Imperial College London
Department of Computing

Robust Multimodal Dense SLAM

Tristan William Laidlow

19th March 2020

Supervised by Dr. Stefan Leutenegger
Co-supervised by Prof. Andrew Davison

Submitted in part fulfilment of the requirements for the degree of PhD in Computing and the Diploma of Imperial College London. This thesis is entirely my own work, and, except where otherwise indicated, describes my own research.

Abstract

To enable increasingly intelligent behaviours, autonomous robots will need to be equipped with a deep understanding of their surrounding environment. It would be particularly desirable if this level of perception could be achieved automatically through the use of vision-based sensing, as passive cameras make a compelling sensor choice for robotic platforms due to their low cost, low weight, and low power consumption.

Fundamental to extracting a high-level understanding from a set of 2D images is an understanding of the underlying 3D geometry of the environment. In mobile robotics, the most popular and successful technique for building a representation of 3D geometry from 2D images is Visual Simultaneous Localisation and Mapping (SLAM). While sparse, landmark-based SLAM systems have demonstrated high levels of accuracy and robustness, they are only capable of producing sparse maps. In general, to move beyond simple navigation to scene understanding and interaction, dense 3D reconstructions are required.

Dense SLAM systems naturally allow for online dense scene reconstruction, but suffer from a lack of robustness due to the fact that the dense image alignment used in the tracking step has a narrow convergence basin and that the photometric-based depth estimation used in the mapping step is typically poorly constrained due to the presence of occlusions and homogeneous textures.

This thesis develops methods that can be used to increase the robustness of dense SLAM by fusing additional sensing modalities into standard dense SLAM pipelines. In particular, this thesis will look at two sensing modalities: acceleration and rotation rate measurements from an inertial measurement unit (IMU) to address the tracking issue, and learned priors on dense reconstructions from deep neural networks (DNNs) to address the mapping issue.

Acknowledgements

I am very thankful for the support given to me by many people over the course of my studies, without which this work would not have been possible.

I am particularly grateful to Dr. Stefan Leutenegger and Prof. Andrew Davison for providing me with the opportunity to pursue a PhD in the Dyson Robotics Laboratory at Imperial College. As my supervisor, Stefan was incredibly patient, understanding, trusting, and generous with his time. He guided me to interesting problems, provided much needed expertise, and kept pushing me to do the best work possible. Andy, my second supervisor and director of the lab, created an environment where I felt safe to take risks in my research and pursue impactful topics.

I am very appreciative of Dyson Technology Ltd. for not only funding my research, but providing an opportunity for many insightful discussions with Charles Collis, Barry Beard, and the rest of the robotics research team.

One of the great benefits of working in the Dyson Robotics Lab was the opportunity to collaborate with many excellent researchers including Michael Bloesch, Ronald Clark, Jan Czarnowski, Wenbin Li, and Andrea Nicastro. I am also thankful for the advice and discussions with all other members of the Dyson Robotics Lab, past and present: Patrick Bardow, Fabian Falck, Ankur Handa, Dorian Hennings, Charlie Houseago, Stephen James, Edward Johns, Zoe Landgraf, Daniel Lenton, Robert Lukierski, John McCormac, Owen Nicholson, Sajad Saeedi, Edgar Sucar, Akis Tsiotsios, Kentaro Wada, Shuaifeng Zhi, and Jacek Zienkiewicz.

Iosifina Pournara was particularly helpful to me, organising travel to conferences and ensuring that I had access to the equipment I needed, as well as providing general support, advice and encouragement.

Finally, thank you to my family and friends. I love you all.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

Contents

1.1 Robot Perception and Scene Understanding

To enable increasingly intelligent behaviours, autonomous robots will need to be equipped with a deep understanding of their surrounding environment. Self-driving cars, for example, will need to identify potential hazards and quickly determine an appropriate response based on factors such as terrain, weather, and the presence of pedestrians, cyclists, or other vehicles. Home-cleaning robots will need to know which areas of the house they have already cleaned, which areas require attention,

and how to complete their tasks without being intrusive or damaging fragile objects. Autonomous search-and-rescue vehicles will need to quickly and efficiently explore unknown and potentially dangerous environments while identifying people who need help and other areas of concern. It would be particularly desirable if this level of perception could be achieved automatically through the use of vision-based sensing, as passive cameras make a compelling sensor choice for robotic platforms due to their low cost, low weight, and low power consumption.

Extracting a high-level understanding from a set of 2D images is one of the core challenges of computer vision and encompasses a broad range of topics such as object detection, semantic segmentation, path planning and motion prediction. Fundamental to all of these, however, is an understanding of the underlying 3D geometry of the environment. For this reason, much research in mobile robotics over the past few decades has focused on building increasingly sophisticated methods to accurately and robustly navigate and reconstruct the observed world. Building a detailed representation of 3D geometry from a sequence of 2D images is the focus of this thesis.

1.2 Visual SLAM

There are a number of computer vision techniques for estimating 3D structure from a set of images. Some, such as Shape-from-Silhouettes [Fitzgibbon et al., 1998, Hernández and Schmitt, 2004, Phothong et al., 2018] and Space Carving [Kutulakos and Seitz, 2000, Tulsiani et al., 2017], are based on using multiple images of the same scene from different viewpoints. Others use multiple images from the same viewpoint but with varying lighting conditions, such as Shape-from-Shading [Zhang et al., 1999, Yang and Deng, 2018], or varying camera intrinsics, such as Depth-from-Defocus [Nayar et al., 1995, Tao et al., 2017]. More recently, a number of techniques have been developed that use machine learning [Saxena et al., 2005, Hoiem et al., 2005, Hoiem et al., 2008], and deep learning, in particular [Eigen et al., 2014, Liu et al., 2015, Laina et al., 2016, Ummenhofer et al., 2017, Fu et al., 2018]. In mobile

robotics, however, the most popular and successful techniques are those based on *Structure from Motion* (SfM). These techniques rely on few assumptions about the environment and, crucially, estimate the pose of the camera along with the 3D geometry, making these methods well suited for robot navigation.

Many state-of-the-art SfM systems [Snavely et al., 2006, Furukawa and Ponce, 2007, Goesele et al., 2007, Agarwal et al., 2009, Frahm et al., 2010, Kazhdan and Hoppe, 2013, Moulon et al., 2013, Wenzel et al., 2013, Wu, 2013, Fuhrmann et al., 2014, Schönberger and Frahm, 2016, Schönberger et al., 2016] give impressive results, but typically rely on multi-stage pipelines and global optimisation, which is problematic for robotic systems needing to estimate the geometry and poses incrementally and in real time. For this reason, *incremental SfM* has been one of the main areas of focus in autonomous mobile robotics research for the past few decades. In the robotics community, this incremental problem is usually known as *Visual Simultaneous Localisation and Mapping* (SLAM), as an autonomous agent in unknown surroundings needs to simultaneously localise its position against the current map of the environment and update the map based on the images it captures at its current position.

The SLAM problem is formulated as a probabilistic model that estimates the unknown 3D structure and camera pose parameters from a sequence of noisy measurements. According to [Durrant-Whyte and Bailey, 2006, Bailey and Durrant-Whyte, 2006], this probabilistic formulation of SLAM dates back to the 1986 IEEE Robotics and Automation Conference; however SfM techniques greatly predate this. In 1913, it was shown by [Kruppa, 1913] that given five manually labelled point correspondences in two overlapping images it was possible to determine the relative poses of the two cameras and the positions of all points up to a scale factor.

The use of cameras in robot navigation also predates visual SLAM, notably with the work of [Moravec, 1977] and the Stanford cart. The Stanford cart had a single camera that would slide along a 50cm rail to capture stereo images. By matching features across stereo pairs as the cart moved, the position of the cart and any

obstacles could be determined. Using this method of *Visual Odometry* (VO), the cart was capable of navigating a 20m course in approximately five hours.

The fundamental difference between the VO used by the Stanford cart and visual SLAM, is that in SLAM the map is incrementally refined with additional measurements. In particular, through a process called *loop closure*, the SLAM algorithm is able to detect when the camera is viewing a previously mapped area and can remove accumulated drift by adding constraints between the current camera pose estimate and the pose estimates from when it previously visited the area.

Mostly due to a lack of processing power, however, early SLAM work focused on other sensing modalities, such as sonars [Leonard and Whyte, 1991], rather than cameras. Early visual SLAM systems using passive cameras relied on stereo setups [Krotkov et al., 1995] which require careful calibration and time synchronisation. It was not until the early 2000s that the first real-time monocular SLAM system was demonstrated with MonoSLAM [Davison, 2003].

MonoSLAM used an Extended Kalman Filter (EKF) [Kalman, 1960] and extracted keypoint features [Shi and Tomasi, 1994] from images to use as landmarks. This was followed by PTAM (Parallel Tracking and Mapping) [Klein and Murray, 2007], which showed significant gains in robustness by tracking many more keypoints and using bundle adjustment [Triggs et al., 1999] to jointly optimise for the camera poses and landmark positions. A key insight of PTAM was to split the tracking and mapping threads so that the tracking could be done in real time while the bundle adjustment could be done at a slower rate in the background. The sparse, feature-based approach of these early systems became the standard in visual SLAM for many years.

In [Engel et al., 2017], the authors introduce a taxonomy for monocular visual SLAM, categorising the systems along two axes. The first axis is *direct vs. indirect*. Like [Irani and Anandan, 1999], this axis differentiates systems based on whether the measurements that are used to estimate the camera pose and structure parameters are based on "measurable image quantities" such as pixel intensities, or on

features abstracted from the image. In MonoSLAM and PTAM, the images are first preprocessed to extract and match keypoints across multiple frames, after which the *geometric error* of the landmark positions is minimised. Therefore, these systems fall into the indirect category. It is also possible for systems to use the pixel intensities directly in the SLAM formulation, for example by using the Lucas-Kanade method [Lucas and Kanade, 1981] for image alignment. In these systems, it is the *photometric error* between reprojected frames that is minimised. DTAM [Newcombe et al., 2011b] is an example of a system that falls into this direct category. These direct methods will be discussed in more detail in the next section.

The other axis in the taxonomy is *sparse vs. dense*. In sparse systems, only a small selected subset of points are tracked and reconstructed (usually those points that produce distinct feature descriptions, such as corners). For example, MonoSLAM only tracks a few hundred keypoints and PTAM only tracks a few thousand. Dense systems, conversely, attempt to use all of the pixels in the image. There is an intermediate approach, called *semi-dense*, that uses a selected subset of the image, but one that is much larger than is typical in sparse systems. LSD-SLAM [Engel et al., 2014] is an example of such a semi-dense system. Semi-dense approaches are usually grouped with dense methods because, unlike sparse methods, the subsets of pixels are usually well-connected regions and cannot be considered independent.

Note that whether a system is sparse or dense does not depend on whether it is direct or indirect. While it is possible for systems to be both sparse and direct by performing direct image alignment on only a small number of pixels [Jin et al., 2003, Engel et al., 2017], or dense and indirect by first computing the optical flow [Ranftl et al., 2016] or by using an optimisable representation [Bloesch et al., 2018], the vast majority of systems are either sparse and indirect or dense and direct.

1.3 From Sparse to Dense

As discussed earlier, the most successful early monocular SLAM systems were sparse and indirect. As processing power continued to grow and more and more keypoints

could be added to the systems, and research on low-level vision led to the development of many excellent feature descriptors (such as SIFT [Lowe, 1999], SURF [Bay et al., 2006], FAST [Rosten and Drummond, 2006], BRIEF [Calonder et al., 2010], ORB [Rublee et al., 2011], and BRISK [Leutenegger et al., 2011]), the accuracy and robustness of state-of-the-art sparse SLAM systems became very high. An example of one of these modern systems is ORB-SLAM [Mur-Artal and Tardós, 2017], which is widely used for camera tracking in many applications.

One of the drawbacks of these sparse systems, however, is that they are only capable of producing sparse maps. That is, the small set of landmarks that are reconstructed by sparse SLAM systems are very useful for camera tracking, but there are many application areas where a dense reconstruction of the environment would be preferred. For example, dense reconstructions may be necessary to determine on which parts of the terrain a robot can safely drive, and to enable free-space mapping for obstacle avoidance. Augmented reality systems require an understanding of surfaces, not only for the placement of virtual objects, but to correctly render occlusions. A dense shape reconstruction may be useful for robot manipulation tasks such as identifying good grasp positions on objects. In general, to move beyond simple navigation to scene understanding and interaction, dense 3D reconstructions are required.

While there have been some attempts to "densify" sparse maps [Lovegrove et al., 2011], dense SLAM systems naturally allow for online dense scene reconstruction. With the emergence of commodity graphics processing units (GPUs), interest in these methods increased. Early GPU-based algorithms showed that it was possible to estimate dense depth maps [Merrell et al., 2007] and 3D mesh models [Newcombe and Davison, 2010] in real time. One of the first real-time monocular dense SLAM systems was DTAM (Dense Tracking and Mapping) [Newcombe and Davison, 2010]. DTAM tracks the camera pose using direct image alignment, minimising the photometric error between the current frame and warped keyframe. Associated with each keyframe is a photometric cost volume built up over many small baseline measurements from which dense depth maps are extracted through a variational optimisa-

tion technique. Other important examples include [Stuehmer et al., 2010], [Graber et al., 2011], MonoFusion [Pradeep et al., 2013], REMODE [Pizzoli et al., 2014], and LSD-SLAM [Engel et al., 2014], a semi-dense method.

The introduction of low cost commodity depth cameras such as the Microsoft Kinect and ASUS Xtion Pro Live also led to increased research on dense methods. As the existence of a high quality depth channel greatly simplifies both tracking and mapping, many of the most impressive online 3D reconstruction systems make use of RGB-D cameras. One of the first and most influential RGB-D SLAM systems is KinectFusion [Newcombe et al., 2011a], which uses a volumetric map based on a signed distance function (SDF). Many other RGB-D SLAM systems follow this formulation [Kerl et al., 2013, Whelan et al., 2012, Kahler et al., 2015]. ElasticFusion [Whelan et al., 2016], which will be discussed in more detail in Chapter 3, enables greater scalability by using a surfel-based map and focusing on global consistency by applying elastic map deformations upon loop closure.

1.4 Robust Dense Visual SLAM

While dense SLAM systems are currently valued for the 3D reconstructions they produce, it is interesting that one of the original motivations behind dense SLAM was to attain higher levels of robustness [Newcombe, 2012, Zienkiewicz, 2017]. The argument was that since the quality of an estimate can only increase with additional measurements, and since dense and direct systems use all of the available information and do not suffer from bad correspondences, they should outperform sparse systems. While dense SLAM does tend to be more robust to motion blur [Engel, 2017], in general, it is quite brittle. In [Concha and Civera, 2017], the authors evaluate two state-of-the-art dense RGB-D SLAM systems, ElasticFusion [Whelan et al., 2015] and RGBDTAM [Concha and Civera, 2017], on the full set of sequences in the TUM RGB-D Dataset [Sturm et al., 2012], a widely used RGB-D SLAM benchmark. Both systems failed on more than a third of the 36 sequences. In the majority of cases, these failures were a result of missing frames (requiring the systems to track across

large baselines) or a lack of sufficient photometric variation.

More generally, there are three main concepts that relate to the lack of robustness in dense SLAM:

1. **Tracking**: Direct image alignment, on which the tracking step is usually based, has a very narrow basin of convergence [Mobahi et al., 2012]. In practice, this restricts camera motions to being slow and smooth as without a prior on the camera motion, the relative pose of the current frame is initialised with the estimated pose of the previous frame. Some methods, such as a coarse-to-fine technique [Baker et al., 2004], can help improve the situation, but not enough for dense methods to track fast motions.

2. **Mapping**: Estimating depth by minimising the photometric error is not well constrained due to the presence of occlusions, homogeneous textures, non-Lambertian surfaces, and dynamic lighting. Dense SLAM systems address this by using regularisers, usually based on smoothness [Newcombe et al., 2011b, Pizzoli et al., 2014] or planar assumptions [Flint et al., 2011, Pradeep et al., 2013, Concha et al., 2014, Concha and Civera, 2014, Concha and Civera, 2015]. These regularisers are often chosen in a generic and data-independent way, and require tuning to get the best performance.

3. **Joint Inference**: Since dense SLAM systems use all of the pixels in the image, it is not feasible to compute the joint probabilistic inference of all poses and map points in a manner similar to sparse SLAM. Instead, dense systems typically alternate between optimising for the camera pose (while assuming that the map is correct) and optimising for the map (while assuming that the pose is correct). This approximation of the probabilistic inference means that not all existing cross-correlations are taken into account, as the accuracy of the map depends on the accuracy of the poses and *vice versa*.

The focus of this thesis is on developing methods that can be used to increase the robustness of dense SLAM by addressing the first two of these issues. While some

work addressing the third issue has been done in conjunction with this thesis [Bloesch et al., 2019, Czarnowski et al., 2020], it will not be directly discussed.

In particular, this thesis will look for ways in which dense SLAM can be improved by fusing additional sensing modalities into standard dense SLAM pipelines. Existing commercial robots that employ visual SLAM methods usually fuse measurements from other sensors such as bump detectors and position sensitive devices [Zienkiewicz, 2017], so keeping the standard pipeline intact to allow for the continued fusion of other sensors is an important consideration. This thesis will look at two sensing modalities, in particular: acceleration and rotation rate measurements from an *inertial measurement unit* (IMU) to address the *tracking* issue, and *learned priors on dense reconstructions* from deep neural networks (DNNs) to address the *mapping* issue.

IMUs are a compelling sensor choice as they have become cheap and abundant in consumer electronic devices. The fusing of inertial measurements has been widely adopted in sparse visual SLAM, leading to high levels of accuracy and robustness [Mourikis and Roumeliotis, 2007, Li and Mourikis, 2013, Tanskanen et al., 2015, Jones and Soatto, 2011, Keivan et al., 2014, Leutenegger et al., 2014, Forster et al., 2015]. This thesis demonstrates that similar improvements in robustness can be gained by fusing inertial measurements into the tracking step of a dense SLAM system.

The application of deep learning techniques to computer vision has led to dramatic increases in performance in many areas, including dense reconstruction [Eigen et al., 2014, Ummenhofer et al., 2017, Laina et al., 2016, Liu et al., 2015]. It was shown in [Fácil et al., 2017] that learning-based and geometry-based approaches have a complementary nature as learning-based systems tend to perform better on the interior points of objects but struggle on edges, and geometry-based systems typically do well on areas with high image gradients but poorly on interior points that may lack texture. The best way to combine these two approaches remains an open problem, however. Some impressive results have been shown by taking

the output of a traditional geometry-based system and passing this through a network [Zhou et al., 2018]. As discussed above, however, it may be desirable to treat a learning-based system as a "sensor" that is fused into a traditional dense SLAM pipeline. The difficulty with this approach is that some measure of the uncertainty associated with each prediction is required. This thesis will examine two methods to learn priors on the reconstruction that can be fused into standard dense SLAM systems.

1.5 Contributions

The contributions made in this thesis resulted in three separate publications. A full list of publications done in conjunction with thesis are provided in the next section.

1.5.1 Paper I: Dense RGB-D Inertial Fusion

Laidlow, T., Bloesch, M., Li, W. and Leutenegger, S. (2017), **Dense RGB-D-Inertial SLAM with Map Deformations**. In *Proceedings of the IEEE/RSJ Conference on Intelligent Robots and Systems (IROS)*. [Laidlow et al., 2017].

Context

As discussed in the previous section, dense tracking works by estimating the relative pose that minimises the photometric error between two frames. However, it is well-known that the associated optimisation is highly nonlinear and susceptible to local minima [Jin et al., 2003, Molton et al., 2004, Silveira et al., 2008, Mobahi et al., 2012]. For this reason, a good initialisation is required for each frame to be tracked. For general camera motion, a strong motion model may not be available and the best initialisation may simply be the estimated pose of the previous frame. In effect, this restricts dense SLAM implementations to slow, smooth trajectories where the pose of the previous frame is a good initialisation. In actual robotic systems, this may be too restrictive as vibrations, collisions and other fast or non-smooth motions may frequently break this underlying assumption.

Even with a good initialisation, the optimisation may still not converge without sufficient photometric variation to constrain the problem. This can easily occur when the camera view is dominated by a large homogeneous region such as a textureless wall.

Contribution

To address these weaknesses in direct photometric alignment, and inspired by the success of sparse visual-inertial systems [Jones and Soatto, 2011, Keivan et al., 2014, Leutenegger et al., 2014, Forster et al., 2015, Shen et al., 2015, Mur-Artal and Tardos, 2017, Qin et al., 2017, von Stumberg et al., 2018], this paper presents a dense SLAM system that fuses acceleration and rotation rate measurements into the tracking step in a tightly-coupled manner. This system has real-time capability while running on a GPU. It jointly optimises for the camera pose, velocity, IMU biases and gravity direction while building up a globally consistent, fully dense surfel-based 3D reconstruction of the environment. Through a series of experiments on both synthetic and real world datasets, it is shown that this dense visual-inertial SLAM system is more robust to fast motions and periods of low texture and low geometric variation than a related RGB-D-only SLAM system.

This dense RGB-D-inertial SLAM system is discussed in detail in Chapter 3.

1.5.2 Paper II: Probabilistic Fusion of Learned Parametric Priors

Laidlow, T., Czarnowski, J. and Leutenegger, S. (2019), **DeepFusion: Real-Time Dense 3D Reconstruction for Monocular SLAM using Single-View Depth and Gradient Predictions**. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*. [Laidlow et al., 2019].

Context

One of the major motivations for the use of dense SLAM systems is the desire for dense 3D reconstructions, as these scene representations may be useful for a number of robotic tasks such as safe robot navigation, augmented reality, and manipulation

tasks. Online dense reconstruction systems such as DTAM [Newcombe et al., 2011b] typically estimate depth values for selected keyframes by minimising the photometric error over many frames. Unfortunately, this optimisation problem is not well constrained due to the presence of occlusions, and homogeneous or repeated texture [Campbell et al., 2008]. To address this, dense systems typically use a strong regulariser based on planar [Flint et al., 2011, Pradeep et al., 2013, Concha et al., 2014, Concha and Civera, 2014, Concha and Civera, 2015] or smoothness assumptions [Newcombe et al., 2011b, Pizzoli et al., 2014].

As it is not possible to determine the translation of a camera using image correspondences alone, monocular reconstruction systems also suffer from an inherent scale ambiguity. It will be shown in Chapter 3 that it is possible to address this scale ambiguity by fusing vision-based measurements with readings from an IMU; however, in situations with low accelerations, scale becomes practically unobservable when using low grade IMUs. Another option to resolve these issues in monocular depth reconstruction is through the use of a depth camera [Newcombe et al., 2011a, Kerl et al., 2013], but depth cameras are limited in range and to indoor spaces.

Contribution

This paper presents a 3D reconstruction system, called DeepFusion, that leverages the output of a DNN to produce fully dense depth maps for keyframes with metric scale. DeepFusion is capable of producing real-time dense reconstructions on a GPU. It fuses the output of a semi-dense multi-view stereo algorithm with the depth and depth gradient predictions of a DNN in a probabilistic fashion, using learned uncertainties produced by the network. While the network only needs to be run once per keyframe, the optimisation for the depth map can be run at frame rate so as to constantly make use of the newest geometric constraints. Based on synthetic and real world datasets, we demonstrate that DeepFusion is capable of performing at least as well as other similar systems that combine deep learning with SLAM while maintaining a standard probabilistic SLAM framework.

DeepFusion is discussed in detail in Chapter 4.

1.5.3 Paper III: Probabilistic Fusion of Learned Nonparametric Priors

Laidlow, T., Czarnowski, J., Nicastro, A., Clark, R. and Leutenegger, S. (2020), **Towards the Probabilistic Fusion of Learned Priors into Standard Pipelines for 3D Reconstruction**. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*. [Laidlow et al., 2020].

Context

As will be shown in Chapter 4, it is possible to probabilistically fuse the outputs of a DNN into a standard dense reconstruction pipeline by having the DNN predict its own aleatoric uncertainty and using a Gaussian likelihood loss function during training. The problem with this approach, however, is that it forces the network to predict a parametric and unimodal distribution. This type of distribution may be particularly ill-suited to dense reconstruction where there is a clear need for a multi-hypothesis prediction [Campbell et al., 2008].

Additionally, the previous paper only estimated multi-view geometric constraints for keyframe pixels with a high image gradient as these are the points where a good estimation is most likely. Unfortunately, these points also happen to be where there are high depth gradients (edges of objects, etc.), which is where the network had learned to predict a high uncertainty. This meant that in the fusion, very little information was passed between semi-dense points and points on the interior of objects. Therefore the depth values for interior points relied almost entirely on the depth values predicted by the network.

Finally, the previous paper required the network to make an explicit prediction of the uncertainty. While the network's predictions matched intuition (the network placed highest uncertainty at depth discontinuities where there are large or rapidly changing gradients), it was difficult to train and it was not clear how the absolute

magnitude of the values should be interpreted.

Contribution

This paper proposes fusing a learned single-view depth prior into a standard 3D reconstruction system. It presents a system that is capable of incrementally producing dense depth maps for a set of keyframes. A DNN is trained to predict a discrete, nonparametric probability distribution for the depth of each pixel from a single image. This *probability volume* is then fused with another probability volume based on the photometric consistency between subsequent frames and the keyframe image. Combining the probability volumes from these two sources results in a volume that is better conditioned. To extract depth maps from the volume, a cost function that includes a regularisation term based on network predicted surface normals and occlusion boundaries is minimised. That each of these components improves the overall performance of the system is demonstrated through a series of experiments.

This is discussed in detail in Chapter 5.

1.6 Publications

The work described in this thesis resulted in the following publications:

- Laidlow, T., Czarnowski, J., Nicastro, A., Clark, R. and Leutenegger, S. (2020), **Towards the Probabilistic Fusion of Learned Priors into Standard Pipelines for 3D Reconstruction**. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*. [Laidlow et al., 2020].

- Laidlow, T., Czarnowski, J. and Leutenegger, S. (2019), **DeepFusion: Real-Time Dense 3D Reconstruction for Monocular SLAM using Single-View Depth and Gradient Predictions**. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*. [Laidlow et al., 2019].

- Laidlow, T., Bloesch, M., Li, W. and Leutenegger, S. (2017), **Dense RGB-D-Inertial SLAM with Map Deformations**. In *Proceedings of the IEEE/RSJ Conference on Intelligent Robots and Systems (IROS)*. [Laidlow et al., 2017].

While not described directly, the following publications were done in conjunction with this thesis:

- Czarnowski, J., Laidlow, T., Clark, R., and Davison, A. J. (2020), **DeepFactors: Real-Time Probabilistic Dense Monocular SLAM**. *IEEE Robotics and Automation Letters (RA-L)*. [Czarnowski et al., 2020].
- Bloesch, M., Laidlow, T., Clark, R., Leutenegger, S. and Davison, A. J. (2019), **Learning Meshes for Dense Visual SLAM**. In *Proceedings of the International Conference on Computer Vision (ICCV)*. [Bloesch et al., 2019].
- Bloesch, M., Sommer, H., Laidlow, T., Burri, M., Nuetzi, G., Fankhauser, P., Bellicoso, D., Gehring, C., Leutenegger, S., Hutter, M. and Siegwart, R. (2016), **A Primer on the Differential Calculus of 3D Orientations**. *CoRR*, arXiv.org. [Bloesch et al., 2016].

The following video material provides a visualisation of one of the algorithms developed in this thesis:

- Dense RGB-D-Inertial SLAM with Map Deformations, https://www.youtube.com/watch?v=-gUdQ0cxDh0.

1.7 Thesis Structure

The remainder of this thesis is structured as follows:

Chapter 2 introduces basic notation, the concepts used in dense SLAM, and provides a primer on both inertial measurement units (IMUs) and deep neural networks (DNNs).

Chapter 3 describes a tightly-coupled dense RGB-D-inertial SLAM system. Through a series of experiments on both synthetic and real world datasets, it is shown that fusing inertial measurements into the tracking step makes the dense SLAM system more robust to fast motions and scenes with low texture.

Chapter 4 presents DeepFusion, a real-time system that probabilistically fuses the output of a semi-dense multi-view stereo algorithm with depth and depth gradient predictions from a DNN to produce dense 3D reconstructions.

Chapter 5 describes another system for incrementally producing dense depth maps for a set of keyframes. This system addresses the limitations found in the previous chapter by using a DNN to predict discrete, nonparametric probability distributions for the depth of each pixel from a single image and then fusing this *probability volume* with another probability volume based on the photometric consistency between subsequent frames and the keyframe image.

Chapter 6 concludes the thesis with a discussion of the research presented and suggestions for future work.

Chapter 2

Preliminaries

Contents

2.1 Notation

This thesis makes use of the following notation:

2.1.1 General Notation

a This font is used for scalars.

$\mathbf{a}$ This font is used for M-dimensional column vectors, where a_i is the i^{th} element of the vector:

$$\mathbf{a} = \begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_M \end{bmatrix}, \quad \mathbf{a}^T = \begin{bmatrix} a_1 & a_2 & \dots & a_M \end{bmatrix}. \tag{2.1}$$

$\mathbf{A}$ This font is for $M \times N$-dimensional matrices, where a_{ij} is the matrix element at the i^{th} row and j^{th} column:

$$\mathbf{A} = \begin{bmatrix} a_{11} & a_{12} & \dots & a_{1N} \\ a_{21} & a_{22} & \dots & a_{2N} \\ \vdots & \vdots & \ddots & \vdots \\ a_{M1} & a_{M2} & \dots & a_{MN} \end{bmatrix}. \tag{2.2}$$

$\boldsymbol{a}$ This font is used for the homogeneous coordinate vector corresponding to the coordinate vector $\mathbf{a}$:

$$\boldsymbol{a} = \begin{bmatrix} \mathbf{a} \\ 1 \end{bmatrix}. \tag{2.3}$$

$\mathbf{1}$ This represents the identity matrix.

$\mathbf{0}$ This represents the zero matrix.

$(\cdot)^\times$ This denotes the cross-product operator that produces a skew-symmetric matrix from a 3-dimensional vector, such that $\mathbf{a} \times \mathbf{b} = \mathbf{a}^\times \mathbf{b}$:

$$\mathbf{a}^\times = \begin{bmatrix} a_1 \\ a_2 \\ a_3 \end{bmatrix}^\times = \begin{bmatrix} 0 & -a_3 & a_2 \\ a_3 & 0 & -a_1 \\ -a_2 & a_1 & 0 \end{bmatrix}. \tag{2.4}$$

2.1.2 Probability

$p(\mathbf{x})$	This represents the probability density of $\mathbf{x}$.
$p(\mathbf{x}\|\mathbf{y})$	This represents the probability density of $\mathbf{x}$ given $\mathbf{y}$.

2.1.3 Spaces and Manifolds

$\mathbb{R}$	This denotes the set of real numbers.
$\mathbb{R}_+$	This denotes the set of positive real numbers.
$\mathbb{R}^M$	This denotes the vector space of real M-dimensional vectors.
$\mathbb{R}^{M\times N}$	This denotes the vector space of real $M \times N$-dimensional matrices.
S^M	This denotes the M-sphere group.
$SO(3)$	This denotes the 3D rotation group.
$SE(3)$	This denotes the Special Euclidean group.
$\exp_{\mathbf{q}}(\cdot)$	This denotes the exponential map from $\mathbb{R}^3$ to S^3.
$\exp_{\mathbf{C}}(\cdot)$	This denotes the exponential map from $\mathbb{R}^3$ to $SO(3)$.
$\boxplus$	This denotes the *box-plus* operator that applies a small perturbation expressed in the tangent space to a manifold state.
$\boxminus$	This denotes the *box-minus* operator that determines the difference between two manifold states in the tangent space.

2.1.4 Frames and Transformations

$\underset{\rightarrow}{\mathcal{F}}_A$	This represents a frame of reference in $\mathbb{R}^3$.
${}_A\mathbf{a}$	The represents the vector $\mathbf{a}$ expressed in $\underset{\rightarrow}{\mathcal{F}}_A$.
${}_A\mathbf{r}_{AB}$	This represents the position vector from the origin of $\underset{\rightarrow}{\mathcal{F}}_A$ to the origin of $\underset{\rightarrow}{\mathcal{F}}_B$, represented in $\underset{\rightarrow}{\mathcal{F}}_A$.
${}_A\mathbf{v}_{BC}$	This represents the velocity of the origin of $\underset{\rightarrow}{\mathcal{F}}_C$ as observed by $\underset{\rightarrow}{\mathcal{F}}_B$ and expressed in $\underset{\rightarrow}{\mathcal{F}}_A$.
$\mathbf{C}_{AB}$	This represents a 3D rotation expressed as a rotation matrix (i.e. $\mathbf{C}_{AB} \in SO(3)$).

$\mathbf{q}_{AB}$ This represents a 3D rotation expressed as a Hamiltonian unit quaternion (i.e. $\mathbf{q}_{AB} \in S^3$).

$\otimes$ This denotes quaternion multiplication, such that $\mathbf{q}_{AC} = \mathbf{q}_{AB} \otimes \mathbf{q}_{BC}$.

$\mathbf{T}_{AB}$ This represents the homogeneous transformation matrix that transforms homogeneous points from $\underrightarrow{\mathcal{F}}_B$ to $\underrightarrow{\mathcal{F}}_A$.

2.1.5 Camera Models and Images

f_x This represents the horizontal focal length of the camera, in pixels.

f_y This represents the vertical focal length of the camera, in pixels.

c_x This represents the horizontal coordinate of the camera centre, in pixels.

c_y This represents the vertical coordinate of the camera centre, in pixels.

$\mathbf{K}$ This represents the intrinsic camera matrix:

$$\mathbf{K} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix}. \tag{2.5}$$

$\boldsymbol{\pi}(\cdot)$ This denotes the perspective projection function:

$$\boldsymbol{\pi}(\mathbf{a}) = \boldsymbol{\pi}\left(\begin{bmatrix} a_1 \\ a_2 \\ a_3 \end{bmatrix}\right) = \frac{1}{a_3}\begin{bmatrix} a_1 \\ a_2 \end{bmatrix}. \tag{2.6}$$

$I^{(n)}(\mathbf{u})$ This represents the intensity of image n at pixel coordinate $\mathbf{u}$.

$D^{(n)}(\mathbf{u})$ This represents the depth value corresponding to the pixel coordinate $\mathbf{u}$ in image n.

2.2 Transformations

The homogeneous transformation matrix, $\mathbf{T}_{AB}$, transforms homogeneous points from $\underrightarrow{\mathcal{F}}_B$ to $\underrightarrow{\mathcal{F}}_A$:

$$ {}_A\boldsymbol{p} = \mathbf{T}_{AB}\, {}_B\boldsymbol{p}. \tag{2.7} $$

This is particularly useful for visual SLAM as the matrix can be used to represent a six degrees-of-freedom (DoF) rigid transformation between two camera frames.

The 4×4 transformation matrix belongs to $SE(3)$, consisting of a 3DoF rotation and a 3DoF translation:

$$ \mathbf{T}_{AB} = \begin{bmatrix} \mathbf{C}_{AB} & {}_A\mathbf{r}_{AB} \\ \mathbf{0}^T & 1 \end{bmatrix}, \tag{2.8} $$

where $\mathbf{C}_{AB}$ is a 3×3 rotation matrix in $SO(3)$ (meaning $\mathbf{C}_{AB}\,\mathbf{C}_{AB}^T = \mathbf{C}_{AB}^T\,\mathbf{C}_{AB} = \mathbf{1}$ and $\det(\mathbf{C}_{AB}) = 1$), and ${}_A\mathbf{r}_{AB} \in \mathbb{R}^3$.

Transformation matrices can be chained together:

$$ \mathbf{T}_{AC} = \mathbf{T}_{AB}\,\mathbf{T}_{BC}, \tag{2.9} $$

and are invertible:

$$ \mathbf{T}_{BA} = \mathbf{T}_{AB}^{-1} = \begin{bmatrix} \mathbf{C}_{AB}^T & -\mathbf{C}_{AB}^T\, {}_A\mathbf{r}_{AB} \\ \mathbf{0}^T & 1 \end{bmatrix}. \tag{2.10} $$

2.3 Camera Model

The camera model describes the relationship between 3D Euclidean points and their projection onto the image plane of the camera. In this thesis, the *pinhole camera model* is used (Figure 2.1), and all geometric distortions are assumed to have been corrected.

The pinhole camera model uses four parameters to model the image formation process (*skew*, a possible fifth parameter, is assumed to be zero): the focal length in the horizontal direction (f_x), the focal length in the vertical direction (f_y), the horizontal coordinate of the camera centre (c_x), and the vertical coordinate of the

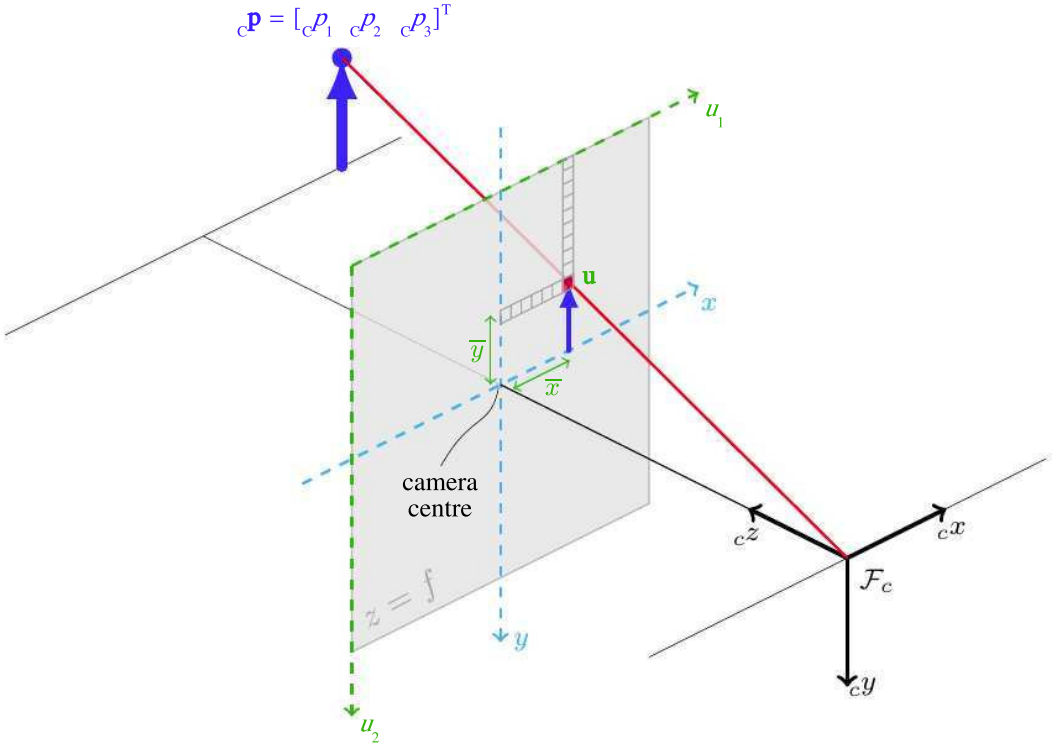


Figure 2.1: Perspective projection of a 3D Eucliean coordinate to a 2D pixel coordinate using the pinhole camera model. Modified from: https://tex.stackexchange.com/questions/96074, original author: https://tex.stackexchange.com/users/22653/perr0. License: CC BY-SA 3.0.

camera centre (c_y). In an ideal pinhole camera, the focal length (which is the distance between the camera centre and image plane) would be the same for the horizontal and vertical directions, but these can differ in reality as the physical height and width of the pixels may not be equal. These camera parameters are typically combined into the intrinsic matrix, $\mathbf{K}$:

$$\mathbf{K} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix}. \tag{2.11}$$

The projection of a 3D Euclidean point, ${}_C\mathbf{p}$, expressed in the camera frame, $\mathcal{F}_{\underrightarrow{C}}$,

to a 2D pixel coordinate, $\mathbf{u}$, is given by:

$$\mathbf{u} = \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} = \begin{bmatrix} \frac{f_x\, {}_C p_1}{{}_C p_3} + c_x \\ \frac{f_y\, {}_C p_2}{{}_C p_3} + c_y \end{bmatrix} = \boldsymbol{\pi}\left(\begin{bmatrix} f_x\, {}_C p_1 + c_x\, {}_C p_3 \\ f_y\, {}_C p_2 + c_y\, {}_C p_3 \\ {}_C p_3 \end{bmatrix}\right) = \boldsymbol{\pi}\left(\begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} {}_C p_1 \\ {}_C p_2 \\ {}_C p_3 \end{bmatrix}\right) = \boldsymbol{\pi}(\mathbf{K}\, {}_C\mathbf{p}), \tag{2.12}$$

where $\boldsymbol{\pi}(\cdot)$ is the perspective projection function:

$$\boldsymbol{\pi}(\mathbf{a}) = \boldsymbol{\pi}\left(\begin{bmatrix} a_1 \\ a_2 \\ a_3 \end{bmatrix}\right) = \frac{1}{a_3} \begin{bmatrix} a_1 \\ a_2 \end{bmatrix}. \tag{2.13}$$

The projection function is not injective and, therefore, not invertible. However, if the depth of the pixel ($D(\mathbf{u}) = {}_C p_3$) is known, the 3D Euclidean point ${}_C\mathbf{p}$ can be recovered from the homogeneous pixel coordinate $\boldsymbol{u}$:

$${}_C\mathbf{p} = \begin{bmatrix} {}_C p_1 \\ {}_C p_2 \\ {}_C p_3 \end{bmatrix} = {}_C p_3 \begin{bmatrix} \frac{1}{f_x} & 0 & \frac{-c_x}{f_x} \\ 0 & \frac{1}{f_y} & \frac{-c_y}{f_y} \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \frac{f_x\, {}_C p_1}{{}_C p_3} + c_x \\ \frac{f_y\, {}_C p_2}{{}_C p_3} + c_y \\ 1 \end{bmatrix} = D(\mathbf{u}) \begin{bmatrix} \frac{1}{f_x} & 0 & \frac{-c_x}{f_x} \\ 0 & \frac{1}{f_y} & \frac{-c_y}{f_y} \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \\ 1 \end{bmatrix} = D(\mathbf{u})\mathbf{K}^{-1}\boldsymbol{u}. \tag{2.14}$$

In addition to standard cameras, this thesis also makes use of *depth* or *RGB-D cameras*. Depth cameras, such as the Microsoft Kinect or Asus Xtion Pro Live, also capture a dense depth reading from the environment for each RGB frame of the incoming video stream. Typically, the cameras register the depth readings to the RGB frame (an assumption made in this thesis). To measure depth, these cameras project a known infrared pattern onto the scene and capture the reflection with an offset infrared sensor. The depth of each scene point can be computed by measuring deviations between the captured and known reference patterns.

2.4 Nonlinear Optimisation

As will be shown later, both the tracking and mapping problems in dense SLAM are typically formulated as nonlinear least-squares optimisation problems. For these

types of problems, a cost function, $c(\cdot)$, is defined based on the (weighted) sum of squared residuals for a given state estimate, $\mathbf{x}$:

$$c(\mathbf{x}) = \frac{1}{2}\sum_{i=1}^{M} \mathbf{r}_i(\mathbf{x})^T \mathbf{W}_i \mathbf{r}_i(\mathbf{x}) = \frac{1}{2}\mathbf{r}(\mathbf{x})^T \mathbf{W}\mathbf{r}(\mathbf{x}) = \frac{1}{2}||\mathbf{r}(\mathbf{x})||^2_{\mathbf{W}}, \tag{2.15}$$

where

$$\mathbf{r}(\mathbf{x}) = \begin{bmatrix} \mathbf{r}_1(\mathbf{x})^T & \mathbf{r}_2(\mathbf{x})^T & \dots & \mathbf{r}_M(\mathbf{x})^T \end{bmatrix}^T, \tag{2.16}$$

$$\mathbf{W} = \begin{bmatrix} \mathbf{W}_1 & \mathbf{0} & \dots & \mathbf{0} \\ \mathbf{0} & \mathbf{W}_2 & \dots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \dots & \mathbf{W}_M \end{bmatrix}, \tag{2.17}$$

$\mathbf{r}_i(\cdot)$ is a nonlinear residual function of arbitrary form, $\mathbf{W}_i$ is a symmetric, positive-definite (and often diagonal) weighting matrix, and $||\cdot||_{\mathbf{W}}$ is the Mahalanobis norm. In SLAM, the weighting matrix often will be the inverse covariance matrix associated with a given measurement ($\mathbf{W}_i = \mathbf{R}^{-1}$).

The suitability of a given state estimate, $\mathbf{x}$, is determined by how well it minimises the cost function:

$$\mathbf{x}^* = \underset{\mathbf{x}}{\operatorname{argmin}}\, c(\mathbf{x}) = \underset{\mathbf{x}}{\operatorname{argmin}}\, \frac{1}{2}||\mathbf{r}(\mathbf{x})||^2_{\mathbf{W}}. \tag{2.18}$$

The Gauss-Newton algorithm is an iterative method that can be used to solve this least squares minimisation problem. With each iteration, the algorithm first approximates the nonlinear formulation by linearising the residual function, $\mathbf{r}(\cdot)$, using a first-order Taylor expansion about the current estimate of the solution, $\bar{\mathbf{x}}$:

$$\min_{\mathbf{x}} \frac{1}{2}||\mathbf{r}(\mathbf{x})||^2_{\mathbf{W}} = \min_{\delta\mathbf{x}} \frac{1}{2}||\mathbf{r}(\bar{\mathbf{x}} + \delta\mathbf{x})||^2_{\mathbf{W}} \approx \min_{\delta\mathbf{x}} \frac{1}{2}||\mathbf{r}(\bar{\mathbf{x}}) + \mathbf{J}\delta\mathbf{x}||^2_{\mathbf{W}}, \tag{2.19}$$

where $\delta\mathbf{x}$ is a perturbation around $\bar{\mathbf{x}}$, and

$$\mathbf{J} = \left.\frac{\partial \mathbf{r}(\mathbf{x})}{\partial \mathbf{x}}\right|_{\mathbf{x}=\bar{\mathbf{x}}}. \tag{2.20}$$

Expanding the Mahalanobis norm and denoting $\mathbf{r}(\bar{\mathbf{x}})$ as $\bar{\mathbf{r}}$ gives:

$$\min_{\delta\mathbf{x}} \frac{1}{2}||\mathbf{r}(\bar{\mathbf{x}}) + \mathbf{J}\delta\mathbf{x}||^2_{\mathbf{W}} = \min_{\delta\mathbf{x}} \frac{1}{2}\bar{\mathbf{r}}^T\mathbf{W}\bar{\mathbf{r}} + \bar{\mathbf{r}}^T\mathbf{W}\mathbf{J}\delta\mathbf{x} + \frac{1}{2}\delta\mathbf{x}^T\mathbf{J}^T\mathbf{W}\mathbf{J}\delta\mathbf{x}. \tag{2.21}$$

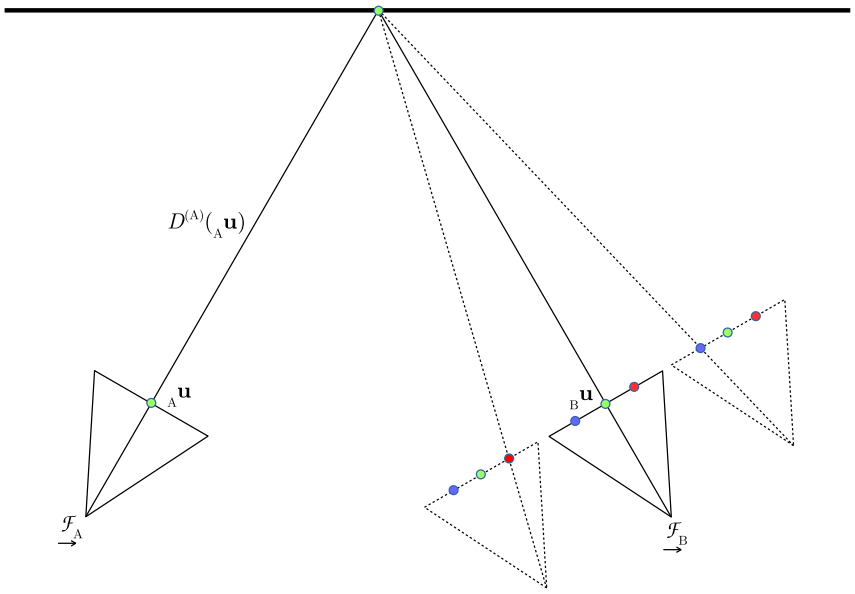


Figure 2.2: In dense tracking, a nonlinear optimisation technique is used to find the relative pose of camera B that minimises the photometric error between the image observed by camera B and the warped image from camera A, given a depth map, $D^{(A)}({}_A\mathbf{u})$. Dense tracking assumes photoconsistency and that all surfaces are sufficiently textured.

The Gauss-Newton algorithm then computes an update to the current estimate by taking the derivative of the resulting quadratic and setting it equal to zero:

$$\delta\mathbf{x}^* = -(\mathbf{J}^T\mathbf{W}\mathbf{J})^{-1}\mathbf{J}^T\mathbf{W}\bar{\mathbf{r}}. \tag{2.22}$$

If all variables in the problem are observable, the system will have full rank and a unique solution will exist. This update is then applied to the current estimate ($\mathbf{x} \leftarrow \bar{\mathbf{x}} + \delta\mathbf{x}^*$), and the process is repeated, relinearising at the new estimate, until convergence.

2.5 Dense Tracking

As previously discussed, dense SLAM does not typically estimate the joint probability of camera and map parameters like is done in sparse SLAM, but, for computational feasibility, alternates between the tracking and mapping steps.

Unlike sparse tracking, which estimates a camera pose by minimising the reprojec-

tion error over a set of 3D landmarks, dense tracking directly optimises over the pixel intensities. The dense tracking problem is formulated as a nonlinear optimisation problem:

$$\boldsymbol{\xi}^*_{BA} = \underset{\boldsymbol{\xi}_{BA}}{\operatorname{argmin}} \sum_{{}_A\mathbf{u} \in \Omega^{(A)}} \psi(I^{(A)}({}_\mathbf{A}\mathbf{u}),\, I^{(B)}(\mathbf{w}({}_\mathbf{A}\mathbf{u},\, D^{(A)}({}_A\mathbf{u}),\, \boldsymbol{\xi}_{BA}))), \tag{2.23}$$

where ${}_A\mathbf{u}$ is the coordinate vector of a pixel in image A, $\mathbf{w}(\cdot)$ is a warping function, $I^{(A)}(\cdot)$ is the intensity value of a given pixel in image A, $I^{(B)}(\cdot)$ is the intensity value of a given pixel in image B, $D^{(A)}(\cdot)$ is the depth value associated with a given pixel in image A and is assumed to be known, $\boldsymbol{\xi}_{BA}$ is the vector of parameters used by the warping function to transform points from $\underset{\rightarrow}{\mathcal{F}}_A$ to $\underset{\rightarrow}{\mathcal{F}}_B$, $\psi(\cdot,\cdot)$ is a function that returns an error metric based on two intensity values, and $\Omega^{(A)}$ is the set of pixels in A that are co-visible in B. The dense tracking problem is illustrated in Figure 2.2.

Typically, the *sum of squared differences* (SSD) function is used for the error metric:

$$\psi\left(I^{(A)}({}_A\mathbf{u}),\, I^{(B)}({}_B\mathbf{u})\right) = \frac{1}{2}\left(I^{(A)}({}_A\mathbf{u}) - I^{(B)}({}_B\mathbf{u})\right)^2, \tag{2.24}$$

as this formulates dense tracking as a nonlinear least-squares optimisation problem and enables using the Gauss-Newton algorithm discussed in Section 2.4.

For general 6DoF camera motion, the transformation parameters, $\boldsymbol{\xi}_{BA}$, must represent a 3D rotation and a 3D translation. Common representations of 3D rotation include quaternions ($\mathbf{q}_{BA}$) and rotation matrices ($\mathbf{C}_{BA}$). Both of these representations are overparameterised, however, as only unit quaternions (vectors in $\mathbb{R}^4$ belonging to S^3) and the subset of 3×3 matrices that belong to $SO(3)$ represent valid rotations. Not only does this mean that the state space would have redundant parameters, but also that these state representations would have to be continuously re-normalised to avoid them becoming degenerate. Alternatively, it is possible to use minimal representations such as Euler angles or angle-axis and avoid this problem, but these representations suffer from singularities where large changes in the parameterisation are required to represent small changes in the state.

Since most state estimation techniques work through iterative methods that continually apply small perturbations to a reference point (such as with Gauss-Newton optimisation), it is possible to use the fact that all points on an n-dimensional manifold have a local neighbourhood with a bidirectional mapping to a tangent space in $\mathbb{R}^n$. This allows for state representations to belong to S^3 ($\mathbf{q}_{BA}$) or $SO(3)$ ($\mathbf{C}_{BA}$), but to have the small iterative updates calculated in a minimal parameterisation belonging to $\mathbb{R}^3$ which are then mapped back to the manifold.

This thesis follows the implementation set out in [Hertzberg et al., 2011] and [Bloesch et al., 2016], where this mapping between an n-dimensional manifold, $\mathcal{M}^n$, and its tangent space, $\mathbb{R}^n$, are performed via the $\boxplus$ ("box-plus") and $\boxminus$ ("box-minus") operators:

$$\begin{aligned} \boxplus &: \mathcal{M}^n \times \mathbb{R}^n \to \mathcal{M}^n, \\ \boxminus &: \mathcal{M}^n \times \mathcal{M}^n \to \mathbb{R}^n. \end{aligned} \tag{2.25}$$

The $\boxplus$ operator applies a perturbation, expressed in $\mathbb{R}^n$, to the state estimate, expressed in $\mathcal{M}^n$. The $\boxminus$ operator determines the difference between two $\mathcal{M}^n$ state representations in the tangent space. Note that the $\boxplus$ and $\boxminus$ operators can apply to any manifold, and if that manifold is $\mathbb{R}^n$ itself, then the operators correspond to standard vector addition and subtraction.

If the 3D rotation is represented by a unit quaternion, $\mathbf{q}_{BA} \in S^3$, then the update is applied by a local perturbation, $\delta\boldsymbol{\alpha} \in \mathbb{R}^3$, around the reference $\bar{\mathbf{q}}_{BA} \in S^3$:

$$\mathbf{q}_{BA} = \bar{\mathbf{q}}_{BA} \boxplus \delta\boldsymbol{\alpha} = \exp_{\mathbf{q}}(\delta\boldsymbol{\alpha}) \otimes \bar{\mathbf{q}}_{BA}, \tag{2.26}$$

where $\exp_{\mathbf{q}}(\cdot)$ is the exponential map from $\mathbb{R}^3$ to a unit quaternion:

$$\exp_{\mathbf{q}}(\delta\boldsymbol{\alpha}) = \begin{bmatrix} \sin\left(\frac{||\delta\boldsymbol{\alpha}||}{2}\right) \frac{\delta\boldsymbol{\alpha}}{||\delta\boldsymbol{\alpha}||} \\ \cos\left(\frac{||\delta\boldsymbol{\alpha}||}{2}\right) \end{bmatrix} \in S^3. \tag{2.27}$$

If the 3D rotation is represented by a rotation matrix, $\mathbf{C}_{BA} \in SO(3)$, then the update is applied by a local perturbation, $\delta\boldsymbol{\alpha} \in \mathbb{R}^3$, around the reference $\bar{\mathbf{C}}_{BA} \in SO(3)$:

$$\mathbf{C}_{BA} = \bar{\mathbf{C}}_{BA} \boxplus \delta\boldsymbol{\alpha} = \exp_{\mathbf{C}}(\delta\boldsymbol{\alpha})\bar{\mathbf{C}}_{BA}, \tag{2.28}$$

where $\exp_{\mathbf{C}}(\cdot)$ is the exponential map from $\mathbb{R}^3$ to $SO(3)$, also known as Rodrigues' formula:

$$\exp_{\mathbf{C}}(\delta\boldsymbol{\alpha}) = \mathbf{1} + \sin(||\delta\boldsymbol{\alpha}||)\frac{\delta\boldsymbol{\alpha}^{\times}}{||\delta\boldsymbol{\alpha}||} + (1 - \cos(||\delta\boldsymbol{\alpha}||))\left(\frac{\delta\boldsymbol{\alpha}^{\times}}{||\delta\boldsymbol{\alpha}||}\right)^2 \in SO(3), \tag{2.29}$$

where

$$\delta\boldsymbol{\alpha}^{\times} = \begin{bmatrix} 0 & -\delta\alpha_3 & \delta\alpha_2 \\ \delta\alpha_3 & 0 & -\delta\alpha_1 \\ -\delta\alpha_2 & \delta\alpha_1 & 0 \end{bmatrix}. \tag{2.30}$$

For tracking the 6DoF camera pose with a 3D Euclidean vector, ${}_B\mathbf{r}_{BA}$, and a rotation matrix, $\mathbf{C}_{BA}$, then, a minimal parameterisation in $\mathbb{R}^6$ is required when optimising Eq. 2.23:

$$\boldsymbol{\xi}_{BA} = \begin{bmatrix} \delta\mathbf{r}^T & \delta\boldsymbol{\alpha}^T \end{bmatrix}^T \in \mathbb{R}^6, \tag{2.31}$$

where ${}_B\mathbf{r}_{BA} = {}_B\bar{\mathbf{r}}_{BA} \boxplus \delta\mathbf{r}$ and $\mathbf{C}_{BA} = \bar{\mathbf{C}}_{BA} \boxplus \delta\boldsymbol{\alpha}$.

The choice of the warping function, $\mathbf{w}(\cdot)$, depends on the assumptions made about camera motion and scene geometry. While Eq. 2.23 includes the depth values, $D^{(A)}$, in the warping function, these are not strictly necessary if the camera motion is purely rotational or if camera motion is translational and the scene is assumed to be planar. In the case of general 6DoF camera motion, however, the following warping function is used:

$$\mathbf{w}({}_{\mathbf{A}}\mathbf{u}, D^{(A)}({}_A\mathbf{u}), \boldsymbol{\xi}_{BA}) = \boldsymbol{\pi}\left(\mathbf{K}\left(\exp_{\mathbf{C}}(\delta\boldsymbol{\alpha})\bar{\mathbf{C}}_{BA}D({}_A\mathbf{u})\mathbf{K}^{-1}{}_A\boldsymbol{u} + {}_B\bar{\mathbf{r}}_{BA} + \delta\mathbf{r}\right)\right). \tag{2.32}$$

Since $\mathbf{w}(\cdot)$ needs to be a continuous function, the intensity value, $I^{(B)}$, is interpolated using bilinear sampling.

The choice of the error metric, $\psi(\cdot,\cdot)$ is also important for dense tracking. As mentioned earlier, a common and simple approach that enables the use of a nonlinear least-squares algorithm to solve Eq. 2.23 is to use the SSD function. Unfortunately, this error metric makes the dense tracking optimisation problem susceptible to outliers, as large differences between the two intensity values will have a significant impact on the least-squares system and potentially lead to spurious results. One

option is to evaluate the error metric on small image patches around the correspondences rather than on single pixels. Another possible solution is to use the *sum of absolute differences* (SAD) function:

$$\psi\left(I^{(A)}({}_A\mathbf{u}),\, I^{(B)}({}_B\mathbf{u})\right) = \left| I^{(A)}({}_A\mathbf{u}) - I^{(B)}({}_B\mathbf{u}) \right|, \tag{2.33}$$

as the penalty on large outliers will be linear rather than quadratic. However, since the SAD function is not differentiable at the origin, it can not easily be used with common optimisation techniques like Gauss-Newton.

One approach that is sometimes used in this thesis to reduce the sensitivity of the optimisation to outliers is to use a differentiable robust cost function like the Huber loss. Compared with the least-squares quadratic penalty, the penalties of these cost functions are less extreme when there are large differences in the intensity values. The Huber loss function is given by:

$$\psi\left(I^{(A)}({}_A\mathbf{u}),\, I^{(B)}({}_B\mathbf{u})\right) = \begin{cases} \frac{1}{2}e^2, & \text{for } |e| \leq \kappa \\ \kappa\left(|e| - \frac{1}{2}\kappa\right), & \text{otherwise} \end{cases} \tag{2.34}$$

where

$$e = I^{(A)}({}_A\mathbf{u}) - I^{(B)}({}_B\mathbf{u}), \tag{2.35}$$

and κ is a hyperparameter to be tuned.

All of these error metrics are based on the assumption that photometric consistency is maintained across frames; that is, it is assumed that the pixel intensities of two true correspondences will be the same. In practice, it is possible to estimate other parameters of the image formation process, such as bias and gain, to account for differences in factors such as exposure time [Engel et al., 2017].

Another major assumption in the error metrics used for dense tracking is that the environment remains static. Any moving objects in the scene could result in large photometric errors. While these errors are often handled through the use of robust cost functions and other methods to reduce the impact of outliers, it is possible to go a step further and use techniques that identify dynamic objects [Jaimez et al.,

2017, Scona et al., 2018, Barnes et al., 2018, Bescós et al., 2018, Xu et al., 2019] and mask the resulting errors.

The dense tracking problem is usually solved by a nonlinear least-squares optimisation method such as Gauss-Newton. It has been shown, however, that the cost landscape of Eq. 2.23 is highly non-convex with a very narrow convergence basin [Mobahi et al., 2012]. For this reason, a good initialisation is required to converge to the true global minimum. Without a prior on the camera motion, the transformation is initialised at identity, limiting the dense tracking to only being able to solve slow and smooth trajectories. One method that can help widen the convergence basin is called *coarse-to-fine* tracking. With coarse-to-fine, the optimisation is first performed on a coarse resolution of the two images and the resulting transformation is used to initialise the optimisation on a finer resolution of the images. This process is carried on up the "pyramid" of images until the original image resolution is reached.

An important point to note is that the tracking discussed so far is based on finding the relative pose between frames. An alternative to this *frame-to-frame* tracking is *model-based* tracking where the global pose of a camera is tracked against a known 3D scene. This form of tracking works similarly to frame-to-frame tracking, but instead of comparing the intensity values of one frame warped into another, the rendered intensity values at the estimated pose are compared against those of the current frame.

If a depth camera is being used, then it is also possible to perform dense tracking by using *geometric* alignment in addition to the photometric alignment. With dense geometric alignment, the goal is to align the point cloud produced by backprojecting the depth channel with either the global model or a similar point cloud from a previous frame. The most commonly used method to achieve this alignment is the *Iterative Closest Point* (ICP) algorithm. This thesis uses the point-to-plane variant of ICP and performs the geometric tracking against the global model. Given a current estimate of the camera position, the ICP algorithm first matches each point

from the current camera frame with the closest point in the global model based on the Euclidean distance. Next, a transformation between the camera frame, $\underrightarrow{\mathcal{F}}_C$, and world frame, $\underrightarrow{\mathcal{F}}_W$, is estimated by minimising a nonlinear cost function:

$$\boldsymbol{\xi}^*_{WC} = \underset{\boldsymbol{\xi}_{WC}}{\operatorname{argmin}} \sum_{\mathbf{u}} {}_W\mathbf{n}_k^T \left(\mathbf{w}(\mathbf{u}, D(\mathbf{u}), \boldsymbol{\xi}_{WC}) - {}_W\mathbf{p}_k \right), \tag{2.36}$$

where $\mathbf{w}(\cdot)$ is a warping function that transforms points from $\underrightarrow{\mathcal{F}}_C$ to $\underrightarrow{\mathcal{F}}_W$ according to the transformation parameters, $\boldsymbol{\xi}_{WC}$, ${}_W\mathbf{p}_k$ is the 3D Euclidean point in the global model that corresponds with the backprojected pixel coordinate, and ${}_W\mathbf{n}_k$ is the surface normal vector associated with that point. Using this update to the camera pose, the points are matched again to the global model and the process is repeated until convergence.

2.6 Dense Mapping

Most dense mapping systems are keyframe-based, estimating depth maps for a subset of the incoming RGB frames. These depth maps are usually produced through the use of a multi-view stereo approach. While depth maps are individually useful for dense tracking, the combination of many depth maps are required for a good 3D reconstruction. This combination can be done in a number of ways. One simple method is to backproject the points to form a 3D point cloud. This can be augmented with knowledge of the surface orientation to produce a surfel-based representation [Whelan et al., 2016]. Alternatively, the depth maps can be fused into a volumetric representation such as a signed distance function (from which a mesh can be extracted using the marching cubes algorithm) [Newcombe et al., 2011a] or occupancy map [Vespa et al., 2018]. While the way in which depth maps are fused is an important research question in dense SLAM, it is not a focus of this thesis. Instead, this thesis will focus on the creation of high quality depth maps that will improve the reconstructions produced by any of the fusion methods.

Whereas dense tracking estimates the relative transformation between two frames assuming the depth is known, multi-view stereo methods estimate the depth assuming the relative transformation between the frames is known. Given the degree to

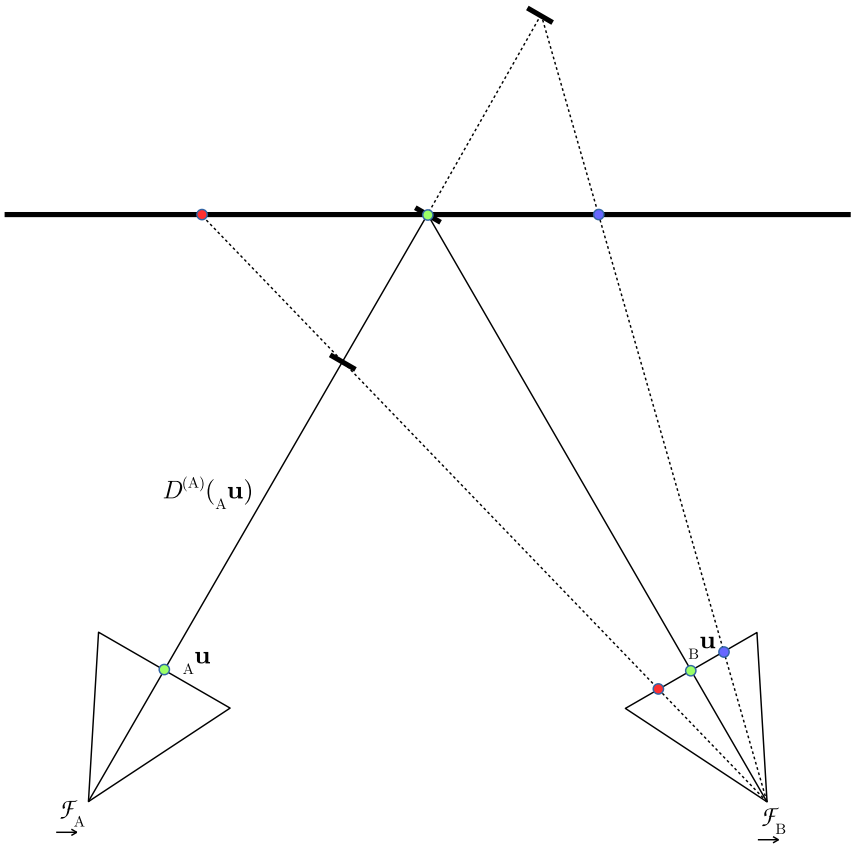


Figure 2.3: Given the relative transformation between two cameras, A and B, the depth map for camera A can be estimated by finding the depth values that minimise the photometric error between the image observed by camera B and the warped image from camera A. Like dense tracking, depth estimation assumes photoconsistency between the two frames and that the surface is sufficiently textured.

which these problems are closely related, it is not surprising that the depth estimation problem has the same formulation as dense tracking but with a different set of minimisation variables:

$$D^{(A)}({}_A\mathbf{u})^* = \underset{D^{(A)}({}_A\mathbf{u})\in\mathbb{R}_+}{\operatorname{argmin}} \; c_{\text{data}}\left(D^{(A)}({}_A\mathbf{u})\right), \tag{2.37}$$

for all ${}_A\mathbf{u} \in \Omega$, where

$$c_{\text{data}}\left(D^{(A)}({}_A\mathbf{u})\right) = \psi(I^{(A)}({}_\mathbf{A}\mathbf{u}),\, I^{(B)}(\mathbf{w}({}_\mathbf{A}\mathbf{u},\, D^{(A)}({}_A\mathbf{u}),\, {}_B\mathbf{r}_{BA},\, \mathbf{C}_{BA}))). \tag{2.38}$$

The depth estimation problem is illustrated in Figure 2.3.

Since the relative transformation between the two cameras $({}_B\mathbf{r}_{BA}, \mathbf{C}_{BA})$ is assumed to be known, the warping function $\mathbf{w}(\cdot)$ becomes:

$$\mathbf{w}({}_\mathbf{A}\mathbf{u},\, D^{(A)}({}_A\mathbf{u}),\, {}_B\mathbf{r}_{BA},\, \mathbf{C}_{BA}) = \boldsymbol{\pi}\left(\mathbf{K}\left(\mathbf{C}_{BA} D^{(A)}({}_A\mathbf{u})\mathbf{K}^{-1}\, {}_A\boldsymbol{u} + {}_B\mathbf{r}_{BA}\right)\right). \tag{2.39}$$

The issues regarding the error metric, $\psi(\cdot,\cdot)$ in Eq. 2.37 are the same as those for dense tracking. While it is possible to use an SSD function and easily estimate the depth values using a nonlinear least-squares technique, the dense mapping minimisation problem is highly non-convex and requires a good initialisation for convergence. An alternative approach is to discretise the depth range and do a direct search over the possible depth hypotheses [Newcombe et al., 2011b].

As formulated in Eq. 2.37, the depth value corresponding to each pixel in image A is estimated independently. Unlike sparse systems that estimate the 3D position of sparsely distributed landmarks, the assumption that points in the scene are independent is not statistically valid in dense systems as it is assumed that the environment is composed of many connected surfaces. That is, the likelihood of a pixel's depth value is dependent on the depth values of other pixels in its local neighbourhood. Such a formulation also does not work well in practise, as many factors that frequently occur in natural environments (such as non-Lambertian surfaces, occlusions, areas of low texture, and repeated texture in the scene) may mean that the cost landscape associated with Eq. 2.37 has an ambiguous minimum or many local minima.

To address these issues, dense SLAM systems typically make use of a *geometry prior*. These priors usually take the form of a regulariser on the cost function, often based on smoothness or planar assumptions:

$$D^{(A)*} = \underset{D^{(A)} \in \mathbb{R}_+}{\operatorname{argmin}} \sum_{{}_A\mathbf{u} \in \Omega} \left[c_{\text{data}}\left(D^{(A)}({}_A\mathbf{u})\right) + \sum_{{}_A\mathbf{v} \in \mathcal{N}({}_A\mathbf{u})} c_{\text{reg}}\left(D^{(A)}({}_A\mathbf{u}),\, D^{(A)}({}_A\mathbf{v})\right)\right], \tag{2.40}$$

where $c_{\text{reg}}(\cdot)$ is a regularisation cost term and $\mathcal{N}({}_A\mathbf{u})$ is the set of pixels in a neighbourhood of ${}_A\mathbf{u}$.

2.7 Inertial Measurement Units (IMUs)

As discussed above, the accuracy of dense tracking can suffer in situations with fast motion (due to the narrow convergence basin of the minimisation problem) or low texture (due to a lack of a photometric information to constrain the optimisation). Sparse SLAM methods, which can also struggle in these situations, have typically addressed this lack of robustness by fusing inertial data into the estimation procedure. Inertial measurement units (IMUs), particularly the low-cost microelectromechanical (MEMS) IMUs, are a popular sensor choice and are commonly found in consumer electronics. An IMU consists of an accelerometer and gyroscope that return measurements of the specific force (acceleration relative to free-fall) and the angular rate, respectively.

The fusion of visual and inertial information, in particular, can provide improvements in both the robustness and accuracy over vision-only tracking because of the complementary nature of the two sensing modalities. Incremental poses can be estimated at a high rate by integrating the incoming inertial measurements, providing good initialisations for the photometric alignment used in the tracking step. These incremental pose estimates will quickly accumulate drift, however, due to the integration of sensor noise, which is typically high in low-grade IMUs. The accelerometer and gyroscope also have time-varying biases that cannot be estimated from the inertial data alone. Visual information, on the other hand, can only be used with a good initialisation, but can help constrain the drift in the pose estimates and make the IMU biases observable.

There are two general methods for fusing inertial data with visual information: loosely-coupled approaches and tightly-coupled approaches. In loosely-coupled systems, the inertial and visual systems run independently of each other, each treated as a sensor model that returns a state estimate and associated covariance which are usually combined together in a probabilistically meaningful manner. While this reduces the computational complexity and allows for the extension of existing vision-only SLAM systems without modification, it does not allow the vision system to gain

any benefit from the inertial measurements.

In tightly-coupled systems, both sets of measurements are included in the model and the pose is estimated jointly from the combined visual-inertial data. This ensures that any correlations between the state variables is kept intact. It has been shown (e.g. [Leutenegger et al., 2014]) that the correlations maintained by a tightly-coupled system are necessary for high-precision tracking.

For inertial estimation, a model of IMU kinematics and bias dynamics is required. The state of the IMU, $\mathbf{x}$, consists of a 6DoF pose (${}_I\mathbf{r}_{IS}$ and $\mathbf{q}_{IS}$) of the IMU frame, $\underrightarrow{\mathcal{F}_S}$, in the inertial frame, $\underrightarrow{\mathcal{F}_I}$, the velocity of the IMU (since the accelerometer measurements are integrated twice), ${}_I\mathbf{v}_{IS}$, and the biases of the gyroscope, $\mathbf{b}_\mathrm{g}$, and accelerometer, $\mathbf{b}_\mathrm{a}$:

$$\mathbf{x} := \begin{bmatrix} {}_I\mathbf{r}_{IS}^T & \mathbf{q}_{IS}^T & {}_I\mathbf{v}_{IS}^T & \mathbf{b}_\mathrm{g}^T & \mathbf{b}_\mathrm{a}^T \end{bmatrix}^T \in \mathbb{R}^3 \times S^3 \times \mathbb{R}^9. \tag{2.41}$$

Assuming the effects of the Earth's rotation can be ignored, the nonlinear continuous-time state-transition model is given by:

$$\begin{aligned} \frac{\partial {}_I\mathbf{r}_{IS}}{\partial t} &= {}_I\mathbf{v}_{IS}, \\ \frac{\partial \mathbf{q}_{IS}}{\partial t} &= \frac{1}{2}\mathbf{q}_{IS} \otimes \begin{bmatrix} -{}_S\tilde{\boldsymbol{\omega}} + \mathbf{b}_\mathrm{g} - \mathbf{w}_g \\ 0 \end{bmatrix}, \\ \frac{\partial {}_I\mathbf{v}_{IS}}{\partial t} &= \mathbf{C}_{IS}\left({}_S\tilde{\mathbf{a}} + \mathbf{w}_a - \mathbf{b}_\mathrm{a}\right) + {}_I\mathbf{g}, \\ \frac{\partial \mathbf{b}_\mathrm{g}}{\partial t} &= \mathbf{w}_{b_g}, \\ \frac{\partial \mathbf{b}_\mathrm{a}}{\partial t} &= -\frac{1}{\tau}\mathbf{b}_\mathrm{a} + \mathbf{w}_{b_a}, \end{aligned} \tag{2.42}$$

where $\mathbf{w}_g$, $\mathbf{w}_a$, $\mathbf{w}_{b_g}$, and $\mathbf{w}_{b_a}$ are uncorrelated zero-mean Gaussian noise processes, ${}_S\tilde{\boldsymbol{\omega}}$ and ${}_S\tilde{\mathbf{a}}$ are the measured rotational velocity and linear acceleration in the IMU frame, ${}_I\mathbf{g}$ is the gravity vector in the world frame (assumed known), and τ is an experimentally chosen time constant.

Using inertial data in a visual-inertial state estimation procedure requires knowledge of the transformation between the IMU and camera frames, as well as parameters for the noise models for the accelerometer and gyroscope. These can be obtained

through the use of open-source calibration software such as Kalibr [Furgale et al., 2013].

2.8 Deep Neural Networks

Deep learning techniques have led to significant increases in performance for many computer vision tasks, such as image classification and semantic segmentation. More relevant to dense SLAM, deep learning algorithms have also produced impressive results for both single- and multi-view depth prediction [Eigen et al., 2014, Ummenhofer et al., 2017, Laina et al., 2016, Liu et al., 2015], and surface normal estimation [Eigen and Fergus, 2015, Ramamonjisoa and Lepetit, 2019].

Deep neural networks (DNNs) are constructed from many biologically-inspired "neurons", each consisting of a weighted summation of inputs that is passed through a nonlinear activation function to produce an output. A DNN, $f_{\boldsymbol{\theta}}(\cdot)$, is used to approximate an unknown function, $f : \mathcal{X} \rightarrow \mathcal{Y}$. This approximation is done by optimising a DNN's weights, $\boldsymbol{\theta}$, to minimise the prediction error over all ($\mathbf{x} \in \mathcal{X}$, $y \in \mathcal{Y}$) pairs in a given training set.

This optimisation is performed using an iterative gradient descent technique. Given the current set of weights, the network makes a prediction, $f_{\boldsymbol{\theta}}(\mathbf{x})$, for each training example, $\mathbf{x}$. Using a method called *backpropagation*, the gradient of the total error with respect to each weight is calculated, and updates are applied to the weights based on this gradient. This is repeated until convergence or until the prediction error on a reserved validation set starts to increase (which suggests *overfitting*). Often it is not computationally feasible to do the minimisation over all pairs at once, and the iterative step is calculated for a small *mini-batch* instead. As the mini-batches are randomly drawn from the full training set, the optimisation procedure resembles stochastic gradient descent which may help avoid local minima [Goodfellow et al., 2016].

The simplest neural networks, called multilayer perceptrons or fully-connected

networks, consist of many neurons organised into layers, where each neuron in a layer is densely connected to all of the neurons in the next layer. Dramatic improvements on many computer vision benchmarks came with the development of convolutional neural networks (CNNs) [LeCun et al., 1998]. Each layer of a CNN learns a small filter that is convolved against the entire input domain. Since the same filter is used throughout the whole layer, the number of weights that need to be estimated is greatly reduced (allowing for bigger networks) and some translational invariance exists between the input and output. Modern networks typically consist of many convolutional layers, with the intuition being that each layer learns increasingly abstract features from the layers below.

One of the most common DNN architectures is an encoder-decoder network with a bottleneck in the middle. With each subsequent layer in the encoder, the spatial resolution of the output is reduced, but the depth of the features is increased, with the minimum spatial resolution at the bottlenck. The decoder reverses this process, decreasing the feature depth but upsampling the input in each layer to produce larger spatial resolutions. While forcing the information through the bottleneck is necessary to have the DNN make use of the global context in the image, fine-grained details are lost. To remedy this, most encoder-decoder networks use some form of *skip connections* that pass feature maps from some of the encoder layers to upsampled decoder layers. A successful example of an encoder-decoder network with skip connections is U-Net [Ronneberger et al., 2015].

DNNs can be applied to both regression and classification tasks. For classification, it is often desirable to have the network produce a probability distribution over the possible classes. This can be enforced by applying the softmax function to the output layer which maps a vector of arbitrary real numbers to be in the interval (0, 1) and sum to 1:

$$\sigma(\mathbf{z})_i = \frac{\exp z_i}{\sum_{j=1}^{n} \exp z_j}. \tag{2.43}$$

Chapter 3

Dense RGB-D Inertial Fusion

Contents

3.1 Introduction

As discussed in Chapter 1, visual SLAM algorithms can be split into two broad categories: sparse landmark-based systems and dense or semi-dense systems. While sparse methods may not directly produce dense maps, pose estimation quality and

robustness of state-of-the-art systems, such as [Mur-Artal et al., 2015] and [Forster et al., 2014], are typically very high. Even higher accuracy and robustness have been attained by the inclusion of inertial measurements in a tightly-coupled fusion. Approaches are formulated either as filters e.g. [Mourikis and Roumeliotis, 2007, Li and Mourikis, 2013, Tanskanen et al., 2015, Bloesch et al., 2015, Sun et al., 2018] or as methods employing iterative minimisation, typically in a sliding window manner, such as [Jones and Soatto, 2011, Keivan et al., 2014, Leutenegger et al., 2014, Forster et al., 2015, Shen et al., 2015, Mur-Artal and Tardos, 2017, Qin et al., 2017, von Stumberg et al., 2018]. Loosely-coupled approaches to visual-inertial fusion, such as [Meier et al., 2011, Weiss et al., 2012, Engel et al., 2012] that separate out either the visual or inertial estimation part have also been proposed. These methods are popular due to their modularity, but disregard correlations in the state estimates, typically leading to lower accuracy and/or robustness.

The maps produced by dense SLAM algorithms offer much more potential for general scene understanding and interaction. As of now, however, vision-only SLAM, and dense SLAM using direct image alignment in particular, suffers from a lack of robustness in the tracking step when initialised too far from the "true" solution; in fact, the tracking optimisation may not converge at all in absence of sufficient texture and/or geometric variation in the depth channel. To address these shortcomings, and inspired by the success of sparse visual-inertial systems, the integration of acceleration and rotation rate measurements into the tracking of a dense SLAM system is advocated. In principle, the tight integration of these complementary sensing modalities should provide robustness in rapid motion, low texture and flat walls. Furthermore, the inclusion of an IMU renders the gravity direction observable, which not only improves map accuracy due to bounded absolute inclination error, but may also be of paramount importance for robot control, most prominently drones.

There have been a few recent examples of dense visual-inertial systems: both [Omari et al., 2015] and [Ma et al., 2015] present loosely-coupled approaches, with the former using the integrated IMU data as a prediction step in a filter to estimate

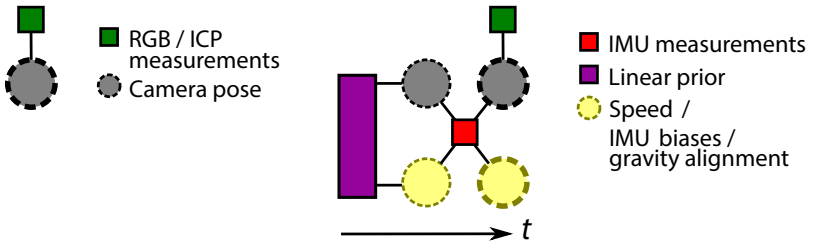
Figure 3.1: Tightly integrating IMU measurements into a dense RGB-D SLAM system leads to more accurate and robust tracking and 3D reconstruction compared with using visual information alone. For example, the scene reconstructed above has several large regions that lack photometric and geometric variation; despite this, the visual-inertial system described in this chapter was able to create a globally consistent reconstruction with flat and aligned walls.

the transformation between image pairs, and the latter fusing relative poses generated by inertial and stereo camera measurements in a manner similar to a pose graph. A tightly-coupled semi-dense monocular visual-inertial odometry system is presented in [Concha et al., 2016]. Unlike other pure monocular odometry systems, it is able to use the inertial data to remove scale ambiguity. Their system uses a semi-dense approach for tracking and, in a separate thread, estimates a fully dense map below frame rate using a piecewise planar prior. Another example of a semi-dense visual-inertial odometry system is described in [Usenko et al., 2016]. This system is implemented within the stereo LSD-SLAM framework [Engel et al., 2015].

Through a series of experiments, they demonstrate that their tightly-coupled approach outperforms both vision-only and loosely-coupled approaches. While their system is closely related to the one described here, this thesis proposes a more map-centric, fully dense approach where the goal is to generate an accurate and complete reconstruction, rather than a camera-centric approach (usually based on pose graph optimisation) that aims to estimate an accurate trajectory. Of course, as previously discussed, the accuracy of the map and the accuracy of the trajectory are closely related. The system described here additionally considers a depth channel and performs map optimisation compliant with gravity alignment.

In this chapter, the RGB-D SLAM system ElasticFusion [Whelan et al., 2016] is extended with tightly-coupled IMU integration, which is capable of more accurate and robust fully dense mapping. Please see Figure 3.1 for an example map output. More specifically, the following contributions are made:

- In the tracking step, the camera pose, velocity, IMU biases and gravity direction from an RGB-D camera and IMU are simultaneously estimated by minimising a joint photometric, geometric, and inertial energy functional.
- Concerning the mapping, the proposed system constructs a globally consistent, fully dense surfel-based 3D reconstruction of the environment. The map is optimised not through a pose graph, but by applying non-rigid space deformations using a sparse deformation graph. An addition to the deformation energy is proposed that ensures consistency with the observable gravity direction.
- Through experiments on both synthetic and real world datasets, the benefits of this approach are demonstrated. It performs well under aggressive motion, fast rotations, and under low texture and geometric variation. Trajectory and map reconstruction accuracy are demonstrated to be higher or on-par with an RGB-D-only ElasticFusion.
- The system maintains real-time capability while running on a GPU. Unlike [Concha et al., 2016] and [Usenko et al., 2016], which achieve real-time



(a) RGB-D-only ElasticFusion (b) RGB-D-inertial ElasticFusion

Figure 3.2: The factor graphs used for tracking optimisation in RGB-D-only and RGB-D-inertial ElasticFusion are compared above. In RGB-D-only ElasticFusion, the current camera pose is optimised against a factor based on the photometric reprojection error with the previous frame and the ICP alignment error with the global model. The inclusion of inertial measurements in RGB-D-inertial ElasticFusion necessitate augmenting the state with speed, IMU biases, and gravity alignment. The temporal nature of IMU measurements requires that both the state of the current frame (variables with thick outlines) and previous frame (variables with thin outlines) be optimised, and for older states to be marginalised and converted into a linear prior.

performance on a CPU, the proposed system constructs a fully dense map at frame rate.

3.2 Coordinate Frames

Four different coordinate frames will be used in this work:

- $\underrightarrow{\mathcal{F}}_W$, the world frame in which the global model is expressed. This frame corresponds with the initial camera frame.
- $\underrightarrow{\mathcal{F}}_I$, the inertial frame that is aligned with gravity and shares an origin with $\underrightarrow{\mathcal{F}}_W$.
- $\underrightarrow{\mathcal{F}}_C$, the camera frame in which the RGB-D data is observed.
- $\underrightarrow{\mathcal{F}}_S$, the sensor frame in which the IMU data is observed.

3.3 System Overview

The system proposed here directly builds upon the vision-only dense RGB-D tracking and mapping approach of ElasticFusion [Whelan et al., 2016]. Like ElasticFusion, tracking and mapping are performed in separate steps. In the tracking step, a joint photometric, geometric and inertial energy functional is constructed. Please see Figure 3.2 for a comparison of the factor-graph representation of the underlying tracking optimisation problem in RGB-D-only and RGB-D-inertial ElasticFusion. Whereas ElasticFusion combined the photometric and geometric terms based on a tuning parameter, λ, here the terms are combined based on the covariances associated with the measurement noise. A nonlinear optimisation formulation is then used to simultaneously estimate the camera pose, velocity, IMU biases and gravity direction. Unlike the original ElasticFusion which only estimated the current camera pose, the proposed system estimates the states associated with both the current and previous camera frames. After the optimisation, the state variables related to the previous frame are marginalised and the remaining variables associated with the current state are used as a prior in the next time step.

In the mapping step, a fully dense surfel-based surface representation is constructed from the camera data and estimated poses obtained from the tracking step. The map is kept globally consistent by applying non-rigid space deformations through a sparse deformation graph. The deformation energy formulation proposed by ElasticFusion is extended to ensure consistency with the observable gravity direction.

3.4 Tracking

3.4.1 States & Local Parameterisation

At the arrival of each new camera frame, the current state, $\mathbf{x}_1$, is estimated while simultaneously refining the previous state, $\mathbf{x}_0$. The system state is comprised of the camera position in the world frame, ${}_W\mathbf{r}_{WC}$, the camera orientation, $\mathbf{q}_{WC}$, the velocity of the IMU in the inertial frame, ${}_I\mathbf{v}_{IS}$, the biases of the gyroscopes, $\mathbf{b}_\mathrm{g}$, and

accelerometers, $\mathbf{b}_\text{a}$, and the orientation of the world frame in the inertial frame, $\mathbf{q}_{IW}$. Therefore the system state, $\mathbf{x}$, for a specific time instance is given by:

$$\mathbf{x} := \begin{bmatrix} {}_W\mathbf{r}_{WC}^T & \mathbf{q}_{WC}^T & {}_I\mathbf{v}_{IS}^T & \mathbf{b}_\text{g}^T & \mathbf{b}_\text{a}^T & \mathbf{q}_{IW}^T \end{bmatrix}^T \in \mathbb{R}^3 \times S^3 \times \mathbb{R}^9 \times S^3. \tag{3.1}$$

While only two degrees of freedom are required to express the gravity direction in the world frame, for simplicity, a 3D implementation with gauge freedom is used. No issues related to this formulation were observed.

The system state exists on a manifold and so is updated by a local perturbation $\delta\mathbf{x}$ in the tangent space through the $\boxplus$ operator, such that $\mathbf{x} = \bar{\mathbf{x}} \boxplus \delta\mathbf{x}$ around a reference $\bar{\mathbf{x}}$. For ${}_W\mathbf{r}_{WC}$, ${}_I\mathbf{v}_{IS}$, $\mathbf{b}_\text{g}$ and $\mathbf{b}_\text{a}$, the $\boxplus$ operator is equivalent to standard vector addition. For $\mathbf{q}_{WC}$ and $\mathbf{q}_{IW}$, a combination of the group operator (quaternion multiplication) and exponential map is used ($\mathbf{q} \boxplus \delta\boldsymbol{\alpha} = \exp_\mathbf{q}(\delta\boldsymbol{\alpha}) \otimes \mathbf{q}$). This results in the following minimal local coordinate representation:

$$\delta\boldsymbol{x} = \begin{bmatrix} \delta\mathbf{r}^T & \delta\boldsymbol{\alpha}^T & \delta\mathbf{v}^T & \delta\mathbf{b}_\text{g}^T & \delta\mathbf{b}_\text{a}^T & \delta\mathbf{g}^T \end{bmatrix}^T \in \mathbb{R}^{18}. \tag{3.2}$$

Similarly, a $\boxminus$ operator can be introduced to compute the difference between two systems states. For regular vector space quantities this corresponds to standard subtraction. For orientations an inverse of the above $\boxplus$ can be constructed ($\mathbf{p} \boxminus \mathbf{q} = \log_\mathbf{q}(\mathbf{p} \otimes \mathbf{q}^{-1})$).

3.4.2 Dense Photometric & Geometric Alignment

The RGB-D subsystem combines dense per-pixel photometric alignment with ICP point-to-plane geometric alignment. The photometric alignment error for a pixel in the current image, ${}_{C_1}\mathbf{u}$, is given by:

$$\begin{aligned} &e_\text{RGB}(\mathbf{x}_0, \mathbf{x}_1, {}_{C_1}\mathbf{u}) = \\ &\quad I^{(0)}(\,\boldsymbol{\pi}(\,\mathbf{K}\,\mathbf{C}_{WC_0}^T\mathbf{C}_{WC_1}D^{(1)}({}_{C_1}\mathbf{u})\,\mathbf{K}^{-1}{}_{C_1}\boldsymbol{u} + \mathbf{C}_{WC_0}^T({}_W\mathbf{r}_{WC_1} - {}_W\mathbf{r}_{WC_0}))) - I^{(1)}({}_{C_1}\mathbf{u}), \end{aligned} \tag{3.3}$$

where $I^{(0)}(\cdot)$ is the intensity of a given pixel in the previous image, $I^{(1)}(\cdot)$ is the intensity of a given pixel in the current image, $\boldsymbol{\pi}(\cdot)$ is the projection function, $\mathbf{C}_{WC} \in$

$SO(3)$ is the rotation matrix representation of $\mathbf{q}_{WC}$, $\mathbf{K}$ is the camera intrinsics matrix, and $D^{(1)}(\cdot)$ is the depth value of a given pixel in the current frame as measured by the depth camera.

The geometric alignment error for a pixel in the current image, ${}_{C_1}\mathbf{u}$, is based on a point-to-plane ICP technique:

$$e_{\text{ICP}}(\mathbf{x}_1, {}_{C_1}\mathbf{u}) = {}_{W}\mathbf{n}_k^T(\mathbf{C}_{WC_1} D^{(1)}({}_{C_1}\mathbf{u})\,\mathbf{K}^{-1}\,{}_{C_1}\boldsymbol{u} + {}_{W}\mathbf{r}_{WC_1} - {}_{W}\mathbf{p}_k). \tag{3.4}$$

where ${}_{W}\mathbf{p}_k$ is the 3D Euclidean point in the global model that corresponds with the backprojected pixel coordinate, and ${}_{W}\mathbf{n}_k$ is the normal vector associated with that point.

3.4.3 Inertial Integration

For the formulation of the IMU measurement error term, the approach of [Leutenegger et al., 2014] is adopted, extending it to include the preintegration technique described by [Forster et al., 2015]. The IMU measurements are integrated numerically between the previous and current camera frames using a linearised and discretised version of the model in Eq. 2.42. The final IMU error term is given by:

$$\mathbf{e}_{\text{IMU}}(\mathbf{x}_0, \mathbf{x}_1) = \hat{\mathbf{x}}_1(\mathbf{x}_0) \boxminus \mathbf{x}_1, \tag{3.5}$$

where $\hat{\mathbf{x}}_1(\cdot)$ is the prediction of the current state made by integrating the IMU measurements onto the previous state, $\mathbf{x}_0$.

3.4.4 Optimisation

The RGB-D-inertial tracking problem is solved using a joint cost function $c_{\text{track}}(\cdot,\cdot)$ that contains the weighted photometric alignment, geometric alignment and inertial

terms:

$$\begin{aligned} c_{\text{track}}(\mathbf{x}_0, \mathbf{x}_1) = & \sum_{{}_{C_1}\mathbf{u} \in \Omega} W_{\text{RGB}}\, e_{\text{RGB}}(\mathbf{x}_0, \mathbf{x}_1, {}_{C_1}\mathbf{u})^2 \\ & + \sum_{{}_{C_1}\mathbf{u}} W_{\text{ICP},{}_{C_1}\mathbf{u}}\, e_{\text{ICP}}(\mathbf{x}_1, {}_{C_1}\mathbf{u})^2 \\ & + \mathbf{e}_{\text{IMU}}(\mathbf{x}_0, \mathbf{x}_1)^T \mathbf{W}_{\text{IMU}}\, \mathbf{e}_{\text{IMU}}(\mathbf{x}_0, \mathbf{x}_1) \\ & + \left(\mathbf{x}_0 \boxminus \bar{\mathbf{x}}_0 - \mathbf{H}^{*-1}\mathbf{b}^*\right)^T \mathbf{H}^* \left(\mathbf{x}_0 \boxminus \bar{\mathbf{x}}_0 - \mathbf{H}^{*-1}\mathbf{b}^*\right), \end{aligned} \tag{3.6}$$

where W_{RGB}, $W_{\text{ICP},{}_{C_1}\mathbf{u}}$, and $\mathbf{W}_{\text{IMU}}$ are the inverse covariance (matrices) associated with the respective measurement uncertainties, Ω is the set of pixels in the current frame that are co-visible with the previous frame, and $\mathbf{H}^*$ and $\mathbf{b}^*$ are priors obtained through the marginalisation step.

The cost function is minimised using a Gauss-Newton iterative method with a three level coarse-to-fine pyramid scheme. After each iteration, the current and previous states are updated using the $\boxplus$ operator.

3.4.5 Partial Marginalisation & Fixation of Variables

The equations for the Gauss-Newton system are constructed from the Jacobians, error terms and information relating to the current and previous states, taking the form:

$$\begin{bmatrix} \mathbf{H}_{00} & \mathbf{H}_{01} \\ \mathbf{H}_{10} & \mathbf{H}_{11} \end{bmatrix} \begin{bmatrix} \delta\mathbf{x}_0 \\ \delta\mathbf{x}_1 \end{bmatrix} = \begin{bmatrix} \mathbf{b}_0 \\ \mathbf{b}_1 \end{bmatrix}. \tag{3.7}$$

After the current and previous states are updated, the previous state is marginalised using the Schur-Complement:

$$\mathbf{H}^*_{11} = \mathbf{H}_{11} - \mathbf{H}_{10}\mathbf{H}^{-1}_{00}\mathbf{H}_{01}, \tag{3.8a}$$

$$\mathbf{b}^*_1 = \mathbf{b}_1 - \mathbf{H}_{10}\mathbf{H}^{-1}_{00}\mathbf{b}_0. \tag{3.8b}$$

The resulting $\mathbf{H}^*$ and $\mathbf{b}^*$ information is used as a prior in the next optimisation step. This partial marginalisation fixes the linearisation point, but with each iteration in the subsequent optimisation scheme, the linearisation point changes. Instead

of relinearising at each step, a first-order correction is applied, $\Delta\mathbf{x}$, based on the difference between the new and old linearisation points as is commonly done in the literature [Leutenegger et al., 2014, Usenko et al., 2016]:

$$\mathbf{H}_{11}^{*'} = \mathbf{H}_{11}^{*}, \tag{3.9a}$$

$$\mathbf{b}_{1}^{*'} = \mathbf{b}_{1}^{*} + \mathbf{H}_{11}^{*}\Delta\mathbf{x}. \tag{3.9b}$$

3.5 Mapping

The mapping system proposed in this chapter is a direct extension of the one proposed in ElasticFusion [Whelan et al., 2016]. As such, the notation used in ElasticFusion to describe the mapping system is adopted. Like the original ElasticFusion, the map is split into *active* and *inactive* areas. The active map is the area most recently observed and is where the tracking and fusing takes place. If a segment of the active map is not observed for a period of time, δ_t, it becomes inactive. The map is kept globally consistent by attempting to match the currently observed portion of the active map with the inactive map. If a match is detected, the loop is closed by applying non-rigid space deformations through a sparse deformation graph.

A deformation graph is a set of nodes, $\mathcal{G}^l \in \mathcal{G}$, that are embedded in the global model, each with a position, $\mathcal{G}^l_{\mathbf{g}}$, and a set of neighboring nodes, $\mathcal{G}^n \in \mathcal{N}(\mathcal{G}^l)$. Each deformation node stores a Euclidean transformation as a rotation, $\mathcal{G}^l_{\mathbf{R}}$, and a translation, $\mathcal{G}^l_{\mathbf{t}}$, that is used to elastically deform surfels in the map from a source position $\mathcal{Q}_{\mathbf{s}}$ to a destination position $\mathcal{Q}_{\mathbf{d}}$ through a deformation function, $\phi(\cdot)$, defined in [Whelan et al., 2016]. This affine transformation is determined by minimising a cost function. In the original ElasticFusion, the cost function consists of five terms. The first encourages rigidity in the deformation:

$$E_{rot} = \sum_{l} \left\| {\mathcal{G}^l_{\mathbf{R}}}^T \mathcal{G}^l_{\mathbf{R}} - \mathbf{1} \right\|_F^2. \tag{3.10}$$

The second encourages smoothness in the deformation:

$$E_{\text{reg}} = \sum_{l} \sum_{n \in \mathcal{N}(\mathcal{G}^l)} \left\| \mathcal{G}^l_{\mathbf{R}}(\mathcal{G}^n_{\mathbf{g}} - \mathcal{G}^l_{\mathbf{g}}) + \mathcal{G}^l_{\mathbf{g}} + \mathcal{G}^l_t - (\mathcal{G}^n_{\mathbf{g}} + \mathcal{G}^n_t) \right\|_2^2. \tag{3.11}$$

The third minimises the distance of each point from the desired deformation:

$$E_{\text{con}} = \sum_{p} \left\| \phi(\mathcal{Q}_s^p) - \mathcal{Q}_d^p \right\|_2^2 . \tag{3.12}$$

The fourth constrains the inactive areas of the map such that the active map is being deformed into the inactive map:

$$E_{\text{pin}} = \sum_{p} \left\| \phi(\mathcal{Q}_d^p) - \mathcal{Q}_d^p \right\|_2^2 . \tag{3.13}$$

The fifth term is only applied to global deformations, and is used to prevent previous registrations, $\mathcal{R}$, from being pulled apart by future global loop closures:

$$E_{\text{rel}} = \sum_{p} \left\| \phi(\mathcal{R}_s^p) - \phi(\mathcal{R}_d^p) \right\|_2^2 . \tag{3.14}$$

As matches between the active and inactive areas of the map are determined only by the RGB-D subsystem, a sixth cost term is included to constrain the graph from deforming the map out of alignment with gravity:

$$E_{\text{imu}} = \sum_{l} \left\| \mathcal{G}_{\mathbf{R}}^l \, {}_W\mathbf{g} - {}_W\mathbf{g}_{\cdot} \right\|_2^2 , \tag{3.15}$$

where ${}_W\mathbf{g}$ denotes the acceleration due to gravity represented in vision-world frame $\underrightarrow{\mathcal{F}}_W$.

Keeping the parameter choices the same as ElasticFusion, the total cost function for local loop closures is given by:

$$E_{\text{loc}} = \omega_f E_{\text{rot}} + \omega_r E_{\text{reg}} + \omega_c (E_{\text{con}} + E_{\text{pin}}) + \omega_i E_{\text{imu}}, \tag{3.16}$$

and the total cost for the global loop closures is given by:

$$E_{\text{glo}} = \omega_f E_{\text{rot}} + \omega_r E_{\text{reg}} + \omega_c (E_{\text{con}} + E_{\text{pin}} + E_{\text{rel}}) + \omega_i E_{\text{imu}}, \tag{3.17}$$

with $\omega_f = 1$, $\omega_r = 10$, and $\omega_c = \omega_i = 100$.

3.6 Experimental Results

The proposed system is evaluated in terms of trajectory estimation and reconstruction accuracy on both synthetic and real world datasets. The living room sequences

of the ICL-NUIM dataset [Handa et al., 2014] are used for the experiments on synthetic data. For the real world experiments, a dataset was recorded along with ground truth poses from a Vicon motion capture system. The synthetic dataset consists of slow, smooth trajectories usually required for dense visual SLAM. The real world dataset contains a mixture of slow trajectories, aggressive motions and sequences with low texture and geometric information where vision-only systems tend to struggle.

Two metrics are considered when examining the performance of the system: the absolute trajectory (ATE) root-mean-square error (RMSE) described in [Sturm et al., 2012], and for reconstruction error, the mean distance from each point in the reconstruction to the nearest surface in the aligned ground truth model. The ATE RMSE is calculated for all sequences, but the reconstruction error is only available for the synthetic dataset. As the behaviour of the loop closure mechanism is non-deterministic, each test is run 10 times and the average result is reported. Tests where either system had lost tracking are denoted by brackets in the tables.

Through these experiments, it is shown that the dense RGB-D-inertial SLAM system performs at least as well as the RGB-D-only system on "easier" trajectories where the problem is well constrained by the visual data alone, but is much more robust when facing sequences with fast motions or little photometric and geometric variation.

3.6.1 Synthetic Data

Both the trajectory estimation and surface reconstruction accuracy of the system are evaluated on a modified version of the living room sequences in the ICL-NUIM dataset. The ICL-NUIM dataset is a benchmark that provides ground truth poses as well as a 3D model with which to evaluate reconstructions of RGB-D SLAM systems. The dataset does not come with inertial data, however, so in a manner similar to [Kerl et al., 2015], splines are fit to the ground truth poses to simulate continuous trajectories. IMU measurements are then generated along these trajectories using

Table 3.1: System Parameters

Noise Parameter	Synthetic Dataset	Real World Dataset	Units
Gyroscope			
Rate	200	200	Hz
Saturation	7.8	7.8	rad s^{-1}
Noise Density	12.0e-4	12.0e-4	rad s^{-1} $Hz^{-0.5}$
Bias Prior	0.03	0.03	rad s^{-1}
Drift Noise Density	4.0e-6	4.0e-6	rad s^{-2} $Hz^{-0.5}$
Accelerometer			
Rate	200	200	Hz
Saturation	176.0	176.0	m s^{-2}
Noise Density	8.0e-3	8.0e-2	m s^{-2} $Hz^{-0.5}$
Bias Prior	0.1	1.0	m s^{-2}
Drift Noise Density	2.0e-5	2.0e-5	m s^{-3} $Hz^{-0.5}$
Static Bias	[0, 0, 0]	[0.060, 0.258, 0.126]	m s^{-2}
Gravitational Acc.	9.81	9.81	m s^{-2}

the model described in [Nikolic et al., 2016] and the noise parameters given in Table 3.1. To make the synthetic dataset as realistic as possible, these noise parameters were chosen to closely match those of the IMU used in real world experiments. Due to the non-smooth trajectories of the dataset, however, it was necessary to sample every 10th frame of the ground truth trajectories when fitting the splines. This resulted in the new ground truth poses being close to but not exactly the same as those in the original dataset. Therefore, the images were rendered at these new poses using POV-Ray and the same noise models were applied to the images as those in the original dataset.

Since the entire ground truth states are known for the synthetic dataset, it is possible to demonstrate that the proposed system can accurately track the velocity and IMU biases. For example, the error in the velocity and bias estimates for Sequence LR0, provided in Fig. 3.3, quickly converges to zero.

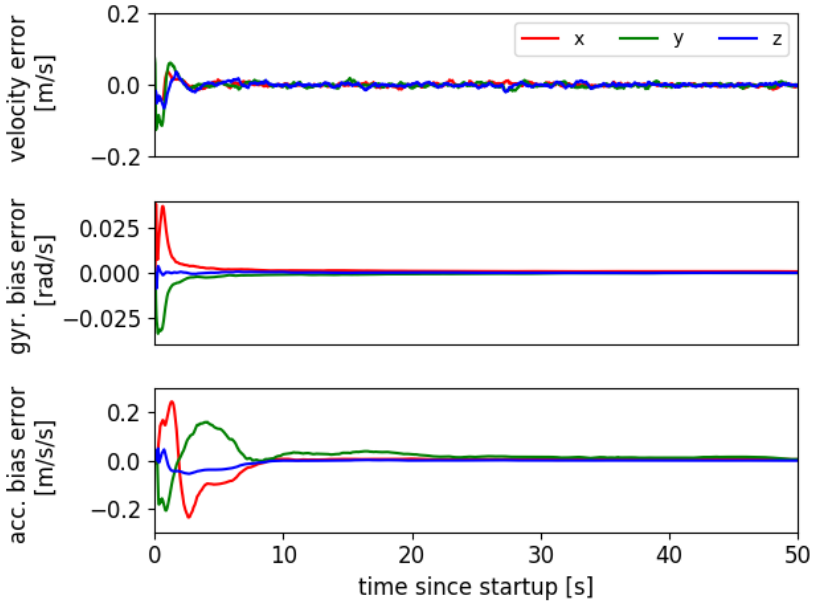


Figure 3.3: Error in the velocity and bias estimates when compared to the ground truth values in synthetic dataset LR0. The proposed system is able to converge to and track the correct values.

Table 3.2: ATE RMSE on the synthetic datasets (brackets indicate a tracking failure)

Sequence	RGB-D-Only	RGB-D-Inertial
LR0	0.032	**0.009**
LR1	**0.009**	0.012
LR2	**0.009**	**0.009**
LR3	(0.906)	**0.019**

The performance of the RGB-D-inertial and RGB-D-only versions of the system are compared on each of the four living room sequences of the modified ICL-NUIM dataset. The results for the ATE RMSE are given in Table 3.2 and for the reconstruction error in Table 3.3.

Although slow, some of the sequences in the modified ICL-NUIM dataset are still difficult for dense SLAM systems to follow, particularly the last sequence. In this sequence, the camera moves slowly along a wall providing little photometric or geo-

Table 3.3: Surface reconstruction accuracy on the synthetic datasets (brackets indicate a tracking failure)

Sequence	RGB-D-Only	RGB-D-Inertial
LR0	0.014	**0.008**
LR1	**0.007**	0.009
LR2	**0.010**	0.011
LR3	(0.118)	**0.010**

metric variation. The results of the original ElasticFusion were not obtained using the same set of internal parameters for each sequence in the ICL-NUIM dataset. However, in this work, to showcase the robustness of the system and to avoid overfitting to a particular sequence, the default set of parameters was used across all datasets. As a result, the RGB-D-only version of ElasticFusion now fails on this sequence. The RGB-D-inertial system, however, is able to use the inertial data to get through the difficult section of the sequence and successfully reconstructs the scene. The RGB-D-inertial system performs approximately as well as the RGB-D-only system on the three easier sequences.

3.6.2 Real World Data

While the synthetic data showed that the RGB-D-inertial system is capable of performing at least as well as the RGB-D-only system on slow, smooth trajectories, the real strength of visual-inertial systems is their robustness to aggressive motions and sequences with little photometric or geometric information. To test this, a new dataset of 21 sequences was collected using the Intel RealSense ZR300 visual-inertial sensor. This sensor captures aligned RGB and depth images as well as inertial measurements. The camera intrinsics, as well as the transformation between the camera and IMU, $\boldsymbol{T}_{CS}$, was obtained using the Kalibr calibration system [Furgale et al., 2013].

To see how the system would perform under different scenarios, a number of different types of datasets were captured. Sequences 1-3 are slow, smooth trajectories

Table 3.4: Comparison of ATE RMSE on the real world datasets (brackets indicate a tracking failure)

Trajectory Type	Sequence	RGB-D-Only	RGB-D-Inertial
slow	1	0.227	**0.066**
	2	0.110	**0.065**
	3	0.225	**0.088**
slow, loop closure	4	0.089	**0.050**
	5	0.106	**0.048**
	6	0.091	**0.051**
medium	7	0.156	**0.077**
	8	0.166	**0.069**
	9	**0.118**	0.124
fast	10	0.098	**0.061**
	11	0.438	**0.354**
	12	0.267	**0.156**
quick rotation	13	0.231	**0.110**
	14	**0.057**	0.063
	15	0.220	**0.064**
low texture	16	(54.238)	**0.682**
	17	(26.306)	**0.498**
	18	(6.536)	(2.141)
long	19	**0.373**	0.560
	20	0.359	**0.216**
	21	0.417	**0.202**

that typical RGB-D SLAM systems could handle. Sequences 4-6 are also slow and smooth, but with a large loop closure. Sequences 7-9 have slightly faster trajectories, and sequences 10-12 have very aggressive trajectories but continue to map the same area, allowing the SLAM system to keep tracking against a previously built up map. Sequences 13-15 are also aggressive trajectories, but include a quick rotation into an unmapped area of the scene. Sequences 16-18 are slow trajectories, but pass close to a white wall such that the RGB-D data provides little photometric or geometric information. Sequences 19-21 are slow, smooth trajectories, but much longer than the other sequences, on the order of 15-20m.

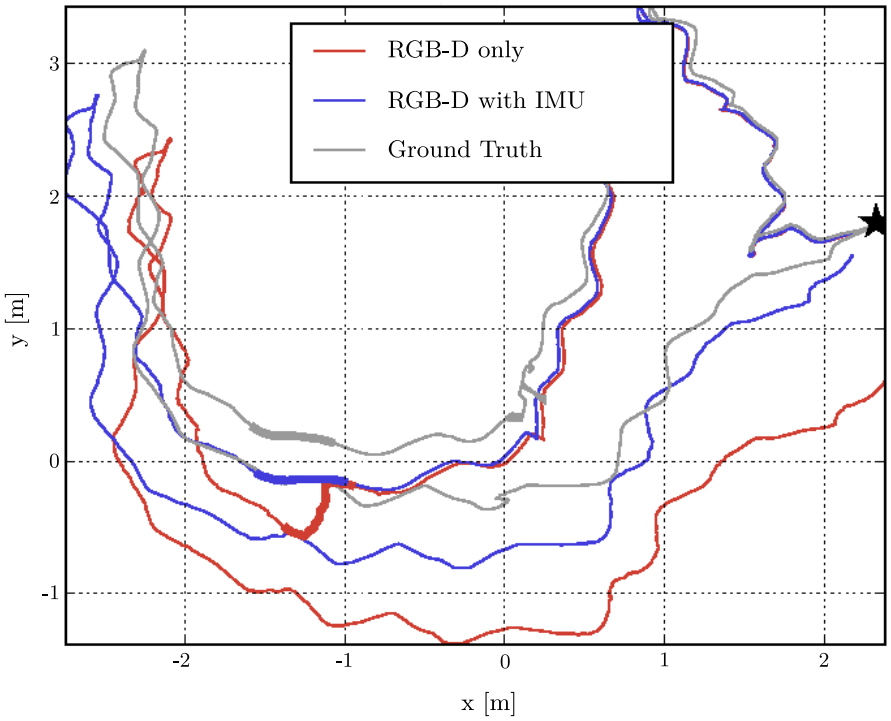


Figure 3.4: Top view of the estimated trajectories for Sequence 21. Integrating the IMU data improves the tracking capabilities of the framework. In particular, it increases robustness against visually degenerate situations which pose a significant problem to the RGB-D-only framework. Such a challenging event, where the majority of the camera's field of view was filled with a white wall, is highlighted by thicker lines in the above trajectories. During this period, the RGB-D-inertial system is able to closely follow the ground truth trajectory, while the RGB-D-only system accumulates significant drift.

For each sequence, a ground truth trajectory was captured using the Vicon motion capture system. As explained in [Sturm et al., 2012], these ground truth poses cannot be used to create a reliable ground truth scene reconstruction through depth image projection, as very small errors in the pose can result in very large errors in the reconstruction. For this reason, the reconstruction error for these sequences is not calculated, only the ATE RMSE.

RGB-D-Only vs. RGB-D-Inertial

For each of the 21 sequences, the performance of the RGB-D-inertial system is compared with the RGB-D-only system. The results of these experiments are presented in Table 3.4. In all but 3 of the sequences the RGB-D-inertial system outperformed the RGB-D-only system, often decisively. In particular, the RGB-D-only system was not capable of tracking the sequences where the camera moves across a white wall. The RGB-D-inertial system is able to rely on the IMU measurements to continue tracking despite the lack of photometric or geometric information, but in the final sequence of that group, the camera moves across the wall for too long and even the RGB-D-inertial system fails. Another example of this occurs in Sequence 21, where halfway through the trajectory the RGB-D-only system struggles when the camera goes across a blank wall. This is visualised in Fig. 3.4.

Qualitatively, a higher degree of map consistency is generally achieved in the RGB-D-inertial system compared with the RGB-D-only system. For example, Fig. 3.5 shows map reconstructions when the system is run on Sequence 7, a moderately difficult sequence in the real world dataset. The top level views show how the inclusion of inertial terms in tracking significantly reduces the amount of drift, as the map is much better aligned for the RGB-D-inertial system. Keeping the map aligned helps ElasticFusion find potential loop closures, but this will still sometimes fail as shown in the pair of images second from the bottom.

Odometry vs. SLAM

To confirm that the proposed formulation of a globally consistent map is improving the trajectory estimation, the performance of the system is compared with an open loop version where the system is restricted to only tracking and fusing against the active map (deformations are not allowed). This comparison is done on a sample sequence from five of the different categories (excluding the quick rotation and low texture scenes due to their difficulty). The results of these tests are shown in Table 3.5. On four out of the five sequences, the closed loop version performed at least as well as the open loop version.

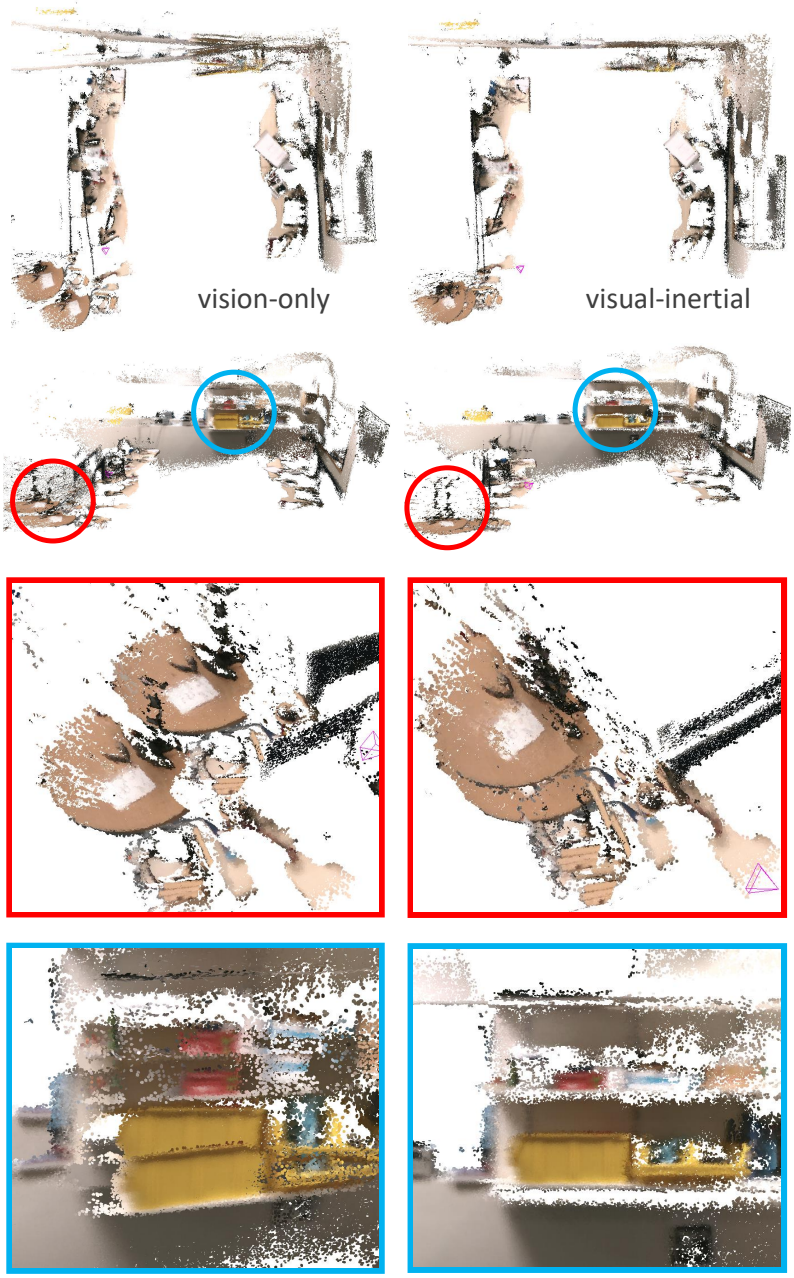


Figure 3.5: Qualitative comparison of map reconstructions in RGB-D-only (left) and RGB-D-inertial (right) ElasticFusion: a higher degree of map consistency is achieved through the inclusion of inertial measurements in the tracking. While loop closure was enabled, the first zoom-in (row second from the bottom) shows that ElasticFusion failed to detect and apply a larger loop closure (in both cases); but it also shows smaller drift as a starting point before potential loop closures. The second zoom-in (bottom row) highlights in more detail the generally higher map consistency with inertial integration.

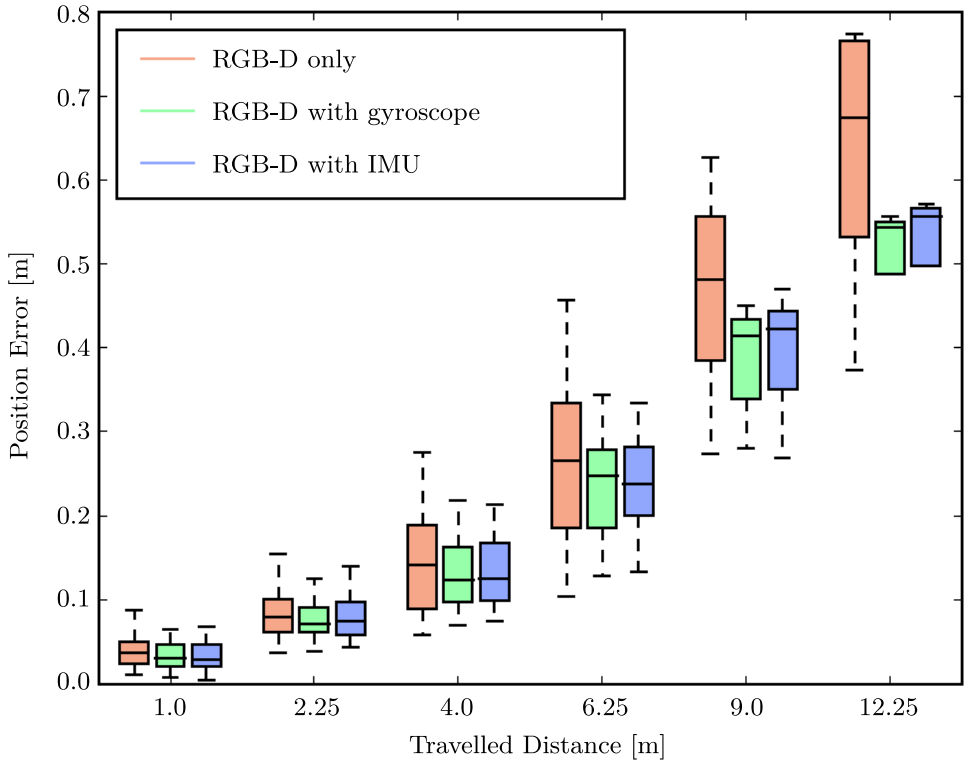


Figure 3.6: Relation between accumulated position error and traveled distance for Sequence 20 with three different setups: no IMU, gyroscope only, and full IMU (gyroscope + accelerometer). Most of the accuracy is gained from the integration of the gyroscopes. While the accelerometers do not significantly improve the accuracy of the system, their integration can contribute to the reliability of the system.

Table 3.5: Comparison of ATE RMSE between open loop odometry and SLAM on the real world datasets

Trajectory Type	**Sequence**	**Odometry**	**SLAM**
slow	1	0.102	**0.066**
slow with loop closure	4	0.051	**0.050**
medium	7	0.078	**0.077**
fast	10	0.062	**0.061**
long	19	**0.541**	0.560

Table 3.6: Comparison of ATE RMSE between RGB-D-inertial and RGB-D with only gyroscopes on the real world datasets

Trajectory Type	Sequence	Gyro Only	Full IMU
long	19	**0.296**	0.560
	20	0.223	**0.216**
	21	0.203	**0.202**
low texture	16	(7.548)	**0.682**
	17	(3.916)	**0.498**
	18	(0.917)	(2.141)

Drift Analysis

In order to examine the relative contributions of the gyroscopes and accelerometers, the system is tested on a number of sequences where the accelerometer related residuals are ignored. Fig. 3.6 shows the position error as a function of the distance travelled for Sequence 20, comparing RGB-D-only to RGB-D-and-gyroscopes-only to the full RGB-D-inertial system. As this figure shows, most of the gain in accuracy comes from the gyroscopes. This is confirmed by the results for the long sequences in Table 3.6. Over such a long sequence, the gyroscopes-only setup can outperform the full IMU due to the high noise levels of the accelerometers. The necessity of the accelerometers, however, is shown by the low texture sequences in Table 3.6. As the camera passes over the white wall, the gyroscope-only system fails because without visual input the relative position is no longer constrained.

3.7 Conclusion

A real-time tightly-coupled dense RGB-D-inertial SLAM system has been presented in this chapter. In the tracking step, a combined photometric, geometric and inertial energy functional is minimised to simultaneously estimate the camera pose, velocity, IMU biases and gravity direction. In the mapping step, the system constructs a fully dense 3D reconstruction of the environment which is not only globally consistent, but gravity aligned due to the addition of an inertial deformation energy applied to

the deformation graph. It is shown through a series of experiments on both synthetic and real world datasets that this RGB-D-inertial system is often more accurate than the RGB-D-only version on slow, smooth trajectories, and is much more robust to aggressive motions and a lack of photometric and geometric variation.

This increase in accuracy and robustness is supported by similar findings in recent work on sparse visual-inertial SLAM. However, these gains do not come without a cost. Adding an IMU to a system means a much more complicated calibration procedure, for example. While Kalibr [Furgale et al., 2013] was helpful for estimating the needed calibration parameters, generating the calibration sequences that had sufficient motion around each axis while keeping the calibration pattern in view and not saturating the accelerometer or gyroscope was difficult, and research towards a calibration system capable of providing real-time feedback to the user would be beneficial. Kalibr also assumes that the timestamps assigned to each camera image and inertial reading were generated by the same clock, so only a static time offset needs to be estimated. While this was true for the system presented in this chapter, most visual-inertial setups will consist of separate cameras and IMUs, each with their own clocks, requiring the drift between them to be estimated as well. As these costs are not negligible, it is important to think about the environment in which a potential system will be deployed. If a system is only expected to operate under slow and smooth motions across reasonably textured scenes, and if the cost of a tracking failure is relatively low, the extra engineering effort required for inertial fusion may not be worth it.

The decision on whether or not to include inertial data does not have to be a binary one. Most of the calibration issues centred around the accelerometer, while most of the gain in accuracy and robustness came from the gyroscope. Unless a lot of low texture scenes are expected, it may be preferable to only fuse gyroscope data into the vision system. Also, using a higher quality IMU would make for an easier calibration because with less noise, less data is required to get good parameter estimates.

While these are some of the limitations of visual-inertial SLAM, some other possible benefits remain to be explored. For example, even during tracking loss, there will be some measure of inclination from the accelerometer. This measurement could be used to help constrain the optimisation during camera relocalisation.

Chapter 4

Probabilistic Fusion of Learned Parametric Priors

Contents

4.1 Introduction

There has been continued research interest in using visual SLAM for the incremental creation of dense 3D scene geometry due to its potential applications in safe robotic

navigation, augmented reality and manipulation. Until recently, most dense monocular reconstruction systems typically estimate depth by minimising the photometric error over several frames. As this minimisation problem is not well-constrained due to occlusion boundaries or regions of low texture, most reconstruction systems employ regularisers based on smoothness [Newcombe et al., 2011b, Pizzoli et al., 2014] or planar assumptions [Concha et al., 2014, Concha and Civera, 2015, Concha et al., 2016].

Monocular reconstruction systems also suffer from an inherent scale ambiguity, as it is not possible to determine a camera's translation from image correspondences alone. As was shown in the previous chapter, it is possible to address this scale ambiguity by fusing vision-based measurements with readings from an inertial measurement unit (IMU); however, in situations with low accelerations (for example, when the camera is still), scale becomes practically unobservable when using low grade IMUs. One option to resolve these issues in monocular dense reconstruction is through the use of a depth camera (the approach taken by [Newcombe et al., 2011a] and [Kerl et al., 2013]), but depth cameras have limited range, consume more power, and do not typically work outdoors or in strong sunlight.

With the continued success of deep learning in computer vision, there have been many suggestions for data-driven approaches to the monocular reconstruction problem. Several of these approaches propose a completely end-to-end framework, predicting the scene geometry from either a single image [Eigen et al., 2014, Laina et al., 2016, Godard et al., 2017, Fu et al., 2018] or several consecutive frames [Ummenhofer et al., 2017, Zhou et al., 2017, Mahjourian et al., 2018, Zhou et al., 2018, Yao et al., 2018, Chang and Chen, 2018]. Most promising, however, are those systems that combine deep learning with standard geometric constraints [Weerasekera et al., 2017, Tateno et al., 2017, Yang et al., 2018, Bloesch et al., 2018, Wang et al., 2018, Tang and Tan, 2019]. It was shown in [Fácil et al., 2017] that learning-based and geometry-based approaches have a complementary nature as learning-based systems tend to perform better on the interior points of objects but blur edges, whereas geometry-based systems typically do well on areas with a high image gradient but

Figure 4.1: Fusing the depth predictions of a semi-dense multi-view stereo system with the depth and gradient predictions of a CNN allows for the creation of fully dense depth maps at scale. The above projected keyframe depth map was created by DeepFusion from only pose estimates and RGB images. Despite a lack of photometric texture on major scene components (such as the monitor and desk), which makes it difficult to obtain good multi-view stereo depth estimates, the use of learned priors allows for an accurate, dense 3D reconstruction.

perform poorly on interior points that may lack texture.

The optimal way to combine these two approaches, however, is not clear. The best current results seem to come from systems that take the output of traditional geometry-based systems and feed these into a deep neural network (DNN). A particularly impressive example of this type of system is DeepTAM [Zhou et al., 2018], which passes a photometric cost volume through a network to extract a depth map.

One problem with using a network as the final stage in a reconstruction pipeline is that an expensive network pass must be computed every time the underlying geo-

metric information is updated, which may be unacceptably expensive for real-time incremental systems. Another problem is that this framework breaks the probabilistic formulation of typical dense SLAM pipelines, which may be necessary for the fusion of additional sensors such as IMUs or wheel encoders. For these reasons, a number of approaches that take network depth predictions and refine them with geometric constraints have been proposed. In [Fácil et al., 2017], the authors compute a network depth prediction for each keyframe and update a semi-dense multi-view stereo depth map with each new frame. The two depth estimates are then interpolated based on a set of tunable weights related to the image structure. Another approach [Zhang and Funkhouser, 2018] predicts surface normals and occlusion boundaries for each keyframe image and then attempts to fill in missing values in the output of a depth camera. Unfortunately, the solve time is too slow to be used on incremental SLAM systems. In [Yin et al., 2017], a CRF is used to refine the regression results. In [Weerasekera et al., 2018], which greatly inspired the work in this chapter, the authors were able to create accurate dense reconstructions using a fully-connected CRF with network predicted uncertainties from a sparse set of 3D points generated by a monocular SLAM system. Since they use the output of a monocular system to constrain the dense reconstructions, however, the resulting depth maps are ambiguous in scale. In this chapter, the idea of maintaining global consistency by linking neighbouring pixels in the reconstruction through depth gradient predictions is used and extended to include the estimation of absolute scale. Perhaps the most comparable work to that presented here is CNN-SLAM [Tateno et al., 2017], which uses a network to predict an at-scale depth map for each keyframe and then refines it through small baseline stereo constraints in real time. The refinement in CNN-SLAM, however, is done on a per-pixel basis and therefore does not preserve global consistency.

This chapter proposes DeepFusion, a 3D reconstruction system that is capable of producing dense depth maps at scale in real time from RGB images and scale-ambiguous poses provided by a monocular SLAM system. Network predicted depth gradients are used as a constraint on neighbouring pixels to ensure global consistency

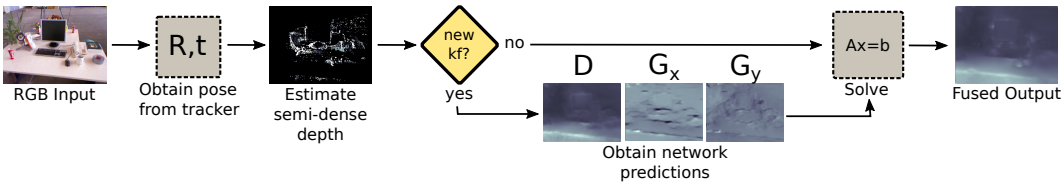


Figure 4.2: The DeepFusion framework.

in the reconstructions, and learned uncertainties to fuse the different modalities in a probabilistic fashion. Please see Figure 4.1 for an example keyframe reconstruction.

4.2 Method

In this section, a system for producing dense reconstructions at scale in real time is described. Please see Figure 4.2 for an overview of the DeepFusion framework.

DeepFusion represents the observed geometry with a series of keyframe depth maps. With each new RGB image, the system obtains the pose from a monocular SLAM system (ORB-SLAM2 [Mur-Artal and Tardós, 2017] in this implementation) and then updates the semi-dense depth estimates for the active keyframe using the method described in [Engel et al., 2013]. If the camera has translated more than λ_{trans} or had fewer than $\lambda_{inliers}$ inliers in the semi-dense estimation, a new keyframe is created.

To maintain a high frame rate, the network outputs are only generated once per keyframe. Using a CNN, the log-depth, log-depth gradients and associated uncertainties are predicted from the new keyframe image. Like [Weerasekera et al., 2018] and [Eigen et al., 2014], log-depths are predicted instead of depths or inverse depths because it is numerically better for network prediction (negative values are meaningful) and it has the convenient property that the difference between two log-depths (the gradient of the log-depth image) is the ratio of two depths, which is scale-invariant. The choice is made to predict log-depth gradients in both the x- and y-directions on the image plane rather than surface normals in order to maintain the linearity of the optimisation problem, as this avoids the need for performing

dot product and normalisation operations. Single-view depth prediction is a highly under-constrained problem, and in practice it seems easier for the network to make accurate predictions about fine-grained local geometry rather than absolute per-pixel depth. For this reason, the absolute log-depth values and the log-depth gradients are predicted separately to obtain separate uncertainties that reflect the difference in the network's ability at these two different tasks.

If a new keyframe is not created, then the current semi-dense depth map and network outputs are fused to update the current depth map.

4.2.1 Network Architecture

For the network, a U-Net [Ronneberger et al., 2015] style architecture is used with the same dimensions as the one in [Bloesch et al., 2018], except that three more identical decoders are added to predict log-depth uncertainties, log-depth gradients, and log-depth gradient uncertainties in addition to just the log-depths. All inputs and outputs have a resolution of 256x192. See Figure 4.3.

In order to fuse the outputs of the network with the estimates coming from the semi-dense multi-view stereo system, an uncertainty associated with each pixel in each of the log-depth and gradient images is required. To obtain this, the method described in [Kendall and Gal, 2017] is used to have the network learn to predict both a mean and a variance with a maximum likelihood cost function:

$$c_{ML}(\boldsymbol{\theta}) = \sum_{\mathbf{u}} \frac{(\mathbf{y}(\mathbf{u}) - f_{\boldsymbol{\theta},\mathbf{u}}(\mathbf{x}))^2}{\sigma_{\boldsymbol{\theta},\mathbf{u}}(\mathbf{x})^2} + \log(\sigma_{\boldsymbol{\theta},\mathbf{u}}(\mathbf{x})^2), \tag{4.1}$$

where $\boldsymbol{\theta}$ is the set of network weights, $\mathbf{x}$ is the input image, $\mathbf{y}(\cdot)$ is the ground truth label for a given pixel, and $f_{\boldsymbol{\theta},\mathbf{u}}(\mathbf{x})$ and $\sigma_{\boldsymbol{\theta},\mathbf{u}}(\mathbf{x})^2$ are the network's predictions for the mean and variance of pixel $\mathbf{u}$, respectively. The total loss is the sum of this loss function for each of the output images.

Like [Bloesch et al., 2018], the network is trained on the SceneNet RGB-D dataset [McCormac et al., 2017], a dataset of 5 million rendered indoor scenes. The network is trained to predict log-depths that have been normalised by the focal length of the

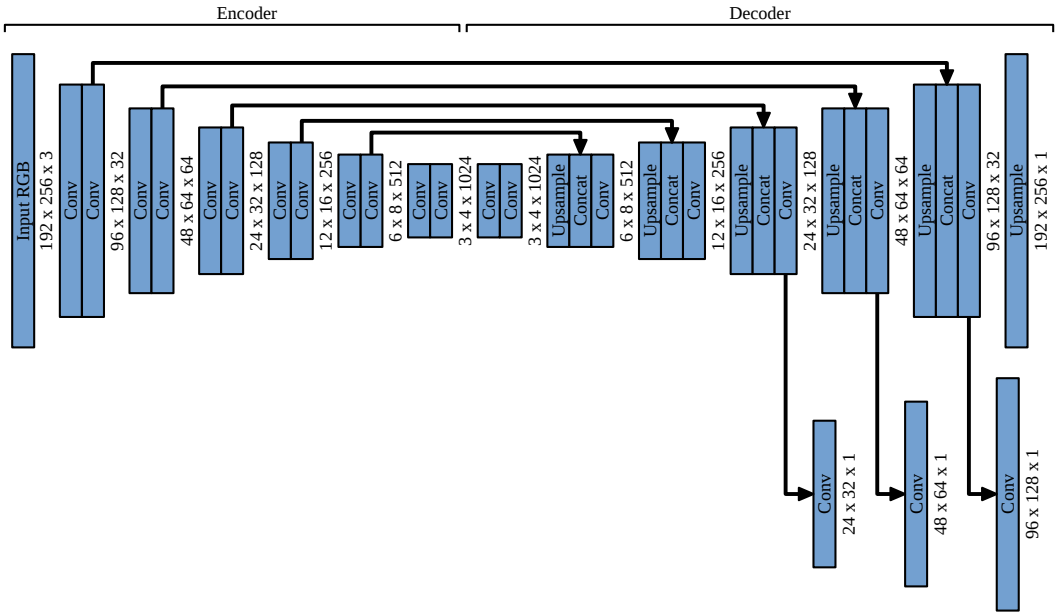


Figure 4.3: The network consists of a U-Net-style encoder and decoder with five skip connections. During training, outputs are extracted at four different resolutions with losses calculated for each of them. While a single encoder is shared across all tasks, a separate decoder is used for each of the network predictions (depth, depth uncertainty, depth gradients and depth gradient uncertainty).

SceneNet camera. In the same manner as CNN-SLAM [Tateno et al., 2017], the log-depth predictions of the network are scaled by the focal length of the camera used at test time so the absolute scale can be recovered for images captured by cameras with different intrinsics. Since all predictions are done in logspace, the gradients (which represent depth ratios) and uncertainties do not need to be scaled.

4.2.2 Semi-Dense Estimation

For the semi-dense multi-view stereo component, the depth estimation method from [Engel et al., 2013] is implemented. For each pixel in the keyframe where there is sufficient texture, a search is made along the epipolar line for the depth value, $D^{(0)}_{\text{semi}}({}_{C_0}\mathbf{u})$ that minimises the sum of squared differences for five equally spaced points. Note that $D^{(0)}_{\text{semi}}$ only has valid depth values at pixels corresponding to a high image gradient. If there is a current depth estimate for pixel ${}_{C_0}\mathbf{u}$, the search is conducted over the interval $D^{(0)}_{\text{semi}}({}_{C_0}\mathbf{u}) \pm 2\sigma_{\text{semi},{}_{C_0}\mathbf{u}}$. Otherwise, the search

is conducted over the entire epipolar line. Given a pixel in the keyframe, ${}_{C_0}\mathbf{u}$, and the poses of the keyframe $({}_W\mathbf{r}_{WC_0}, \mathbf{C}_{WC_0})$ and reference frame $({}_W\mathbf{r}_{WC_1}, \mathbf{C}_{WC_1})$, the photometric error is given by:

$$e(D^{(0)}_{\text{semi}}, {}_{C_0}\mathbf{u}) = I^{(1)}(\boldsymbol{\pi}(\mathbf{K}\,\mathbf{C}^T_{WC_1}\mathbf{C}_{WC_0}D^{(0)}_{\text{semi}}({}_{C_0}\mathbf{u})\mathbf{K}^{-1}{}_{C_0}\boldsymbol{u} + \mathbf{C}^T_{WC_1}({}_W\mathbf{r}_{WC_1} - {}_W\mathbf{r}_{WC_0}))) - I^{(0)}({}_{C_0}\mathbf{u}), \tag{4.2}$$

where $I^{(0)}(\cdot)$ is the intensity of a given pixel in the keyframe, $I^{(1)}(\cdot)$ is the intensity of a given pixel in the reference frame, $\boldsymbol{\pi}(\cdot)$ is the projection function, and $\mathbf{K}$ is the camera intrinsics matrix.

The Jacobian of the error function, $\mathbf{J}_{{}_{C_0}\mathbf{u}}$, is approximated with finite differences:

$$\mathbf{J}_{{}_{C_0}\mathbf{u}} \approx \frac{1}{\Delta D^{(0)}_{\text{semi}}({}_{C_0}\mathbf{u})}\Delta\mathbf{e}_{{}_{C_0}\mathbf{u}} \tag{4.3}$$

where $\mathbf{e}_{{}_{C_0}\mathbf{u}}$ is the 5x1 vector containing the photometric error associated with each of the five points. When searching for the minimum value along the epipolar line, even sized steps of 1 pixel length are taken. Once the minimum is found, the optimal depth at sub-pixel resolution is found by interpolating between two steps. The difference in the photometric error at the two endpoints of the interpolation is $\Delta\mathbf{e}_{{}_{C_0}\mathbf{u}}$, and the difference between the depths at those points is $\Delta D^{(0)}_{\text{semi}}({}_{C_0}\mathbf{u})$.

The uncertainty of each semi-dense measurement is approximated by:

$$\sigma^2_{{}_{C_0}\mathbf{u}} = (\mathbf{J}^T_{{}_{C_0}\mathbf{u}}\mathbf{J}_{{}_{C_0}\mathbf{u}})^{-1}. \tag{4.4}$$

The semi-dense depth estimates and uncertainties are then converted into logspace to match the network outputs.

4.2.3 Optimisation

To update the current depth prediction, the following cost function consisting of three terms is minimised with each new frame:

$$c(D^{(0)}, s) = c_{\text{semi}}(D^{(0)}, s) + c_{\text{net}}(D^{(0)}) + c_{\text{grad}}(D^{(0)}), \tag{4.5}$$

where $D^{(0)}$ is the set of log-depth values to be estimated and s is the scale correction factor.

The semi-dense cost term imposes a unary constraint over the set of pixels where valid semi-dense log-depth values have been estimated:

$$\begin{aligned} c_{\text{semi}}(D^{(0)}, s) &= \sum_{{}_{C_0}\mathbf{u} \in \Omega_{\text{semi}}} \frac{1}{\sigma^2_{{}_{C_0}\mathbf{u}}} e_{\text{semi}}(D^{(0)}, s, {}_{C_0}\mathbf{u})^2 \\ e_{\text{semi}}(D^{(0)}, s, {}_{C_0}\mathbf{u}) &= \log D^{(0)}({}_{C_0}\mathbf{u}) - \log s - \log D^{(0)}_{\text{semi}}({}_{C_0}\mathbf{u}), \end{aligned} \tag{4.6}$$

where $\sigma^2_{{}_{C_0}\mathbf{u}}$ is the uncertainty estimated by the approximation given in Eq. 4.4 and $D^{(0)}_{\text{semi}}(\cdot)$ is the scale-ambiguous log-depth predicted by the semi-dense system for a given pixel. Since the semi-dense log-depth estimates are calculated based on the poses provided by a monocular SLAM system, they have an arbitrary scale. Because the depth map to be estimated, $D^{(0)}$, is to scale, a scale correction factor, s is needed to solve. With the fully dense per-pixel cost term, $c_{\text{net}}(\cdot)$, this scale correction factor becomes observable.

The network depth cost term imposes an additional unary constraint over all pixels of the fused depth map:

$$\begin{aligned} c_{\text{net}}(D^{(0)}) &= \sum_{{}_{C_0}\mathbf{u} \in \Omega} \frac{1}{\sigma^2_{\boldsymbol{\theta},\text{depth},{}_{C_0}\mathbf{u}}(\mathbf{x})} e_{\text{net}}(D^{(0)}, {}_{C_0}\mathbf{u})^2 \\ e_{\text{net}}(D^{(0)}, {}_{C_0}\mathbf{u}) &= \log D^{(0)}({}_{C_0}\mathbf{u}) - f_{\boldsymbol{\theta},\text{depth},{}_{C_0}\mathbf{u}}(\mathbf{x}) \end{aligned} \tag{4.7}$$

where $f_{\boldsymbol{\theta},\text{depth},{}_{C_0}\mathbf{u}}(\mathbf{x})$ and $\sigma^2_{\boldsymbol{\theta},\text{depth},{}_{\mathbf{u}}}(\mathbf{x})$ are the log-depth and uncertainty predictions made by the network (see Eq. 4.1). While the network may have a high uncertainty about the absolute depth at any pixel, this cost term provides a weak prior on the absolute scale of the scene and allows for the estimation of the scale correction factor, s.

In order to maintain global consistency while fusing together the semi-dense and network depth values, an additional cost term that imposes pairwise constraints

between a given pixel and each of its four neighbours is included:

$$
\begin{aligned}
c(D^{(0)}) &= \sum_{C_0\mathbf{u}\in\Omega} \left[\frac{e_{\text{grad,x}}(D^{(0)}, C_0\mathbf{u})^2}{\sigma^2_{\boldsymbol{\theta},\text{grad,x},C_0\mathbf{u}}(\mathbf{x})} + \frac{e_{\text{grad,y}}(D^{(0)}, C_0\mathbf{u})^2}{\sigma^2_{\boldsymbol{\theta},\text{grad,y},C_0\mathbf{u}}(\mathbf{x})} \right] \\
e_{\text{grad,x}}(D^{(0)}, C_0\mathbf{u}) &= \log D^{(0)}(C_0\mathbf{u}_E) - \log D^{(0)}(C_0\mathbf{u}) - f_{\boldsymbol{\theta},\text{grad,x},C_0\mathbf{u}}(\mathbf{x}) \\
e_{\text{grad,y}}(D^{(0)}, C_0\mathbf{u}) &= \log D^{(0)}(C_0\mathbf{u}_S) - \log D^{(0)}(C_0\mathbf{u}) - f_{\boldsymbol{\theta},\text{grad,y},C_0\mathbf{u}}(\mathbf{x})
\end{aligned}
\tag{4.8}
$$

where $\mathbf{u}_E$ is the neighbouring pixel of $\mathbf{u}$ in the positive horizontal direction ("East"), $\mathbf{u}_S$ is the neighbouring pixel of $\mathbf{u}_S$ in the positive vertical direction ("South"), $f_{\boldsymbol{\theta},\text{grad,x},C_0\mathbf{u}}(\mathbf{x})$ and $f_{\boldsymbol{\theta},\text{grad,y},C_0\mathbf{u}}(\mathbf{x})$ are the log-depth gradients in the x- and y-directions predicted by the network, and $\sigma^2_{\boldsymbol{\theta},\text{grad,x},C_0\mathbf{u}}(\mathbf{x})$ and $\sigma^2_{\boldsymbol{\theta},\text{grad,y},C_0\mathbf{u}}(\mathbf{x})$, the associated predicted uncertainties (see Eq. 4.1).

The semi-dense depth estimates can be very noisy and the network predictions can have extreme outliers which have a significant impact on the final reconstruction. For this reason, the Huber loss function on each of the cost terms is used.

The system is solved using the Opt [DeVito et al., 2017] optimisation framework. With Opt, an energy function is defined for each term in the cost function, which are then automatically compiled into GPU optimisation kernels. Ten Gauss-Newton iterations are performed, alternating between solving for the depth and scale. With this setup, it is possible to solve the system for each new frame in real time.

4.3 Experimental Results

4.3.1 Qualitative Results

Figure 4.4 shows some qualitative results from selected keyframes on the ICL-NUIM [Handa et al., 2014] and TUM RGB-D [Sturm et al., 2012] datasets. Comparing the network depth predictions with the final fused depth maps shows that including geometric constraints from the semi-dense depth estimation and pairwise pixel constraints from the gradient predictions produces depth maps that are more globally consistent and have fewer blurring artifacts.

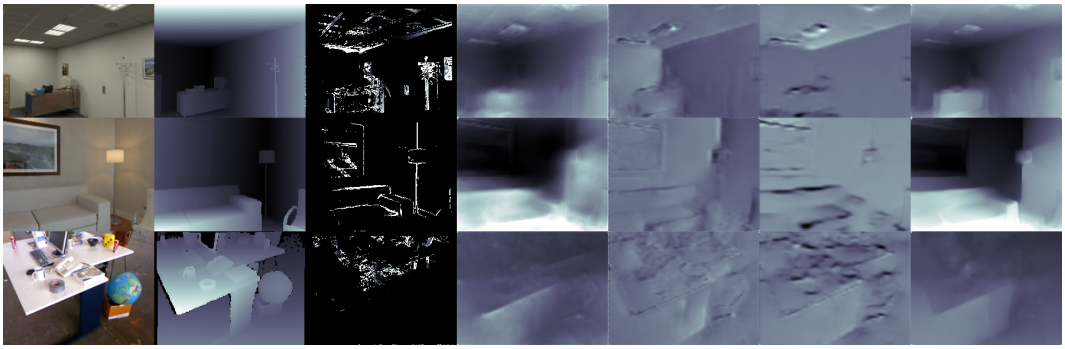


Figure 4.4: Qualitative results for selected keyframes on the ICL-NUIM Office2 (top), ICL-NUIM LivingRoom1 (middle) and TUM RGB-D fr2_desk (bottom) sequences. From left to right: input image, ground truth depth, semi-dense depth estimate, network depth prediction, network depth gradient prediction in the x-direction, network depth gradient prediction in the y-direction, and the optimised depth map.

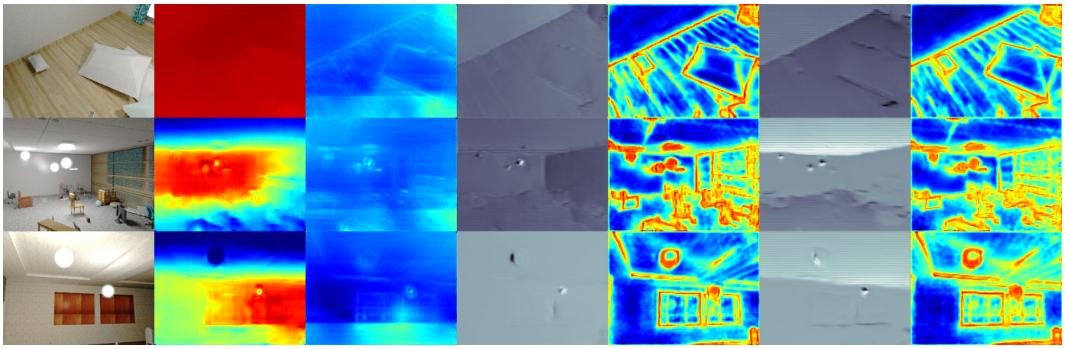


Figure 4.5: Example network predictions on sample images from the SceneNet RGB-D dataset. From left to right: input image, log-depth prediction, log-depth uncertainty prediction, log-depth gradient prediction in the x-direction, log-depth gradient in the x-direction uncertainty prediction, log-depth gradient prediction in the y-direction, log-depth gradient in the y-direction uncertainty prediction.

Figure 4.5 shows the network output for sample images in the SceneNet RGB-D dataset [McCormac et al., 2017]. The uncertainties associated with the gradient images are clearly largest on areas of high image gradient suggesting that the network has learned that these regions tend to correspond with depth discontinuities or other rapid changes in the depth gradient.

Table 4.1: Comparison of reconstruction accuracy in terms of percentage of correct depth values (within 10% of ground truth) on ICL-NUIM and TUM RGB-D datasets (TUM/seq1: *fr3/long_office_household*, TUM/seq2: *fr3_nostructure_texture_near_withloop*, TUM/seq3: *fr3/structure_texture_far*). LSD-SLAM (BS) is LSD-SLAM bootstrapped with a ground truth depth map, and REMODE uses LSD-SLAM (BS) poses and keyframes.

Sequence	DeepFusion	CNN-SLAM	LSD-SLAM (BS)	LSD-SLAM	ORB-SLAM	Laina	REMODE
ICL/office0	**21.090**	19.410	0.603	0.335	0.018	17.194	4.479
ICL/office1	**37.420**	29.150	4.759	0.038	0.023	20.838	3.132
ICL/office2	30.180	**37.226**	1.435	0.078	0.040	30.639	16.708
ICL/living0	**24.223**	12.840	1.443	0.360	0.027	15.008	4.479
ICL/living1	**14.001**	13.038	3.030	0.057	0.021	11.449	2.427
ICL/living2	25.235	26.560	1.807	0.167	0.014	**33.010**	8.681
TUM/seq1	8.069	12.477	3.797	0.086	0.031	**12.982**	9.548
TUM/seq2	14.774	**24.077**	3.966	0.882	0.059	15.412	12.651
TUM/seq3	27.200	**27.396**	6.449	0.035	0.027	9.450	6.739
Avg.	**22.466**	22.464	3.032	0.226	0.029	18.452	7.649

4.3.2 Reconstruction Evaluation

The reconstruction pipeline is evaluated by comparing it to the results obtained by CNN-SLAM [Tateno et al., 2017], a state-of-the-art system that fuses together network predictions and geometric constraints to produce fully dense depth maps. Following the evaluation procedure of CNN-SLAM, the results are also compared to the depth maps produced by a sparse feature-based monocular system (ORB-SLAM2 [Mur-Artal and Tardós, 2017]), a semi-dense geometric system (LSD-SLAM [Engel et al., 2014]), a fully dense reconstruction method (REMODE [Pizzoli et al., 2014]), and a pure deep learning approach (Laina, et al. [Laina et al., 2016]).

As there was no open-source version of CNN-SLAM available at the time of writing, DeepFusion is evaluated on the same sequences used in [Tateno et al., 2017] and compared with their reported results. The sequences used for the comparison come from two different datasets: the synthetic ICL-NUIM RGB-D dataset [Handa et al., 2014] and the real world TUM RGB-D SLAM dataset [Sturm et al., 2012]. The ICL-NUIM dataset provides rendered depth maps as a ground truth comparison and the TUM RGB-D dataset approximates this with Kinect depth camera images.

As proposed by [Tateno et al., 2017], the percentage of estimated depth values

that are within 10% of the corresponding ground truth depth values are measured in order to evaluate both the reconstruction accuracy and density.

The results are presented in Table 4.1. While DeepFusion and CNN-SLAM have approximately the same performance overall, this performance is not evenly distributed over the two datasets. DeepFusion performs the best on four out of the six ICL-NUIM sequences, whereas CNN-SLAM performs the best on two out of the three TUM RGB-D sequences. The reason for this most likely has to do with the training data used by the two systems. The network is trained on the synthetic SceneNet dataset [McCormac et al., 2017], whereas CNN-SLAM uses the network described in [Laina et al., 2016] which is trained on the Kinect-captured NYUv2 [Silberman et al., 2012]. In addition, two of the TUM RGB-D sequences used in the comparison consist of the camera moving over flat planes covered in posters, with seemingly little semantic information for the networks to leverage. These results speak to the importance of keeping the network's training data as close as possible to the domain in which the system will be deployed. While both DeepFusion and CNN-SLAM clearly outperform the geometry-only systems, the learning-based method proposed by Laina, et al. [Laina et al., 2016] also does well, performing the best on two of the nine sequences. This demonstrates the power of learning-based approaches, but ultimately the systems that can make use of both modalities perform the best overall.

4.3.3 Scale Optimisation Evaluation

While the network predicted log-depth values are included in the optimisation problem to solve for the absolute scale of the reconstruction, there are alternative methods. For instance, in [Weerasekera et al., 2017], the authors first predict a scale-ambiguous reconstruction and then scale their reconstruction by finding a least-squares fit with a network predicted depth map. The advantage of the proposed method is that it is able to use the relative uncertainties between the semi-dense estimates and network predictions when performing the fusion.

Table 4.2: Comparison of scale estimation methods in DeepFusion in terms of percentage of correct depth values (within 10% of ground truth) on ICL-NUIM and TUM RGB-D datasets (TUM/seq1: *fr3/long_office_household*, TUM/seq2: *fr3_nostructure_texture_near_withloop*, TUM/seq3: *fr3/structure_texture_far*). "Least Squares for Scale Estimation" shows the results when using least squares to align a scale-ambiguous estimation with the network depth prediction to estimate a scaled depth map for each keyframe.

Sequence	DeepFusion	Least Squares for Scale Estimation
ICL/office0	**21.090**	18.135
ICL/office1	**37.420**	26.415
ICL/office2	30.180	**31.359**
ICL/living0	**24.223**	23.861
ICL/living1	**14.001**	10.372
ICL/living2	**25.235**	22.082
TUM/seq1	8.069	**9.690**
TUM/seq2	**14.774**	14.490
TUM/seq3	**27.200**	24.047

To demonstrate that the proposed method produces better results, a version of DeepFusion is implemented that optimises for a scale-ambiguous depth map using only the semi-dense and gradient terms of the cost function and then finds a least-squares fit with the network predicted depth and measured its performance on sequences used for the reconstruction evaluation. The results are presented in Table 4.2. In seven of the nine sequences, DeepFusion outperforms the least-squares post-processing method.

4.3.4 Global Consistency Evaluation

One of the primary differences between DeepFusion and CNN-SLAM [Tateno et al., 2017] is that DeepFusion includes a pairwise constraint on neighbouring pixels in order to enforce global consistency on the optimised depth maps.

To show that including these constraints does in fact help produce more accurate reconstructions, a version of DeepFusion is implemented that optimises for a depth

Table 4.3: Analysis on the importance of pairwise constraints on reconstruction accuracy in terms of percentage of correct depth values (within 10% of ground truth) on ICL-NUIM and TUM RGB-D datasets (TUM/seq1: *fr3/long_office_household*, TUM/seq2: *fr3_nostructure_texture_near_withloop*, TUM/seq3: *fr3/structure_texture_far*). "No Pairwise Constraints" shows the results when fusing together only the semi-dense and network log-depth values.

Sequence	DeepFusion	No Pairwise Constraints
ICL/office0	**21.090**	16.641
ICL/office1	**37.420**	24.633
ICL/office2	30.180	**30.899**
ICL/living0	**24.223**	21.643
ICL/living1	**14.001**	12.774
ICL/living2	**25.235**	21.772
TUM/seq1	8.069	**9.469**
TUM/seq2	**14.774**	14.187
TUM/seq3	**27.200**	23.584

map with absolute scale using only the network predicted log-depths, the semi-dense log-depth estimates, and their associated uncertainties. The performance of this version is evaluated on the sequences used in the reconstruction evaluation. The results are presented in Table 4.3. In seven of the nine sequences, DeepFusion outperforms the method that does not enforce global consistency.

4.3.5 Timing Evaluation

To demonstrate the real-time capability of the proposed system, the approximate runtimes of each of the components are shown in Table 4.4. These runtimes were based on an implementation using an Intel Core i7-5820K CPU and a GeForce GTX 980 GPU.

4.4 Conclusion

This chapter has presented DeepFusion, a system capable of producing dense 3D reconstructions at scale in real time. By formulating a cost function that includes

Table 4.4: Approximate timing information for key components in the DeepFusion system.

	Semi-Dense	**Optimisation**	**Network Prediction**
Mean	16ms	33ms	45ms
Min	1ms	26ms	44ms
Max	43ms	47ms	47ms

per-pixel losses based on network depth predictions and sparse semi-dense depth estimates with pairwise constraints from network depth gradient predictions, it is possible to estimate both the shape of the observed scene and its absolute scale. By predicting both the per-pixel mean and variance, it is possible to obtain uncertainties for all network outputs and fuse the outputs together with the geometric constraints in a probabilistic fashion. Through a series of experiments on synthetic and real datasets, it is demonstrated that DeepFusion performs at least as well as other comparable systems. Furthermore, through a series of ablation studies, the value of estimating the scale of the depth maps by including the network depth output as a per-pixel constraint in the optimisation and using pairwise constraints to enforce global consistency is demonstrated.

A weakness of the approach presented in this chapter is that it forces the network to predict a parametric and unimodal depth distribution which may be ill-suited to dense reconstruction where there can be a clear need for a multi-hypothesis prediction. Additionally, multi-view geometric constraints were only estimated for keyframe pixels with a high image gradient. Unfortunately, these points also tend to be where there are high depth gradients (as they often correspond with edges of objects), and this is where the DNN has learned to predict high uncertainty. In practice, this means that very little information is passed between semi-dense points and points on the interior of objects. Therefore, the depth values of these interior points rely almost entirely on the depth values predicted by the network and do not benefit from the fusion.

The next chapter will attempt to address some of these issues by training a DNN to predict a nonparametric probability distribution for the depth of all pixels.

Chapter 5

Probabilistic Fusion of Learned Nonparametric Priors

Contents

5.1 Introduction

As discussed in the previous chapter, there has been a lot of recent research on data-driven approaches to the depth estimation problem, and the best current results

Figure 5.1: Fusing a single-view nonparametric depth probability distribution predicted by a DNN with standard photometric error terms helps to resolve ambiguities in the photometric error due to occlusions or lack of texture. The above projected keyframe depth map was created by the system. Despite a lack of photometric texture on the desk, walls and computer monitor, the system is able to create an accurate 3D reconstruction through the use of the data-based priors.

seem to come from systems that take the output of traditional geometry-based pipelines and feed these into a DNN to produce a final output (e.g. DeepTAM [Zhou et al., 2018]). It may be desirable, however, to treat learning-based systems as an additional component that is fused into the pipeline of a traditional system. Such a framework would keep the probabilistic formulation intact and prevent the necessity of having to perform an expensive neural network pass every time the geometric information is updated. Also, as DNNs perform best on images close to the training dataset, it would be possible to switch the network component on or off or switch between different networks depending on the environment being reconstructed. The difficulty of this approach, however, is that to probabilistically fuse the network outputs into a 3D reconstruction system, some measure of the uncertainty associated

with each prediction is required.

In general, uncertainty can be classified into two categories: model or epistemic uncertainty and statistical or aleatoric uncertainty. In [Gal and Ghahramani, 2016], the authors suggest using a Monte Carlo dropout technique to estimate the model uncertainty of a network, but this requires multiple expensive network passes.

Like [Bishop, 1994], the authors of [Kendall and Gal, 2017] propose having the network predict its own aleatoric uncertainty and using a Gaussian or Laplacian likelihood as the loss function during training, which was the method used in the previous chapter. The problem with this approach is that it forces the network to predict a parametric and unimodal distribution. As shown in [Campbell et al., 2008], this type of distribution may be particularly ill-suited to dense reconstruction where there may be a clear need for a multi-hypothesis prediction.

One proposal has been to use a multi-headed network with each head making a separate prediction [Zhou et al., 2018, Peretroukhin et al., 2019]. From these many predictions, one can calculate the mean and covariance to use in a probabilistic fusion algorithm. The drawbacks of this approach are that it increases the size of the network and requires a careful balancing of the relative size of the network body and heads.

Recently, both [Fu et al., 2018] and [Liu et al., 2019] achieved impressive results by having their networks predict discrete, nonparametric probability distributions. While [Liu et al., 2019] uses these distributions to fuse the output with other network predictions, this method has not been used to fuse the predictions of networks with the output of standard reconstruction pipelines.

In this chapter, a dense mapping system is proposed that fuses together the output of a DNN with a standard photometric cost volume to create dense depth maps for a set of keyframes. A network is trained to predict a discrete, nonparametric probability distribution for the depth of each pixel over a given range from a single image. Following [Liu et al., 2019], this collection of probability distributions for each pixel

in the keyframe is referred to as a *probability volume*. Then, with each subsequent frame, a probability volume based on the photometric consistency between the current frame and the keyframe image is created and fused into the keyframe volume. It is argued that combining the probability volumes from these two sources will result in a better conditioned probability volume. Depth maps are extracted from the probability volume by optimising a cost function that includes a regularisation term based on network predicted surface normals and occlusion boundaries. Please see Figure 5.1 for an example keyframe reconstruction created by this system.

5.2 Method

The proposed dense mapping system represents the observed geometry as a collection of keyframe-based *probability volumes*. That is, instead of representing the surface as a depth map with a single depth estimate per pixel, the depth is represented with a per-pixel discrete probability distribution over a given depth range. These probability volumes are initialised with the output of a monocular depth prediction network. With each additional RGB image, the system computes a cost volume based on the photometric consistency. This cost volume is then converted to a probability volume and fused into the volume of the current keyframe. Once the number of inliers drops below a given threshold, a new keyframe is created. To propagate information from one keyframe to another, the previous distribution is warped and fused into the new one.

When extracting a depth map from the probability volume, it would be possible to take the depth values with the highest probability, but in featureless regions where there is also high network uncertainty this would be susceptible to false minima and cause local inconsistencies in the prediction. Also, as the probability distribution is discrete, taking the maximum would result in a quantisation of the final depth prediction. To overcome these shortcomings, a smooth probability density function (PDF) is first constructed from the volume using a kernel density estimation (KDE) technique. Then negative log probability of this PDF along with a regularisation

term is minimised. While many dense systems propose using regularisers based on smoothness [Newcombe et al., 2011b, Pizzoli et al., 2014] or planar assumptions [Concha et al., 2014, Concha and Civera, 2014, Concha and Civera, 2015], the examples of [Weerasekera et al., 2017] and [Laidlow et al., 2019] are followed, and the reconstruction is penalised for deviating from the surface normals predicted by a DNN.

5.2.1 Multi-Hypothesis Monocular Depth Prediction

Rather than predict a single depth value for each pixel, the network predicts a discrete depth probability distribution over a given range, similar to [Fu et al., 2018] and [Liu et al., 2019]. Not only does this allow the network to express uncertainty about its prediction, but it also allows the network to make a multi-hypothesis depth prediction. As discussed in [Fu et al., 2018], the prediction of the depth probability distribution can be improved by having a variable resolution over the depth range. A log-depth parameterisation is chosen, following the examples of [Weerasekera et al., 2018] and [Eigen et al., 2014]. By uniformly dividing the depth range in log-space, the desired result of having higher resolution in the areas close to the camera and lower resolution farther away is achieved.

For the network architecture (see Figure 5.2), a ResNet-50 encoder is used [He et al., 2016], followed by three upsample blocks, each consisting of a bilinear upsampling layer, a concatenation with the input image, and then two convolutional layers to bring the output back up to the input resolution. All inputs and outputs have a resolution of 256x192.

As the network predicts a discrete distribution rather than a depth map, it is not possible to use a standard loss function based on the sum of squared errors. A cross-correlation loss would not be ideal either, as it would not penalise the network less for predicting high probabilities in incorrect bins that are close to the true bin than in bins farther away. Instead,the ordinal loss function proposed in [Fu et al.,

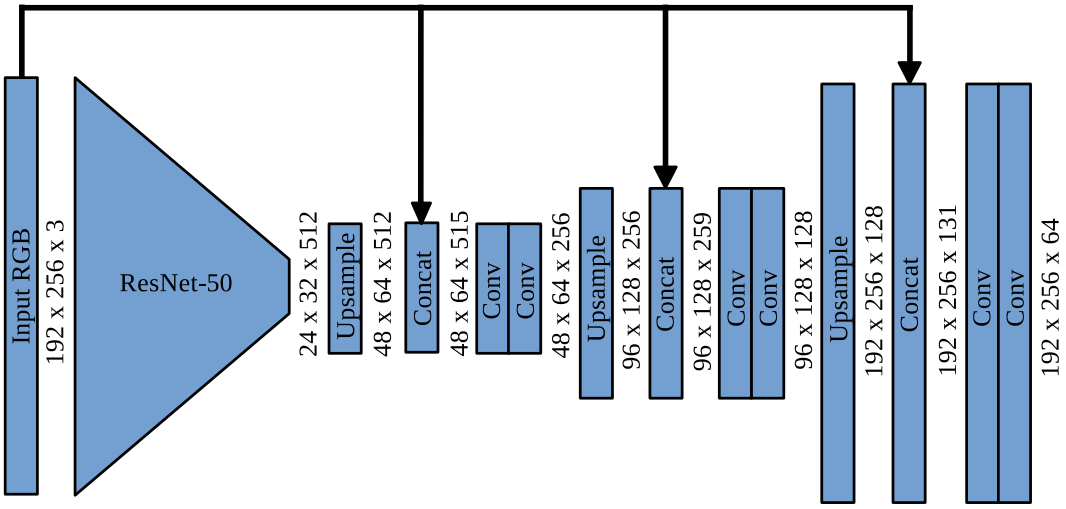


Figure 5.2: The network consists of a ResNet-50 encoder with an output stride size of 8 and no global pooling layer.The output of the encoder is then passed through three upsample blocks consisting of a bilinear resize, concatenation with the input image, and then two convolutional layers to match the output resolution to the input. The probability distribution that the network outputs is discretised over 64 channels.

2018] is chosen:

$$c(\boldsymbol{\theta}) = -\sum_i \left[\sum_{k=0}^{k_i^*} \log(p_{\boldsymbol{\theta},i}(k_i^* \geq k)) + \sum_{k=k_i^*+1}^{K-1} \log(1 - p_{\boldsymbol{\theta},i}(k_i^* \geq k)) \right], \tag{5.1}$$

where

$$p_{\boldsymbol{\theta},i}(k_i^* \geq k) = \sum_{j=k}^{K-1} p_{\boldsymbol{\theta},i}(k_i^* = j), \tag{5.2}$$

$\boldsymbol{\theta}$ is the set of network weights, K is the number of bins over which the depth range is discretised, k_i^* is the index of the bin containing the ground truth depth for pixel i, and $p_{\boldsymbol{\theta},i}(k_i^* = j)$ is the network prediction of the probability that the ground truth depth is in bin j.

Like [Liu et al., 2019], the network is trained on the ScanNet RGB-D dataset [Dai et al., 2017]. The depth range is set to be between 10cm and 12m and the log-depth values are grouped uniformly into 64 bins.

Each keyframe created by the system is initialised with this network output.

5.2.2 Fusion with Photometric Error Terms

For each additional reference frame, a DTAM-style cost volume is constructed [Newcombe et al., 2011b]. First, both the keyframe and reference frame images are normalised by subtracting their means and dividing by their standard deviations. Then the photometric error is calculated by warping the normalised keyframe image into the reference frame for each depth value in the cost volume and taking the sum of squared differences on 3x3 patches. To simplify the later fusion, the midpoint of each of the depth bins used for the network prediction are used as the depth values in the cost volume. Poses are obtained from an oracle, such as a separate tracking system like ORB-SLAM2 [Mur-Artal and Tardós, 2017].

To convert to a probability volume, the negative of the photometric error is separately scaled for each pixel such that it sums to one over the ray. This new probability volume is then fused into the current keyframe volume:

$$p_i(k_i^* = k) = p_{\mathrm{KF},i}(k_i^* = k) p_{\mathrm{RF},i}(k_i^* = k), \tag{5.3}$$

for each pixel i, which is then scaled to sum to one.

5.2.3 Kernel Density Estimation

To avoid a quantisation of the final depth prediction and to have a smooth function to use in the optimisation step, a PDF is constructed for the depth of each pixel, $\mathbf{u}$, using a KDE technique with Gaussian basis functions:

$$f_{\mathbf{u}}(d) = \sum_{k=0}^{K-1} p_{\mathbf{u}}(k_{\mathbf{u}}^* = k)\, \phi\left(d(k), \sigma\right), \tag{5.4}$$

where $\phi(\mu, \sigma)$ is the probability density of the Gaussian distribution with mean μ and standard deviation σ, $d(k)$ is the depth value at the midpoint of bin k, and σ is a constant smoothing parameter across all pixels and depth values. The value of σ is a hyperparameter that needs to be tuned empirically; it was found that $\sigma = 0.1$ worked well on the test cases.

An example of a discrete PDF produced by the system and the smoothed result after applying the KDE technique is shown in Figure 5.3.

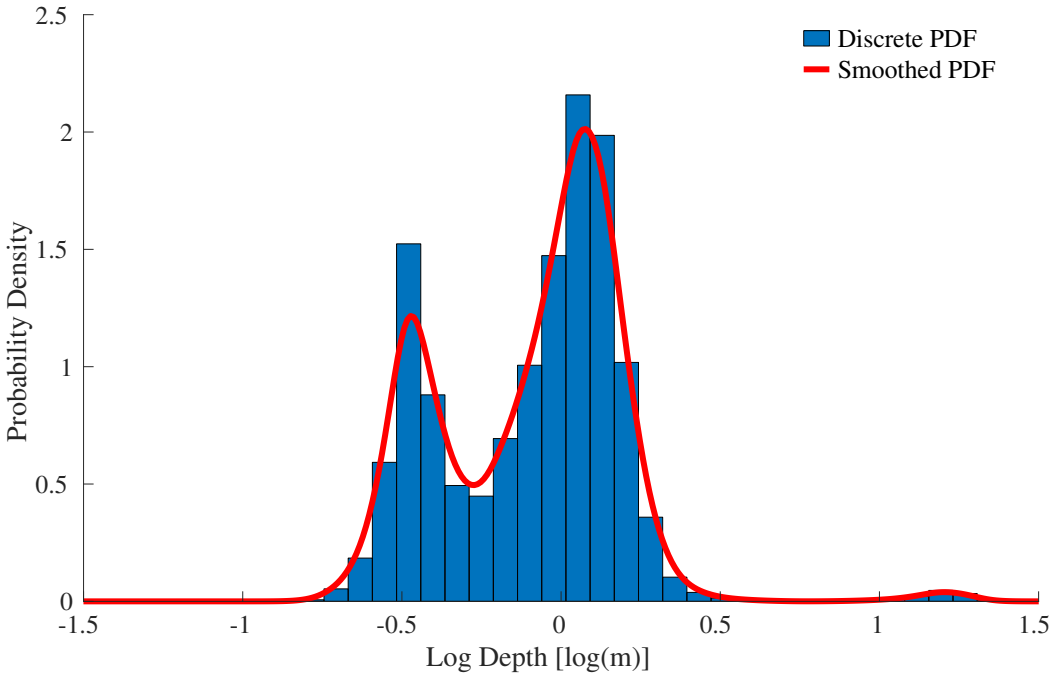


Figure 5.3: The fusion algorithm produces a discrete probability distribution for each pixel in the keyframe. To reduce discretisation errors and to have a continuous cost function for the optimiser, the probability values along each ray are converted into a smooth probability density function using a kernel density estimation technique.

5.2.4 Regularisation

Although the fused probability volume will have more local consistency than using the photoconsistency terms alone, the result can still be improved by adding a regularisation term to the optimisation used to extract the depth map. While most dense reconstruction systems base their regularisers on smoothness or planar assumptions, the proposed system uses the surface normals predicted by a DNN as was done in both [Weerasekera et al., 2017] and [Laidlow et al., 2019] as this may allow for better preservation of fine-grained local geometry. To predict the surface normals from the keyframe image, the state-of-the-art network SharpNet is used [Ramamonjisoa and Lepetit, 2019]. As the local surface orientation of the depth estimation is determined from neighbouring pixels and it is not desirable to incur high costs at depth discontinuities, the regularisation term is masked at occlusion boundaries, which are also predicted by SharpNet. Since SharpNet actually predicts a probability of each

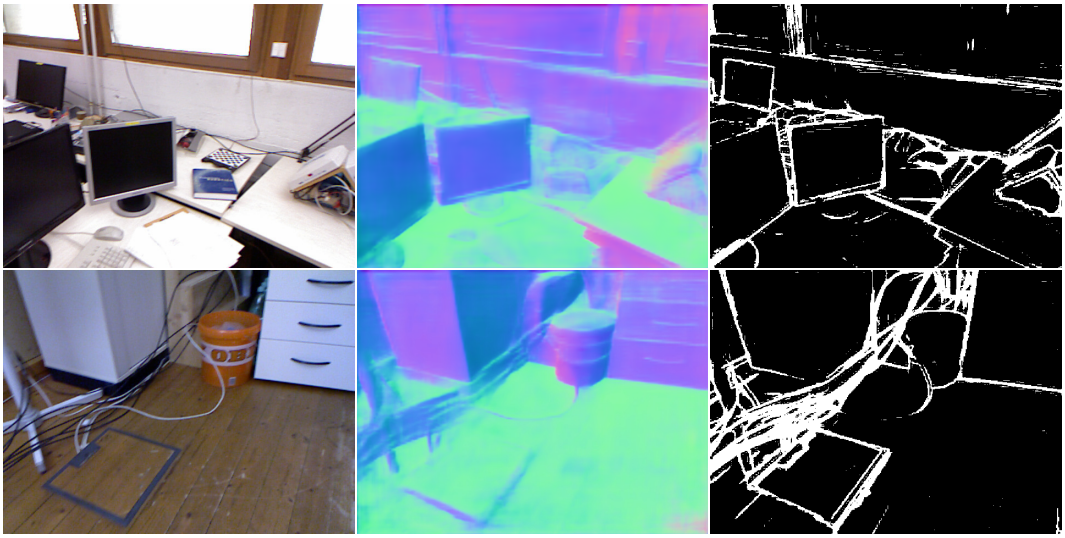


Figure 5.4: To regularise the depth estimate, the surface normals and occlusion boundaries predicted by SharpNet are used [Ramamonjisoa and Lepetit, 2019]. Some examples of the predictions made by SharpNet on the TUM RGB-D dataset [Sturm et al., 2012] are shown above. From left to right: input RGB images, predicted normals, predicted occlusion boundaries with a probability greater than 0.4.

pixel belonging to an occlusion boundary, all pixels with a probability higher than 0.4 are included in the mask. Example predictions of surface normals and occlusion boundaries made by SharpNet on the TUM RGB-D dataset [Sturm et al., 2012] are shown in Figure 5.4.

5.2.5 Optimisation

To extract a depth map from the probability volume, a cost function consisting of two terms is minimised:

$$c(D) = c_f(D) + \lambda c_{\mathbf{n}}(D), \tag{5.5}$$

where D is the depth map to be estimated, and λ is a hyperparameter used to adjust the strength of the regularisation term. Empirically, a value of $\lambda = 1.0 \cdot 10^7$ was found to work well.

The first term, c_f, imposes a unary constraint on each of the pixels:

$$c_f(D) = -\sum_{\mathbf{u}\in\Omega} \log\left(f_{\mathbf{u}}(d = D(\mathbf{u}))\right) \tag{5.6}$$

where $f_{\mathbf{u}}(d = D(\mathbf{u}))$ is the smoothed PDF of pixel $\mathbf{u}$ evaluated at depth $D(\mathbf{u})$.

The second term, $c_{\mathbf{n}}(\cdot)$, is a regularisation term that combines two pairwise constraints:

$$\begin{aligned} c_{\mathbf{n}}(D) = \sum_{\mathbf{u}\in\Omega} & b_{\mathbf{u}} \left(\langle \mathbf{n}_{\mathbf{u}}, D(\mathbf{u})\mathbf{K}^{-1}\boldsymbol{u} - D(\mathbf{u}_E)\mathbf{K}^{-1}\boldsymbol{u}_E \rangle \right)^2 \\ & + b_{\mathbf{u}} \left(\langle \mathbf{n}_{\mathbf{u}}, D(\mathbf{u})\mathbf{K}^{-1}\boldsymbol{u} - D(\mathbf{u}_S)\mathbf{K}^{-1}\boldsymbol{u}_S \rangle \right)^2, \end{aligned} \tag{5.7}$$

where $\mathbf{u}_E$ is the neighbouring pixel of $\mathbf{u}$ in the positive horizontal direction ("East"), $\mathbf{u}_S$ is the neighbouring pixel of $\mathbf{u}$ in the positive vertical direction ("South"), $b_{\mathbf{u}} \in \{0, 1\}$ is the value of the occlusion boundary mask for pixel $\mathbf{u}$, $\langle \cdot, \cdot \rangle$ is the dot product operator, $\mathbf{n}_{\mathbf{u}}$ is the normal vector predicted by SharpNet for pixel $\mathbf{u}$, and $\mathbf{K}$ is the camera intrinsics matrix.

The cost function is minimised by applying 100 iterations of gradient descent with a step size of 0.2, and the optimisation is initialised with the maximum probability depth values from the fused probability volume. The process of going from a fused probability volume through the smoothing and optimisation to an extracted depth map is currently only able to run at a few Hz; however, this could be improved significantly by using Newton's method or the primal-dual algorithm. As the main contribution of this work is to show the benefit of the fusion, this is left for future work.

5.2.6 Keyframe Warping

To avoid throwing away information on the creation of each new keyframe, the probability volume of the current keyframe is warped into the new one. As the probability volume is a distribution over the depth values of a pixel, however, warping the probability volume is not trivial. To do this, the proposed system uses a discrete variation of the method described in [Loop et al., 2016], where the depth probability distribution is first converted to an occupancy-based probability volume, where for each depth bin along the ray there is a probability that the associated point in space is occupied. This occupancy grid is then warped into the new frame and converted back to a depth probability distribution.

The probability that the voxel $S_{k,\mathbf{u}}$ (corresponding to depth bin k along the ray of pixel $\mathbf{u}$) is occupied, conditioned on the depth belonging to bin j:

$$p(S_{k,\mathbf{u}} = 1 | k^*_{\mathbf{u}} = j) = \begin{cases} 0 & \text{if } k < j \\ 1 & \text{if } k = j \\ \frac{1}{2} & \text{if } k > j \end{cases}. \tag{5.8}$$

To convert a depth probability distribution into an occupancy probability, the conditional is marginalised out:

$$p(S_{k,\mathbf{u}} = 1) = \sum_{k=0}^{K-1} p_{\mathbf{u}}(k^*_{\mathbf{u}} = k) p(S_{k,\mathbf{u}} = 1 | k^*_{\mathbf{u}} = k), \tag{5.9}$$

where $p_{\mathbf{u}}(k^*_{\mathbf{u}} = k)$ is the value of the depth probability volume in bin k for pixel $\mathbf{u}$.

As the occupancy grid represents probabilities for locations in 3D space, it can be directly warped into the new keyframe, filling in any unknown values with a default occupancy probability. In this work, a default probability of 0.01 is used.

After warping, the occupancy grid can be converted back into a depth probability distribution:

$$p_{\mathbf{u}}(k^*_{\mathbf{u}} = k) = \prod_{j<k} \left[1 - p(S_{j,\mathbf{u}} = 1)\right] p(S_{k,\mathbf{u}} = 1), \tag{5.10}$$

and scaled so that the distribution sums to one along the ray.

5.3 Experimental Results

The system is evaluated on the Freiburg 1 sequences of the TUM RGB-D dataset [Sturm et al., 2012]. Please note that only the RGB images are processed by the system and the depth channel is only used as a "ground truth" with which to validate the results against.

5.3.1 Qualitative Results

Figure 5.5 shows the various PDFs for a sample of four pixels taken from a keyframe in the TUM RGB-D sequence *fr1/desk*. The PDFs in the first row are those predicted

by the DNN. Note that the network is able to make multi-hypothesis predictions and can have varying degrees of certainty. The PDFs in the second row are those that result from the photometric cost volume. For some of the pixels (such as pixels A and C), the photometric error results in a clear peak. This situation is most often found on corners and edges in the image where there are large intensity gradients. For pixels in textureless regions or on occlusion boundaries or areas with repeating patterns, the photometric PDF may have many peaks (such as pixel B) or no peak at all (such as Pixel D). The final row of the figure shows the fused PDF for each of the pixels. By fusing the two PDFs together, uncertainty can be reduced and ambiguous photometric data can be resolved.

An example reconstruction for a single keyframe with various ablations is shown in Figure 5.6.

5.3.2 Quantitative Evaluation

The value of fusion on the reconstruction pipeline is demonstrated by comparing the performance of the system on each of the Freiburg 1 TUM RGB-D sequences under three different scenarios: using only the network probability volume, using only the photometric probability volume and using the fused probability volume. To isolate the performance of the reconstruction system, the ground truth poses provided in the dataset are used. The performance is evaluated using three metrics defined in [Eigen et al., 2014]: the absolute relative difference (L1-rel), the squared relative difference (L2-rel) and the root mean squared error (RMSE). Note that since the photometric probability volume has extremely noisy results on textureless surfaces, it was found that the results were improved by initialising the optimisation with the expected value of the depth from the probability volume rather than the highest probability depth.

The results are presented in Table 5.1. In six of the sequences, the best result is achieved by fusing together the network and photometric probability volumes. While there is a large performance gain in using the network over the photomet-

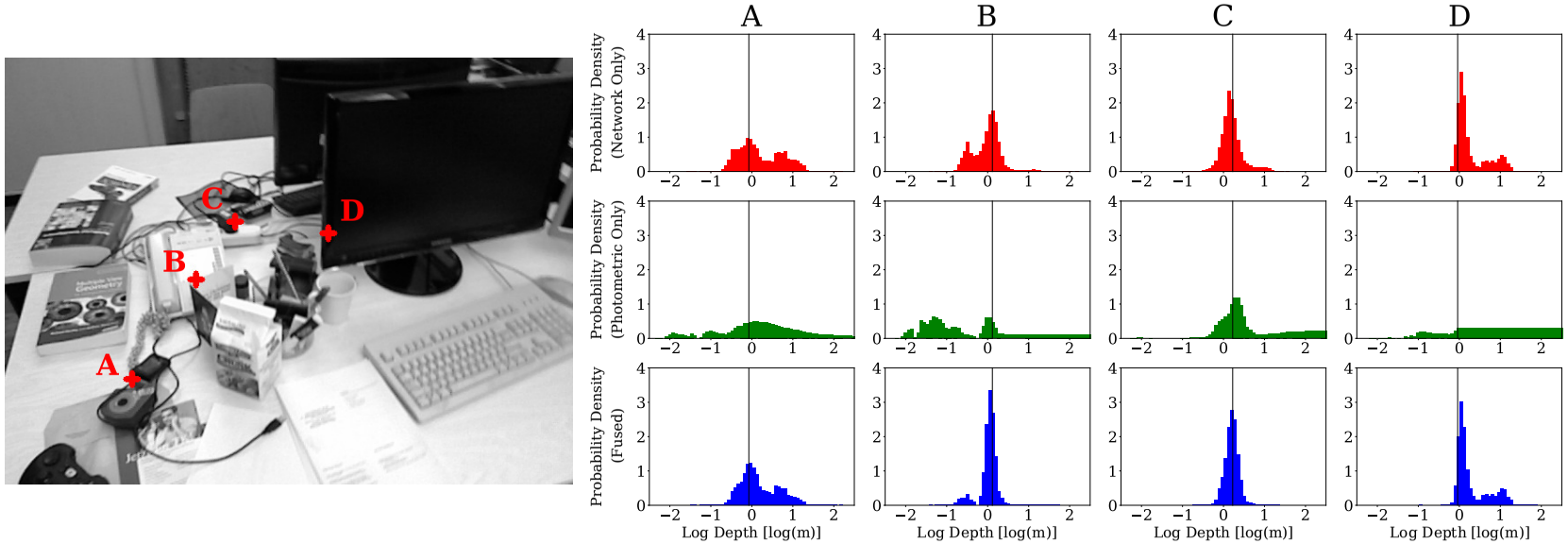


Figure 5.5: This figure shows a grid of probability densities for a sample of four pixels from a keyframe (left). The first row, in red, shows the probability densities predicted by the network. The second row, in green, shows the probability densities estimated from the photometric error after the addition of 25 reference frames. The final row, in blue, shows the fused probability densities that result from the algorithm. Note that both the network and the photometric error are capable of producing multiple peaks. In some cases (such as pixel C), both the network and the photometric methods produce good estimates. In others (such as pixel A), both the network and photometric error are relatively uncertain, but together produce a strong peak. In pixels B and D, the network helps resolve ambiguous photometric peaks from either a repetition or lack of texture. The vertical black bars show the location of the ground truth depth.

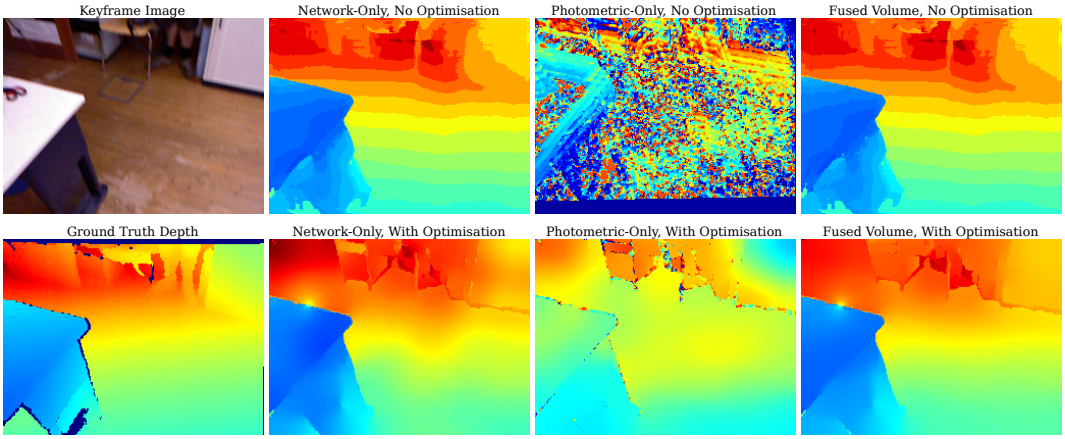


Figure 5.6: Qualitative results from an example keyframe and 6 additional reference frames in the TUM RGB-D *fr1/360* sequence. The top left image is the keyframe image, and the bottom left is the ground truth depth. The remaining images on the top row are the depth estimates obtained by taking the maximum probability depth from each corresponding probability volume. The bands of colour show the quantisation that results from using this method. The remaining images in the bottom row are the depth estimates that result after performing the optimisation step. Note that the photometric error is only capable of estimating the depth at pixels with a high image gradient (the repeated edges are the result of pose error). While using only the network prediction results in a good reconstruction, the best reconstruction is obtained by fusing the network and photometric volumes together.

ric probability volume, the best outcome is achieved by fusing the two together. In one of the sequences (*fr1/floor*), the best result is achieved by using only the photometric probability volume. For this entire sequence, the camera is aimed at a bare wooden floor, and, being well outside the training distribution, the network produces particularly bad priors. In the remaining two sequences, the best result is achieved using only the network probability volume. In one of these sequences (*fr1/rpy*), the camera motion is purely rotational and the photoconsistency-based subsystem is not able to produce meaningful depth estimates.

To show the benefit of the regularisation method, the performance of the full system is compared against three other regularisation schemes: using no optimisation at all (taking the depth values that maximise the discrete probability distribution), optimising without any regularisation (this will allow for the smoothing of the depth maps based on the continuous PDF, but provide no regularisation), and regularising

Sequence	System	L1-rel	L2-rel	RMSE
fr1/360	Network-Only	0.193	0.147	**0.555**
	Photometric-Only	0.633	1.008	1.514
	Fused	**0.191**	**0.143**	**0.555**
fr1/desk	Network-Only	0.295	0.201	0.447
	Photometric-Only	0.541	0.503	0.859
	Fused	**0.278**	**0.177**	**0.427**
fr1/desk2	Network-Only	0.237	0.139	**0.423**
	Photometric-Only	0.522	0.494	0.890
	Fused	**0.236**	**0.138**	0.424
fr1/floor	Network-Only	0.806	0.727	0.821
	Photometric-Only	**0.488**	**0.303**	**0.562**
	Fused	0.785	0.691	0.796
fr1/plant	Network-Only	0.426	0.502	**0.816**
	Photometric-Only	0.726	1.422	1.983
	Fused	**0.416**	**0.485**	0.833
fr1/room	Network-Only	0.231	0.155	0.493
	Photometric-Only	0.605	0.762	1.187
	Fused	**0.226**	**0.147**	**0.488**
fr1/rpy	Network-Only	**0.242**	**0.199**	**0.577**
	Photometric-Only	0.514	0.577	1.047
	Fused	0.255	0.212	0.614
fr1/teddy	Network-Only	**0.294**	**0.271**	**0.773**
	Photometric-Only	0.748	1.569	2.108
	Fused	0.297	0.277	0.792
fr1/xyz	Network-Only	0.241	0.162	0.432
	Photometric-Only	0.517	0.403	0.764
	Fused	**0.225**	**0.137**	**0.401**

Table 5.1: Comparison of reconstruction errors on Freiburg 1 TUM RGB-D [Sturm et al., 2012] sequences showing the relative performance of using only the network-predicted probability volume, only the photometric probability volume, and the fused probability volume.

using the total variation. To make the comparison with total variation be as fair as possible, the hyperparameters of the system were tuned for the best performance on the test sequences ($\lambda = 1.0 \cdot 10^2$ and a step size of 0.05). The results are presented in Table 5.2. In eight of the nine cases the best performance is achieved when using the surface normals and occlusion masks predicted by SharpNet.

Finally, to evaluate the method for warping probability volumes between keyframes, the system is compared against a version without warping where each keyframe is initialised only with the network output and does not receive any information from other keyframes. The results are presented in Table 5.3. Using the warping method improves the performance of the system in eight of the nine cases.

5.4 Conclusion

A method for fusing learned monocular depth priors into a standard dense mapping pipeline has been presented. By training a DNN to predict nonparametric probability distributions, the network is allowed to express uncertainty and make multi-hypothesis depth predictions.

Through a series of experiments, it was demonstrated that by fusing the discrete probability volume predicted by the network with a probability volume computed from the photometric data, a better performance is achieved than using either on its own. Unfortunately, the gap in performance between a system that uses only network predictions and a system that uses the fused volumes is not significant. This is largely due to the probability density of the network predicted distributions being significantly more clustered than the probability density of the distributions estimated from photoconsistency. While the relative certainty of the network predictions may be justified by the accuracy of the depth estimates it produces, it is likely that this confidence is a result of the network overfitting during training. Developing DNNs that are better able to predict appropriate confidence measures or finding a better way to balance the fusion of the two modalities remains for future work.

Sequence	System	L1-rel	L2-rel	RMSE
fr1/360	No Optimisation	0.210	0.184	0.608
	Smoothing-Only	0.207	0.179	0.601
	Total Variation	0.194	0.152	0.565
	Normals + Occlusions	**0.191**	**0.143**	**0.555**
fr1/desk	No Optimisation	0.324	0.292	0.537
	Smoothing-Only	0.323	0.289	0.533
	Total Variation	0.296	0.226	0.470
	Normals + Occlusions	**0.278**	**0.177**	**0.427**
fr1/desk2	No Optimisation	0.283	0.240	0.537
	Smoothing-Only	0.280	0.235	0.532
	Total Variation	0.254	0.181	0.470
	Normals + Occlusions	**0.236**	**0.138**	**0.424**
fr1/floor	No Optimisation	0.806	0.772	0.861
	Smoothing-Only	0.807	0.771	0.860
	Total Variation	0.801	0.738	0.836
	Normals + Occlusions	**0.785**	**0.691**	**0.796**
fr1/plant	No Optimisation	0.436	0.558	0.863
	Smoothing-Only	0.435	0.555	0.857
	Total Variation	0.425	0.520	**0.830**
	Normals + Occlusions	**0.416**	**0.485**	0.833
fr1/room	No Optimisation	0.267	0.227	0.583
	Smoothing-Only	0.265	0.223	0.577
	Total Variation	0.243	0.179	0.522
	Normals + Occlusions	**0.226**	**0.147**	**0.488**
fr1/rpy	No Optimisation	0.327	0.434	0.781
	Smoothing-Only	0.320	0.390	0.755
	Total Variation	0.265	0.231	0.621
	Normals + Occlusions	**0.255**	**0.212**	**0.614**
fr1/teddy	No Optimisation	0.296	0.300	0.799
	Smoothing-Only	0.295	0.296	0.792
	Total Variation	**0.290**	**0.276**	**0.771**
	Normals + Occlusions	0.297	0.277	0.792
fr1/xyz	No Optimisation	0.299	0.303	0.595
	Smoothing-Only	0.296	0.298	0.590
	Total Variation	0.255	0.212	0.493
	Normals + Occlusions	**0.225**	**0.137**	**0.401**

Table 5.2: Comparison of reconstruction errors on Freiburg 1 TUM RGB-D [Sturm et al., 2012] sequences showing the relative performance of different regularisation schemes. No Optimisation: results from taking the depth value with the maximum probability in the probability volume. Smoothing-Only: results from minimising the smoothed negative log probability density function without including a regularisation term. Total Variation: results from using the total variation of the depth as a regulariser. Normals + Occlusions: the pipeline as described in this chapter.

Sequence	System	L1-rel	L2-rel	RMSE
fr1/360	No Keyframe Warping	0.202	0.157	0.575
	Keyframe Warping	**0.191**	**0.143**	**0.555**
fr1/desk	No Keyframe Warping	0.316	0.236	0.474
	Keyframe Warping	**0.278**	**0.177**	**0.427**
fr1/desk2	No Keyframe Warping	0.283	0.195	0.480
	Keyframe Warping	**0.236**	**0.138**	**0.424**
fr1/floor	No Keyframe Warping	**0.776**	**0.684**	**0.787**
	Keyframe Warping	0.785	0.691	0.796
fr1/plant	No Keyframe Warping	0.420	0.490	0.845
	Keyframe Warping	**0.416**	**0.485**	**0.833**
fr1/room	No Keyframe Warping	0.256	0.189	0.528
	Keyframe Warping	**0.226**	**0.147**	**0.488**
fr1/rpy	No Keyframe Warping	0.297	0.263	0.654
	Keyframe Warping	**0.255**	**0.212**	**0.614**
fr1/teddy	No Keyframe Warping	0.302	0.286	**0.791**
	Keyframe Warping	**0.297**	**0.277**	0.792
fr1/xyz	No Keyframe Warping	0.315	0.247	0.521
	Keyframe Warping	**0.225**	**0.137**	**0.401**

Table 5.3: Comparison of reconstruction errors on Freiburg 1 TUM RGB-D [Sturm et al., 2012] sequences showing the performance gain from using the method to warp keyframe probability volumes.

While predicting nonparametric probability distributions for each pixel helps to overcome some of the weaknesses of the method described in the previous chapter, it comes at the cost of a much higher computational load. The system described here does not have real-time performance, although it may be possible to achieve that with significant engineering effort. Even so, the resources required to compute, store and warp the probability volumes are much higher than the unimodal Gaussian distributions used in Chapter 4. This suggests that there may be some trade off between accuracy and speed for these types of systems. Depending on the intended application of the system, one of these methods may be better suited than the other.

Chapter 6

Conclusions and Future Work

A number of contributions have been presented in this thesis that aim to increase the robustness of dense visual SLAM through the fusion of additional sensing modalities. In particular, acceleration and rotation rate measurements from an IMU were proposed to address convergence issues in dense tracking, and learned priors on dense reconstructions from DNNs were proposed to help constrain the depth estimation problem. Using a sensor fusion approach enables keeping the probabilistic formulation of standard pipelines, which is particularly important in mobile robotics where the fusion of many sensors is common and often necessary.

In Chapter 3, a dense RGB-D SLAM system was presented where inertial measurements were fused into the tracking step in a tightly-coupled fashion. Through the joint optimisation of the camera pose, velocity, IMU biases and gravity direction it was shown that such a system was capable of building a globally consistent, fully dense 3D reconstruction of the environment while being more robust to fast motions and periods of low texture and low geometric variation.

While some of the inertial information was passed to the mapping step in the form of a constraint ensuring consistency with the observable gravity direction, the complete set of cross-correlations with the inertial measurements could not be taken into account due to the separation of tracking and mapping that exists in dense SLAM. There has been some promising work using compact, optimisable

representations through which a joint inference can be done [Bloesch et al., 2018, Tang and Tan, 2019, Bloesch et al., 2019, Czarnowski et al., 2020], and it is an open research question as to whether inertial fusion in these systems would result in the levels of accuracy and robustness seen in sparse SLAM.

Chapter 4 proposed using learned parametric priors to help constrain the depth estimation problem. A real-time system was presented that fused the output of a semi-dense multi-view stereo algorithm with the depth and depth gradient predictions of a DNN using a standard probabilistic framework. The DNN predicted its own aleatoric uncertainties for the outputs using a Gaussian likelihood loss function during training. While the uncertainty predictions matched intuition (the highest uncertainty occurred at depth discontinuities where there are large or rapidly changing gradients), it was difficult to train and it was not clear how the magnitude of the values should be interpreted. The uncertainty predictions were also forced to conform with a parametric and unimodal distribution which may be ill-suited for dense reconstruction where there is often a need for a multi-hypothesis prediction. Also, the system only estimated multi-view geometric constraints for pixels with a high image gradient as these are the points where a good estimation is most likely. Unfortunately, these points typically happened to be where there were high depth gradients and the network had learned to predict high uncertainties. This meant that in practice, very little information was passed between the semi-dense points and points on the interior of objects during the fusion step and the depth estimation relied almost entirely on the network prediction.

Chapter 5 addressed some of these shortcomings by training a DNN to predict a discrete, nonparametric probability distribution for the depth of each pixel from a single image. By having a nonparametric formulation, the network was able to make use of multi-hypothesis predictions. This predicted probability volume was then fused with another probability volume based on the photometric consistency between subsequent frames and the keyframe image. Unlike the previous chapter, the photometric probability distributions were calculated for every pixel ensuring that some photometric information was available at every pixel during the fusion step. Depth

maps were extracted from the fused volume, using a regularisation method based on network predicted surface normals and occlusion boundaries. While this system did not quite achieve real-time performance, additional engineering effort could be made to speed up the optimisation step by implementing a primal-dual algorithm or Newton's method. In any case, the computational requirements for this method were much higher than the method described in Chapter 4, and a trade off between accuracy and computation needs to be navigated.

Unfortunately, the gap in performance between using the fused predictions and network-only predictions was not significant. This was mostly due to the photometric probability distributions only having strong peaks on pixels with high image gradients and the DNN being very confident in its own predictions. Since the test data was close to the data on which the DNN was trained, the network may have been justified in making such confident predictions. However, it is likely that this level of confidence is related to an issue that was also noticed in Chapter 4: during training, the network begins to overfit to the training data and learns to become overconfident in its own predictions. The network has no way of knowing if the images seen at test time are close to those it observed in the training set and therefore does nothing to temper its confidence. Developing DNNs that are able to predict appropriate confidence measurements remains an ongoing research problem, although there have been some promising recent results [Carvalho et al., 2020].

The bigger question to be asked is whether it is worth keeping the probabilistic formulation at all given that other methods of combining deep learning and dense reconstruction are capable of producing much more accurate results. The best results currently seem to come from systems that take the outputs of classic, photometric-based approaches and feed them into a DNN for regularisation. DeepTAM [Zhou et al., 2018] is the state-of-the-art example of such a system; it passes a photometric cost volume through a DNN to extract a depth map for a given keyframe. While the depth maps it produces are very accurate in comparison to other methods, there is no measure of uncertainty and it is far from being a real-time system. However, as mentioned previously, a probabilistic formulation has been a mainstay of robotics

research for decades and it seems unlikely that the correct course is to abandon this formulation now. Most decisions that robots need to make require some measure of uncertainty, especially for those that are safety critical. For this reason, it is important to continue research in the direction of probabilistic fusion of learned priors into standard systems even if the results of such systems are not yet capable of obtaining such high levels of accuracy as DeepTAM and other similar approaches.

In general, there is no best way to build a dense SLAM system. The many design decisions that go into constructing one will be heavily based on the goals and requirements of the system and the expected application area. For example, if real-time capability is important, it may not be possible to use systems like DeepTAM that need to run an expensive DNN pass for each update. Depending on the computational budget, it may not be possible to run any DNN at all, and instead a reconstruction system would need to rely on more basic priors, such as planes, for regularisation. Similarly, different systems may have different needs regarding the accuracy of the reconstructions and their ability to handle uncertainty. All of these factors would have significant influence into how a dense reconstruction system is put together.

In both Chapter 4 and Chapter 5, the depth was regularised based on network predictions of the surface orientation (depth gradients in Chapter 4 and surface normals in Chapter 5). It may be useful to explore other methods for regularisation using network predictions, as the surface orientation estimates were often noisy, especially around object boundaries. It may be better, for instance, to have networks predict planar regions or other large geometric primitives as these may be more robustly estimated.

Perhaps the more interesting research direction, however, would be to use DNNs to enable the building of SLAM systems that operate at much more abstract levels of scene understanding and interaction. Most likely, these higher levels of understanding would involve the inclusion of semantic information in the map representation. While semantic labels have been used to augment existing dense reconstructions,

it seems likely that developing truly joint geometric and semantic representations will be an important part of incremental scene understanding. For example, if a reconstruction system believed that a specific region represented a floor, tabletop or wall, then it could apply a strong planar prior to the reconstruction. Similarly, more complicated shape-based priors could be used for objects in the scene. It seems likely that the next generation of SLAM systems will find a way to combine semantics and geometry in an elegant and meaningful way, as not only will this result in better reconstructions, but may also provide other things that are useful for robotic tasks such as object segmentations and an understanding of traversable terrain.

Finally, throughout this thesis, many different SLAM algorithms were evaluated. The evaluation metrics were chosen from standard ones used in the SLAM community, such as the absolute trajectory error or average per-pixel RMSE on depth maps. It is not clear that these are the best metrics for evaluating SLAM algorithms, however. Like the design of a SLAM system itself, the best metrics to use should depend on the goals of that system. For example, with augmented reality applications, the reduction of local tracking error that causes the "swimming" of virtual objects placed in a scene is of the utmost importance, whereas the drift accumulated over an entire sequence might not. It may be the opposite case for ground vehicles expected to navigate autonomously over long distances: the local tracking error may not be nearly as important as ensuring that the global drift is small enough that the vehicle can reach its final objective. Furthermore, SLAM systems are complicated software engineering projects that consist of many design choices and tunable parameters that may interact in complicated ways. This makes any real comparison of SLAM systems difficult, if not impossible. The development of better evaluation techniques and a movement towards publishing full results (i.e. frame-by-frame pose estimates rather than final scores on a given metric) would greatly benefit the SLAM community.

Bibliography

[Agarwal et al., 2009] Agarwal, S., Snavely, N., Simon, I., Seitz, S. M., and Szeliski, R. (2009). Building Rome in a Day. In *Proceedings of the International Conference on Computer Vision (ICCV)*. 13

[Bailey and Durrant-Whyte, 2006] Bailey, T. and Durrant-Whyte, H. (2006). Simultaneous Localisation and Mapping (SLAM): Part II. *IEEE Robotics and Automation Magazine*, 13(3):108–117. 13

[Baker et al., 2004] Baker, S., Patil, R., Cheung, K. M., and Matthews, I. (2004). Lucas-Kanade 20 Years On: Part 5. Technical report, Robotics Institute, Carnegie Mellon University. Technical Report CMU-RI-TR-04-64. 18

[Barnes et al., 2018] Barnes, D., Maddern, W., Pascoe, G., and Posner, I. (2018). Driven to Distraction: Self-Supervised Distractor Learning for Robust Monocular Visual Odometry in Urban Environments. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*. 39

[Bay et al., 2006] Bay, H., Tuytelaars, T., and Van Gool, L. (2006). SURF: Speeded Up Robust Features. In *Proceedings of the European Conference on Computer Vision (ECCV)*. 16

[Bescós et al., 2018] Bescós, B., Fácil, J. M., Civera, J., and Neira, J. (2018). Detecting, Tracking and Eliminating Dynamic Objects in 3D Mapping using Deep

Learning and Inpainting. In *Proceedings of the Workshop on Multimodal Robot Perception at the IEEE International Conference on Robotics and Automation (ICRA)*. 39

[Bishop, 1994] Bishop, C. M. (1994). *Mixture Density Networks*. Aston University. 93

[Bloesch et al., 2018] Bloesch, M., Czarnowski, J., Clark, R., Leutenegger, S., and Davison, A. J. (2018). CodeSLAM – Learning a Compact, Optimisable Representation for Dense Visual SLAM. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 15, 74, 78, 110

[Bloesch et al., 2019] Bloesch, M., Laidlow, T., Clark, R., Leutenegger, S., and Davison, A. J. (2019). Learning Meshes for Dense Visual SLAM. In *Proceedings of the International Conference on Computer Vision (ICCV)*. 19, 25, 110

[Bloesch et al., 2015] Bloesch, M., Omari, S., Hutter, M., and Siegwart, R. (2015). Robust Visual Inertial Odometry using a Direct EKF-Based Approach. In *Proceedings of the IEEE/RSJ Conference on Intelligent Robots and Systems (IROS)*. 50

[Bloesch et al., 2016] Bloesch, M., Sommer, H., Laidlow, T., Burri, M., Nützi, G., Fankhauser, P., Bellicoso, D., Gehring, C., Leutenegger, S., Hutter, M., and Siegwart, R. (2016). A Primer on the Differential Calculus of 3D Orientations. *CoRR*, abs/1606.0. 25, 37

[Calonder et al., 2010] Calonder, M., Lepetit, V., Strecha, C., and Fua, P. (2010). BRIEF: Binary Robust Independent Elementary Features. In *Proceedings of the European Conference on Computer Vision (ECCV)*. 16

[Campbell et al., 2008] Campbell, N. D. F., Vogiatzis, G., Hernández, C., and Cipolla, R. (2008). Using Multiple Hypotheses to Improve Depth-Maps for Multi-View Stereo. In *Proceedings of the European Conference on Computer Vision (ECCV)*. 22, 23, 93

[Carvalho et al., 2020] Carvalho, E. D. C., Clark, R., Nicastro, A., and Kelly, P. H. J. (2020). Scalable Uncertainty for Computer Vision with Functional Variational Inference. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 111

[Chang and Chen, 2018] Chang, J.-R. and Chen, Y.-S. (2018). Pyramid Stereo Matching Network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 74

[Concha and Civera, 2014] Concha, A. and Civera, J. (2014). Using Superpixels in Monocular SLAM. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 365–372. IEEE. 18, 22, 95

[Concha and Civera, 2015] Concha, A. and Civera, J. (2015). DPPTAM: Dense Piecewise Planar Tracking and Mapping from a Monocular Sequence. In *Proceedings of the IEEE/RSJ Conference on Intelligent Robots and Systems (IROS)*. IEEE. 18, 22, 74, 95

[Concha and Civera, 2017] Concha, A. and Civera, J. (2017). RGBDTAM: A Cost-Effective and Accurate RGB-D Tracking and Mapping System. In *Proceedings of the IEEE/RSJ Conference on Intelligent Robots and Systems (IROS)*. 17

[Concha et al., 2014] Concha, A., Hussain, W., Montano, L., and Civera, J. (2014). Manhattan and Piecewise-Planar Constraints for Dense Monocular Mapping. In *Proceedings of Robotics: Science and Systems (RSS)*. 18, 22, 74, 95

[Concha et al., 2016] Concha, A., Loianna, G., Kumar, V., and Civera, J. (2016). Visual-Inertial Direct SLAM. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*. 51, 52, 74

[Czarnowski et al., 2020] Czarnowski, J., Laidlow, T., Clark, R., and Davison, A. J. (2020). DeepFactors: Real-Time Probabilistic Dense Monocular SLAM. In *IEEE Robotics and Automation Letters*. 19, 25, 110

[Dai et al., 2017] Dai, A., Chang, A. X., Savva, M., Halber, M., Funkhouser, T., and Nießner, M. (2017). ScanNet: Richly-Annotated 3D Reconstructions of Indoor

Scene. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 96

[Davison, 2003] Davison, A. J. (2003). Real-Time Simultaneous Localisation and Mapping with a Single Camera. In *Proceedings of the International Conference on Computer Vision (ICCV)*. 14

[DeVito et al., 2017] DeVito, Z., Mara, M., Zollöfer, M., Bernstein, G., Theobalt, C., Hanrahan, P., Fisher, M., and Nießner, M. (2017). Opt: A Domain Specific Language for Non-linear Least Squares Optimization in Graphics and Imaging. In *ACM Transactions on Graphics*. 82

[Durrant-Whyte and Bailey, 2006] Durrant-Whyte, H. and Bailey, T. (2006). Simultaneous Localisation and Mapping (SLAM): Part I The Essential Algorithms. *IEEE Robotics and Automation Magazine*, 13(2):99–110. 13

[Eigen and Fergus, 2015] Eigen, D. and Fergus, R. (2015). Predicting Depth, Surface Normals and Semantic Labels with a Common Multi-Scale Convolutional Architecture. In *Proceedings of the International Conference on Computer Vision (ICCV)*. 46

[Eigen et al., 2014] Eigen, D., Puhrsch, C., and Fergus, R. (2014). Depth Map Prediction from a Single Image using a Multi-Scale Deep Network. In *Neural Information Processing Systems (NIPS)*. 12, 19, 46, 74, 77, 95, 102

[Engel et al., 2017] Engel, J., Koltun, V., and Cremers, D. (2017). Direct Sparse Odometry. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*. 14, 15, 39

[Engel et al., 2014] Engel, J., Schoeps, T., and Cremers, D. (2014). LSD-SLAM: Large-Scale Direct Monocular SLAM. In *Proceedings of the European Conference on Computer Vision (ECCV)*. 15, 17, 84

[Engel et al., 2015] Engel, J., Stückler, J., and Cremers, D. (2015). Large-Scale Direct SLAM with Stereo Cameras. In *Proceedings of the IEEE/RSJ Conference on Intelligent Robots and Systems (IROS)*. 51

[Engel et al., 2012] Engel, J., Sturm, J., and Cremers, D. (2012). Camera-Based Navigation of a Low-Cost Quadrocopter. In *Proceedings of the IEEE/RSJ Conference on Intelligent Robots and Systems (IROS)*. 50

[Engel et al., 2013] Engel, J., Sturm, J., and Cremers, D. (2013). Semi-Dense Visual Odometry for a Monocular Camera. In *Proceedings of the International Conference on Computer Vision (ICCV)*. 77, 79

[Engel, 2017] Engel, J.-J. (2017). *Large-Scale Direct SLAM and 3D Reconstruction in Real-Time*. PhD thesis, Technical University of Munich. 17

[Fácil et al., 2017] Fácil, J. M., Concha, A., Montesano, L., and Civera, J. (2017). Single-View and Multi-View Depth Fusion. *IEEE Robotics and Automation Letters*, 2(4):1994–2001. 19, 74, 76

[Fitzgibbon et al., 1998] Fitzgibbon, A. W., Cross, G., and Zisserman, A. (1998). Automatic 3D Model Construction for Turn-Table Sequences. In *Proceedings of the Workshop on Structure from Multiple Images of Large Scale Environments (SMILE), in conjunction with ECCV*. 12

[Flint et al., 2011] Flint, A., Murray, D. W., and Reid, I. (2011). Manhattan Scene Understanding Using Monocular, Stereo, and 3D Features. In *Proceedings of the International Conference on Computer Vision (ICCV)*. 18, 22

[Forster et al., 2015] Forster, C., Carlone, L., Dellaert, F., and Scaramuzza, D. (2015). IMU Preintegration on Manifold for Efficient Visual-Inertial Maximum-a-Posteriori Estimation. In *Proceedings of Robotics: Science and Systems (RSS)*. 19, 21, 50, 56

[Forster et al., 2014] Forster, C., Pizzoli, M., and Scaramuzza, D. (2014). SVO: Fast Semi-Direct Monocular Visual Odometry. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*. 50

[Frahm et al., 2010] Frahm, J.-M., Georgel, P. F., Gallup, D., Johnson, T., Raguram, R., Wu, C., Jen, Y., Dunn, E., Clipp, B., and Lazebnik, S. (2010).

Building Rome on a Cloudless Day. In *Proceedings of the European Conference on Computer Vision (ECCV)*. 13

[Fu et al., 2018] Fu, H., Gong, M., Wang, C., Batmanghelich, K., and Tao, D. (2018). Deep Ordinal Regression Network for Monocular Depth Estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 12, 74, 93, 95

[Fuhrmann et al., 2014] Fuhrmann, S., Langguth, F., and Goesele, M. (2014). MVE — A Multi-View Reconstruction Environment. In *EUROGRAPHICS Workshops on Graphics and Cultural Heritage*. 13

[Furgale et al., 2013] Furgale, P., Rehder, J., and Siegwart, R. (2013). Unified Temporal and Spatial Calibration for Multi-Sensor Systems. In *Intelligent Robots and Systems (IROS), 2013 IEEE/RSJ International Conference on*, pages 1280–1286. IEEE. 46, 63, 70

[Furukawa and Ponce, 2007] Furukawa, Y. and Ponce, J. (2007). Accurate, Dense, and Robust Multi-View Stereopsis. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 13

[Gal and Ghahramani, 2016] Gal, Y. and Ghahramani, Z. (2016). Dropout as a Bayesian Approximation: Representing Model Uncertainty in Deep Learning. In *Proceedings of the International Conference on Machine Learning (ICML)*. 93

[Godard et al., 2017] Godard, C., Mac Aodha, O., and Brostow, G. J. (2017). Unsupervised Monocular Depth Estimation with Left-Right Consistency. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 74

[Goesele et al., 2007] Goesele, M., Snavely, N., Curless, B., Hoppe, H., and Seitz, S. M. (2007). Multi-View Stereo for Community Photo Collections. In *Proceedings of the International Conference on Computer Vision (ICCV)*. 13

[Goodfellow et al., 2016] Goodfellow, I., Bengio, Y., and Courville, A. (2016). *Deep Learning*. MIT Press. http://www.deeplearningbook.org. 46

[Graber et al., 2011] Graber, G., Pock, T., and Bischof, H. (2011). Online 3D Reconstruction using Convex Optimization. In *Workshop on Live Dense Reconstruction from Moving Cameras at ICCV*. 17

[Handa et al., 2014] Handa, A., Whelan, T., McDonald, J. B., and Davison, A. J. (2014). A Benchmark for RGB-D Visual Odometry, 3D Reconstruction and SLAM. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*. 60, 82, 84

[He et al., 2016] He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep Residual Learning for Image Recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 95

[Hernández and Schmitt, 2004] Hernández, C. and Schmitt, F. (2004). Silhouette and Stereo Fusion for 3D Object Modeling. *Computer Vision and Image Understanding (CVIU)*, 96(3):367–392. 12

[Hertzberg et al., 2011] Hertzberg, C., Wagner, R., Frese, U., and Schröder, L. (2011). Integrating Generic Sensor Fusion Algorithms with Sound State Representations through Encapsulation of Manifolds. *Information Fusion*, 14(1):57–77. 37

[Hoiem et al., 2005] Hoiem, D., Efros, A. A., and Hebert, M. (2005). Geometric Context from a Single Image. In *Proceedings of the International Conference on Computer Vision (ICCV)*. 12

[Hoiem et al., 2008] Hoiem, D., Efros, A. A., and Hebert, M. (2008). Putting Objects in Perspective. *International Journal of Computer Vision (IJCV)*, 80(1):3–15. 12

[Irani and Anandan, 1999] Irani, M. and Anandan, P. (1999). All About Direct Methods. In *Proceedings of the International Workshop on Vision Algorithms, in association with ICCV*. 14

[Jaimez et al., 2017] Jaimez, M., Kerl, C., Gonzalez-Jimenez, J., and Cremers, D. (2017). Fast Odometry and Scene Flow from RGB-D Cameras Based on Geometric

Clustering. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*. 39

[Jin et al., 2003] Jin, H., Favaro, P., and Soatto, S. (2003). A Semi-Direct Approach to Structure from Motion. *The Visual Computer*, 19(6):377–394. 15, 20

[Jones and Soatto, 2011] Jones, E. S. and Soatto, S. (2011). Visual-Inertial Navigation, Mapping and Localization: A Scalable Real-Time Causal Approach. *International Journal of Robotics Research (IJRR)*, 30(4):407–430. 19, 21, 50

[Kahler et al., 2015] Kahler, O., Prisacariu, V. A., Ren, C. Y., Sun, X., Torr, P. H. S., and Murray, D. W. (2015). Very High Frame Rate Volumetric Integration of Depth Images on Mobile Device. In *Proceedings of the International Symposium on Mixed and Augmented Reality (ISMAR)*. 17

[Kalman, 1960] Kalman, R. (1960). A New Approach to Linear Filtering and Prediction Problems. *Journal of Basic Engineering*, 82(1):35–45. 14

[Kazhdan and Hoppe, 2013] Kazhdan, M. and Hoppe, H. (2013). Screened Poisson Surface Reconstruction. *ACM Transactions on Graphics*. 13

[Keivan et al., 2014] Keivan, N., Patron-Perez, A., and Sibley, G. (2014). Asynchronous Adaptive Conditioning for Visual-Inertial SLAM. In *Proceedings of the International Symposium on Experimental Robotics (ISER)*. 19, 21, 50

[Kendall and Gal, 2017] Kendall, A. and Gal, Y. (2017). What Uncertainties Do We Need in Bayesian Deep Learning for Computer Vision? In *Neural Information Processing Systems (NIPS)*. 78, 93

[Kerl et al., 2015] Kerl, C., Stückler, J., and Cremers, D. (2015). Dense Continuous-Time Tracking and Mapping with Rolling Shutter RGB-D Cameras. In *Proceedings of the International Conference on Computer Vision (ICCV)*. 60

[Kerl et al., 2013] Kerl, C., Sturm, J., and Cremers, D. (2013). Dense Visual SLAM for RGB-D Cameras. In *Proceedings of the IEEE/RSJ Conference on Intelligent Robots and Systems (IROS)*. 17, 22, 74

[Klein and Murray, 2007] Klein, G. and Murray, D. W. (2007). Parallel Tracking and Mapping for Small AR Workspaces. In *Proceedings of the International Symposium on Mixed and Augmented Reality (ISMAR)*. 14

[Krotkov et al., 1995] Krotkov, E., Hebert, M., and Simmons, R. (1995). Stereo Perception and Dead-Reckoning for a Prototype Lunar Rover. *Autonomous Robots*, 2(4):313–331. 14

[Kruppa, 1913] Kruppa, E. (1913). Zur Ermittlung eines Objektes aus zwei Perspektiven mit innerer Orientierung. *Sitzungsberichte der math.-naturw. Kl. der kaiserlichen Akademie der Wissenschaften, Abt. Ila*, 122:1939–1948. 13

[Kutulakos and Seitz, 2000] Kutulakos, K. N. and Seitz, S. M. (2000). A Theory of Shape by Space Carving. *International Journal of Computer Vision (IJCV)*, 38(3):199–218. 12

[Laidlow et al., 2017] Laidlow, T., Bloesch, M., Li, W., and Leutenegger, S. (2017). Dense RGB-D-Inertial SLAM with Map Deformations. In *Proceedings of the IEEE/RSJ Conference on Intelligent Robots and Systems (IROS)*. 20, 25

[Laidlow et al., 2019] Laidlow, T., Czarnowski, J., and Leutenegger, S. (2019). DeepFusion: Real-Time Dense 3D Reconstruction for Monocular SLAM using Single-View Depth and Gradient Predictions. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*. 21, 24, 95, 98

[Laidlow et al., 2020] Laidlow, T., Czarnowski, J., Nicastro, A., Clark, R., and Leutenegger, S. (2020). Towards the Probabilistic Fusion of Learned Priors into Standard Pipelines for 3D Reconstruction. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*. 23, 24

[Laina et al., 2016] Laina, I., Rupprecht, C., Belagiannis, V., Tombari, F., and Navab, N. (2016). Deeper Depth Prediction with Fully Convolutional Residual Networks. In *Proceedings of the International Conference on 3D Vision (3DV)*. 12, 19, 46, 74, 84, 85

[LeCun et al., 1998] LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. (1998). Gradient-Based Learning Applied to Document Recognition. *Proceedings of the IEEE*, 86(11):2278–2324. 47

[Leonard and Whyte, 1991] Leonard, J. J. and Whyte, D. H. (1991). Simultaneous Map Building and Localization for an Autonomous Mobile Robot. In *Proceedings of the IEEE/RSJ Conference on Intelligent Robots and Systems (IROS)*. 14

[Leutenegger et al., 2011] Leutenegger, S., Chli, M., and Siegwart, R. (2011). BRISK: Binary Robust Invariance Scalable Keypoints. In *Proceedings of the International Conference on Computer Vision (ICCV)*. 16

[Leutenegger et al., 2014] Leutenegger, S., Lynen, S., Bosse, M., Siegwart, R., and Furgale, P. (2014). Keyframe-Based Visual-Inertial Odometry using Nonlinear Optimization. *The International Journal of Robotics Research*, 34(3):314–334. 19, 21, 45, 50, 56, 58

[Li and Mourikis, 2013] Li, M. and Mourikis, A. (2013). High-Precision Consistent EKF-Based Visual-Inertial Odometry. *International Journal of Robotics Research (IJRR)*, 32:690–711. 19, 50

[Liu et al., 2019] Liu, C., Gu, J., Kim, K., Narasimhan, S. G., and Kautz, J. (2019). Neural RGB-D Sensing: Depth and Uncertainty From a Video Camera. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 93, 95, 96

[Liu et al., 2015] Liu, F., Shen, C., and Lin, G. (2015). Deep Convolutional Neural Fields for Depth Estimation from a Single Image. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 12, 19, 46

[Loop et al., 2016] Loop, C., Cai, Q., Chou, P., and Orts-Escolano, S. (2016). A Closed-Form Bayesian Fusion Equation using Occupancy Probabilities. In *Proceedings of the International Conference on 3D Vision (3DV)*. 100

[Lovegrove et al., 2011] Lovegrove, S. J., Davison, A. J., and Ibanez-Guzmán, J. (2011). Accurate Visual Odometry from a Rear Parking Camera. In *Proceedings of the IEEE Intelligent Vehicles Symposium (IV)*. 16

[Lowe, 1999] Lowe, D. G. (1999). Object Recognition from Local Scale-Invariant Features. In *Proceedings of the International Conference on Computer Vision (ICCV)*. 16

[Lucas and Kanade, 1981] Lucas, B. D. and Kanade, T. (1981). An Iterative Image Registration Technique with an Application to Stereo Vision. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*. 15

[Ma et al., 2015] Ma, L., Falquez, J. M., McGuire, S., and Sibley, G. (2015). Large Scale Dense Visual Inertial SLAM. In *Proceedings of the International Symposium on Experimental Robotics (ISER)*. 50

[Mahjourian et al., 2018] Mahjourian, R., Wicke, M., and Angelova, A. (2018). Unsupervised Learning of Depth and Ego-Motion from Monocular Video Using 3D Geometric Constraints. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 74

[McCormac et al., 2017] McCormac, J., Handa, A., Leutenegger, S., and Davison, A. J. (2017). SceneNet RGB-D: Can 5M Synthetic Images Beat Generic ImageNet Pre-Training on Indoor Segmentation? In *Proceedings of the International Conference on Computer Vision (ICCV)*. 78, 83, 85

[Meier et al., 2011] Meier, L., Tanskanen, P., Fraundorfer, F., and Pollefeys, M. (2011). Pixhawk: A System for Autonomous Flight using Onboard Computer Vision. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*. 50

[Merrell et al., 2007] Merrell, P., Akbarzadeh, A., Wang, L., Mordohai, P., Frahm, J.-M., Yang, R., Nistér, D., and Pollefeys, M. (2007). Real-Time Visibility-Based Fusion of Depth Maps. In *Proceedings of the International Conference on Computer Vision (ICCV)*. 16

[Mobahi et al., 2012] Mobahi, H., Zitnick, C. L., and Ma, Y. (2012). Seeing through the Blur. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 18, 20, 40

[Molton et al., 2004] Molton, N. D., Davison, A. J., and Reid, I. (2004). Locally Planar Patch Features for Real-Time Structure from Motion. In *Proceedings of the British Machine Vision Conference (BMVC)*. 20

[Moravec, 1977] Moravec, H. P. (1977). Towards Automatic Visual Obstacle Avoidance. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, volume 2, page 584. 13

[Moulon et al., 2013] Moulon, P., Monasse, P., and Marlet, R. (2013). Global Fusion of Relative Motions for Robust, Accurate and Scalable Structure from Motion. In *Proceedings of the International Conference on Computer Vision (ICCV)*. 13

[Mourikis and Roumeliotis, 2007] Mourikis, A. I. and Roumeliotis, S. I. (2007). A Multi-State Constraint Kalman Filter for Vision-Aided Inertial Navigation. In *Robotics and Automation, 2007 IEEE International Conference on*, pages 3565–3572. IEEE. 19, 50

[Mur-Artal et al., 2015] Mur-Artal, R., Montiel, J. M. M., and Tardós, J. D. (2015). ORB-SLAM: a Versatile and Accurate Monocular SLAM System. *IEEE Transactions on Robotics (T-RO)*, 31(5):1147–1163. 50

[Mur-Artal and Tardós, 2017] Mur-Artal, R. and Tardós, J. D. (2017). ORB-SLAM2: An Open-Source SLAM System for Monocular, Stereo, and RGB-D Cameras. *IEEE Transactions on Robotics (T-RO)*, 33(5):1255–1262. 16, 77, 84, 97

[Mur-Artal and Tardos, 2017] Mur-Artal, R. and Tardos, J. D. (2017). Visual-Inertial Monocular SLAM with Map Reuse. *IEEE Robotics and Automation Letters*, 2:796–803. 21, 50

[Nayar et al., 1995] Nayar, S., Watanabe, M., and Noguchi, M. (1995). Real-Time Focus Range Sensor. In *Proceedings of the International Conference on Computer Vision (ICCV)*. 12

[Newcombe, 2012] Newcombe, R. A. (2012). *Dense Visual SLAM*. PhD thesis, Imperial College London. 17

[Newcombe and Davison, 2010] Newcombe, R. A. and Davison, A. J. (2010). Live Dense Reconstruction with a Single Moving Camera. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 16

[Newcombe et al., 2011a] Newcombe, R. A., Izadi, S., Hilliges, O., Molyneaux, D., Kim, D., Davison, A. J., Kohli, P., Shotton, J., Hodges, S., and Fitzgibbon, A. (2011a). KinectFusion: Real-Time Dense Surface Mapping and Tracking. In *Proceedings of the International Symposium on Mixed and Augmented Reality (ISMAR)*. 17, 22, 41, 74

[Newcombe et al., 2011b] Newcombe, R. A., Lovegrove, S., and Davison, A. J. (2011b). DTAM: Dense Tracking and Mapping in Real-Time. In *Proceedings of the International Conference on Computer Vision (ICCV)*. 15, 18, 22, 43, 74, 95, 97

[Nikolic et al., 2016] Nikolic, J., Furgale, P., Melzer, A., and Siegwart, R. (2016). Maximum Likelihood Identification of Inertial Sensor Noise Model Parameters. *IEEE Sensors Journal*, 16(1):163–176. 61

[Omari et al., 2015] Omari, S., Bloesch, M., Gohl, P., and Siegwart, R. (2015). Dense Visual-Inertial Navigation System for Mobile Robots. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*. 50

[Peretroukhin et al., 2019] Peretroukhin, V., Wagstaff, B., and Kelly, J. (2019). Deep Probabilistic Regression of Elements of SO(3) using Quaternion Averaging and Uncertainty Injection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*. 93

[Phothong et al., 2018] Phothong, W., Wu, T.-C., Lai, J.-Y., Yu, C.-Y., Wang, D., and Liao, C.-Y. (2018). Quality Improvement of 3D Models Reconstructed from Silhouettes of Multiple Images. *Computer-Aided Design and Applications*, 15:288–299. 12

[Pizzoli et al., 2014] Pizzoli, M., Forster, C., and Scaramuzza, D. (2014). REMODE: Probabilistic, Monocular Dense Reconstruction in Real Time. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*. 17, 18, 22, 74, 84, 95

[Pradeep et al., 2013] Pradeep, V., Rhemann, C., Izadi, S., Zach, C., Bleyer, M., and Bathiche, S. (2013). MonoFusion: Real-Time 3D Reconstruction of Small Scenes with a Single Web Camera. In *Proceedings of the International Symposium on Mixed and Augmented Reality (ISMAR)*, pages 83–88. 17, 18, 22

[Qin et al., 2017] Qin, T., Li, P., and Shen, S. (2017). VINS-Mono: A Robust and Versatile Monocular Visual-Inertial State Estimator. *IEEE Transactions on Robotics (T-RO)*, 34:1004–1020. 21, 50

[Ramamonjisoa and Lepetit, 2019] Ramamonjisoa, M. and Lepetit, V. (2019). SharpNet: Fast and Accurate Recovery of Occluding Contours in Monocular Depth Estimation. In *Proceedings of the International Conference on Computer Vision Workshops (ICCVW)*. 46, 98, 99

[Ranftl et al., 2016] Ranftl, R., Vineet, V., Chen, Q., and Koltun, V. (2016). Dense Monocular Depth Estimation in Complex Dynamic Scenes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 15

[Ronneberger et al., 2015] Ronneberger, O., Fischer, P., and Brox, T. (2015). U-Net: Convolutional Networks for Biomedical Image Segmentation. In *Proceedings of the International Conference on Medical Image Computing and Computer Assisted Intervention (MICCAI)*. 47, 78

[Rosten and Drummond, 2006] Rosten, E. and Drummond, T. (2006). Machine Learning for High-Speed Corner Detection. In *Proceedings of the European Conference on Computer Vision (ECCV)*. 16

[Rublee et al., 2011] Rublee, E., Rabaud, V., Konolige, K., and Bradski, G. (2011). ORB: An Efficient Alternative to SIFT or SURF. In *Proceedings of the International Conference on Computer Vision (ICCV)*, pages 2564–2571. IEEE. 16

[Saxena et al., 2005] Saxena, A., Chung, S., and Ng, A. (2005). Learning Depth from Single Monocular Images. In *Neural Information Processing Systems (NIPS)*. 12

[Schönberger and Frahm, 2016] Schönberger, J. L. and Frahm, J.-M. (2016). Structure-from-Motion Revisited. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 13

[Schönberger et al., 2016] Schönberger, J. L., Zheng, E., Pollefeys, M., and Frahm, J.-M. (2016). Pixelwise View Selection for Unstructured Multi-View Stereo. In *Proceedings of the European Conference on Computer Vision (ECCV)*. 13

[Scona et al., 2018] Scona, R., Jaimez, M., Petillot, Y. R., Fallon, M., and Cremers, D. (2018). StaticFusion: Background Reconstruction for Dense RGB-D SLAM in Dynamic Environments. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*. 39

[Shen et al., 2015] Shen, S., Michael, N., and Kumar, V. (2015). Tightly-Coupled Monocular Visual-Inertial Fusion for Autonomous Flight of Rotorcraft MAVs. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*. 21, 50

[Shi and Tomasi, 1994] Shi, J. and Tomasi, C. (1994). Good Features to Track. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 14

[Silberman et al., 2012] Silberman, N., Hoiem, D., Kohli, P., and Fergus, R. (2012). Indoor Segmentation and Support Inference from RGBD Images. In *Proceedings of the European Conference on Computer Vision (ECCV)*. 85

[Silveira et al., 2008] Silveira, G., Malis, E., and Rives, P. (2008). An Efficient Direct Approach to Visual SLAM. *IEEE Transactions on Robotics (T-RO)*, 24(5):969–979. 20

[Snavely et al., 2006] Snavely, N., Seitz, S. M., and Szeliski, R. (2006). Photo Tourism: Exploring Photo Collections in 3D. In *Proceedings of SIGGRAPH*. 13

[Stuehmer et al., 2010] Stuehmer, J., Gumhold, S., and Cremers, D. (2010). Real-Time Dense Geometry from a Handheld Camera. In *Proceedings of the DAGM Symposium on Pattern Recognition*. 17

[Sturm et al., 2012] Sturm, J., Engelhard, N., Endres, F., Burgard, W., and Cremers, D. (2012). A Benchmark for the Evaluation of RGB-D SLAM Systems. In *Proceedings of the IEEE/RSJ Conference on Intelligent Robots and Systems (IROS)*. 17, 60, 65, 82, 84, 99, 101, 105, 107, 108

[Sun et al., 2018] Sun, K., Mohta, K., Pfrommer, B., Watterson, M., Liu, S., Mulgaonkar, Y., Taylor, C. J., and Kumar, V. (2018). Robust Stereo Visual Inertial Odometry for Fast Autonomous Flight. *IEEE Robotics and Automation Letters*, 3(2):965–972. 50

[Tang and Tan, 2019] Tang, C. and Tan, P. (2019). BA-Net: Dense Bundle Adjustment Network. In *Proceedings of the International Conference on Learning Representations (ICLR)*. 74, 110

[Tanskanen et al., 2015] Tanskanen, P., Naegeli, T., Pollefeys, M., and Hilliges, O. (2015). Semi-Direct EKF-Based Monocular Visual-Inertial Odometry. In *Proceedings of the IEEE/RSJ Conference on Intelligent Robots and Systems (IROS)*. 19, 50

[Tao et al., 2017] Tao, M. W., Srinivasan, P. P., Hadap, S., Rusinkiewicz, S., Malik, J., and Ramamoorthi, R. (2017). Shape Estimation from Shading, Defocus, and

Correspondence Using Light-Field Angular Coherence. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*. 12

[Tateno et al., 2017] Tateno, K., Tombari, F., Laina, I., and Navab, N. (2017). CNN-SLAM: Real-Time Dense Monocular SLAM with Learned Depth Prediction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 74, 76, 79, 84, 86

[Triggs et al., 1999] Triggs, B., McLauchlan, P., Hartley, R., and Fitzgibbon, A. (1999). Bundle Adjustment — A Modern Synthesis. In *Proceedings of the International Workshop on Vision Algorithms, in association with ICCV*. 14

[Tulsiani et al., 2017] Tulsiani, S., Zhou, T., Efros, A. A., and Malik, J. (2017). Multi-View Supervision for Single-View Reconstruction via Differentiable Ray Consistency. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 12

[Ummenhofer et al., 2017] Ummenhofer, B., Zhou, H., Uhrig, J., Mayer, N., Ilg, E., Dosovitskiy, A., and Brox, T. (2017). DeMoN: Depth and Motion Network for Learning Monocular Stereo. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 12, 19, 46, 74

[Usenko et al., 2016] Usenko, V., Engel, J., Stückler, J., and Cremers, D. (2016). Direct Visual-Inertial Odometry with Stereo Cameras. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*. 51, 52, 58

[Vespa et al., 2018] Vespa, E., Nikolov, N., Grimm, M., Nardi, L., Kelly, P. H., and Leutenegger, S. (2018). Efficient octree-based volumetric SLAM supporting signed-distance and occupancy mapping. *IEEE Robotics and Automation Letters*. 41

[von Stumberg et al., 2018] von Stumberg, L., Usenko, V., and Cremers, D. (2018). Direct Sparse Visual-Inertial Odometry using Dynamic Marginalization. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*. 21, 50

[Wang et al., 2018] Wang, C., Buenaposada, J. M., Zhu, R., and Lucey, S. (2018). Learning Depth from Monocular Videos using Direct Methods. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 74

[Weerasekera et al., 2018] Weerasekera, C. S., Dharmasiri, T., Garg, R., Drummond, T., and Reid, I. (2018). Just-in-Time Reconstruction: Inpainting Sparse Maps using Single View Depth Predictors as Priors. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*. 76, 77, 95

[Weerasekera et al., 2017] Weerasekera, C. S., Latif, Y., Garg, R., and Reid, I. (2017). Dense Monocular Reconstruction using Surface Normals. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*. 74, 85, 95, 98

[Weiss et al., 2012] Weiss, S., Achtelik, M. W., Lynen, S., Chli, M., and Siegwart, R. (2012). Real-Time Onboard Visual-Inertial State Estimation and Self-Calibration of MAVs in Unknown Environments. In *Robotics and Automation (ICRA), 2012 IEEE International Conference on*, pages 957–964. IEEE. 50

[Wenzel et al., 2013] Wenzel, K., Rothermel, M., Fritsch, D., and Haala, N. (2013). Image Acquisition and Model Selection for Multi-View Stereo. *Int. Arch. Photogramm. Remote Sens. Spatial Inf. Sci.* 13

[Whelan et al., 2015] Whelan, T., Leutenegger, S., Salas-Moreno, R. F., Glocker, B., and Davison, A. J. (2015). ElasticFusion: Dense SLAM Without A Pose Graph. In *Proceedings of Robotics: Science and Systems (RSS)*. 17

[Whelan et al., 2012] Whelan, T., McDonald, J. B., Kaess, M., Fallon, M., Johannsson, H., and Leonard, J. J. (2012). Kintinuous: Spatially Extended KinectFusion. In *Workshop on RGB-D: Advanced Reasoning with Depth Cameras, in conjunction with Robotics: Science and Systems*. 17

[Whelan et al., 2016] Whelan, T., Salas-Moreno, R. F., Glocker, B., Davison, A. J., and Leutenegger, S. (2016). ElasticFusion: Real-Time Dense SLAM and

Light Source Estimation. *International Journal of Robotics Research (IJRR)*, 35(14):1697–1716. 17, 41, 52, 54, 58

[Wu, 2013] Wu, C. (2013). Towards Linear-Time Incremental Structure from Motion. In *Proceedings of the International Conference on 3D Vision (3DV)*. 13

[Xu et al., 2019] Xu, B., Li, W., Tzoumanikas, D., Bloesch, M., Davison, A., and Leutenegger, S. (2019). MID-Fusion: Octree-Based Object-Level Multi-Instance Dynamic SLAM. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*. 39

[Yang and Deng, 2018] Yang, D. and Deng, J. (2018). Shape from Shading through Shape Evolution. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 12

[Yang et al., 2018] Yang, N., Wang, R., Stückler, J., and Cremers, D. (2018). Deep Virtual Stereo Odometry: Leveraging Deep Depth Prediction for Monocular Direct Sparse Odometry. In *Proceedings of the European Conference on Computer Vision (ECCV)*. 74

[Yao et al., 2018] Yao, Y., Luo, Z., Li, S., Fang, T., and Quan, L. (2018). MVSNet: Depth Inference for Unstructured Multi-View Stereo. In *Proceedings of the European Conference on Computer Vision (ECCV)*. 74

[Yin et al., 2017] Yin, X., Wang, X., Du, X., and Chen, Q. (2017). Scale Recovery for Monocular Visual Odometry Using Depth Estimated with Deep Convolutional Neural Fields. In *Proceedings of the International Conference on Computer Vision (ICCV)*. 76

[Zhang et al., 1999] Zhang, R., Tsai, P.-S., Cryer, J., and Shah, M. (1999). Shape from Shading: A Survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*. 12

[Zhang and Funkhouser, 2018] Zhang, Y. and Funkhouser, T. (2018). Deep Depth Completion of a Single RGB-D Image. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 76

[Zhou et al., 2018] Zhou, H., Ummenhofer, B., and Brox, T. (2018). DeepTAM: Deep Tracking and Mapping. In *Proceedings of the European Conference on Computer Vision (ECCV)*. 20, 74, 75, 92, 93, 111

[Zhou et al., 2017] Zhou, T., Brown, M., Snavely, N., and Lowe, D. G. (2017). Unsupervised Learning of Depth and Ego-Motion from Video. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 74

[Zienkiewicz, 2017] Zienkiewicz, J. (2017). *Dense Monocular Perception for Mobile Robotics*. PhD thesis, Imperial College London. 17, 19