect. 5.3.1 for details.

ground. Note we take the most generic object detector as it is *without fine-tuning on specific object instances*, which differs from other object instance detection systems. The benefit is scalability: No extra training is required when novel object instances are spotted. For the weakly semantic-aware edge detection network, we adapt SegNet [BKC17] to the task of edge detection: The last layer of SegNet is modified to predict the probability of each pixel being an edge pixel. Weights pre-trained on ImageNet are fine-tuned on BSDS [MFT01]. Fig. 5.4 shows sample results of our edge detection network.

5.3.3 Occlusion and Multiple Objects

We turn to some heuristics to handle occlusion due to its combinatorial nature. Fortunately, this is not a problem because we explicitly model the shape of objects, of which a Z-Buffer of the scene can be constructed with each object represented as its most likely shape at expected pose (Fig. 5.4 and 5.5). Only the visible portion of the edge map is used to measure the edge likelihood while Faster R-CNN still runs on the whole image, because object detectors should have seen enough samples with occlusion during the training phase and thus robust

to occlusion.

5.3.4 Initialization

An object proposal from Faster R-CNN is marked as "explained" if it overlaps with the predicted projection mask by a large margin. For those "unexplained" proposals, we initialize an object attached to the current camera frame by spawning a new set of particles. For each particle: The bearing vector $[x_c, y_c]^\top$ is initialized as the direction from the optical center to the bounding box center with a Gaussian perturbation. The log depth is initialized at a nominal depth value with added Gaussian noise. Both the azimuth and the shape label are sampled from uniform priors. More informative priors enabled by data-driven approaches are left for future investigation.

5.3.5 The Semantic Filter

We summarize our joint pose estimation and shape identification algorithm in Alg. 1, which is a hybrid bootstrap filter [GSS93] with Gaussian kernel for dynamics and a discrete proposal distribution for shape identification: The shape label stays the same with high probability and jumps to other labels equally likely to avoid particle impoverishment. A breakdown of the computational cost of each component can be found in the Sup. Mat.

5.3.6 Computational Cost

Visual-inertial SLAM runs at $\sim$ 300Hz. Edge extraction runs at $\sim$ 300Hz. Faster R-CNN runs at $\sim$ 10Hz in both proposal generation and hypothesis scoring mode. The bottleneck is the naive implementation of our rendering pipeline in the prediction step: Rendering contour maps of 1K particles takes $\sim$ 300ms. Typically a budget of 500 particles is allocated to each object in the scene to achieve reliable estimation. Once the likelihood terms are gathered, overhead to update the posterior is negligible. All the timings are done on 640×480 imagery and a laptop with a GTX1080 GPU, an i7 CPU @ 2.7GHz and 32GB RAM. We expect a reduction in computational time through more advanced rendering techniques and parallel

Algorithm 1 Semantic Filter

1. Initialization

When an unexplained bottom-up proposal is found at time $t = t_r$, sample $\{k, g\}_{t_r}^{(i)} \sim p(\{k, g\}_{t_r})$ and attach object to camera frame t_r. (Sect. 5.3.4, Initialization)

2. Importance Sampling

At time $t \geq t_r$, sample $\{k, g\}_t^{(i)} \sim q(k_t^{(i)}|k_{t-1}^{(i)})\mathcal{N}(g_t^{(i)}; g_{t-1}^{(i)}, \Sigma_{t-1})$ and compute weights $w_t^{(i)} = \exp\left(\alpha \cdot \Phi_{\text{CNN}} + \beta \cdot \Phi_{\text{edge}}\right)$. (Sect. 5.2.4)

3. Resampling

Resample particles $\{k, g\}_t^{(i)}$ with respect to the normalized importance weights $w_t^{(i)}$ to obtain equally weighted particles $\{k, g\}_t^{(i)}$.

4. Occlusion handling

Construct Z-Buffer at mean state to explain away bottom-up object proposals. (Sect. 5.3.3, Occlusion)

Set $t \leftarrow t + 1$ and go to step 1.

processing of particles.

5.4 Experiments

We evaluate our system thoroughly in terms of mapping and object detection. While there are several benchmarks for each domain, very few allow measuring simultaneously localization and reconstruction accuracy, as well as 3D object detection.

In particular, [SEE12, HWM14] are popular for benchmarking RGB-D SLAM: one is real, the other synthetic. KITTI [GLS13] enables benchmarking SLAM as well as object detection and optical flow. Two recent visual-inertial SLAM benchmarks are [BNG16] and [PSD17]. Unfortunately, we find these datasets unsuitable to evaluate the performance of our system: Either there are very few objects in the dataset [SEE12, HWM14, BNG16, PSD17], or there are many, but no ground truth shape annotations are available [GLS13].

On the other hand, object detection datasets [EVW10, RDS15, LMB14] focus on objects as regions of the image plane, rather than on the 3D scene. [XMS14, XKC16] are among

the few exploring object attributes in 3D, but are single-image based. Not only does our method leverage video imagery, but it requires a Euclidean reference, in our case provided by inertial sensors, making single-image benchmarks unsuitable.

Therefore, to measure the performance of our method, we had to construct a novel dataset, aimed at measuring performance in visual-inertial semantic mapping. We call this the VISMA set, which will be made publicly available upon completion of the anonymous review process, together with the implementation of our method.

VISMA contains 8 richly annotated videos of several office scenes with multiple objects, together with time-stamped inertial measurements. We also provide ground truth annotation of several objects (mostly furniture, such as chairs, couches and tables) (Sect. 5.4.2.1). Over time we will augment the dataset with additional scanned objects, including moving ones, and outdoor urban scenes. The reason for selecting indoors at first is because we could use RGB-D sensors for cross-modality validation, to provide us with pseudo-ground truth. Nevertheless, to demonstrate the outdoor-applicability of our system, we provide illustrative results on outdoor scenes in Fig. 5.3.

We also looked for RGB-D benchmarks and datasets, where we could compare our performance with independently quantified ground truth. SceneNN [HPN16] is a recently released RGB-D dataset, suitable for testing at least the semantic mapping module of our system, even though originally designed for deep learning. Sect. 5.4.3 describes the experiments conducted on SceneNN.

5.4.1 VISMA Dataset

A customized sensor platform is used for data acquisition: An inertial measurement unit (IMU) is mounted atop camera equipped with a wide angle lens. The IMU produces time-stamped linear acceleration and rotational velocity readings at 100Hz. The camera captures 500×960 color images at 30Hz. We have collected 8 sequences in different office settings, which cover $\sim$ 200m in trajectory length and consist of $\sim$ 10K frames in total.

To construct the database of 3D models, we rely on off-the-shelf hardware and software,

Figure 5.2: **Top** *Sample objects in the VISMA dataset.* Each mesh has ∼5000 faces and is placed in an object-centric canonical frame, simplified, and texture-mapped. **Bottom** *(Pseudo) ground truth* from different viewpoints with the last panel showing an augmented view with models aligned to the original scene.

specifically an Occipital Structure Sensor [1] on an iPad, to reconstruct furniture objects in office scenes with the built-in 3D scanner application. This is a structured light sensor that acts as an RGB-D camera to yield water-tight surfaces and texture maps. We place the 3D meshes in an object-centric canonical frame and simplify the meshes via quadratic edge collapse decimation using MeshLab [2]. Top row of Fig. 5.2 shows samples from our database. While the database will eventually be populated by numerous shapes, we use a small dictionary of objects in our experiments, following the setup of [SNS13]. An optional shape retrieval [SYS16] process can be adopted for larger dictionaries, but this is beyond the scope of this paper and not necessary given the current model library.

[1] http://www.structure.io

[2] http://www.meshlab.net

5.4.2 Evaluation

Comparing dense surface reconstruction is non-trivial, and several approaches have been proposed for RGB-D SLAM: Sturm *et al.* [SEE12] use pose error (RPE) and absolute trajectory error (ATE) to evaluate RGB-D odometry. To ease the difficulty of ground truth acquisition, Handa *et al.* [HWM14] synthesized a realistic RGB-D dataset for benchmarking both pose estimation and surface reconstruction, according to which, the state of the art RGB-D SLAM systems have typical ATE of $1.1 \sim 2.0$cm and average surface error of $0.7 \sim 2.8$cm [WLS15], which renders RGB-D SLAM a strong candidate as our (pseudo) ground truth for the purpose of evaluating visual-inertial-semantic SLAM system.

5.4.2.1 Ground Truth

To obtain (pseudo) ground truth reconstruction of experimental scenes, we run ElasticFusion [WLS15], which is at state-of-the-art in RGB-D SLAM, on data collected using a Kinect sensor. In cases where only partial reconstruction of objects-of-interest was available due to failures of ElasticFusion, we align meshes from our database to the underlying scene via the following procedure: Direction of gravity is first found by computing the normal to the ground plane which is manually selected from the reconstruction. Ground truth alignment of objects is then found by rough manual initialization followed by orientation-constrained ICP [ZPK18] where only rotation around gravity is allowed. Bottom row of Fig. 5.2 shows a reconstructed scene from different viewpoints where the last panel shows an augmented view.

5.4.2.2 Metrics and Results

We adopt the surface error metric proposed by [HWM14] for quantitative evaluation. First, a scene mesh is assembled by retrieving 3D models from the database according to the most likely shape label, to which the pose estimate is applied. A point cloud is then densely sampled from the scene mesh and aligned to the ground truth reconstruction from RGB-D SLAM via ICP, because both our reconstructed scene and the ground truth scene are up to

Error Metric		Clutter1	Clutter2	Occlusion1	Occlusion2
Surface	Median(cm)	1.37	1.11	1.30	2.01
	Mean(cm)	1.99	1.39	1.73	2.79
	Std.(cm)	1.96	1.12	1.45	2.54
	Max(cm)	17.6	9.88	14.3	17.9
Pose	Mean Trans. (cm)	4.39	2.42	3.94	13.64
	Mean Rot. (degree)	6.16	4.66	4.86	9.12

Table 5.1: *Surface error and pose error* measured over 4 sequences from the VISMA dataset. Qualitative results on the other 4 sequences with coarse annotations can be found in the Sup. Mat. Translational error reads $\|T_{gt} - \hat{T}\|_2$ and rotational error reads $\|\log^{\vee}(\hat{R}^{\top} R_{gt})\|_2$, where $\log : \mathrm{SO}(3) \mapsto so(3)$ and $^{\vee} : so(3) \mapsto \mathbb{R}^3$. (R_{gt}, T_{gt}) and $(\hat{R}, \hat{T})$ are ground truth and estimated object pose respectively.

an arbitrary rigid-body transformation. Finally, for each point in the aligned scene mesh, the closest triangle in the ground truth scene mesh is located and the normal distance between the point and the closest triangle is recorded. Following [HWM14], four standard statistics are computed over the distances for all points in the scene mesh: Mean, median, standard deviation, and max (Table 5.1). In addition to surface error, Table 5.1 also includes pose estimation error which consists of translational and rotational part. Fig. 5.4 shows how common failures of an image-based object detector have been resolved by memory (state of the semantic filter) and inference in a globally consistent spatial frame.

5.4.3 Experiments on SceneNN Dataset

For independent validation, we turn to recent RGB-D scene understanding datasets to test at least the semantic mapping part of our system. Although co-located monocular and inertial sensors are ubiquitous, hence our choice of sensor suite, any SLAM alternative can be used in our system as the backbone localization subsystem as long as reliable metric scale and gravity estimation are provided. This makes SceneNN suitable for testing the semantic mapping part of our system, although originally designed for RGB-D scene understanding. It

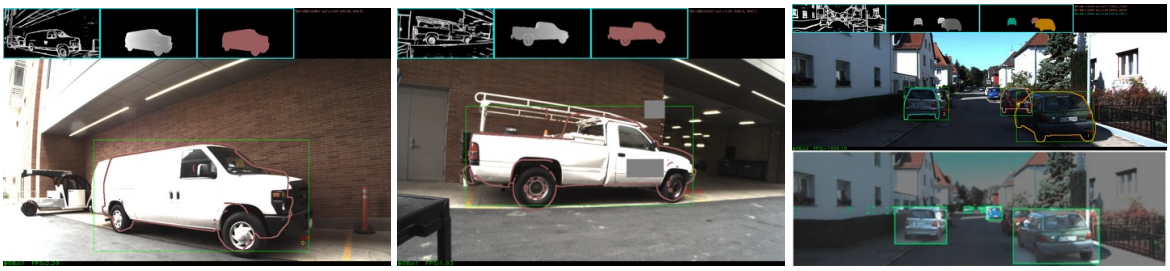


Figure 5.3: *Exemplary outdoor results* (best viewed in color at 5×). In each panel, top inset shows (left to right): edge map, Z-buffer, projection masks; bottom shows input RGB with predicted mean object boundary and CNN detection. Rightmost panel shows a visual comparison of ours (top) against Fig. 1 of [DFS17] (bottom), where we capture the boundaries of the cars better. Though only generic models from ShapeNet are used in these examples, pose estimates are fairly robust to shape variations.

provides ground truth camera trajectories in a gravity-aligned reference frame. Raw RGB-D streams and ground truth meshes reconstructed from several object-rich real world scenes are provided in SceneNN.

To test the semantic mapping module on SceneNN, we take the ground truth camera trajectory and color images as inputs. *Note the depth images are not used in our experiments.* The database is constructed by manually selecting and cropping object meshes from the ground truth scene mesh. A subset scenes of SceneNN with various chairs is selected for our experiments. Except the fact that the camera trajectory and gravity are from the ground truth instead of from our visual-inertial SLAM, the rest of the experiment setup are the same as those in the experiment on our own dataset. Table 5.2 shows statistics of surface error of our semantic mapping on SceneNN. Typical mean surface error is around 3cm. Fig. 5.5 shows some qualitative results on SceneNN.

5.5 Discussion

Our method exploits monocular images and time-stamped inertial measurements to construct a point-cloud model of the environment, populated by object models that were recognized,

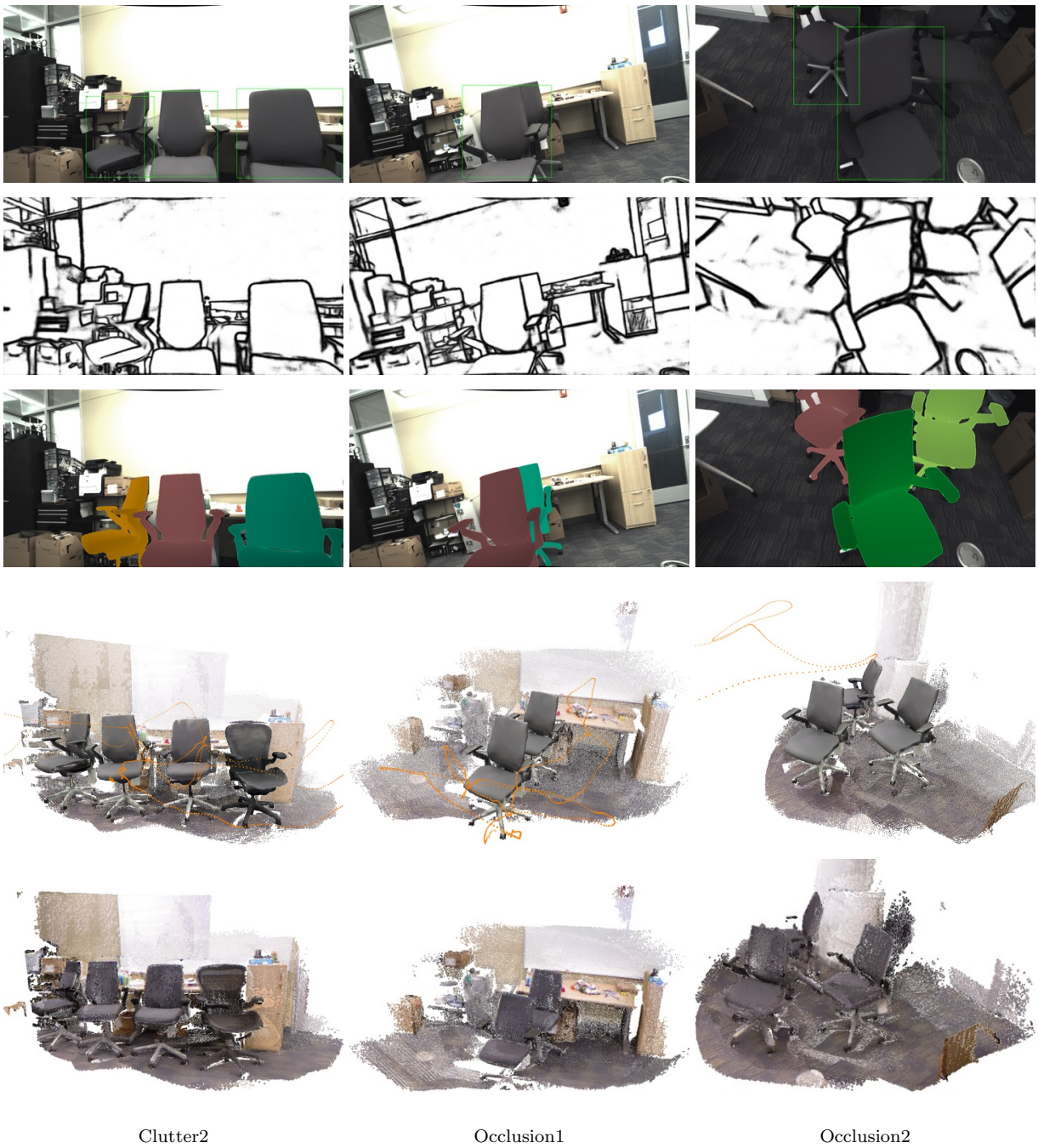


Figure 5.4: *Qualitative results* (best viewed in color at 5×). Each column shows (top to bottom): One frame of the input video with CNN bounding box proposals with confidence > 0.8; Extracted edge map; Frame overlaid with predicted instance masks shaded according to Z-Buffer – darker indicates closer; Background reconstruction augmented with camera trajectory (orange dots) and semantic reconstruction from our visual-inertial-semantic SLAM; Ground truth dense reconstruction. Missed detections due to heavy occlusion (middle column) and indistinguishable background (right column) are resolved by memory and inference in a globally consistent spatial frame.

Sequence	005	025	032	036	043	047	073	078	080	082	084	096	273	522	249
Median(cm)	1.84	0.726	3.08	2.25	3.66	3.10	2.59	3.04	2.82	2.35	1.29	0.569	2.06	1.31	0.240
Mean(cm)	3.47	0.756	6.28	4.10	4.24	4.11	3.04	3.51	3.15	3.32	1.70	0.684	2.15	1.69	0.299
Std.(cm)	3.48	0.509	6.95	5.10	3.11	3.52	2.17	2.60	2.09	2.99	1.51	0.518	1.24	1.39	0.217
Max(cm)	13.7	3.07	36.3	34.3	11.9	18.5	8.72	17.4	13.9	22.7	8.33	4.41	5.75	5.60	1.27

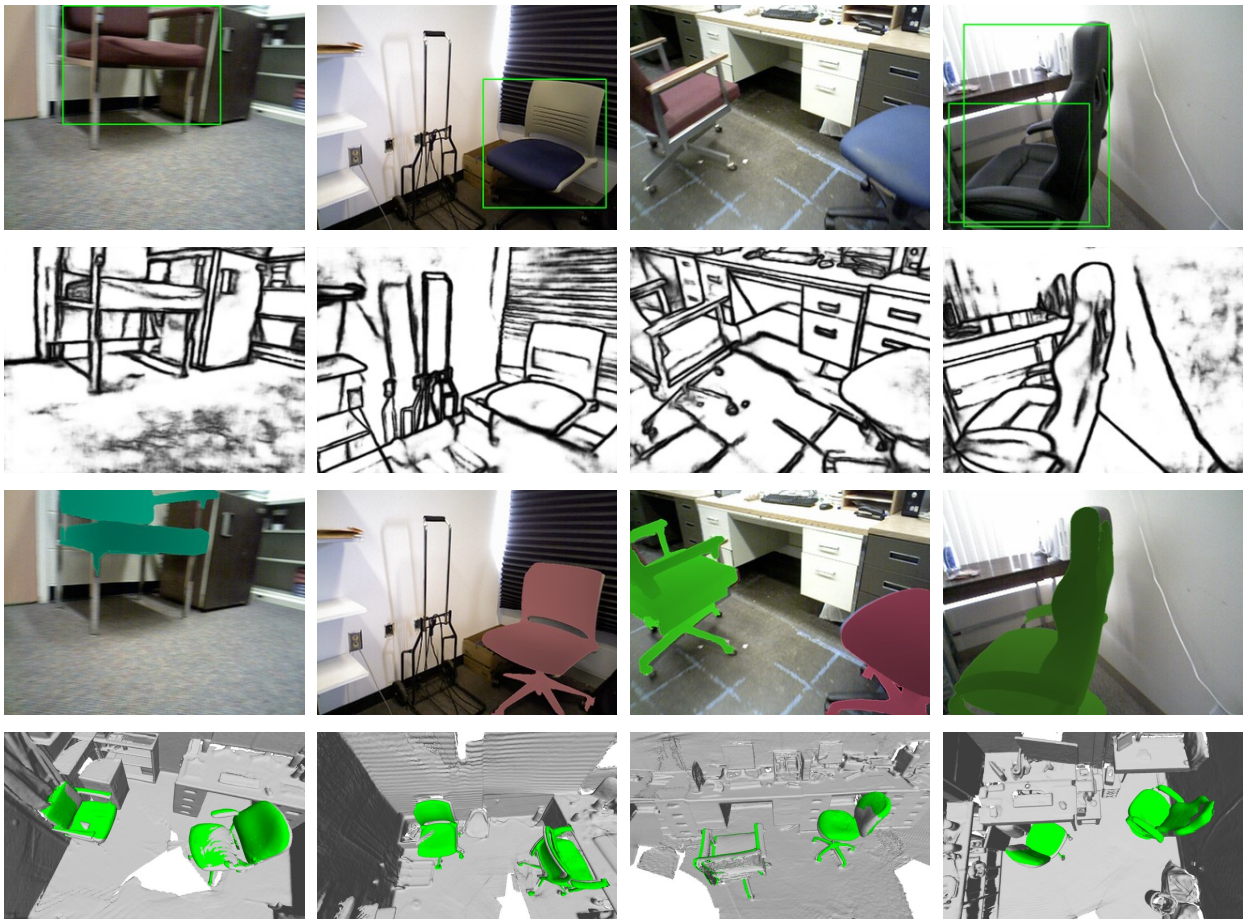
Table 5.2: *Surface error* measured on a subset of the SceneNN dataset.

along with the camera trajectory in an Euclidean frame. We target indoor and outdoor mobility scenarios, and focus on indoor for evaluation due to the availability of benchmark. Yet no benchmark has inertial and semantic ground truth, so we have introduced VISMA.

We believe most mapping and navigation methods in the near future will utilize this modality as it is ubiquitous (*e.g.,*, in every smart phone or car, even some vacuum cleaners). Yet, at present, ours is one of few methods to exploit inertials for semantic mapping in the literature.

Our method has several limitations: It is limited to rigid objects and static scenes; it is susceptible to failure of the low-level processing modules, such as the detection or edge networks. It works for object instances, but cannot handle intra-class variability. It is not operating in real time at present, although it has the potential to.

Future extensions of this work include expansions of the VISMA dataset, the addition of synthetic scenes with rich ground truth. Extensions to independently moving objects, and deforming objects, is also an open area of investigation.



025 (motion blur) 043 (distraction) 036 (missed detection) 096 (duplicate)

Figure 5.5: *Qualitative results on SceneNN.* (best in color at 5×) Each panel has the same meaning as Fig. 5.4. Last row shows estimated shape & pose (green) overlaid on ground truth mesh (gray). Partial projections due to broken models provided by SceneNN. 1st col: Moderate motion blur does not affect edge extraction. 2nd col: Background distraction does not affect shape & pose inference thanks to the holistic and semantic knowledge injected into low-level edge features. 3rd col: Missed detections due to truncation resolved by memory. 4th col: Duplicate detection from Faster R-CNN eliminated by memory and inference in a consistent spatial frame.

CHAPTER 6

Efficient Large-Scale Loop Closure Detection

6.1 Introduction

We tackle the problem of *loop closure* in vision-based navigation. This is a particular classification task whereby a training set of images is indexed by location, and given a test image one wants to query the database to decide whether the former is present in the latter, and if so return the indexed location. This is closely related to *scene recognition*, where the focus is on a particular instance, as opposed to an object class (we want to determine whether we are at particular intersection in a given city, not whether we are at *some* intersection of *some* urban area). As such, test images are only subject to *nuisance variability* due to viewpoint, illumination and partial occlusion from moving objects, but otherwise there is no *intrinsic* (intra-class) variability.

The state-of-the-art for image retrieval is based on convolutional neural network (CNN) architectures, trained to marginalize nuisance and intrinsic variability. In a discriminatively trained network, the compositionality property afforded by linear convolutions, while critical to model intra-class variability, is unhelpful for loop closure, as there is no intrinsic variability. At the same time, a CNN does not respect the topology of data space at higher levels of the hierarchy, since filters at any given layer are supported on the entire feature map of the previous layer. In loop closure, locality is key, and while one could retrieve from the feature map the locations that correspond to active units, this requires some effort [SVZ14].

Given the critical importance of loop closure in location services ranging from smartphones to autonomous vehicles, we focus on its peculiarities, and attempt to harvest some of the components of neural networks to improve the state-of-the-art. Stripped of the lin-

ear convolutions (we do not need to model intrinsic variability) and ReLu, what we have left is a *hierarchical spatially pooled data structure built upon local photometric descriptors [GID14, MV15].* There are no filters, and no learning other than the trivial pooling of local descriptors. Motivated by this intuition, we propose a new hierarchical representation for loop closure, detailed in Sec. 6.2.

Loop closure is also closely related to location, or "place," recognition [UN00, TMF03, CN09] and large-scale visual search [NS06, CPS07, JDS08], but with some important restrictions.

First, both previous data (training images) and current (test, or query) data are usually available as time-indexed sequences, even if they are captured by different agents, and training images may be aggregated into a "map" [JS11] or reduced to a collection of "keyframes" [ND10]. Second, as a binary classification task (at each instant of time, a loop closure is either detected or not), the cost of missed detections and false alarms are highly asymmetric: We pay a high price for declaring a loop closure that isn't, but there is minor harm in missing one, as temporal continuity affords many second chances in subsequent images. This is unlike large-scale image retrieval, where we wish to find what we are looking for (few missed detections, or high recall) even if we have to wade through some irrelevant hits (many false alarms, or low precision).

Like image retrieval, however, the challenge with loop closure is scaling. In navigation applications, it may be hours before we return to a previously seen portion of the scene. Therefore, we have to store, and search through, hundreds of thousands to millions of images. Our goal in this paper is to *design a hierarchical data structure that helps speed up matching by leveraging on the two domain-specific constraints above:* temporal adjacency, and high precision.

Assuming continuous trajectories, the first translates to proximity in pose space SE(3) (position and orientation). For the second, the best trade-off with missed detections can be achieved by testing every datum in the training set via *linear search accelerated via an inverted index.* Our goal is to achieve similar performance at a fraction of the cost compared

to inverted index search. This cannot be achieved in a worst-case setting. What matters instead is *average performance* trading off precision with computational cost. We evaluate such average performance empirically on the *KITTI* [GLS13], *Oxford* [CN09] and *TUM RGB-D* [SEE12] datasets, as well as demonstrate extensions to general image retrieval on the *ukbench* [NS06] and *INRIA Holidays* [JDS08] datasets. To demonstrate scalability, we also evaluate our algorithm on augmented datasets with around 40K images.

We propose a simple data structure based on hierarchical pooling of location likelihoods – in the form of sample distributions of BoW descriptors – with respect to the topology of pose space. In practice, this means simply constructing BoW descriptors, that represent the likelihood of the locations that generated them, and pooling them temporally in a fine-to-coarse fashion, either by averaging, summing, or taking the index-wise maximum.

While averaging likelihoods may seem counter-productive, in Sec. 6.2 we show it makes sense in the context of the classical theories of sampling and anti-aliasing. In Sec. 6.3 we show that, despite its simplicity, it works as well as sophisticated agglomerative schemes at a fraction of the effort

6.1.1 Related work

Loop closure is a key component in robotic mapping (SLAM) [WCN09], autonomous driving, location services on hand-held devices, and for wearables such as virtual reality displays. Loop closure methods can be roughly divided into 3 categories: appearance-only, map-only and methods in between. Appearance-only methods [CN09] are essentially large-scale image retrieval algorithms, influenced by [NS06] and more in general the literature of BoW object recognition and categorization [SZ03]. Map-only methods [KM07] use the data (images, but most often range sensors) to infer the configuration of points in 3D space, and then seek to match subsets of these points, often using variants of ICP [CSS02] as a building block. These methods do not scale beyond a few hundreds of thousands of points, or thousands of keyframes, and are often limited to what is referred to as "short-term" loop closure [KM07], necessary for instances when complete loss of visual reference occurs while tracking. There

are also a variety of map-to-image and image-to-map [SLK11] methods that show great promise, but have yet to prove scalability to the point where the map spans tens if not hundreds of kilometers [CN09].

For scalability, the most common choice is to combine quantized local descriptors into a BoW and then use an inverted index. FAB-MAP [CN09] extends the basic setup by learning a generative model of the visual words using a Chow-Liu tree to model the probability of co-occurrence of visual words. FAB-MAP 2.0 scales further by exploiting sparsity to make the inverted index retrieval architecture more efficient. Starting from [GT11], SIFT or SURF descriptors were replaced by more efficient binary descriptors such as BRIEF [CLS10] and ORB [RRK11] to achieve comparable precision and recall to FAB-MAP 2.0 with an order of magnitude speed increase. Several recent mapping and localization systems adopt it as a module, including [LLK14] and ORB-SLAM [MMT15].

In addition to the specific loop closure literature, general ideas from spatial data structures and agglomerative clustering [TPB00] are also relevant to this work, including k-d trees [Sam90], dual trees and decision trees [GJ96], as well as data structures used for retrieval such as pyramid matching [GD05] and its spatial version [LSP06]. In more general terms, this work also relates to visual navigation and mapping, structure-from-motion, and location recognition, including the use of global descriptors [TMF03].

Our method can be considered appearance-only, but it is loosely informed by geometry, in the sense that the scene domain (pose space) provides the topology with respect to which we pool descriptors. Also closely related to our approach are [TL09, TSP11], which present techniques for merging only pairs of BoWs; in [CPS07] queries are expanded by using retrieved and verified images, which is orthogonal to and can be viewed as a *query-end* version of our method.

6.2 Methodology

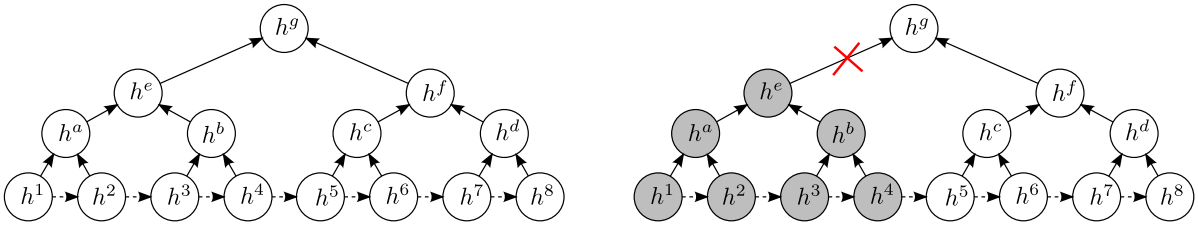
Since our focus is on a spatial structure that facilitates accelerated loop closure queries, we integrate components from recent state-of-the-art methods within our data structure and

adopt such methods as a baseline, against which we compare our method. Specifically, we adopt [MMT15] as a baseline, consisting of a BoW where each word is an element of a dictionary of descriptors obtained off-line by hierarchical k-means clustering, with each word weighted by its inverse document frequency. FAST detectors [RD06] and BRIEF descriptors [CLS10] are employed, and TF-IDF [BNJ03, Aiz03, SZ03] is used to weigh the BoW relative to the inverse document frequency. This standard pipeline, with different clustering procedures to generate the dictionary and different features, comprises most basic large-scale retrieval systems, including appearance-only loop closure. However, the number of false alarms in large-scale settings is crippling, so temporal consistency and geometric verification are typically used as correction mechanisms.

6.2.1 Hierarchical testing

6.2.1.1 Construction of hierarchy

Our data structure can be interpreted as a hierarchical version of TF-IDF. To illustrate the method, we first assume that every frame is a "keyframe" and therefore we have a time-series of BoWs, obtained as described above, and organized into a linear structure or *un-oriented list*, as we wish to retrieve frames regardless of the direction of traversal. Each node is associated with a histogram, in the form of a BoW, representing the likelihood of a pose $g(t) \in \mathrm{SE}(3)$ (position $T(t) \in \mathbb{R}^3$ and orientation $R(t) \in \mathrm{SO}(3)$) given the data (the image at time t, $I(t)$): $h^t \doteq \mathrm{BoW}(t) \sim p(I(t)|g(t))$, where the equivalence is up to normalization, and the density function is approximated with a histogram with N bins, equal to the size of the dictionary.



(a) Construction of hierarchy (b) Hierarchical testing

Figure 6.1: (a) **Construction of hierarchy** for an 8-long sequence of (key)frames and constant branching factor of 2. Dashed lines indicate temporal order. (b) **Hierarchical testing**: If h^e does not score higher than the threshold, the whole sub-tree rooted at h^e (shaded) will not be searched. In the case of sum- or max-pooling, this would *not* introduce loss of precision compared to searching only the lowest level nodes.

We now construct a second level, or "layer", of the data structure, simply by pooling adjacent histograms (Fig. 6.1a). This is repeated for higher layers until either a maximum depth is reached, or until a single root node is left. Several standard choices for the pooling operation are available which allow us to trade off between precision and cost (Sec. 6.3). Suppose h^p is the parent histogram which has child histograms $\{h^k\}, k = 1, 2 \ldots K$. Both h^p and $h^k \in \mathbb{R}^N$. Mean- or average-pooling refers to $h^p = \frac{1}{K} \sum_{k=1}^{K} h^k$, sum-pooling refers to $h^p = \sum_{k=1}^{K} h^k$, and max-pooling refers to $h_i^p = \max_k \{h_i^k\}$, where $i = 1, 2 \ldots N$. Once we have constructed the hierarchy for database histograms, raw histograms are used as queries for loop closure detection.

6.2.1.2 Query processing

Similarities between pooled and query histograms are computed using the *intersection kernel* [SB91], that is the area of the intersection of the two histograms. Thus, if h^q (a query histogram) has bin values $h_1^q, \ldots, h_N^q$, and similarly for h^p, we have that

$$\mathbb{I}(h^q, h^p) = \sum_{i=1}^{N} \min\{h_i^q, h_i^p\} \tag{6.1}$$

The intersection kernel is related to many divergence functions [Vas04] as well as to metrics used in optimal transport problems.

Sum- and max-pooling operators have the following upper bound property when intersection kernel is applied: For a query histogram h^q, a parent histogram h^p and its child h^k in the database,

$$\mathbb{I}(h^q, h^p) > \mathbb{I}(h^q, h^k) \tag{6.2}$$

therefore if $\mathbb{I}(h^q, h^p) < \tau$, $\mathbb{I}(h^q, h^k) < \tau$ must hold.

Since our goal is to search for the closest match, or at least for all matches that exceed a threshold $\tau > 0$ (we seek large values of $\mathbb{I}$), if $\mathbb{I}(h^q, h^p) < \tau$, the chance of any of h^p's descendants exceeding the threshold is rare (or impossible, in the case of max- or sum-pooling as shown by the upper bound property), therefore we stop searching the sub-tree rooted at h^p (Fig. 6.1b).

Therefore, search in a hierarchical TF-IDF setting simply boils down to *greedy breadth-first search, while maintaining an inverted index for each layer.* If only one layer is used, this reduces to standard linear search using an inverted index.

A key point is that with sum- or max-pooling, the proposed method *has exactly the same precision-recall behavior as standard inverted index search* while still achieving a substantial speedup. With mean-pooling, a large speedup can be achieved with only a minimal loss of precision (Sec. 6.3).

Different trees with different depths and different branching factors can be constructed, trading off expected risk and computation time, characterized empirically in Sec. 6.3.4. In addition to a fixed depth and branching factor, one could devise more clever schemes to determine the topology of the tree, discussed in Sec. 6.2.2. However, we find that the benefit is limited compared to the straightforward fixed-topology architecture.

6.2.2 Keyframes and adaptive tree topology

So far we have assumed that the time-series of data $\{h^t\}_{t=1}^T$ is sampled regularly (at constant time or space intervals), but it can also be sampled adaptively, by exploiting statistics of the

data stream to decide which samples, or *keyframes*, to use. The data structure above does not change, since all that is required is a topology or adjacency structure to construct the tree.

Adaptive (sub)-sampling can be done in many ways, and there are a wide variety of standard heuristics for selecting keyframes. Our goal here is not to determine the best method for selecting keyframes, but to focus on the data structure regardless of the sub-sampling mechanism. Consequently, we limit ourselves to constructing it either on the raw time series, or on any subsampling of it, as generated by standard keyframe selection methods.

Just like selecting keyframes, building the hierarchy can be understood as a form of (sub)-sampling. Regardless of whether subsampling is regular (as in building the tree above) or adaptive (as in selecting keyframes), classical sampling theory [SZ05] suggests that what should be stored at the samples is *not* the value of the function, but the local average relative to the topology of the domain where the data are defined (*anti-aliasing*). This lends credence to the use of mean-pooling, which initially may seem counter-intuitive since our goal is to maintain high precision.

In our case, the domain is time, or the order of keyframes, as a proxy of location in SE(3). The range of the data is the space of likelihood functions, approximated by histograms h^t. Therefore, anti-aliasing simply reduces to averaging neighboring histograms. The study of the optimal averaging, both in terms of support and weights, is beyond our scope here, where for mean-pooling we simply average nearest neighbors in the tree topology relative to a uniform prior. We do not delve into considering more sophisticated anti-aliasing schemes, since we have found that simple topologies yield attractive precision-computational cost trade-off, which is unlikely to be significantly disrupted by fine-tuning the weights.

The practice of averaging likelihood functions as a way of anti-aliasing descriptors has also been recently shown by [DS15] in the context of pooling local descriptors for correspondences in wide-baseline matching. Our method can be considered an extension (or special case) where the correspondence and pooling are performed in time, and the descriptors are

histograms of visual words, a mid-level representation, rather than histograms of gradient orientation, the result of low-level processing.

While the choice of heuristics for keyframe selection has no effect on our method, which can be applied to the raw time series or to the sequence of keyframes, the same (adaptive sampling) heuristics used to (down)-sample keyframes from the regularly sampled images could be used to aggregate nodes at one level into parents one level above. This would give rise to trees having different levels of connectivity at different layers, and indeed potentially at each node.

We have found that, in practice, these heuristics fail to yield significant performance improvements when compared to trees with fixed topology having constant splitting factors that match the average of their adaptive counter-part. Representative experiments are shown in Sec. 6.3.4.

6.3 Evaluation

The most important evaluation for the proposed method is to test performance in-the-loop when incorporated into a real system (ORB-SLAM [MMT15], in this case), discussed in Sec. 6.3.2 where we find a 65% reduction in mean query time with no loss in localization performance and no missed loop closures relative to the baseline. We investigate query-time reduction and precision-recall behavior while varying vocabulary size and tree topology in Sec. 6.3.3 and Sec. 6.3.4, respectively. In Sec. 6.3.5 we augment standard datasets to explore various test-time scenarios, and Sec. 6.3.6 presents a generalization of our method to other image retrieval tasks. Sec. 6.3.1 discusses the datasets and methodology used throughout the evaluation.

6.3.1 Datasets and methodology for loop closure

We perform experiments using the common loop closure datasets of *KITTI*, *Oxford City Centre*, and *Oxford New College* [GLS13, CN09]. The *KITTI* dataset consists of several

sequences on the order of 1000 stereo pairs in length. To provide additional experimental evaluation at large scale, we augment *KITTI* by concatenating all sequences, to form the *concatenated KITTI* dataset consisting of approx. 40K images. For all sequences we construct the data structure using all frames unless otherwise noted, in which case we adopt the keyframe selection strategy of our baseline (Sec. 6.3.2).

Unless otherwise stated, we build the hierarchical data structure using the left stereo images of the sequences (when stereo is available) and evaluate loop closure correctness using the provided ground truth poses. The evaluation protocol is as follows: traverse the sequence and insert BoW of images into the database incrementally, while using each image to query the database before it is added. Two images are regarded as a correct match if they were taken within 15 meters of each other. To avoid trivial matches, we prevent the query from matching temporally adjacent images. This evaluation protocol mimics loop closure in a practical SLAM system, which we test in Sec. 6.3.2.

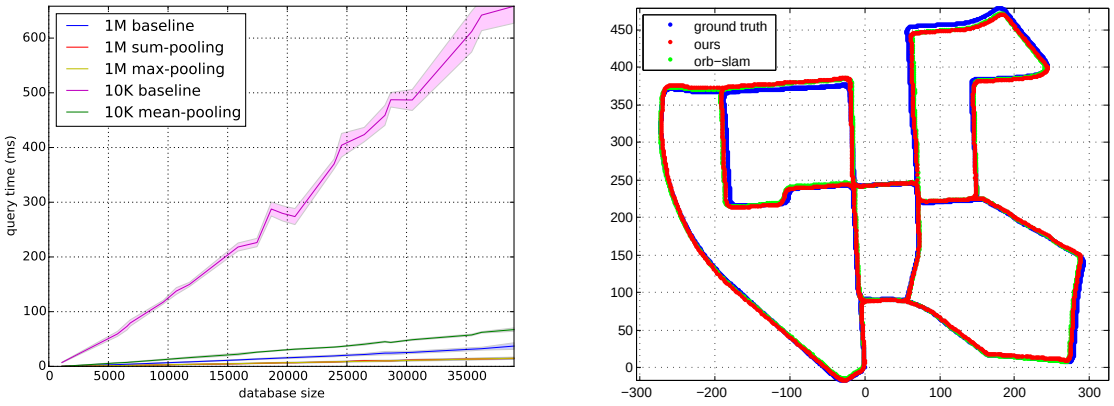
To evaluate matching, missed detection and false alarms are traded off by an arbitrary choice of threshold, as in any detection algorithm. Since the threshold affects the average query time (we can make that quite short by choosing a threshold that yields no false alarms while rejecting every hypothesis) we must come to a reasonable choice. Unless otherwise stated, we adopt the following policy: We generate precision-recall curves on *KITTI* 00. Then, we select the smallest threshold that yields zero false alarms and use it on other sequences. Of course, that may yield a non-zero false alarm rate in datasets that are not used in setting the threshold, but this (as is customary) can be handled by verification steps afterwards. This is a limitation inherent to the choice of image representation, in this case Bag-of-Words, and not a sensitivity that our hierarchical data structure is designed to circumvent.

6.3.2 In-the-loop with the baseline

We use components of ORB-SLAM [MMT15], made available by the authors, as the baseline for our experiments. We use this as a *black box* and implement our hierarchy atop its

single-layer inverted index architecture for performing image queries. As a result, we also inherit some of the limitations of its components (e.g. keyframe selection, discriminability of quantized descriptors and BoW representations, sensitivity to matching threshold selection), which are common to the majority of SLAM systems.

We first show that when using ORB-SLAM *as is*, with no change in thresholds or tuning, a significant reduction in image query time can be achieved simply by applying our max-pooling hierarchy, which by construction achieves identical precision-recall performance to the original system, missing no loop closures that may be critical to pose-graph optimization algorithms. In Fig. 6.2b, we compare the trajectories estimated by ORB-SLAM with and without our max-pooling hierarchy on *KITTI*. Errors relative to ground truth are similar (within 1σ of each other over multiple trials); mean query times are reduced by 65% (2.04ms from 5.80ms). No loop closures are missed by our max-pooling method that would not be missed without our data structure, confirming that improvement in speed comes at no loss of classification performance. In Fig. 6.2a we show this speedup holds with increasing scale by showing query times for the *concatenated KITTI* dataset for different vocabulary sizes (Sec. 6.3.3) and various pooling strategies using the methodology of Sec. 6.3.5.



(a) Scaling (b) Comparison to ORB-SLAM

Figure 6.2: (a) **Scaling:** Timings for concatenated *KITTI* sequences (approx. 40K images) with 1M and 10K vocabularies. (b) Comparison to **ORB-SLAM** with and without our data structure. Multiple trials yield nearly identical trajectories with and without our data structure, with no loop closures missed while achieving a 2-3x speedup.

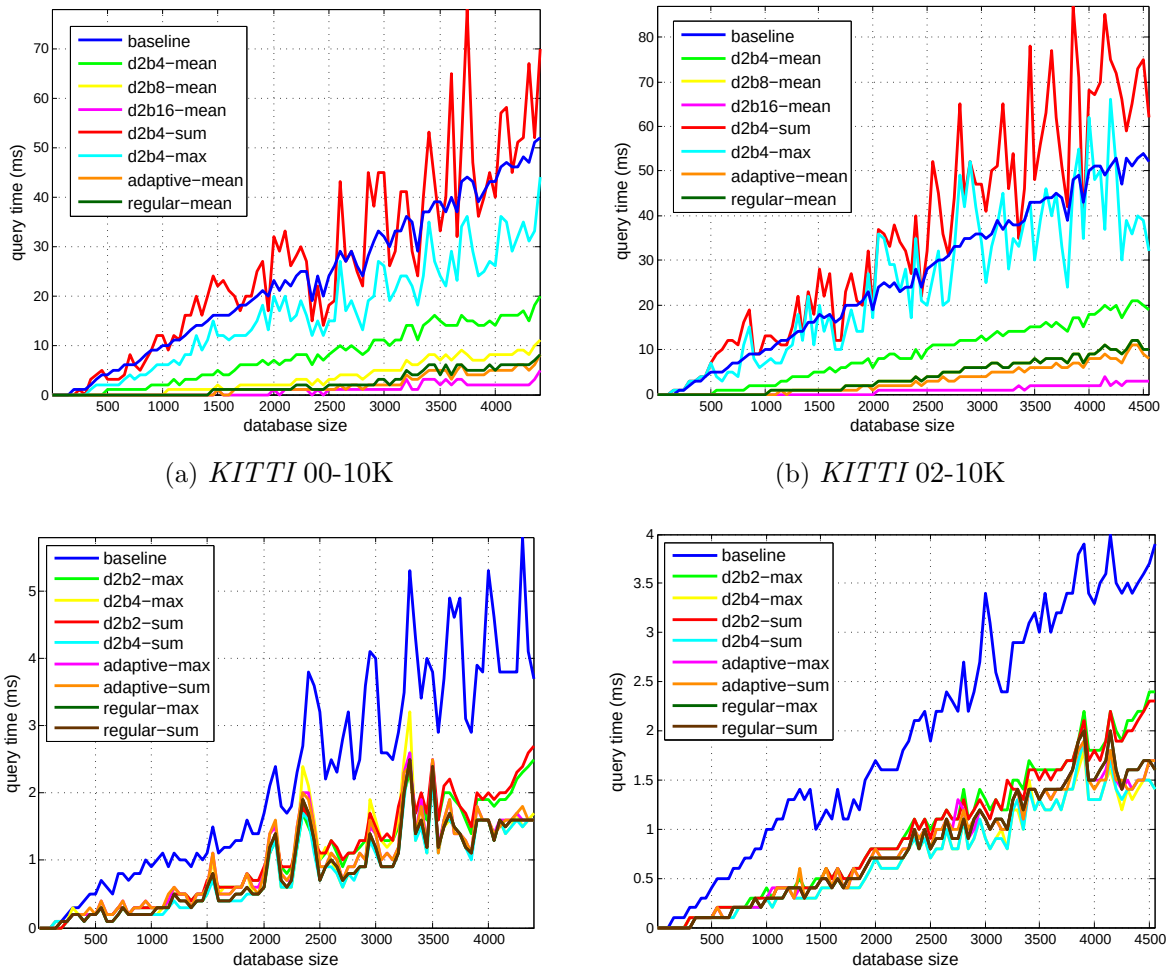
6.3.3 Varying vocabulary size

Some may argue that a speedup could be easily gained by just using a larger vocabulary. It is true that with a larger vocabulary, each visual word is associated with a much smaller list of documents in the inverted index system which leads to shorter query time. However, the vocabulary size should be determined by the performance of the specific task as well as the volume of the data and a larger vocabulary is not always better. A larger vocabulary has finer division of feature space compared to a smaller vocabulary but is also more sensitive to quantization errors (two slightly different images may have completely different histograms). In this case, mean-pooling may not be ideal as shown in Fig. 6.4c and 6.4d. However, sum/max-pooling can still be applied to gain further speedup while maintaining same precision-recall as shown in Fig. 6.3c and 6.3d, and also on augmented dataset as shown in Fig. 6.2a.

6.3.4 Varying tree topology

6.3.4.1 Variable depth and branching factor

Fig. 6.3 shows timings of the baseline and our algorithm with different topologies and pooling schemes at the same threshold on two of the *KITTI* sequences with many loop closures. Only time to query the database is counted, time for feature extraction and descriptor quantization are excluded. Fig. 6.4a and Fig. 6.4b show precision-recall curves for the mean-pooling variants. We use $d_i b_j$-X to denote a hierarchy with i layers, a branching factor of j and pooling strategy X. Note that for baseline and our proposed algorithm with configuration $d_2 b_4$-mean and $d_2 b_8$-mean, the precision-recall curves are nearly identical, while our approach is 2-5 times faster. For configuration $d_2 b_{16}$-mean, while its performance is slightly worse, it achieves *an order of magnitude speedup* relative to the baseline.



(a) *KITTI* 00-10K

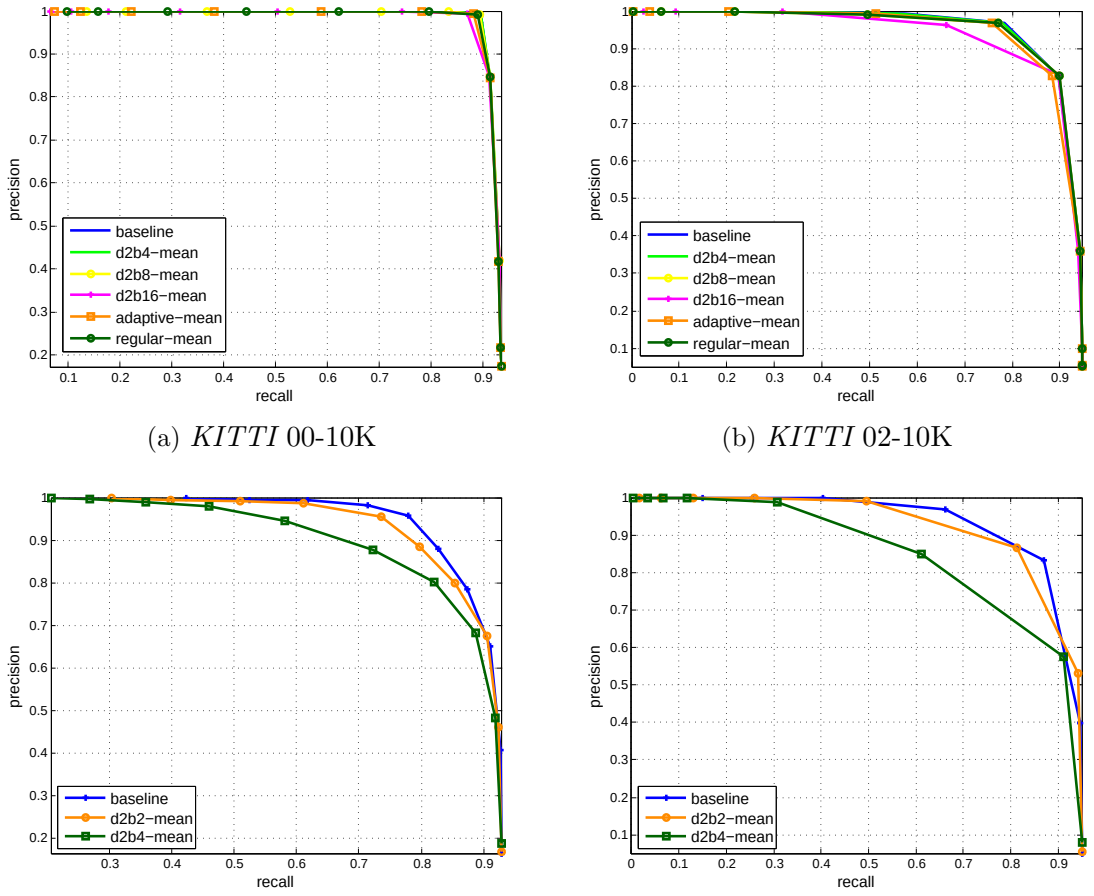
(b) *KITTI* 02-10K

(c) *KITTI* 00-1M

(d) *KITTI* 02-1M

Figure 6.3: Timings of baseline and proposed algorithm with different topologies and pooling strategies on *KITTI* dataset 00 and 02 using *all* frames. $d_i b_j$ -X: a hierarchy with i layers, a branching factor of j and pooling strategy X. Adaptive sampling: spectral clustering in SE(3). Regular sampling: sampling at the average rate of adaptive sampling scheme. Baseline: inverted index search. Two different vocabulary sizes (10K and 1M) are considered.

As mentioned in Sec. 6.2.1.2, sum/max-pooling have *exactly the same precision-recall behavior* as the baseline. In these two datasets, sum/max-pooling are slightly slower than inverted index search. Since both of these operations rapidly reduce sparsity in the histograms, we expect slower performance relative to mean-pooling. However, sum/max-pooling have their advantages when a much larger vocabulary is used as shown in Sec. 6.3.3.



(a) *KITTI* 00-10K (b) *KITTI* 02-10K

(c) *KITTI* 00-1M (d) *KITTI* 02-1M

Figure 6.4: Precision-recall curves of baseline and proposed algorithm with different topologies on *KITTI* dataset 00 and 02 using *all* frames. Two different vocabulary sizes (10K and 1M) are considered. Notations have the same meanings as in Fig. 6.3.

6.3.4.2 Adaptive domain-based clustering

In addition to the baseline algorithm, we generate a second baseline by applying the same algorithm to keyframes, rather than to all stored images. In principle, the heuristics involved in the selection of keyframes could be propagated to all nodes of the data structure, as discussed in Sec. 6.2.2. However, our experiments indicate that this yields minor benefits compared to simple averaging. The second row of Tab. 6.2a shows average time-cost rate [1]

[1]Time-cost rate is defined as the increase of query time per thousand (1k) images in the database. Average time-cost rate is the average of time-cost rates computed for all sequences in each dataset.

for searching via an inverted index among keyframes, which is worse than searching in a simple hierarchy built on raw images, as shown in the second row of Tab. 6.1c. A simple regular sampling strategy on top of keyframes can speedup searching by a large margin as shown in Tab. 6.2a.

Instead of a fixed topology of the data structure, corresponding to regular grouping, we can consider adaptive grouping based on a variety of criteria. Adaptive sampling, or grouping, based on *geometry* includes performing spectral clustering in SE(3). Curves in Fig. 6.3 indicate that adaptive sampling achieves marginal improvements compared to regular sampling at a constant rate equal to the average of the adaptive sampling rate. Similarly, parallax-based sampling, based on clustering only the translational component of pose, also yields underwhelming improvements. We do, however, expect adaptive sampling to win in some cases, as it has in a number of smaller-scale experiments we conducted with different motion characteristics from smooth driving, for instance the *TUM RGB-D* dataset (Fig. 6.5) [SEE12].

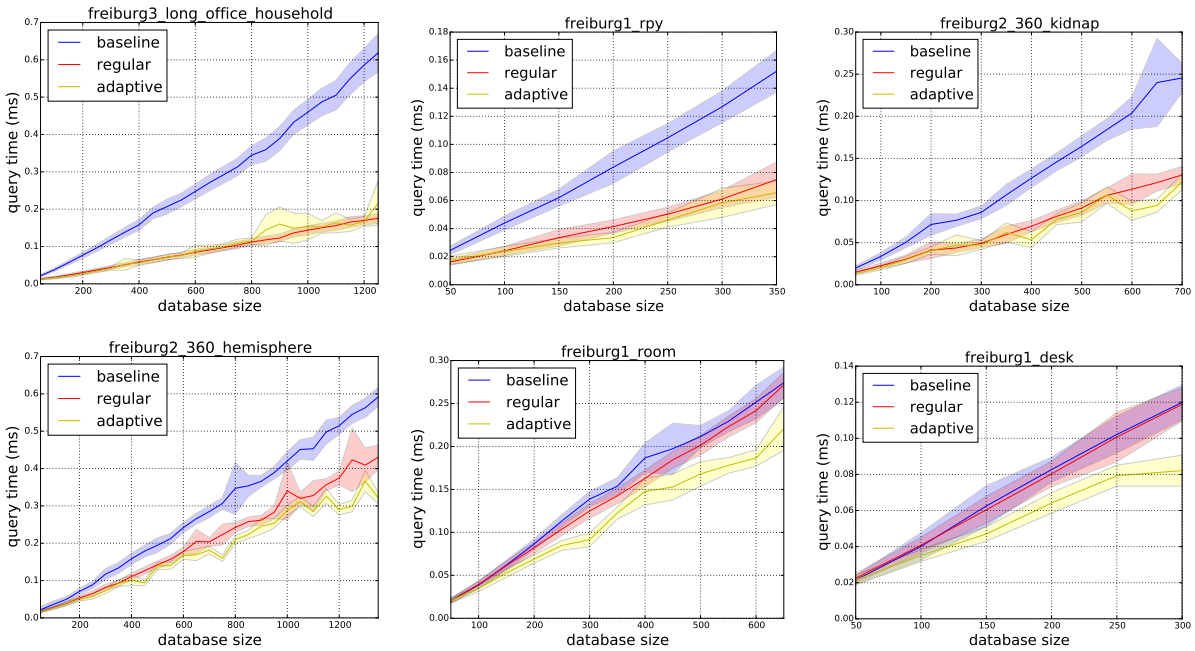


Figure 6.5: Sample results on the *TUM RGB-D* dataset using adaptive domain clustering (Sec. 6.2.2). The experiment setup is similar to that for the *Oxford* dataset in Sec. 6.3.5. Adaptive (yellow) improves with more exciting motion (left to right, up to down). Limited speedup relative to baseline due to very small dataset size. Variance shown is derived from multiple trials with sightly differing cluster assignments.

6.3.5 Quantifying speedup using synthetic ground-truth

Depending on the particular query image, our method could reduce or increase search time relative to the mean. The former occurs when correspondence fails early allowing us to rule out subsequent tests at finer scales. However, in the worst-case we may end up performing more comparisons than inverted index search when the test reaches the finest scale too often. In practice, what matters is that our algorithm shortens test time *on average* during long sequences. Since most *KITTI* sequences contain few or no loop closures, we generate synthetic positive and negative queries as follows: For sequences 01 to 21, we generate positive queries by sampling the right stereo images of each sequence (slightly different from the left images from which we constructed the database), and generate negative queries by sampling images from sequence 00. For the *Oxford* datasets, we construct the database

using odd-numbered images, generate positive queries from the even-numbered images, and negative queries again from *KITTI* 00.

Overall performance is measured by combining both sets of queries. Of course, even in the negative case our algorithm could find erroneous correspondences, which are then labeled as false alarms. Similarly, we may find no correspondence in the former case (missed detection). We use average time-cost rate to evaluate how the searching algorithm scales with size of the database. Tab. 6.1 reports experiment results on raw *KITTI*. Tab. 6.2 reports average speedup when keyframe selection is applied on both *KITTI* and *Oxford*.

structure	pooling	rate(ms/1k)	speedup
inverted index	N/A	10.07	1.00
	mean	**0.69**	**14.59**
hierarchical	sum	8.70	1.16
	max	6.65	1.52

(a) positive queries; KITTI - 10K

structure	pooling	rate(ms/1k)	speedup
inverted index	N/A	9.86	1.00
	mean	**0.34**	**29.00**
hierarchical	sum	6.28	1.57
	max	5.04	1.96

(b) negative queries; KITTI - 10K

structure	pooling	rate(ms/1k)	speedup
inverted index	N/A	9.88	1.00
	mean	**0.38**	**26.00**
hierarchical	sum	7.92	1.25
	max	6.06	1.63

(c) overall; KITTI - 10K

structure	pooling	rate(ms/1k)	speedup
inverted index	N/A	0.64	1.00
	mean	N/A	N/A
hierarchical	sum	**0.30**	**2.13**
	max	**0.30**	**2.13**

(d) overall; KITTI - 1M

Table 6.1: Average time-cost rate and speedup over 21 sequences of *KITTI* using *all* frames. 1st col: grouping strategies. 2nd col: pooling operations. 3rd col: average time-cost rate, which describes how the query time increases per 1k images inserted into the database. In 6.1a, 6.1b and 6.1c, a 10K vocabulary is used; in 6.1d, a 1M vocabulary is used.

Fig. 6.2a shows linear scaling of average query time on the much larger *concatenated KITTI*. Practical deployment on even larger datasets typically comes with context (*e.g.* GPS or odometry) that limits the data volume.

structure	pooling	rate(ms/1k)	speedup
inverted index	N/A	8.97	1.00
	mean	**0.88**	**10.14**
hierarchical	sum	7.87	1.14
	max	6.00	1.50

(a) overall; KITTI - 10K

structure	pooling	rate(ms/1k)	speedup
inverted index	N/A	6.98	1.00
	mean	**1.61**	**4.34**
hierarchical	sum	4.71	1.48
	max	4.20	1.66

(b) overall; Oxford - 10K

Table 6.2: A comparison of search in flat and hierarchical structure on *KITTI* and *Oxford* dataset. Notations have the same meanings as in Tab. 6.1 except that 3rd column describes average time-cost rate over the 21 *KITTI keyframe* sequences and all 4 sequences in the *Oxford* dataset respectively. The keyframes are generated by running ORB-SLAM.

6.3.6 Experiments in image retrieval tasks

Although our approach is geared towards the loop closure scenario, its usage is not restricted to it. A hierarchical structure of this form could be built on top of any histogram-based representation of images where some proxy of topology is available. In more general settings when a temporal stream of images is unavailable, extra labeling information, such as geotags, class labels, or textual annotations could be used. A hierarchy can be constructed using affinity between these alternate forms of metadata, provided that affinity implies proximity in the solution space. We test this using two publicly available image retrieval benchmarks: *ukbench* [NS06] and *INRIA Holidays* [JDS08].

ukbench [2] consists of 2550 groups of 4 images each (10200 total). Each group contains the same object under different viewpoint, rotation, scale and lighting conditions. We use the same evaluation protocol provided by the author: Count how many of 4 images are top-4 when using a query image from that set of four images. We use pre-computed visual words provided by the authors, which are quantized SIFT descriptors using a 1M vocabulary.

INRIA Holidays [3] contains 500 image groups (1491 total), each of which represents a

[2] http://vis.uky.edu/~stewe/ukbench/

[3] https://lear.inrialpes.fr/~jegou/data.php

distinct scene under different rotations, viewpoint and illumination changes, blurring, etc. Performance is measured by mean average precision (mAP) averaged over all 500 queries. We use the 4.5 million SIFT descriptors and 100K vocabulary provided by the authors.

structure	pooling	time(ms)	speedup	score
inverted index	N/A	1.47	1.00	2.72
Random hierarchical	mean	0.38	3.87	2.80
	sum	**0.37**	**3.97**	**2.83**
	max	0.39	3.77	2.82
Greedy affinity hierarchical	mean	0.38	3.87	2.80
	sum	0.38	3.87	**2.83**
	max	**0.37**	**3.97**	2.82

(a) ukbench

structure	pooling	time(ms)	speedup	mAP
inverted index	N/A	9.11	1.00	0.56
Random hierarchical	mean	**5.57**	**1.63**	0.58
	sum	6.19	1.47	**0.63**
	max	6.24	1.46	0.62
Greedy affinity hierarchical	mean	**5.58**	**1.63**	0.57
	sum	6.82	1.34	**0.63**
	max	6.53	1.40	0.62

(b) INRIA Holidays

Table 6.3: A comparison of search in flat and hierarchical structure on *ukbench* and *INRIA Holidays*. 1st col: grouping strategies. 2nd col: pooling operations. 3rd col: average query time. *ukbench* takes average number of top-4 retrieved images as score. *INRIA Holidays* takes mAP as evaluation metric.

The baseline remains to search using an inverted index system. We use a three-layer hierarchy with the original histograms at the bottom layer. At the second layer, histograms belonging to the same object/scene are pooled (pooling based on prior information available about the data and problem space). At the top layer, we compare two different strategies to build the hierarchy: Random grouping and greedy affinity grouping. Random grouping: We randomly group every N histograms from the second layer. Greedy affinity grouping: We greedily group every N histograms based on their nearest neighbors in affinity (which is the histogram intersection score). In each setup, we also compare the different choices of pooling operators. Tab. 6.3a and Tab. 6.3b show results on the *ukbench* and *INRIA Holidays* datasets with $N = 16$.

In these image retrieval tasks, we *completely discard the threshold and only search down those nodes which have top 10 highest scores*. Thus even for sum/max-pooling, the precision-recall behavior should be different from the baseline. All hierarchical approaches, regardless

of pooling operation and grouping scheme, are faster than the baseline. The observation that speedup is available even for the random grouping scheme shows that the speedup does not just hinge on grouping similar images, though grouping similar images can boost the speedup further as we have shown in previous experiments on the driving data. We also notice improved score/mAP in these two experiments, likely due to the grouping of histograms of the same object/scene at the second layer of our hierarchy and the top-4 scoring mechanism imposed by the benchmark.

6.4 Discussion

We have presented a hierarchical data structure consisting of pooled local descriptors representing the likelihood of locations given the images they generate, while maintaining an inverted index at each level of the data structure. While mean-pooling of histograms may seem counter-productive, it is a sensible choice when considered an anti-aliasing procedure in the context of classical sampling theory, where the data structure, as well as keyframes, are tasked with *down-sampling* the native rate. We have compared several pooling strategies, and found that mean-pooling provides the most speedup at a small cost to performance; sum-pooling has the upper-bound property and accelerates search to a reasonable degree without loss of performance; and max-pooling shares the same property with sum-pooling but exhibits a larger speedup due better approximating the nodes below it.

For simplicity, we chose a fixed topology (depth and branching factor) and studied the resulting performance empirically. We have found that sophisticated heuristics do not improve performance enough to justify the added complexity. We have benchmarked our scheme on public datasets, where we have shown that even a shallow tree can significantly cut down on test time with minimal impact to precision, which is the main goal of loop closure.

CHAPTER 7

Discussion

In Chapters 2 and 3, we give two examples of using visual and inertial data to improve deep learning models. Both are preliminary attempts towards learning-based multi-sensor fusion and have their pros and cons in contrast to conventional approaches. On the bright side, deep neural networks have more representation power, which, compared to conventional approaches, leads to less information loss in processing the raw sensory data. This is due to the networks' capability to discover the prior knowledge hidden in large datasets, and encode such knowledge in the weights. One may argue that such advantage comes at the cost of more computation. However, the trade-off between modeling capability and computational complexity always exists, and in particular, in conventional methods. For instance, rather than using point features and filters, one can perform optimization-based SLAM by directly minimizing pixel-wise photometric discrepency [ESC14, EKC17] which results in a more detailed reconstruction (a semi-dense model instead of a sparse one from filtering-based approaches). These optimization-based SLAM fall in the category of non-learning based approaches, but have computational costs higher than, or at least on par with, learning-based approaches since learning-based approaches only perform optimization during training, not inference. Another drawback of using learning-based approaches is that it's relatively hard to fully characterize the uncertainty of the estimate, often, only a point estimate is available. But again, this is due to the curse of dimensionality not a flaw of deep neural networks as inference machinery: A fair amount of samples might be sufficient to describe the probability distribution of a scalar random variable, but a faithful probabilistic characterization of a network of millions of parameters is computationally infeasible. Computationally tractable uncertainty characterization [KG17] of deep neural networks is an exciting future topic.

Besides the general merits and limitations of using deep learning to perform sensor fusion, we note more specific points regarding the "deep learning versus SLAM" relationship. In our current development, SLAM is only loosely involved: In Chapter 2, only gravity inferred by VIO is used to improve depth prediction; Chapter 3 goes one step further, where both sparse depth and relative motion produced by a visual-inertial SLAM are fused with raw imagery data by a deep neural network. Both methods only use the output of an off-the-shelf SLAM system to facilitate deep learning. A question arisen naturally is whether deep learning and SLAM can be tightly coupled to benefit each other, and, if so, how? There are end-to-end learning approaches to address localization and reconstruction [ZBS17, KGC15]. All these methods learn a mapping from the raw sensory data to camera poses and completely discard the solution structure of the SFM/SLAM problem. However, the SFM/SLAM problem has been studied for decades, and mathematically elegant results exist, which should be cherished. A promising future direction is to deeply encode domain-specific knowledge in deep learning models to reduce sample complexity and improve system performance further. Some works along this line include [RK18, LDR19, BKN17, CBC18, HJF18], where networks are embedded in existing SFM/SLAM pipelines as replacements of more conventional modules, such as Kanade-Lucas-Tomasi tracker, RANSAC-based outlier rejection, and Gauss-Newton solver, etc., and are trained end-to-end to improve the end performance of the underlying tasks.

In Chapters 4 and 5, by trading off the modeling power and generality of the approach, we present two different implementations of a semantic mapping system which can infer both geometric and semantic attributes of 3D objects. In Chapter 4, a simplified object representation – scaled and oriented 3D bounding boxes – is employed, whereas detailed CAD models are used in Chapter 5. While the 3D bounding boxes lack the details, they are computationally cheaper and more generic – any shape can be modeled as a 3D bounding box of different size as a first-order approximation. On the other hand, the CAD models capture the finest details of previously seen objects, yet they are instance-specific and cannot be deployed unless an exact, or at least a more or less similar, shape model is available. A very promising future direction is to seek more flexible shape representations in the middle of the

spectrum: Ideally, the shape model should be generic enough to accommodate novel shapes and, in the same time, discriminative enough to capture the subtleties in possible shape variation. Ideas from the variational auto-encoders [KW13], differentiable rendering [LB14], and differentiable shape representation [LDG18] can be drawn together to produce compact generative differentiable shape models. Such models should be ideal to fulfill our requirements: Being compact renders the model computationally affordable; being generative to enable prediction which can then be evaluated in a hypothesis-testing framework; being differentiable makes it possible to adapt the representation to the data during inference efficiently. One good example of using such a shape representation is [BCC18] where short latent codes are mapped to depth maps via a differentiable decoder and optimized by minimizing the photometric discrepancy between the original image and a warped version of it – enabled by the the generated depth. Compared to more traditional dense reconstruction methods where depth maps rather than latent codes are the optimization variables, their method is more efficient and gracefully encodes priors learned from data into the latent codes leading to its robustness to textureless regions and challenging illumination.

We believe, to better fuse information from multiple sensor modalities, one should also fuse the representation power of deep neural networks with the long-standing wisdom of visual geometry as demonstrated by this thesis and various related works mentioned above.

Despite all the exciting open problems at the intersection of deep learning and conventional visual geometry, some fundamental problems in SFM/SLAM are not perfectly addressed. Loop closure is one of them. In Chapter 6, we present an efficient way to find loop closures – an indispensable component of large-scale mapping systems – with the intent to reduce localization drift and hence improve mapping accuracy. However, to actually reduce drift and improve accuracy, one needs nonlinear optimization with the detected loops as constraints – known as pose graph optimization [KGS11] or bundle adjustment [TMH99][1] in

[1]In pose graph optimization, relative motion constraints are used as measurements to obtain absolute poses, whereas in bundle adjustment, the camera poses and 3D point cloud are jointly optimized by minimizing the reprojection error of the 3D points. However, bundle adjustment is a very general framework which also accommodates relative motion constraints and hence pose graph optimization is really a specialized version of bundle adjustment but with no feature measurements. The proposed loop closure detection algorithm produces pose-to-pose constraints, which can be used in both algorithms.

the SFM/SLAM literature. Similar to loop closure detection, the real challenge for nonlinear optimization also resides in scalability [ASS10]. Good open-source large-scale optimization packages include Ceres [AM12], g2o [KGS11] and GTSAM [Del12], etc. While Ceres is a slightly more general nonlinear optimization framework, both g2o and GTSAM are specialized to SFM/SLAM.

Nevertheless, Ceres is very popular among the SFM/SLAM community where various commercial products employ it (e.g., Google Street View, PhotoTours, and Tango, etc.) and quality open-source SFM/SLAM packages (e.g., COLMAP [SF16], OpenMVG [MMM], and Theia [Swe], etc.) use it under the hood. One interesting research direction is to develop more efficient and robust large-scale nonlinear optimization algorithms to exploit the special structure of the SFM/SLAM problem. Typically, the SFM/SLAM objective is iteratively minimized by solving a series of large but sparse linear systems [Dav06], which emphasizes the importance of developing solvers exploiting the sparse structure of the SFM/SLAM problem.

REFERENCES

[ABC16] Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, et al. "TensorFlow: A System for Large-Scale Machine Learning." In *OSDI*, volume 16, pp. 265–283, 2016.

[ABS16] Umar Asif, Mohammed Bennamoun, and Ferdous Sohel. "Simultaneous dense scene reconstruction and object labeling." In *IEEE International Conference on Robotics and Automation*, 2016.

[Aiz03] Akiko Aizawa. "An information-theoretic perspective of tf–idf measures." *Information Processing & Management*, **39**(1):45–65, 2003.

[AK17] Brandon Amos and J Zico Kolter. "OptNet: Differentiable Optimization as a Layer in Neural Networks." In *International Conference on Machine Learning*, pp. 136–145, 2017.

[AM12] Sameer Agarwal, Keir Mierle, et al. "Ceres solver." 2012.

[ASS10] Sameer Agarwal, Noah Snavely, Steven M Seitz, and Richard Szeliski. "Bundle adjustment in the large." In *European conference on computer vision*, pp. 29–42. Springer, 2010.

[AYB15] S. Aditya, Y. Yang, C. Baral, C. Fermuller, and Y. Aloimonos. "Visual commonsense for scene understanding using perception, semantic parsing and reasoning." In *2015 AAAI Spring Symposium Series*, 2015.

[AZD14] Nikolay Atanasov, Menglong Zhu, Kostas Daniilidis, and George Pappas. "Semantic localization via the matrix permanent." In *Robotics: Science and Systems*, 2014.

[BAD17] Sean L Bowman, Nikolay Atanasov, Kostas Daniilidis, and George J Pappas. "Probabilistic data association for semantic slam." In *Robotics and Automation (ICRA), 2017 IEEE International Conference on*, pp. 1722–1729. IEEE, 2017.

[BCC18] Michael Bloesch, Jan Czarnowski, Ronald Clark, Stefan Leutenegger, and Andrew J Davison. "CodeSLAMlearning a compact, optimisable representation for dense visual SLAM." In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2560–2568, 2018.

[BDD96] C Bradford Barber, David P Dobkin, David P Dobkin, and Hannu Huhdanpaa. "The quickhull algorithm for convex hulls." *ACM Transactions on Mathematical Software (TOMS)*, **22**(4):469–483, 1996.

[BGC15] L. Baraldi, C. Grana, and R. Cucchiara. "Scene segmentation using temporal clustering for accessing and re-using broadcast video." In *Multimedia and Expo (ICME), 2015 IEEE International Conference on*, pp. 1–6. IEEE, 2015.

[BI97] Andrew Blake and Michael Isard. "The condensation algorithm-conditional density propagation and applications to visual tracking." In *Advances in Neural Information Processing Systems*, pp. 361–367, 1997.

[BKC17] Vijay Badrinarayanan, Alex Kendall, and Roberto Cipolla. "Segnet: A deep convolutional encoder-decoder architecture for image segmentation." *IEEE transactions on pattern analysis and machine intelligence*, **39**(12):2481–2495, 2017.

[BKN17] Eric Brachmann, Alexander Krull, Sebastian Nowozin, Jamie Shotton, Frank Michel, Stefan Gumhold, and Carsten Rother. "DSAC-differentiable RANSAC for camera localization." In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 6684–6692, 2017.

[BNG16] Michael Burri, Janosch Nikolic, Pascal Gohl, Thomas Schneider, Joern Rehder, Sammy Omari, Markus W Achtelik, and Roland Siegwart. "The EuRoC micro aerial vehicle datasets." *The International Journal of Robotics Research*, **35**(10):1157–1163, 2016.

[BNJ03] David M Blei, Andrew Y Ng, and Michael I Jordan. "Latent dirichlet allocation." *the Journal of machine Learning research*, **3**:993–1022, 2003.

[BS15] D. Banica and C. Sminchisescu. "Second-order constrained parametric proposals and sequential search-based structured prediction for semantic segmentation in RGB-D images." In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3517–3526, 2015.

[BSF08] G. Brostow, J. Shotton, J. Fauqueur, and R. Cipolla. "Segmentation and recognition using structure from motion point clouds." In *Computer Vision–ECCV 2008*, pp. 44–57. Springer, 2008.

[BVR15] Maroš Bláha, Christoph Vogel, Audrey Richard, Jan D Wegner, Thomas Pock, and Konrad Schindler. "Large-Scale Semantic 3D Reconstruction: an Adaptive Multi-Resolution Model for Multi-Class Volumetric Labeling." In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.

[Can87] John Canny. "A computational approach to edge detection." In *Readings in Computer Vision*, pp. 184–203. Elsevier, 1987.

[CBC18] Ronald Clark, Michael Bloesch, Jan Czarnowski, Stefan Leutenegger, and Andrew J Davison. "LS-Net: Learning to Solve Nonlinear Least Squares for Monocular Stereo." *arXiv preprint arXiv:1809.02966*, 2018.

[CC12] Changhyun Choi and Henrik I Christensen. "3d textureless object detection and tracking: An edge-based approach." In *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*, pp. 3877–3884. IEEE, 2012.

[CDM08] Javier Civera, Andrew J Davison, and JM Martinez Montiel. "Inverse depth parametrization for monocular SLAM." *IEEE transactions on robotics*, **24**(5):932–945, 2008.

[CFN14] C. Couprie, C. Farabet, L. Najman, and Y. Lecun. "Convolutional nets and watershed cuts for real-time semantic labeling of rgbd videos." *The Journal of Machine Learning Research*, **15**(1):3489–3511, 2014.

[CGR11] Javier Civera, Dorian Gálvez-López, Luis Riazuelo, Juan D Tardós, and JMM Montiel. "Towards semantic SLAM using a monocular camera." In *Intelligent Robots and Systems (IROS), 2011 IEEE/RSJ International Conference on*, pp. 1277–1284. IEEE, 2011.

[CK13] C. Cadena and J. Košecka. "Semantic parsing for priming object detection in RGB-D scenes." In *3rd Workshop on Semantic Perception, Mapping and Exploration*, 2013.

[CKM10] Robert O Castle, Georg Klein, and David W Murray. "Combining monoSLAM with object recognition for scene augmentation using a wearable camera." *Image and Vision Computing*, **28**(11):1548–1556, 2010.

[CKZ15] Xiaozhi Chen, Kaustav Kundu, Yukun Zhu, Andrew G Berneshawi, Huimin Ma, Sanja Fidler, and Raquel Urtasun. "3d object proposals for accurate object class detection." In *Advances in Neural Information Processing Systems*, pp. 424–432, 2015.

[CKZ16] Xiaozhi Chen, Kaustav Kundu, Ziyu Zhang, Huimin Ma, Sanja Fidler, and Raquel Urtasun. "Monocular 3d object detection for autonomous driving." In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2147–2156, 2016.

[CLC08] N. Cornelis, B. Leibe, K. Cornelis, and L. Van Gool. "3D urban scene modeling integrating recognition and reconstruction." *International Journal of Computer Vision*, **78**(2-3):121–141, 2008.

[CLS10] Michael Calonder, Vincent Lepetit, Christoph Strecha, and Pascal Fua. "Brief: Binary robust independent elementary features." In *Computer Vision–ECCV 2010*, pp. 778–792. Springer, 2010.

[CLT10] M. Choi, J. Lim, A. Torralba, and A. Willsky. "Exploiting hierarchical context on a large database of object categories." In *Computer vision and pattern recognition (CVPR), 2010 IEEE conference on*, pp. 129–136. IEEE, 2010.

[CN09] Mark Cummins and Paul Newman. "Highly scalable appearance-only SLAM-FAB-MAP 2.0." In *Robotics: Science and Systems*, volume 5. Seattle, USA, 2009.

[COR16] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele. "The cityscapes dataset for semantic urban scene understanding." *arXiv preprint arXiv:1604.01685*, 2016.

[CPS07] Ondrej Chum, James Philbin, Josef Sivic, Michael Isard, and Andrew Zisserman. "Total recall: Automatic query expansion with a generative feature model for

object retrieval." In *Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on*, pp. 1–8. IEEE, 2007.

[CRU16] Falak Chhaya, Dinesh Reddy, Sarthak Upadhyay, Visesh Chari, M Zeeshan Zia, and K Madhava Krishna. "Monocular Reconstruction of Vehicles: Combining SLAM with Shape Priors." In *IEEE International Conference on Robotics and Automation (ICRA)*, 2016.

[CSS02] Dmitry Chetverikov, Dmitry Svirko, Dmitry Stepanov, and Pavel Krsek. "The trimmed iterative closest point algorithm." In *Pattern Recognition, 2002. Proceedings. 16th International Conference on*, volume 3, pp. 545–548. IEEE, 2002.

[CWL18] Nathaniel Chodosh, Chaoyang Wang, and Simon Lucey. "Deep Convolutional Compressed Sensing for LiDAR Depth Completion." In *Asian Conference on Computer Vision (ACCV)*, 2018.

[Dav06] Timothy A Davis. *Direct methods for sparse linear systems*, volume 2. Siam, 2006.

[DC02] Tom Drummond and Roberto Cipolla. "Real-time visual tracking of complex structures." *IEEE Transactions on pattern analysis and machine intelligence*, **24**(7):932–946, 2002.

[Del12] Frank Dellaert. "Factor graphs and GTSAM: A hands-on introduction." Technical report, Georgia Institute of Technology, 2012.

[DFS17] Jingming Dong, Xiaohan Fei, and Stefano Soatto. "Visual-Inertial-Semantic Scene Representation for 3D Object Detection." In *IEEE Conference on Computer Vision and Pattern Recognition*, 2017.

[DS15] J. Dong and S. Soatto. "Domain size pooling in local descriptors: DSP-SIFT." In *Proc. IEEE Conf. on Comp. Vision and Pattern Recogn.*, 2015.

[DTL15] Z. Deng, S. Todorovic, and L. Latecki. "Semantic Segmentation of RGBD Images with Mutex Constraints." In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 1733–1741, 2015.

[DVP18] Martin Dimitrievski, Peter Veelaert, and Wilfried Philips. "Learning morphological operators for depth completion." In *Advanced Concepts for Intelligent Vision Systems*, 2018.

[EFK18] Abdelrahman Eldesokey, Michael Felsberg, and Fahad Shahbaz Khan. "Propagating confidences through cnns for sparse data regression." In *Proceedings of British Machine Vision Conference (BMVC)*, 2018.

[EKC17] Jakob Engel, Vladlen Koltun, and Daniel Cremers. "Direct sparse odometry." *IEEE transactions on pattern analysis and machine intelligence*, **40**(3):611–625, 2017.

[EPF14] David Eigen, Christian Puhrsch, and Rob Fergus. "Depth map prediction from a single image using a multi-scale deep network." In *NIPS*, pp. 2366–2374, 2014.

[ESC14] Jakob Engel, Thomas Schöps, and Daniel Cremers. "LSD-SLAM: Large-scale direct monocular SLAM." In *European Conference on Computer Vision*, pp. 834–849. Springer, 2014.

[EVW10] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. "The Pascal Visual Object Classes (VOC) Challenge." *International Journal of Computer Vision*, **88**(2):303–338, June 2010.

[FB81] Martin A Fischler and Robert C Bolles. "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography." *Communications of the ACM*, **24**(6):381–395, 1981.

[FDU12] Sanja Fidler, Sven Dickinson, and Raquel Urtasun. "3d object detection and viewpoint estimation with a deformable 3d cuboid model." In *Advances in neural information processing systems*, pp. 611–619, 2012.

[FHG15] D. F Fouhey, W. Hussain, A. Gupta, and M. Hebert. "Single Image 3D Without a Single 3D Image." In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 1053–1061, 2015.

[FS18] X. Fei and S. Soatto. "Visual-Inertial Object Detection and Mapping." In *Proceedings of the European Conference on Computer Vision*, 2018.

[FWS19] Xiaohan Fei, Alex Wong, and Stefano Soatto. "Geo-supervised visual depth prediction." *IEEE Robotics and Automation Letters*, **4**(2):1661–1668, 2019.

[GAG15] S. Gupta, P. Arbeláez, R. Girshick, and J. Malik. "Indoor scene understanding with RGB-D images: Bottom-up segmentation, object detection and semantic segmentation." *International Journal of Computer Vision*, **112**(2):133–149, 2015.

[GAM13] S. Gupta, P. Arbelaez, and J. Malik. "Perceptual organization and recognition of indoor scenes from RGB-D images." In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 564–571, 2013.

[GBC16] Ravi Garg, Vijay Kumar BG, Gustavo Carneiro, and Ian Reid. "Unsupervised cnn for single view depth estimation: Geometry to the rescue." In *ECCV*. Springer, 2016.

[GBS15] G. Graber, J. Balzer, S. Soatto, and T. Pock. "Efficient minimal surface regularization of perspective depth maps in variational stereo." In *Proc. IEEE Conf. on Comp. Vision and Pattern Recogn.*, 2015.

[GD05] Kristen Grauman and Trevor Darrell. "The pyramid match kernel: Discriminative classification with sets of image features." In *Computer Vision, 2005. ICCV 2005. Tenth IEEE International Conference on*, volume 2, pp. 1458–1465. IEEE, 2005.

[GDD14] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. "Rich feature hierarchies for accurate object detection and semantic segmentation." In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 580–587, 2014.

[GID14] Ross Girshick, Forrest Iandola, Trevor Darrell, and Jitendra Malik. "Deformable part models are convolutional neural networks." *arXiv preprint arXiv:1409.5403*, 2014.

[Gir15] Ross Girshick. "Fast r-cnn." In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 1440–1448, 2015.

[GJ96] Donald Geman and Bruno Jedynak. "An active testing model for tracking roads in satellite images." *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, **18**(1):1–14, 1996.

[GLS13] Andreas Geiger, Philip Lenz, Christoph Stiller, and Raquel Urtasun. "Vision meets robotics: The kitti dataset." *The International Journal of Robotics Research*, p. 0278364913491297, 2013.

[GMB17] Clément Godard, Oisin Mac Aodha, and Gabriel J Brostow. "Unsupervised monocular depth estimation with left-right consistency." In *CVPR*, 2017.

[GRG18] Ross Girshick, Ilija Radosavovic, Georgia Gkioxari, Piotr Dollár, and Kaiming He. "Detectron." https://github.com/facebookresearch/detectron, 2018.

[GSS93] Neil J Gordon, David J Salmond, and Adrian FM Smith. "Novel approach to nonlinear/non-Gaussian Bayesian state estimation." In *IEE Proceedings F (Radar and Signal Processing)*, volume 140, pp. 107–113. IET, 1993.

[GT11] Dorian Galvez-Lopez and Juan D Tardos. "Real-time loop detection with bags of binary words." In *Intelligent Robots and Systems (IROS), 2011 IEEE/RSJ International Conference on*, pp. 51–58. IEEE, 2011.

[HFL14] Alexander Hermans, Georgios Floros, and Bastian Leibe. "Dense 3d semantic mapping of indoor scenes from rgb-d images." In *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 2631–2638. IEEE, 2014.

[HFY18] Zixuan Huang, Junming Fan, Shuai Yi, Xiaogang Wang, and Hongsheng Li. "Hms-net: Hierarchical multi-scale sparsity-invariant network for sparse depth completion." *arXiv preprint arXiv:1808.08685*, 2018.

[HGD17] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. "Mask R-CNN." In *Proceedings of the International Conference on Computer Vision (ICCV)*, 2017.

[HHX15] D. Hoiem, J. Hays, J. Xiao, and A. Khosla. "Guest editorial: Scene understanding." *International Journal of Computer Vision*, **112**(2):131–132, 2015.

[HJF18] Lei Han, Mengqi Ji, Lu Fang, and Matthias Nießner. "RegNet: Learning the Optimization of Direct Image-to-Image Pose Registration." *arXiv preprint arXiv:1812.10212*, 2018.

[HKB13] J. Hesch, D. Kottas, S. Bowman, and S. Roumeliotis. "Towards consistent vision-aided inertial navigation." In *Algorithmic Foundations of Robotics X*, pp. 559–574. Springer, 2013.

[HPN16] Binh-Son Hua, Quang-Hieu Pham, Duc Thanh Nguyen, Minh-Khoi Tran, Lap-Fai Yu, and Sai-Kit Yeung. "Scenenn: A scene meshes dataset with annotations." In *3D Vision (3DV), 2016 Fourth International Conference on*, pp. 92–101. IEEE, 2016.

[HWM14] Ankur Handa, Thomas Whelan, John McDonald, and Andrew J Davison. "A benchmark for RGB-D visual odometry, 3D reconstruction and SLAM." In *Robotics and automation (ICRA), 2014 IEEE international conference on*, pp. 1524–1531. IEEE, 2014.

[HZC13] C. Hane, C. Zach, A. Cohen, R. Angst, and M. Pollefeys. "Joint 3D scene reconstruction and class segmentation." In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 97–104, 2013.

[HZR16] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. "Deep residual learning for image recognition." In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 770–778, 2016.

[ISS16] Hamid Izadinia, Qi Shan, and Steven M Seitz. "IM2CAD." *arXiv preprint arXiv:1608.05137*, 2016.

[Jaz70] A.H. Jazwinski. *Stochastic Processes and Filtering Theory*. Academic Press, 1970.

[JCW18] Maximilian Jaritz, Raoul de Charette, Emilie Wirbel, Xavier Perrotton, and Fawzi Nashashibi. "Sparse and Dense Data with CNNs: Depth Completion and Semantic Segmentation." In *International Conference on 3D Vision (3DV)*, 2018.

[JDS08] Herve Jegou, Matthijs Douze, and Cordelia Schmid. "Hamming embedding and weak geometric consistency for large scale image search." In *Computer Vision–ECCV 2008*, pp. 304–317. Springer, 2008.

[JGK17] Omid Hosseini Jafari, Oliver Groth, Alexander Kirillov, Michael Ying Yang, and Carsten Rother. "Analyzing modular cnn architectures for joint depth prediction and semantic segmentation." In *Robotics and Automation (ICRA), 2017 IEEE International Conference on*, pp. 4620–4627. IEEE, 2017.

[Jon09] Eagle Sunrise Jones. *Large scale visual navigation and community map building*. PhD thesis, Citeseer, 2009.

[JS11] E. Jones and S. Soatto. "Visual-Inertial Navigation, Localization and Mapping: A scalable real-time large-scale approach." *Intl. J. of Robotics Res.*, april 2011.

[KAJ11] H. Koppula, A. Anand, T. Joachims, and A. Saxena. "Semantic Labeling of 3D Point Clouds for Indoor Scenes." In *NIPS*, volume 1, p. 4, 2011.

[KB14] Diederik Kingma and Jimmy Ba. "Adam: A method for stochastic optimization." *arXiv preprint arXiv:1412.6980*, 2014.

[KG17] Alex Kendall and Yarin Gal. "What uncertainties do we need in bayesian deep learning for computer vision?" In *Advances in neural information processing systems*, pp. 5574–5584, 2017.

[KGC15] Alex Kendall, Matthew Grimes, and Roberto Cipolla. "Posenet: A convolutional network for real-time 6-dof camera relocalization." In *Proceedings of the IEEE international conference on computer vision*, pp. 2938–2946, 2015.

[KGS11] Rainer Kummerle, Giorgio Grisetti, Hauke Strasdat, Kurt Konolige, and Wolfram Burgard. "g 2 o: A general framework for graph optimization." In *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pp. 3607–3613. IEEE, 2011.

[KK11] Philipp Krähenbühl and Vladlen Koltun. "Efficient inference in fully connected crfs with gaussian edge potentials." In *Advances in neural information processing systems*, pp. 109–117, 2011.

[KKS13] B. Kim, P. Kohli, and S. Savarese. "3D scene understanding by Voxel-CRF." In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 1425–1432, 2013.

[KLD14] Abhijit Kundu, Yin Li, Frank Dellaert, Fuxin Li, and James M Rehg. "Joint semantic segmentation and 3d reconstruction from monocular video." In *European Conference on Computer Vision*, pp. 703–718. Springer, 2014.

[KLK12] Kevin Karsch, Ce Liu, and Sing Bing Kang. "Depth extraction from video using non-parametric sampling." In *European Conference on Computer Vision*, pp. 775–788. Springer, 2012.

[KM06] Georg Klein and David W Murray. "Full-3D Edge Tracking with a Particle Filter." In *BMVC*, pp. 1119–1128, 2006.

[KM07] Georg Klein and David Murray. "Parallel tracking and mapping for small AR workspaces." In *Mixed and Augmented Reality, 2007. ISMAR 2007. 6th IEEE and ACM International Symposium on*, pp. 225–234. IEEE, 2007.

[KMF13] A. Karpathy, S. Miller, and L. Fei-Fei. "Object discovery in 3D scenes via shape analysis." In *Robotics and Automation (ICRA), 2013 IEEE International Conference on*, pp. 2088–2095. IEEE, 2013.

[KMT16] Wadim Kehl, Fausto Milletari, Federico Tombari, Slobodan Ilic, and Nassir Navab. "Deep Learning of Local RGB-D Patches for 3D Object Detection and 6D Pose Estimation." In *European Conference on Computer Vision*, pp. 205–220. Springer, 2016.

[Kol11] Vladlen Koltun. "Efficient inference in fully connected crfs with gaussian edge potentials." *Adv. Neural Inf. Process. Syst*, 2011.

[KW13] D. Kingma and M. Welling. "Auto-encoding variational bayes." *arXiv preprint arXiv:1312.6114*, 2013.

[KWI13] Janusz Konrad, Meng Wang, Prakash Ishwar, Chen Wu, and Debargha Mukherjee. "Learning-based, automatic 2D-to-3D image and video conversion." *IEEE Transactions on Image Processing*, **22**(9):3485–3496, 2013.

[LAE16] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg. "SSD: Single shot multibox detector." In *European Conference on Computer Vision*, pp. 21–37. Springer, 2016.

[LB14] Matthew M Loper and Michael J Black. "OpenDR: An approximate differentiable renderer." In *European Conference on Computer Vision*, pp. 154–169. Springer, 2014.

[LBR11] Kevin Lai, Liefeng Bo, Xiaofeng Ren, and Dieter Fox. "A large-scale hierarchical multi-view rgb-d object dataset." In *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pp. 1817–1824. IEEE, 2011.

[LBR12] K. Lai, L. Bo, X. Ren, and D. Fox. "Detection-based object labeling in 3d scenes." In *Robotics and Automation (ICRA), 2012 IEEE International Conference on*, pp. 1330–1337. IEEE, 2012.

[LDG18] Yiyi Liao, Simon Donne, and Andreas Geiger. "Deep marching cubes: Learning explicit surface representations." In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2916–2925, 2018.

[LDR19] Zhaoyang Lv, Frank Dellaert, James M Rehg, and Andreas Geiger. "Taking a Deeper Look at the Inverse Compositional Algorithm." In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4581–4590, 2019.

[LF05] Vincent Lepetit, Pascal Fua, et al. "Monocular model-based 3d tracking of rigid objects: A survey." *Foundations and Trends® in Computer Graphics and Vision*, **1**(1):1–89, 2005.

[LFU13] D. Lin, S. Fidler, and R. Urtasun. "Holistic scene understanding for 3d object detection with rgbd cameras." In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 1417–1424, 2013.

[LLK14] Hyon Lim, Jongwoo Lim, and H Jin Kim. "Real-time 6-DOF monocular visual SLAM in a large-scale environment." In *Robotics and Automation (ICRA), 2014 IEEE International Conference on*, pp. 1532–1539. IEEE, 2014.

[LM14] M. Li and A. Mourikis. "Online temporal calibration for camera–IMU systems: Theory and algorithms." *The International Journal of Robotics Research*, **33**(7):947–964, 2014.

[LMB14] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. "Microsoft coco: Common objects in context." In *European conference on computer vision*, pp. 740–755. Springer, 2014.

[LMF09] Vincent Lepetit, Francesc Moreno-Noguer, and Pascal Fua. "Epnp: An accurate o (n) solution to the pnp problem." *International journal of computer vision*, **81**(2):155, 2009.

[LRB16] Iro Laina, Christian Rupprecht, Vasileios Belagiannis, Federico Tombari, and Nassir Navab. "Deeper depth prediction with fully convolutional residual networks." In *3D Vision (3DV), 2016 Fourth International Conference on*, pp. 239–248. IEEE, 2016.

[LSD15] Jonathan Long, Evan Shelhamer, and Trevor Darrell. "Fully convolutional networks for semantic segmentation." In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3431–3440, 2015.

[LSH16] G. Lin, C. Shen, A. Hengel, and I. Reid. "Exploring context with deep structured models for semantic segmentation." *arXiv preprint arXiv:1603.03183*, 2016.

[LSL16] Fayao Liu, Chunhua Shen, Guosheng Lin, and Ian Reid. "Learning depth from single monocular images using deep convolutional neural fields." *IEEE transactions on pattern analysis and machine intelligence*, **38**(10):2024–2039, 2016.

[LSP06] Svetlana Lazebnik, Cordelia Schmid, and Jean Ponce. "Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories." In *Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on*, volume 2, pp. 2169–2178. IEEE, 2006.

[LYC18] Chen Liu, Jimei Yang, Duygu Ceylan, Ersin Yumer, and Yasutaka Furukawa. "PlaneNet: Piece-wise Planar Reconstruction from a Single RGB Image." In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2579–2588, 2018.

[LZD16] Spyridon Leonardos, Xiaowei Zhou, and Kostas Daniilidis. "Distributed Consistent Data Association." *arXiv preprint arXiv:1609.07015*, 2016.

[MCK19] Fangchang Ma, Guilherme Venturelli Cavalheiro, and Sertac Karaman. "Self-supervised Sparse-to-Dense: Self-supervised Depth Completion from LiDAR and Monocular Camera." In *International Conference on Robotics and Automation (ICRA)*. IEEE, 2019.

[MFT01] D. Martin, C. Fowlkes, D. Tal, and J. Malik. "A Database of Human Segmented Natural Images and its Application to Evaluating Segmentation Algorithms and Measuring Ecological Statistics." In *Proc. 8th Int'l Conf. Computer Vision*, volume 2, pp. 416–423, July 2001.

[MHD17] John McCormac, Ankur Handa, Andrew Davison, and Stefan Leutenegger. "SemanticFusion: Dense 3D semantic mapping with convolutional neural networks." In *Robotics and Automation (ICRA), 2017 IEEE International Conference on*, pp. 4628–4635. IEEE, 2017.

[MLC14] X. Mottaghi, R.and Chen, X. Liu, N. Cho, S. Lee, S. Fidler, R. Urtasun, and A. Yuille. "The role of context for object detection and semantic segmentation in the wild." In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 891–898, 2014.

[MMM] Pierre Moulon, Pascal Monasse, Renaud Marlet, and Others. "OpenMVG. An Open Multiple View Geometry library." https://github.com/openMVG/openMVG.

[MMT15] Raul Mur-Artal, JMM Montiel, and Juan D Tardos. "ORB-SLAM: a Versatile and Accurate Monocular SLAM System." *Robotics, IEEE Transactions on*, **31**(5):1147–1163, October 2015.

[MR07] Anastasios I Mourikis and Stergios I Roumeliotis. "A multi-state constraint Kalman filter for vision-aided inertial navigation." In *Robotics and automation, 2007 IEEE international conference on*, pp. 3565–3572. IEEE, 2007.

[MSK12] Yi Ma, Stefano Soatto, Jana Kosecka, and Shankar Sastry. *An invitation to 3-d vision: from images to geometric models*, volume 26. Springer Science & Business Media, 2012.

[MV15] Aravindh Mahendran and Andrea Vedaldi. "Understanding deep image representations by inverting them." In *Computer Vision and Pattern Recognition (CVPR), 2015 IEEE Conference on*, pp. 5188–5196. IEEE, 2015.

[MWA18] Reza Mahjourian, Martin Wicke, and Anelia Angelova. "Unsupervised Learning of Depth and Ego-Motion from Monocular Video Using 3D Geometric Constraints." In *CVPR*, 2018.

[ND10] Richard A Newcombe and Andrew J Davison. "Live dense reconstruction with a single moving camera." In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pp. 1498–1505. IEEE, 2010.

[NS06] David Nister and Henrik Stewenius. "Scalable recognition with a vocabulary tree." In *Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on*, volume 2, pp. 2161–2168. IEEE, 2006.

[OBG16] Ali Osman Ulusoy, Michael J Black, and Andreas Geiger. "Patches, planes and probabilities: A non-local prior for volumetric 3d reconstruction." In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3280–3289, 2016.

[PL15] S. Pillai and J. Leonard. "Monocular SLAM Supported Object Recognition." In *Proceedings of Robotics: Science and Systems (RSS)*, Rome, Italy, July 2015.

[PR12] Victor A Prisacariu and Ian D Reid. "PWP3D: Real-time segmentation and tracking of 3D objects." *International journal of computer vision*, **98**(3):335–354, 2012.

[PSD17] Bernd Pfrommer, Nitin Sanket, Kostas Daniilidis, and Jonas Cleveland. "PennCOSYVIO: A challenging Visual Inertial Odometry benchmark." In *2017 IEEE International Conference on Robotics and Automation, ICRA 2017, Singapore, Singapore, May 29 - June 3, 2017*, pp. 3847–3854, 2017.

[QLL18] Xiaojun Qi, Renjie Liao, Zhengzhe Liu, Raquel Urtasun, and Jiaya Jia. "GeoNet: Geometric Neural Network for Joint Depth and Surface Normal Estimation." In *CVPR*, 2018.

[RD06] Edward Rosten and Tom Drummond. "Machine learning for high-speed corner detection." In *Computer Vision–ECCV 2006*, pp. 430–443. Springer, 2006.

[RDG16] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. "You only look once: Unified, real-time object detection." In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 779–788, 2016.

[RDS15] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. "ImageNet Large Scale Visual Recognition Challenge." *International Journal of Computer Vision (IJCV)*, 2015.

[rev16] Anonymized for review. "Demo." In *Anonymized for review*, 2016.

[RHG15] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. "Faster R-CNN: Towards real-time object detection with region proposal networks." In *Advances in neural information processing systems*, pp. 91–99, 2015.

[RK18] René Ranftl and Vladlen Koltun. "Deep fundamental matrix estimation." In *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 284–299, 2018.

[RKT09] Chris Russell, Pushmeet Kohli, Philip HS Torr, et al. "Associative hierarchical crfs for object class image segmentation." In *Computer Vision, 2009 IEEE 12th International Conference on*, pp. 739–746. IEEE, 2009.

[RRK11] Ethan Rublee, Vincent Rabaud, Kurt Konolige, and Gary Bradski. "ORB: an efficient alternative to SIFT or SURF." In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pp. 2564–2571. IEEE, 2011.

[RS15] Zhile Ren and Erik B Sudderth. "Three-dimensional object detection and layout prediction using clouds of oriented gradients." In *IEEE Conference on Computer Vision and Pattern Recognition*, 2015.

[Sam90] Hanan Samet. *The design and analysis of spatial data structures*, volume 85. Addison-Wesley Reading, MA, 1990.

[SB91] Michael J Swain and Dana H Ballard. "Color indexing." *International journal of computer vision*, **7**(1):11–32, 1991.

[SC16] S. Soatto and A. Chiuso. "Visual Representations: Defining properties and deep approximations." *Proc. of the Intl. Conf. on Learning Representations (ICLR); ArXiv: 1411.7676*, May 2016.

[SCN06] Ashutosh Saxena, Sung H Chung, and Andrew Y Ng. "Learning depth from single monocular images." In *Advances in neural information processing systems*, pp. 1161–1168, 2006.

[SEE12] Jürgen Sturm, Nikolas Engelhard, Felix Endres, Wolfram Burgard, and Daniel Cremers. "A benchmark for the evaluation of RGB-D SLAM systems." In *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*, pp. 573–580. IEEE, 2012.

[SF16] Johannes Lutz Schönberger and Jan-Michael Frahm. "Structure-from-Motion Revisited." In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[SGD18] David Schubert, Thore Goll, Nikolaus Demmel, Vladyslav Usenko, Jörg Stückler, and Daniel Cremers. "The TUM VI Benchmark for Evaluating Visual-Inertial Odometry." In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 1680–1687. IEEE, 2018.

[SGS13] S. Sengupta, E. Greveson, A. Shahrokni, and P. Torr. "Semantic Modelling of Urban Scenes." In *International Conference on Robotics and Automation*, 2013.

[SHK12] Nathan Silberman, Derek Hoiem, Pushmeet Kohli, and Rob Fergus. "Indoor segmentation and support inference from rgbd images." In *European Conference on Computer Vision*, pp. 746–760. Springer, 2012.

[SHL16] Nikolay Savinov, Christian Haene, Lubor Ladicky, and Marc Pollefeys. "Semantic 3D Reconstruction with Continuous Regularization and Ray Potentials Using a Visibility Consistency Constraint." *arXiv preprint arXiv:1604.02885*, 2016.

[SHP15] N. Savinov, C. Hane, M. Pollefeys, et al. "Discrete optimization of ray potentials for semantic 3D reconstruction." In *Computer Vision and Pattern Recognition (CVPR), 2015 IEEE Conference on*, pp. 5511–5518. IEEE, 2015.

[SJC08] Jamie Shotton, Matthew Johnson, and Roberto Cipolla. "Semantic texton forests for image categorization and segmentation." In *Computer vision and pattern recognition, 2008. CVPR 2008. IEEE Conference on*, pp. 1–8. IEEE, 2008.

[SK13] G. Singh and J. Kosecka. "Nonparametric scene parsing with adaptive feature relevance and semantic context." In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3151–3157, 2013.

[SLK11] Torsten Sattler, Bastian Leibe, and Leif Kobbelt. "Fast image-based localization using direct 2D-to-3D matching." In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pp. 667–674. IEEE, 2011.

[SNC19] Shreyas S Shivakumar, Ty Nguyen, Steven W. Chen, and Camillo J Taylor. "DFuseNet: Deep Fusion of RGB and Sparse Depth Information for Image Guided Dense Depth Completion." *arXiv preprint arXiv:1902.00761*, 2019.

[SNS13] R. Salas-Moreno, R. Newcombe, H. Strasdat, P. Kelly, and A. Davison. "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, 2013.

[SSN09] Ashutosh Saxena, Min Sun, and Andrew Y Ng. "Make3d: Learning 3d scene structure from a single still image." *IEEE transactions on pattern analysis and machine intelligence*, **31**(5):824–840, 2009.

[SSP16] Nick Schneider, Lukas Schneider, Peter Pinggera, Uwe Franke, Marc Pollefeys, and Christoph Stiller. "Semantically Guided Depth Upsampling." In *German Conference on Pattern Recognition*. Springer, 2016.

[STL14] A. Sharma, O. Tuzel, and M. Liu. "Recursive context propagation network for semantic scene labeling." In *Advances in Neural Information Processing Systems*, pp. 2447–2455, 2014.

[SVZ14] Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. "Deep inside convolutional networks: Visualising image classification models and saliency maps." In *Workshop at International Conference on Learning Representations*, 2014.

[Swe] Chris Sweeney. "Theia Multiview Geometry Library: Tutorial & Reference." http://theia-sfm.org.

[SX15] Shuran Song and Jianxiong Xiao. "Deep Sliding Shapes for amodal 3D object detection in RGB-D images." *arXiv preprint arXiv:1511.02300*, 2015.

[SYS16] Manolis Savva, Fisher Yu, Hao Su, M Aono, B Chen, D Cohen-Or, W Deng, Hang Su, Song Bai, Xiang Bai, et al. "Shrec16 track large-scale 3d shape retrieval from shapenet core55." In *Proceedings of the eurographics workshop on 3D object retrieval*, 2016.

[SZ03] Josef Sivic and Andrew Zisserman. "Video Google: A text retrieval approach to object matching in videos." In *Computer Vision, 2003. Proceedings. Ninth IEEE International Conference on*, pp. 1470–1477. IEEE, 2003.

[SZ05] S. Smale and D.-X. Zhou. "Shannon sampling II: Connections to learning theory." *Applied and Computational Harmonic Analysis*, **19**(3):285–302, 2005.

[SZ14] Karen Simonyan and Andrew Zisserman. "Very deep convolutional networks for large-scale image recognition." *arXiv preprint arXiv:1409.1556*, 2014.

[TCS15] Konstantine Tsotsos, Alessandro Chiuso, and Stefano Soatto. "Robust inference for visual-inertial sensor fusion." In *IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2015.

[TL09] Panu Turcot and David G Lowe. "Better matching with fewer features: The selection of useful features in large database recognition problems." In *Computer Vision Workshops (ICCV Workshops), 2009 IEEE 12th International Conference on*, pp. 2109–2116. IEEE, 2009.

[TMF03] Antonio Torralba, Kevin P Murphy, William T Freeman, and Mark A Rubin. "Context-based vision system for place and object recognition." In *Computer Vision, 2003. Proceedings. Ninth IEEE International Conference on*, pp. 273–280. IEEE, 2003.

[TMH99] Bill Triggs, Philip F McLauchlan, Richard I Hartley, and Andrew W Fitzgibbon. "Bundle adjustmenta modern synthesis." In *International workshop on vision algorithms*, pp. 298–372. Springer, 1999.

[TPB00] N. Tishby, F. C. Pereira, and W. Bialek. "The information bottleneck method." In *Proc. of the Allerton Conf.*, 2000.

[TSP11] Akihiko Torii, Josef Sivic, and Tomas Pajdla. "Visual localization by linear combination of image descriptors." In *Computer Vision Workshops (ICCV Workshops), 2011 IEEE International Conference on*, pp. 102–109. IEEE, 2011.

[TSS17] Henning Tjaden, Ulrich Schwanecke, and Elmar Schömer. "Real-Time Monocular Pose Estimation of 3D Objects using Temporally Consistent Local Color Histograms." In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 124–132, 2017.

[TTD12] A. Toshev, B. Taskar, and K. Daniilidis. "Shape-based object detection via boundary structure segmentation." *International journal of computer vision*, **99**(2):123–146, 2012.

[UN00] Iwan Ulrich and Illah Nourbakhsh. "Appearance-based place recognition for topological localization." In *Robotics and Automation, 2000. Proceedings. ICRA'00. IEEE International Conference on*, volume 2, pp. 1023–1029. Ieee, 2000.

[USS17] Jonas Uhrig, Nick Schneider, Lukas Schneider, Uwe Franke, Thomas Brox, and Andreas Geiger. "Sparsity invariant cnns." In *2017 International Conference on 3D Vision (3DV)*, pp. 11–20. IEEE, 2017.

[Vas04] Nuno Vasconcelos. "On the efficient evaluation of probabilistic similarity functions for image retrieval." *Information Theory, IEEE Transactions on*, **50**(7):1482–1496, 2004.

[VML15] Vibhav Vineet, Ondrej Miksik, Morten Lidegaard, Matthias Nießner, Stuart Golodetz, Victor A Prisacariu, Olaf Kähler, David W Murray, Shahram Izadi,

Patrick Pérez, et al. "Incremental dense semantic stereo fusion for large-scale semantic scene reconstruction." In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 75–82. IEEE, 2015.

[Wal81] D. Waltz. *Understanding and generating scene descriptions*. 1981.

[WBS04] Zhou Wang, Alan C Bovik, Hamid R Sheikh, Eero P Simoncelli, et al. "Image quality assessment: from error visibility to structural similarity." *IEEE transactions on image processing*, **13**(4):600–612, 2004.

[WBZ18] Chaoyang Wang, Jose Miguel Buenaposada, Rui Zhu, and Simon Lucey. "Learning Depth from Monocular Videos using Direct Methods." In *CVPR*, 2018.

[WCN09] B. Williams, M. Cummins, J. Neira, P. Newman, I. Reid, and J. Tardós. "A comparison of loop closing techniques in monocular SLAM." *Robotics and Autonomous Systems*, 2009.

[WFU15] Shenlong Wang, Sanja Fidler, and Raquel Urtasun. "Holistic 3d scene understanding from a single geo-tagged image." In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3964–3972. IEEE, 2015.

[WLS14] C. Wu, I. Lenz, and A. Saxena. "Hierarchical semantic labeling for task-relevant rgb-d perception." In *Robotics: Science and systems (RSS)*, 2014.

[WLS15] Thomas Whelan, Stefan Leutenegger, Renato F Salas-Moreno, Ben Glocker, and Andrew J Davison. "ElasticFusion: Dense SLAM without a pose graph." *Proc. Robotics: Science and Systems, Rome, Italy*, 2015.

[WMZ18] Chaoyang Wang, José Miguel Buenaposada, Rui Zhu, and Simon Lucey. "Learning depth from monocular videos using direct methods." In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2022–2030, 2018.

[WSR16] Peng Wang, Xiaohui Shen, Bryan Russell, Scott Cohen, Brian Price, and Alan L Yuille. "Surge: Surface regularized geometry estimation from a single image." In *Advances in Neural Information Processing Systems*, pp. 172–180, 2016.

[XCL15] Yu Xiang, Wongun Choi, Yuanqing Lin, and Silvio Savarese. "Data-driven 3d voxel patterns for object category recognition." In *IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2015.

[XCL16] Yu Xiang, Wongun Choi, Yuanqing Lin, and Silvio Savarese. "Subcategory-aware Convolutional Neural Networks for Object Proposals and Detection." *arXiv preprint arXiv:1604.04693*, 2016.

[XKC16] Yu Xiang, Wonhui Kim, Wei Chen, Jingwei Ji, Christopher Choy, Hao Su, Roozbeh Mottaghi, Leonidas Guibas, and Silvio Savarese. "ObjectNet3D: A Large Scale Database for 3D Object Recognition." In *European Conference Computer Vision (ECCV)*, 2016.

[XMS14] Yu Xiang, Roozbeh Mottaghi, and Silvio Savarese. "Beyond PASCAL: A Benchmark for 3D Object Detection in the Wild." In *IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2014.

[XOT13] Jianxiong Xiao, Andrew Owens, and Antonio Torralba. "Sun3d: A database of big spaces reconstructed using sfm and object labels." In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 1625–1632, 2013.

[YFU12] J. Yao, S. Fidler, and R. Urtasun. "Describing the scene as a whole: Joint object detection, scene classification and semantic segmentation." In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pp. 702–709. IEEE, 2012.

[YS18] Zhichao Yin and Jianping Shi. "GeoNet: Unsupervised Learning of Deep Depth, Optical Flow and Camera Pose." In *CVPR*, 2018.

[YWS19] Yanchao Yang, Alex Wong, and Stefano Soatto. "Dense Depth Posterior (DDP) from Single Image and Sparse Range." *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019.

[ZBS17] Tinghui Zhou, Matthew Brown, Noah Snavely, and David G Lowe. "Unsupervised learning of depth and ego-motion from video." In *CVPR*, 2017.

[ZCV15] R. Zhang, S. Candra, K. Vetter, and A. Zakhor. "Sensor fusion for semantic segmentation of urban scenes." In *Robotics and Automation (ICRA), 2015 IEEE International Conference on*, pp. 1850–1857. IEEE, 2015.

[ZF18] Yinda Zhang and Thomas Funkhouser. "Deep depth completion of a single rgb-d image." In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 175–185, 2018.

[ZGW18] Huangying Zhan, Ravi Garg, Chamara Saroj Weerasekera, Kejie Li, Harsh Agarwal, and Ian Reid. "Unsupervised Learning of Monocular Depth Estimation and Visual Odometry with Deep Feature Reconstruction." In *CVPR*, 2018.

[ZPK18] Qian-Yi Zhou, Jaesik Park, and Vladlen Koltun. "Open3D: A Modern Library for 3D Data Processing." *arXiv:1801.09847*, 2018.

[ZSQ17] Hengshuang Zhao, Jianping Shi, Xiaojuan Qi, Xiaogang Wang, and Jiaya Jia. "Pyramid scene parsing network." In *CVPR*, pp. 2881–2890, 2017.

[ZSS15] M Zeeshan Zia, Michael Stark, and Konrad Schindler. "Towards scene understanding with detailed 3d object representations." *International Journal of Computer Vision*, **112**(2):188–203, 2015.

[ZZD15] Menglong Zhu, Xiaowei Zhou, and Kostas Daniilidis. "Single Image Pop-Up from Discriminatively Learned Parts." In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 927–935, 2015.

[ZZP17] Bolei Zhou, Hang Zhao, Xavier Puig, Sanja Fidler, Adela Barriuso, and Antonio Torralba. "Scene parsing through ade20k dataset." In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 633–641, 2017.