DEEP MULTI-VIEW RGB-D FUSION FOR ROBUST AND ACCURATE 3D PERCEPTION

Presentata da: Andrea Conti

Coordinatore Dottorato

Ilaria Bartolini

Supervisore

Stefano Mattoccia

Co-supervisore

Matteo Poggi

Valerio Cambareri

Paolo Bellavista

Esame finale anno 2025

Abstract

In the last decade, depth sensing has become a prominent technology in fields such as robotics, automotive, mobile, and augmented reality. In such systems an *active* sensor is employed, *i.e.*, a device exploiting illumination to infer the 3D structure of the framed scene. Time-of-Flight sensors are mainly used indoors on mobile devices, while Light Detection and Ranging sensors are employed in automotive for landscape-like scenarios. Despite the reconstruction accuracy of active depth sensors, their technical limitations demand their integration with other sensors to achieve high accuracy and the technical properties required for specific deployments. This PhD thesis aims to deeply analyze existing technologies for depth estimation and active sensor employment with the ultimate goal of improving and innovating the current technological scenario with novel depth sensing frameworks able to overcome the current limitations. First of all, this is achieved through a detailed analysis of the inherent limitations of active sensors and the proposal of effective solutions. Then, the integration with single or multiple RGB sensors is deeply analyzed in different applicative scenarios, improving the current state of the art with innovative frameworks. Eventually, a fully integrated pipeline able to exploit effectively multi-modal information is proposed.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Depth Sensing Applications

Accurate 3D reconstruction is crucial across a wide spectrum of applications, where the ability to precisely capture and replicate the three-dimensional structure of objects and environments opens up many possibilities. In the realm of mixed reality and 3D content creation, for example, the demand for highly detailed shape reconstructions is paramount. These reconstructions enable the seamless integration of digital objects into real-world environments, allowing for immersive experiences in virtual and augmented reality (VR/AR). For instance, a virtual object in an AR application can be accurately placed on a real-world surface, such as a table, if the table's exact dimensions and surface details are known. This level of detail is essential for creating convincing and interactive digital experiences that respond to the real world.

Historical preservation is another field where accurate 3D reconstruction plays a critical role. Museums and cultural heritage organizations increasingly rely on digital modelling to archive and analyze works of art, archaeological sites, and historical artefacts. By creating precise digital replicas, these institutions can preserve the intricate details of fragile or deteriorating objects, ensuring that they remain accessible for future generations. Moreover, these digital models allow for detailed scientific analysis, such as studying the surface wear of ancient sculptures to understand their history and usage. Digital archives also make it possible to share and exhibit rare or delicate items globally, expanding access to cultural heritage.

In robotics and autonomous driving, precise depth estimation is vital for navigation, object recognition, and interaction with the environment. Autonomous vehicles, for instance, need to accurately perceive their surroundings to make safe and informed decisions in real time. This involves detecting and tracking other vehicles, pedestrians, and obstacles, as well as understanding the road's topography. Robotics, on the other hand, often requires 3D reconstruction for tasks like object manipulation, where a robot arm must grasp objects with precision, or for navigation in complex, dynamic environments. In these cases, depth information is crucial for understanding the spatial relationships between objects and planning safe and efficient paths.

In summary, depth sensing applications are diverse and far-reaching, encompassing fields such as cultural heritage, robotics, and mobile technologies. Therefore, it is crucial to develop robust, versatile, and lightweight methodologies to effectively support these varied use cases.

1.2 Active Depth Sensing

Traditionally, active 3D sensing technologies have been the preferred solution for depth sensing applications. LiDAR (Light Detection and Ranging) and ToF (Time of Flight) sensors are the prominent technologies used in this field. These technologies actively scan scenes using modulated laser illumination or structured light patterns to infer depth. LiDAR systems, for example, emit laser pulses and measure the time it takes for the light to return after reflecting off surfaces, thereby calculating the distance to various points in the scene. This approach provides highly accurate depth measurements, making LiDAR an invaluable tool in applications like autonomous driving, where it is critical to detect objects and obstacles at varying distances, even in challenging lighting conditions.

Similarly, ToF sensors emit infrared light and measure the time it takes for the light to come back from surfaces in the scene. This allows for real-time depth estimation, which is particularly useful in applications such as gesture recognition in gaming or user interaction in smart devices. For instance, ToF cameras are integrated into some smartphones and tablets to enable features like facial recognition or augmented reality experiences. By providing real-time depth maps, ToF sensors help these devices accurately track and interpret user movements and interactions with their surroundings.

Finally it's worth mentioning Structured Light techniques. These method evaluates 3D geometry by projecting a known pattern, often grids, stripes, or dot arrays, onto a scene. The way these patterns deform when striking surfaces provides information about the depth and shape of the scene and can lead to accurate depth sensing. Nonetheless, these techniques are hampered by the limited capability of the projector used – *i.e.* they might not work in full daylight or with distant objects.

However, despite their accuracy, active 3D sensing technologies are not without limitations. LiDAR systems, while effective in outdoor environments and at long ranges, tend to be bulky and expensive. Additionally, the mechanical components involved in LiDAR systems, such as rotating mirrors, limit their frame rates and can introduce reliability concerns over time due to wear and tear. ToF sensors, on the other hand, are more compact and have been increasingly integrated into consumer electronics. However, they also face challenges, particularly in environments with strong ambient light, such as outdoor settings. The infrared signals used by ToF sensors can be overwhelmed by sunlight, leading to reduced accuracy or even failure to generate depth data. They also consume significant amounts of energy, which can be a drawback for mobile applications that require prolonged battery life, such as drones or handheld devices. Moreover, both LiDAR and ToF sensors produce sparse depth data, meaning they provide depth information for only a limited number of points in the scene. This sparsity can hinder their effectiveness in applications requiring detailed and continuous depth information across the entire scene and requires supporting techniques to recover global knowledge of the scenario.

1.3 Multi-Modal Depth Sensing

These limitations have prompted the exploration of alternative approaches to support and integrate active technologies. Particularly, passive sensing approaches – *i.e.* using only standard RGB cameras – are extremely interesting for a wide range of reasons. Indeed, unlike active sensors, RGB cameras do not emit any light beam; instead, they rely on ambient light and capture images in a more energy-efficient manner. This makes them ideal for a variety of applications where power consumption, size, and flexibility are critical considerations. For example, in mobile augmented reality applications, the compactness and low power consumption of RGB cameras are crucial for providing a seamless user experience without draining the device's battery too quickly.

Among passive sensing techniques, stereo vision is a widely used approach. It involves using two calibrated cameras to capture images of the scene from slightly different angles. By analyzing the differences between these two images, known as disparity, it is possible to estimate the depth of objects in the scene. This technique restricts the depth estimation problem to a one-dimensional search space, making it computationally efficient compared to other methods. However, stereo vision has its own set of challenges. It requires the cameras to be positioned such that they share the majority of the field of view and they must be calibrated to enforce coplanarity. After such a step, their relative position must not change. This latter limitation may lead to failure if the cameras move due to external agents. Moreover, stereo vision can struggle in scenes with low texture or poor lighting, where it becomes difficult to match corresponding points between the two images.

The most flexible, yet challenging, passive sensing approach involves using a single monocular RGB camera in motion. This method, known as structure-from-motion (SfM), estimates depth by analyzing how objects in the scene move relative to the camera as it changes position. While this approach offers greater flexibility in terms of camera placement and movement, it is computationally intensive and can be less accurate than stereo vision or active sensing techniques. SfM is particularly useful in scenarios where using multiple cameras or active sensors is impractical.

Passive sensing approaches can also be used to support active devices and ameliorate the issues of the latter still retaining their peculiar accuracy. An example of such integration is depth completion. Depth completion algorithms aim to fill in the gaps in depth information, especially when passive sensors are combined with the sparse outputs of active sensors like LiDAR and ToF. However, current depth completion methods face significant challenges. They are often fragile and fail to accurately reconstruct scenes in areas where no depth points are available or when the distribution of sparse points differs significantly from the conditions under which the system was trained. This fragility limits the practical deployment of these methods, particularly in dynamic or complex environments where sensor conditions can change rapidly. The reliance on training data tailored to specific sensors or environments can prevent the seamless application of depth completion techniques across different contexts. For example, a depth completion algorithm trained on data from an expensive, high-resolution LiDAR system may not perform well when applied to data from a lower-cost, lower-resolution active sensor. This limitation poses a challenge for applications that require adaptability, such as autonomous vehicles operating in diverse setups.

1.4 Thesis Proposal

Given these challenges, there is a pressing need to develop more robust and adaptable solutions for depth perception that can effectively integrate the strengths of both active and passive sensing technologies. Indeed, combining data from multiple sensors – such as using LiDAR data to enhance the depth information from RGB cameras – can lead to more accurate and reliable 3D reconstructions. Such advancements would significantly enhance the performance and applicability of 3D reconstruction in critical fields. For instance, in autonomous driving, improved depth perception could lead to safer navigation and more reliable obstacle detection in a wider range of conditions, from bright sunlight to dark, rainy nights. In augmented reality, more accurate and detailed 3D reconstructions would enable richer and more immersive user experiences, where digital objects interact with the real world naturally and believably. Ultimately, the continued development of these technologies will be key to unlocking the full potential of 3D reconstruction across various industries and applications.

This thesis endeavours to deliver a comprehensive and in-depth exploration of the current state of the art in 3D reconstruction technologies, encompassing both passive and active sensing methods. It seeks to address the complexities inherent in these technologies by proposing and delineating solutions aimed at their integration. The study begins with an extensive examination of the existing literature and the various technological advancements available for depth sensing. This includes a critical analysis of the different approaches, highlighting their strengths and limitations. Subsequently, the thesis addresses the primary challenges associated with depth perception, particularly when dealing with sparse depth data. For each identified challenge, effective and innovative solutions are proposed and thoroughly discussed.

In the final part of the thesis, attention shifts to the integration of multi-view cues, leading to the development of a comprehensive and general framework for 3D reconstruction capable of synergistically leveraging multi-view and monocular cues derived from passive RGB cameras in conjunction with sparse depth cues obtained from active sensors. This framework is designed to seamlessly incorporate both passive and active sensor data, offering a robust solution that enhances the accuracy and reliability of depth perception in diverse applications.

Through this research, the thesis aims to contribute significant advancements to the field, offering both theoretical insights and practical methodologies for the integration of diverse sensing technologies in 3D reconstruction.

Chapter 2

Related Work

2.1 Monocular Depth Prediction

Except for a few attempts to solve monocular depth prediction through non-parametric approaches [65], the practical ability to solve this ill-posed problem has been achieved only with the deep learning revolution. At first, deploying plain convolutional neural networks [34] and then, through more complex approaches. Specifically, [38] casts the problem as a classification task, [2] exploits a bidirectional attention mechanism, [73] introduces novel local planar guidance layers to better perform the decoding phase, [112] jointly computes panoptic segmentation to improve depth prediction performance, [115] unifies multiple depth sources to coherently train a neural network to better generalize. The previous methods required a massive quantity of training data to achieve proper performance in unknown environments thus self-supervised paradigms gained much attention. For instance, [46] relies on a supervisory signal extracted from a monocular video stream.

2.2 Depth Completion

Depth completion is a crucial task in computer vision that aims to densify a sparse depth map, typically obtained from an active depth sensor, by filling in missing depth values. This process is essential because multi-modal samples captured by an RGB camera coupled with a LiDAR or ToF sensor often result in a sparse depth map with valid measurements at only a limited number of pixels in the associated RGB image. The challenge of filling these missing depth values to generate a complete and accurate dense depth map is referred to as depth completion.

Approaches to depth completion can be broadly categorized into unguided and RGB-guided techniques. Unguided methods, which operate solely on the sparse depth map without any additional information from an RGB image, generally underperform compared to RGB-guided methods. This performance gap arises because unguided methods lack critical information, such as discontinuities and monocular cues, which can significantly enhance the accuracy of depth estimation [146, 36, 88].

RGB-guided methods, which have become state-of-the-art, utilize an aligned RGB image to guide the depth completion process. These methods can be further classified based on the fusion strategy employed to combine RGB and depth information. Early-fusion techniques directly aggregate the image and sparse depth map in the initial convolutional layers [32, 60, 128], while late-fusion techniques delay this integration until after an initial encoding of each modality [76, 163, 37]. Alternatively, some methods manipulate explicit 3D representations, such as surface normals or 3D point clouds, to directly infer depth [184], while residual models learn to predict the residual difference between a coarse initial depth map and the final dense map [47].

Among the various techniques, Spatial Propagation Networks (SPN) based models have emerged as the most effective. These models learn an affinity matrix that is iteratively applied to propagate and refine depth values across the sparse map. The pioneering works in this area [18, 19] introduced the use of 3×3 local affinity matrices, learned through a UNet architecture, to perform depth refinement. Subsequent advancements, such as the introduction of deformable sampling in the propagation process [99] and attention mechanisms [82], have further improved the performance of these models. Most recently, novel approaches like the Geometric SPN module [162] and unsupervised frameworks for self-supervised depth completion [164] have continued to push the boundaries of what can be achieved with SPN-based methods.

In the broader context of deep learning-based approaches to depth completion, the standard practice involves concatenating the RGB image and sparse depth map, then passing this combined input through a 2D convolutional neural network to predict the dense depth map [92, 136, 57].

Overall, the evolution of depth completion methods highlights the growing sophistication and effectiveness of techniques designed to integrate RGB and depth information, with SPN-based models currently leading the field in terms of performance and versatility.

2.3 Stereo Depth Prediction

Stereo perception is a fundamental task in computer vision, focused on predicting depth from a pair of calibrated, rectified images. Traditionally, this task was approached using hand-crafted algorithms, which have been extensively explored and categorized in the literature. These conventional stereo algorithms relied heavily on priors and handcrafted features to infer dense disparity maps from stereo pairs, with various strategies ranging from local [140] and global reasoning [135] to semi-global techniques [54]. These approaches have been deeply studied and thoroughly surveyed in seminal works such as [121], and they formed the backbone of stereo matching for many years [175, 148, 170, 169, 80, 137, 71, 55, 9].

The advent of deep learning brought a significant paradigm shift in stereo perception, starting with the pioneering work of Zbontar and LeCun [176], which introduced the idea of replacing hand-crafted features in stereo pipelines with learned features. This initial integration of learning-based techniques led to a marked improvement in performance, eventually paving the way for more sophisticated end-to-end stereo networks. These deep learning approaches have since become the dominant paradigm, offering substantial performance improvements over traditional methods [111].

Deep stereo networks are typically divided into two main categories: 2D and 3D architectures. The former relies on an encoder-decoder design, inspired by the U-Net architecture [117], to process stereo pairs and estimate disparity maps [95, 98, 81, 119, 132, 167, 174, 138]. On the other hand, 3D architectures create a cost volume from the features extracted from the image pair and then estimate disparity using 3D convolutions [67, 13, 68, 178, 19, 21, 33, 165, 155, 51, 129]. While 3D approaches tend to offer more accurate results, they also demand significantly higher memory and computational resources.

More recent advancements in deep stereo perception have introduced innovative techniques, such as iterative refinement, inspired by the RAFT model initially developed for optical flow [139], and Vision Transformers, which capture broader contextual information from images [78, 50, 84, 77, 183, 177, 160, 154]. Despite their success, these learning-based methods face challenges, particularly in their need for massive amounts of annotated data for training and their limited generalization capabilities when applied to out-of-domain distributions [179, 11, 3, 180, 85, 25, 157, 145].

To address these limitations, self-supervised techniques have been developed, enabling the training of deep stereo models without the need for ground-truth annotations. These methods typically leverage photometric losses on stereo pairs or video sequences or incorporate traditional algorithms and confidence measures [186, 143, 142, 72, 156, 22, 106, 141, 4]. Furthermore, some approaches have been designed to perform continuous self-supervised adaptation, allowing models to cope with domain-shift issues by continuously adjusting to the input data [143, 109, 110].

Overall, the evolution of stereo perception has been marked by a transition from traditional, hand-crafted methods to sophisticated, learning-based approaches. These advancements have not only improved the accuracy and efficiency of disparity estimation but have also introduced new challenges, particularly in terms of data requirements and generalization. Ongoing research continues to explore self-supervised and adaptive techniques to overcome these challenges and further enhance the capabilities of stereo-perception systems.

2.4 Multi-View Stereo Depth Prediction

Multi-view depth sensing, essential for 3D reconstruction, extends the principles of stereo matching to an arbitrary number of images captured from known viewpoints. This task aims to recover the 3D structure of scenes, whether for objects or environments like indoor spaces, by leveraging overlapping 2D projections of the 3D world. Traditional methods approached this problem through triangulation and manually engineered features, producing dense reconstructions through voxel-based methods, surface evolution, patch matching, or depth map estimation [131, 147, 126, 75, 39, 12, 41, 123]. Among these, the depth map-based approach has proven the most practical and efficient, computing multiple per-view depth maps and later projecting them into 3D space to generate a point cloud. This method offers advantages in memory footprint and processing time and has been widely adopted in modern deep learning-based multi-view stereo (MVS) architectures.

With the rise of deep learning, MVS has seen significant advancements, primarily through cost-volume-based methods. These methods build a 3D cost volume by integrating pixel-matching scores from multiple views, guided by depth hypotheses that span the scene's depth range. Regularization is then performed using 3D convolutional layers to produce a depth map aligned with the RGB view [171]. The first major breakthrough in this direction was MVSNet [171], which warps multi-view images to construct a 3D cost volume relative to the reference camera. Despite its success, MVSNet's reliance on 3D convolutions resulted in high memory and time consumption. Subsequent methods sought to alleviate these constraints through innovative strategies. For example, R-MVSNet [172], D^{2}HC-RMVSNet [161], and AA-RMVSNet [158] replaced 3D convolutions with 2D GRU units to reduce computational overhead, while other approaches like CAS-MVSNet [48] and UCSNet [17] introduced multi-stage architectures capable of coarse-to-fine inference or used pyramidal cost volumes for more efficient processing.

In addition to these methods, volumetric-based approaches offer another pathway for multi-view depth perception, particularly in scenarios requiring global scene structure reconstruction. These methods back-project rays of deep features into a global voxel grid, using 3D recurrent layers to refine and eventually extract the scene's mesh structure [134, 10, 133, 116, 23]. While effective, these techniques often require processing the entire scene simultaneously, making real-time or online reconstruction challenging. To address these issues, some methods incorporate complex sparsification [134] or attention mechanisms [10] to approximate real-time updates, albeit with increased complexity.

Another significant advancement in multi-view depth sensing is the introduction of neural rendering techniques. These methods learn an implicit representation of the 3D structure and appearance of a scene as a continuous 5D radiance field. Initially popularized for novel view synthesis and geometric reconstruction, neural rendering has also been integrated into MVS pipelines. For instance, MVSNeRF [15] combines cost volumes with a sparser set of images, while GeoNeRF employs cascade cost volumes [64], and Depth-Supervised NeRF [31] integrates sparse depth supervision into the 3D reconstruction process. However, these approaches are not suitable for real-time applications, as they often require test-time optimization and cannot match the online capabilities of voxel grid-based methods.

In summary, multi-view depth sensing has evolved from traditional triangulation methods to sophisticated deep learning frameworks that leverage cost volumes, volumetric representations, and neural rendering. These advancements have significantly improved the accuracy and efficiency of 3D reconstruction, although challenges remain in balancing computational demands with real-time performance.

Chapter 3

Sparse Depth Confidence Analysis

3.1 Introduction

As highlighted in Chapter 2 the 3D structure of a sensed environment can be inferred through passive and active sensing technologies. The former has been deployed for decades using stereo [121, 111], structure-from-motion [122] or multi-view-stereo [127]. Each of these methods has its flaws and constraints. For instance, stereo depth perception requires two calibrated cameras and struggles where the scene lacks texture. On the other hand, active sensing relies on specialized sensors, and in the case of LiDARs (Light Detection And Ranging), flooding the scene with a laser beam and computing the distance of each point by measuring the travelling time of the ray. Despite being accurate, LiDARs struggle, for instance, when sensing not Lambertian surfaces due to multi-path interference or subsurface scattering. Moreover, the resulting point cloud is sparse and not coupled with any visual information. Thus, it is common to jointly use it with a standard camera and project the LiDAR point cloud over the camera image plane, resulting in a sparse depth map. However, this procedure raises a fundamental issue due to the different points of view of the two devices and the intrinsic sparsity of the LiDAR's output. Specifically, it leads to wrong depth values in the final RGB-D image, as shown in Figure 3.1. Therefore in this chapter, the LiDAR depth map filtering problem is explicitly tackled and a technique to solve such a problem without input information other than the RGB-D sample is

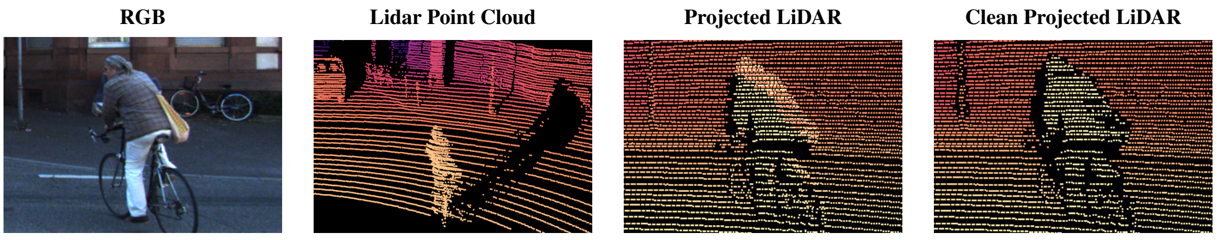


Figure 3.1. **LiDAR Projection Over a RGB Camera.** The sparse point cloud of a LiDAR sensor can be projected over the image plane of an RGB camera, however, this process usually leads to errors due to occlusion and sparsity that must be removed.

proposed.

To date, LiDAR sensors are massively used to source ground-truth data in primary scientific datasets, such as KITTI [43], DrivingStereo [166], and many others [14, 151, 42], powering state-of-the-art deep learning techniques in computer vision. However, when projecting the depth map over the camera image plane, the issue mentioned above is usually tackled by enforcing consistency between the depth map and the values obtained through a stereo algorithm [55, 146] or deep stereo network [166]. Nonetheless, these approaches have some flaws as well: i) they require stereo cameras during acquisition with the LiDAR and ii) they do not filter out only the LiDAR errors, but also the stereo algorithm errors, thus affecting the cleaned data with the intrinsic limitations of the stereo setup (*e.g.* filtering out depth measures in textureless areas). Despite these limitations, such approaches are viable when pre-processing a dataset beforehand is feasible. However, they might not be applicable in real applications where a stereo setup is unavailable or when the depth labels for training are needed at runtime, for instance, to adapt stereo networks online [109].

To address all of these limitations, in this chapter a fast deep neural network framework is proposed, trained in an unsupervised manner, capable of predicting accurately the uncertainty of the projected LiDAR sparse depth map using a simple RGB-D setup. Such uncertainty, or complementary confidence, can be then deployed to filter out the errors, for instance, by enforcing a percentile to be removed or by using an absolute threshold. To this aim a peculiar supervision scheme enabling unsupervised training – thus not requiring any expensive ground-truth depth annotation – is deployed.

Moreover, the results provided in this chapter are upheld by experimenting over two splits of KITTI [43] i) assessing the effectiveness of this method versus existing alternatives [185, 35], constantly outperforming even supervised techniques [35] and ii) illustrating how filtering LiDAR depth maps with this approach yields consistent improvements in applications such as depth completion [92, 91], guided stereo [108] and sensor-guided optical flow [103] frameworks.

3.2 Proposed approach

In this section the reasons giving rise to outliers in LiDAR depth maps are deeply analyzed, and then the framework specifically designed to filter them out is described.

3.2.1 Outliers in LiDAR depth maps

There exist two leading causes of errors in LiDAR depth maps: i) erroneous measurements consequence of the LiDAR technology, for instance, originated by reflective or dark surfaces – over which the behavior of the emitted beams become unpredictable – or by other technological limitations (for instance, the mechanical rotation performed by the Velodyne HDL-64E used in KITTI [44]) and ii) incorrect projection of depth values near object boundaries, due to

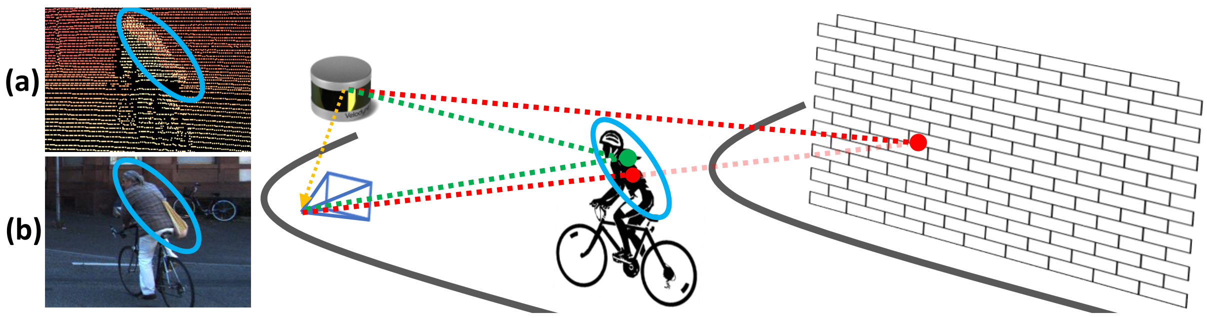


Figure 3.2. **Outliers formation process due to occlusions.** When a LiDAR and an RGB camera acquire from different viewpoints, projecting the point cloud into a depth map (a) on the image (b) introduces outliers (blue oval), e.g. points visible by the LiDAR occluded to the camera (red), yet projected near foreground points visible to both (green).

occlusions originated by the different viewpoints of the LiDAR sensor and the RGB camera.

Figure 3.2 provides an intuitive overview of the second issue: concerning an urban scene acquired by a LiDAR and a camera, with a cyclist in the foreground and a wall far in the background (example available in the KITTI dataset). The different position of the two sensors causes some background regions to be visible to one of the two while occluded to the other. For instance, the red point on the wall is perceived by the LiDAR, but the cyclist occludes it in the image acquired by the camera. On the other hand, regions in the foreground are visible to both sensors, as the green point on the cyclist. When projecting LiDAR points into a depth map, specifically by mapping them over the camera image plane, depth values from occluded points in the background are projected into 2D pixel coordinates of foreground regions. The sparse nature of LiDAR points makes them visible in the resulting depth map shown in Figure 3.2 (a), labeling the RGB image acquired by the camera (b) with wrong depth values in regions occluding the background points sensed by the LiDAR.

This bleeding effect, agnostic to the sensor accuracy, occurs in all LiDAR-camera setups, including the depth maps made available by the KITTI completion dataset [146], thus affecting the methods competing over the completion benchmark itself. Therefore, a carefully unsupervised deep learning framework is designed to deal with this issue, only using the RGB image coupled with the LiDAR depth.

3.2.2 Architecture

Figure 3.3 depicts the architecture developed to estimate LiDAR confidence; it is composed of a multi-scale encoder and a prediction MLP (Multi Layer Perceptron) head. The encoder is fed with the concatenation of the RGB image and the sparse LiDAR depth map.

Features extraction. The encoder of the network consists of three 3×3 conv2D layers with 32, 64 and 64 output channels, followed by five encoding blocks made by a 2×2 MaxPool operator with stride 2 and two 3×3 conv2D layers having the same number of output channels, respectively 128, 256, 512, 512 and 512 for the five blocks. The encoder extracts features at

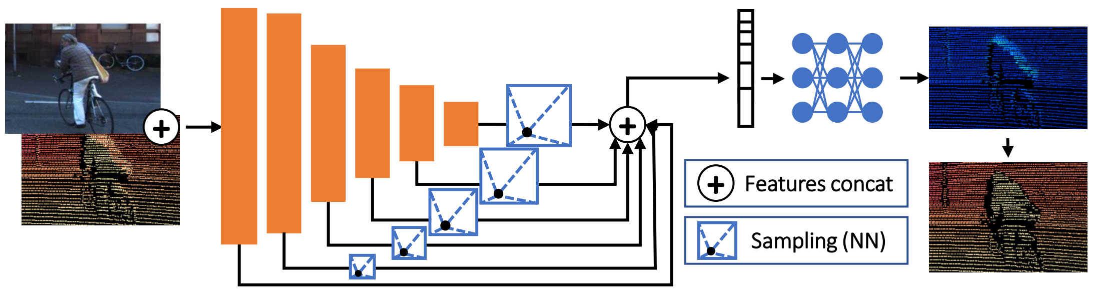


Figure 3.3. **Proposed architecture.** A convolutional encoder (orange) extracts features at different resolutions. We query features for each pixel with a valid LiDAR value and concatenate them (+) in a vector, fed to an MLP (blue) to estimate confidence.

full resolution from the first three conv2D layers, and at five more resolutions from the five aforementioned blocks, respectively at $\frac{1}{2}$, $\frac{1}{4}$, $\frac{1}{8}$, $\frac{1}{16}$ and $\frac{1}{32}$. Extracting features at multiple scales allows for increasing the receptive field and considering complex features from large areas in the image. This strategy, for instance, allows evaluating the shape of a large complex object such as a car to tell which depth measurements are outliers. Furthermore, the multi-scale extraction plays a crucial role since an MLP head – that does not consider the local information around each feature vector – is applied. Leaky ReLUs follow each convolutional/fully connected layer.

Confidence estimation. Once multi-scale features have been extracted, a sampling process occurs at each scale (using nearest-neighbor interpolation to sample at smaller scales) to compose a feature vector of size $64 + 128 + 256 + 512 + 512 + 512$ for each depth measurement contained in the LiDAR sparse depth map. The MLP head infers confidence by processing high-level information regarding the image context and the original sparse depth distribution. Such an estimation comes in the form of variance, similar to predictive uncertainty strategies [66] (the lower, the more confident). It is worth noting that a plain convolutional decoder could be used in place of an MLP. However, this specific task does not require generating a dense output. Thus a simple MLP can estimate the confidence only for the meaningful pixels in the input depth map.

3.2.3 Unsupervised learning procedure

Proxy labels generation. To train the model can be observed that nearby pixels should share similar depth values [107] except for points near discontinuities. Therefore, we take into account for each LiDAR depth point d the other valid depth points inside a patch $P_N(d)$ of size $N \times N$ and compute a *proxy label* representing a plausibly correct depth for each original LiDAR depth value available:

$$d_d^* = f(\{d' : d' \in P_N(d),\ d' > 0\}) \tag{3.1}$$

To speed up the training procedure and obtain a faster convergence, a fixed f function is used. Precisely, the minimum depth among the valid depths contained in the patch is computed. Another approach might be to use the average of the valid depths in the patch. However, in the presence of occlusions, this strategy would cause both background and foreground depths to be detected as outliers since both are far from the average depth occurring between the two. In contrast, using the minimum depth value correctly selects the foreground points as reliable in the presence of occlusions. As a drawback, it may lead to indiscriminately detecting as outliers most of the pixels in the background, even if not occluded. However, in practice, it will be shown that the network learns to ameliorate this issue and that the patch size affects the performance to a lesser extent. To support the effectiveness of the proposal, in Sec. 3.3 the behaviour using the minimum, the average, and the KITTI ground-truth depth itself as proxy labels are compared.

Loss function. The LiDAR depth confidence d is modeled assuming a Gaussian distribution centred in the proxy label d^* with variance σ^2, the latter encoding the depth uncertainty. Thus, during training, the network learns to regress σ by minimizing the negative log-likelihood of the distribution.

$$\mathcal{L}_G = -\ln\left(\frac{1}{\sigma\sqrt{2\pi}}e^{-\frac{(d-d^*)^2}{2\sigma^2}}\right) \tag{3.2}$$

We can rewrite (3.2) as follows:

$$\mathcal{L}_G \approx \ln(\sigma) + \frac{(d-d^*)^2}{2\sigma^2} \tag{3.3}$$

However, (3.3) becomes unstable when $\sigma \ll 1$ since it leads to enormous loss values hampering the learning procedure. Therefore, the network output is constrained to $\sigma \geq 1$, obtaining the following additional advantages. The regularization term $\ln(\sigma)$ is 0 when σ reaches its minimum value ($\sigma = 1$). Besides, small σ values no longer magnify $(d-d^*)^2$, an unwelcome event since the network aims to minimise this term as much as possible.

Nonetheless, by taking into account the derivative of (3.3)

$$\frac{d}{d\sigma}\mathcal{L}_G = \frac{1}{\sigma} - \frac{(d-d^*)^2}{\sigma^3} \tag{3.4}$$

and solving for the minimum, can be obtained that $\sigma_{min} = |d-d^*|$. Hence, to constraint the minimum of the loss function in the chosen domain (i.e. $\sigma \geq 1$), there is also the need to enforce $(d-d^*)^2 \geq 1$ in (3.3). Consequently, the final loss becomes:

$$\mathcal{L}_{G*} = \ln(\sigma) + \frac{(|d-d^*|+1)^2}{2\sigma^2}, \quad \sigma \geq 1 \tag{3.5}$$

In the next section the performance of (3.3), with $\sigma \geq 1$, and (3.5) is compared to measure the impact of this strategy.

Window Size	CV split [146]	142 split [96]	Avg.
5×5	0.1517	0.2291	0.1904
7×7	0.1318	0.1975	0.1647
9×9	0.1292	0.1985	0.1639
11×11	0.1316	0.1999	0.1658
13×13	0.1372	0.2055	0.1714

(a)

Sampled Feat.	CV split [146]	142 split [96]
$\frac{1}{32}$	0.2436	0.3220
$\frac{1}{32}+\frac{1}{16}$	0.1941	0.2597
$\frac{1}{32}+...+\frac{1}{8}$	0.1680	0.2302
$\frac{1}{32}+...+\frac{1}{4}$	0.1486	0.2109
$\frac{1}{32}+...+\frac{1}{2}$	0.1362	0.2010
All	0.1292	0.1985

(b)

	Proxy	Head	Loss	CV Split [146]	142 Split [96]	Avg.
	d^*_{avg}	Decoder	$\mathcal{L}_L$	0.5286	0.8796	0.7041
	d^*_{avg}	Decoder	$\mathcal{L}_{L^*}$	0.2023	0.3730	0.2877
	d^*_{avg}	Decoder	$\mathcal{L}_G$	0.3692	0.5064	0.4378
	d^*_{avg}	Decoder	$\mathcal{L}_{G^*}$	0.1805	0.2403	0.2104
‡	d^*_{avg}	MLP	$\mathcal{L}_L$	0.5890	0.9624	0.7757
	d^*_{avg}	MLP	$\mathcal{L}_{L^*}$	0.1565	0.2649	0.2107
‡	d^*_{avg}	MLP	$\mathcal{L}_G$	0.2430	0.2811	0.2621
	d^*_{avg}	MLP	$\mathcal{L}_{G^*}$	0.1546	0.2197	0.1872

(c)

	Proxy	Head	Loss	CV Split [146]	142 Split [96]	Avg.
‡	d^*_{min}	Decoder	$\mathcal{L}_L$	0.5569	0.7641	0.6605
	d^*_{min}	Decoder	$\mathcal{L}_{L^*}$	0.1558	0.3693	0.2626
‡	d^*_{min}	Decoder	$\mathcal{L}_G$	0.8715	1.2760	1.0738
	d^*_{min}	Decoder	$\mathcal{L}_{G^*}$	0.1382	0.2548	0.1965
‡	d^*_{min}	MLP	$\mathcal{L}_L$	0.6457	1.1370	0.8914
	d^*_{min}	MLP	$\mathcal{L}_{L^*}$	0.1267	0.2446	0.1857
	d^*_{min}	MLP	$\mathcal{L}_G$	0.4801	0.6744	0.5773
	d^*_{min}	MLP	$\mathcal{L}_{G^*}$	0.1292	0.1985	0.1639

(d)

Table 3.1. **Experimental Results – ablation study.** In these ablation studies different parameters affecting the behaviour of the framework are investigated. Respectively from left to right, the impact of the window size used to extract proxy labels d^*, multi-resolution sampling and the three main design strategies (proxy labels, decoding head and loss function). The AUC values on the KITTI CV [146] and 142 [96] splits are reported. In each sub-table, the best, second-best, and third-best are highlighted. ‡ means 10^{-6} learning rate to avoid divergence.

3.3 Experimental Results

In this section, the effectiveness of the proposed framework in comparison with state-of-the-art is assessed.

3.3.1 Evaluation dataset and training protocol

The framework is evaluated on the KITTI dataset [44], a standard benchmark in the field providing both images and raw LiDAR depth maps obtained from 151 video sequences, as well as accurate ground-truth labels. Such annotation, based on semi-automatic procedures, is highly time-consuming and requires stereo images. For instance, the KITTI completion dataset [146] provides ground-truth maps obtained by accumulating 11 consecutive LiDAR pointclouds. Then, outliers (due to noise or moving objects) are removed by looking at inconsistency with respect to the output of the Semi-Global Matching (SGM) stereo algorithm [55]. This labeling strategy allows generating massive data (about 44.5K samples, 93K if considering stereo pairs) with the side-effect of losing several labels where LiDAR and SGM are not consistent. An even more accurate and laborious strategy consists of manually refining the labeling process, as done for the KITTI 2015 stereo dataset [96]. In this case, 3D CAD models have been used to obtain an accurate annotation for cars at the cost of much more effort (indeed, only 200 annotated samples are available).

In the following experiments, two different splits are evaluated:

- **CV split**: composed of 1K images from the KITTI Completion Validation set [146]

Method	CV split [146]	142 split [96]
Surface [185]	0.7014	1.4446
$\lvert d - d^*_{avg}\rvert$	0.2161	0.2641
$\lvert d - d^*_{min}\rvert$	0.2053	0.2665
Ours	0.1292	0.1985
Optimal	0.0271	0.0393

(a) unsupervised

Method	CV split [146]	142 split [96]
NCNN-Conf-L1 [35]	0.1530	0.3093
NCNN-Conf-L2 [35]	0.4624	0.8329
pNCNN-Exp [35]	0.8131	1.5430
Ours †	0.1172	0.2094
Optimal	0.0271	0.0393

(b) supervised

Table 3.2. **Experimental results – outliers removal.** AUC values on the KITTI CV [146] and 142 [96], comparing with unsupervised (a) and supervised (b) methods. † means $d^* =$ ground-truth depth. In each sub-table, the best , second -best, and third -best are highlighted.

- **142 split**: a subset of 142 images from KITTI 2015 overlapping with KITTI completion, thus providing both raw LiDAR depth maps and manually annotated ground-truth

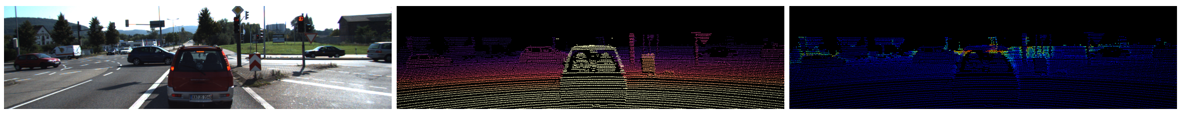
Models are trained using the 113 video sequences that do not overlap with any of the two splits. Moreover, since the framework quickly converges, just a few samples are required to achieve state-of-the-art results; thus, a subset of about 6K (one every five frames) samples is used. Nonetheless, for a fair comparison, the supervised competitor [35] is re-trained over the whole available training set, yet avoiding overlapping with the 142 split (over which the weights released by the authors have been trained on).

The framework is trained for 3 epochs only, using the ADAM optimizer with a learning rate of 10^{-5}, with batches of 2 samples made of 320×1216 crops on a single NVIDIA RTX 3090. To train the models by Eldesokey et al. [35] the authors' code following the recommended settings is used.

3.3.2 Outliers detection

Evaluation metrics. Let us start by evaluating the performance of the method and existing approaches [35, 185] at detecting outliers in LiDAR depth maps. Purposely, the Area Under the sparsification Curve (AUC) is computed, a standard metric for this task [59, 106, 104]. Namely, for each depth map in the dataset, pixels with both LiDAR and ground-truth depth available are sorted in increasing order of confidence score and gradually removed (2% each time). The Root Mean Squared Error (RMSE) over the remaining pixels is computed each time and a curve is drawn. The area under the curve quantitatively assesses the effectiveness at removing outliers (the lower, the better). Optimal AUC is obtained by removing pixels in decreasing order of depth error.

Ablation study. In Table 3.1 (a), the effect of the window size used to compute proxy labels d^* and multi-resolution features sampling on the final model is measured. For these experiments, Eq. 3.5 is used as a loss function. Table 3.1 (a) shows that a 9×9 patch allows to train the framework at its best, while models trained on proxy labels computed on smaller or larger windows gradually achieve worse results. Intuitively, tiny windows lead the network toward over-fitting on high confidence values (i.e., more LiDAR values are likely to be close to d^*). While using larger windows leads to the opposite behavior (i.e., most LiDAR values will have



(a) RGB Image (b) LiDAR Depth (c) Estimated Confidence

Figure 3.4. **Qualitative results (142 split).** RGB image (a), raw LiDAR depth (b) and estimated confidence maps (c).

Filtering Method	RMSE (20.85% filt.)	% filtered (∼0.68 RMSE)
SGM [55]	0.6886	28.85
Ours	0.2085	3.90

Filtering Method	RMSE (20.08% filt.)	% filtered (∼0.92 RMSE)
Reversing [4]	0.9171	20.08
Ours	0.2518	2.39

Filtering Method	RMSE (1.47% filt.)	% filtered (∼1.42 RMSE)
Surface [185]	1.4189	1.47
Ours	1.1841	0.97

Filtering Method	RMSE (12.99% filt.)	% filtered (∼0.72 RMSE)
LiDARStereoNet [20]	0.7176	12.99
Ours	0.3261	3.75

Table 3.3. **Experimental results – semi-automatic annotation.** Annotation performance of the proposed method on the 142 split, either by fixing the % of filtered points or the final RMSE achieved by the competitors.

a high difference compared to d^* and drive, for instance, the network to predict low confidence in the presence of any depth discontinuity). Table 3.1 (b) reports that, not surprisingly, the best results are obtained by sampling features from any resolutions, from full to $\frac{1}{32}$. Then, the impact of the different design choices used to implement the model are measured in Tables 3.1 (c-d). Specifically: i) different proxy label generation functions (d^*_{min} and d^*_{avg} for respectively the minimum and the average among the valid depths in the patch), ii) the prediction head (MLP or a five layers decoder with skip connections, where each layer has two 3×3 convolutional blocks followed by 2×2 nearest-neighbor upsampling) and iii) the distribution function underlying the loss term between Gaussian $\mathcal{L}_G$, Laplacian $\mathcal{L}_L$ [66] and the modified version of both $\mathcal{L}_{G^*}$ and $\mathcal{L}_{L^*}$ as described in Sec. 3.2.3. Table 3.1 (c-d) collects results by several variants of the framework on both CV and 142 splits, using a 9×9 window and sampling features at all resolutions following the outcomes from Table 3.1 (a-b). The scores are generally lower on the CV split because of the many missing labels from ground-truth maps obtained semi-automatically, resulting in several outliers being missing in the AUC evaluation. Can be also noticed that $\mathcal{L}_{L^*}$ and $\mathcal{L}_{G^*}$ always outperform their original counterparts, assessing the quality of the proposed formulation. Moreover, the synergy between the minimum proxy label strategy and the MLP yields the best results overall. Finally, even if both $\mathcal{L}_{L^*}$ and $\mathcal{L}_{G^*}$ are competitive, $\mathcal{L}_{G^*}$ has been chosen since it leads to the best overall results.

Comparison with state-of-the-art. Table 3.2 reports a comparison with existing approaches, namely Surface [185] and NCNN variants [35] on both splits, grouping unsupervised and non-learned methods on the left (a) and supervised ones on the right (b). Since the proposed framework can be trained on ground-truth labels as well, this additional experiment is reported, marked with †, to compare it with supervised methods directly. While this slightly improves the performance on the CV split, which is labeled with the same semi-automatic procedure of the training set, it leads to worse results on the accurate ground-truth maps of the 142 split. This outcome is not surprising since several outliers do not have a corresponding ground-truth value on KITTI CV and are never observed during supervised training over it. In contrast, as reported in the table, the unsupervised strategy is intrinsically unaffected by this bias.

Overall, the framework turns out the best approach, both when trained with and without ground-truth supervision. Indeed, even when trained in an unsupervised manner, it already outperforms supervised methods [35]. Moreover, can be noticed that the absolute difference between LiDAR values and proxy labels is already a good cue to remove outliers, easily outperforming Surface [185] and often being better than supervised approaches [35]. The proposed framework learns to leverage such proxy labels and steadily exceeds their limitations, leading to even better results. Focusing on the former Table 3.2 (a), can be noticed how [185] performs poorly at sparsification. Indeed, Surface performs a binary classification of inliers and outliers, removing only a small set of pixels (respectively 1.60% and 1.47% of the pixels with available ground-truth on CV split and 142 split), yet leaving many outliers on stage. Nonetheless, since a binary method is penalized by AUC evaluation, a more fair comparison is provided in Sec. 3.3.3.

Concerning supervised methods in Table 3.2 (b), can be interestingly noticed that among NCNN variants, the one performing better at modeling uncertainty after completion according to [35], i.e. pNCNN-Exp, is the worst at detecting outliers in the input. On the contrary, NCNN-Conf-L1 is the best variant on raw LiDAR – although always outperformed by the proposed approach, either supervised or unsupervised.

The superior accuracy achieved comes at the cost of slightly higher complexity. the network counts 16M weights versus the 300K of NCNN variants [35], leading to higher runtime on both 3090 and Jetson TX2 GPUs – respectively 0.02 and 1.02 seconds by the model versus 0.01 and 0.31 required by NCNN variants [35]. However, the proposed model achieves better results and does not require any ground-truth depth label for training. For completeness, the runtime required by Surface [185] has been analyzed. Although implemented on CPU, thus not directly comparable with the other methods, [185] takes, respectively, 0.43 and 2.63 seconds on the same desktop PC equipped with a 3090 GPU – and an i9-10900X – and the Jetson TX2 CPU.

Figure 3.4 shows qualitative examples of confidence maps estimated by the unsupervised framework.

3.3.3 Applications

Finally, the unsupervised model's impact on some relevant applications making use of LiDAR data is evaluated.

Semi-automatic annotation. The first direct application consists of filtering LiDAR depth maps to obtain accurate, per-pixel depth annotations. Semi-automatic processes [146] usually rely on an external stereo setup and check for consistency between LiDAR values and disparity maps. The proposed model is compared to this approach, either using a hand-crafted algorithm [55] or state-of-the-art self-supervised stereo networks [4], Surface [185] and a LiDAR-stereo fusion framework [152]. The comparison is performed on the 142 split since it provides manually annotated and refined ground truth, in contrast to the CV split obtained semi-automatically. The experiment is limited to single depth map filtering, not accumulating point clouds over time to avoid issues with moving objects. When filtering using stereo methods, the LiDAR depth is converted into disparity and pixels having a difference with the stereo disparity > 1 are

Model	LiDAR filtering	Removed points (%)	RMSE (mm)	MAE (mm)	iRMSE (1/km)	iMAE (1/km)
Self-Sparse-to-Dense [91]	None	None	1102.062	303.007	4.316	1.670
Self-Sparse-to-Dense [91]	Surface [185]	3.74	974.443	295.587	4.313	1.665
Self-Sparse-to-Dense [91]	Ours	1.20	959.100	288.457	4.187	1.640

(a)

Model	LiDAR filtering	Removed points (%)	RMSE (mm)	MAE (mm)	iRMSE (1/km)	iMAE (1/km)
Sparse-to-Dense [92]	None	None	676.061	274.109	3.097	1.705
Sparse-to-Dense [92]	Surface [185]	3.74	703.597	276.701	3.087	1.689
Sparse-to-Dense [92]	Ours	0.70	647.473	270.554	3.060	1.695

(b)

Model	LiDAR filtering	Removed points (%)	RMSE (mm)	MAE (mm)	iRMSE (1/km)	iMAE (1/km)
PENet [57]	None	None	593.196	178.869	2.242	0.940
PENet [57]	Surface [185]	3.74	616.753	182.139	2.285	0.943
PENet [57]	Ours	0.70	569.449	177.057	2.223	0.936

(c)

Table 3.4. **Experimental results – Depth Completion.** Results on CV split by different completion models processing LiDAR filtered according to different strategies. Range: 50m.

removed.

In Table 3.3, a sub-table for each competitor is reported, measuring the percentage of pixels with both available LiDAR and ground-truth values that are discarded, as well as the filtered RMSE. The RMSE without filtering is 2.5698 meters. Since the four competitors rely on a binary criterion to remove outliers, this experiment is committed to i) remove the same amount of pixels they do and prove that the proposed framework better reduces the error, ii) reduce the RMSE to the same value as the competitors and prove that the framework can achieve such an error by removing fewer points. Thus, in each comparison, respectively i) is removed the same percentage of the competitor and measured the final RMSE (first column), ii) are filtered pixels as long as the same RMSE of the competitor is obtained and is measured the % of pixels removed to obtain it (second column). The proposed method consistently achieves a much lower error when filtering the same percentage of pixels as the competitors. Moreover, it can reach the same final RMSE by removing a fraction of points, i.e. about 7-8 times less compared to stereo methods [55, 4] yet not requiring two cameras as they do. Moreover, by committing to a single fixed threshold as one would do in a real application – e.g., by constantly removing only 5% pixels – the proposed model outperforms all the competitors, with 0.5850 RMSE. Finally, can be noticed how leveraging stereo matching generally removes a high percentage of points (20-30%) because of the several regions where stereo methods struggle, such as occlusions or untextured regions. In contrast, Surface [185] removes very few points but yields a significantly higher RMSE.

Self-supervised/supervised depth completion. In this section is shown that filtering outliers improves the performance of networks for depth completion – the most iconic task performed starting from LiDAR depth maps – without specifically retraining either the proposed framework or the depth completion network. Following [185], Table 3.4 shows results achieved by the Sparse-to-Dense framework – using the weights released by the authors trained either without (a) [91] or with (b) [92] supervision – when processing inputs that have been filtered through unsupervised techniques like ours and Surface [185]. Standard depth completion metrics are used, such as Root Mean Squared Error (RMSE) and Mean Absolute Error (MAE), inverse RMSE and inverse MAE on points up to 50m, to focus on the foreground objects (mostly affected by the outliers). Concerning the self-supervised variant (a), can be noticed how filtering with Sur-

Model	LiDAR Filtering	Removed Points (%)	> 2 (%)	> 3 (%)	> 4 (%)	> 5 (%)	MAE (px)
PSMNet-ft-gd-tr	None	None	5.14	3.39	2.69	2.29	1.08
PSMNet-ft-gd-tr	Surface [185]	4.37	4.75	3.01	2.34	1.95	1.01
PSMNet-ft-gd-tr	Ours	25.00	4.60	2.80	2.13	1.75	0.94

Table 3.5. **Experimental results – Guided Stereo.** Results on 142 split, with PSMNet weights provided by [108] and guided with LiDAR filtered according to different strategies.

Guide Source	LiDAR Filtering	Removed Points (%)	EPE (px)	Fl (%)	Density (%)
Ego +RIC +MaskRCNN [52]	None	None	0.80	2.35	3.16
Ego +RIC +MaskRCNN [52]	Surface [185]	4.37	0.74	2.35	3.06
Ego +RIC +MaskRCNN [52]	Ours	7.00	0.73	2.17	2.93

(a)

Guide Source	LiDAR Filtering	Removed Points (%)	EPE (px)	Fl (%)
guided-QRAFT [103]	None	None	2.08	5.97
guided-QRAFT [103]	Surface [185]	4.37	2.07	5.98
guided-QRAFT [103]	Ours	7.00	2.05	5.82

(b)

Table 3.6. **Experimental results – Sensor-Guided Optical Flow.** Results on 142 split. (a) Accuracy of flow hints, obtained from LiDAR filtered according to different strategies, (b) accuracy of guided QRAFT [103] (CTK).

face [185] improves all metrics by removing nearly 4% of the total pixels with available LiDAR values. Concerning the proposed method, can be achieved a larger improvement by limiting this percentage to 1.20%, hinting that more precise filtering of the outliers, yet limited to fewer pixels, is more effective for the depth completion task. This is confirmed by experiments on the supervised variant (b): in this case, using Surface [185] only improves inverse metrics, while the proposed method always improves all metrics by removing 0.70% pixels only, resulting slightly worse only in iMAE compared to Surface [185]. Finally, 3.4 (c) experiments with PENet [57], a state-of-the-art framework for supervised completion, which confirms the previous findings.

Guided Stereo Matching. The proposed framework can also boost the performance of a sensor fusion pipeline combining passive stereo with LiDAR sensors filtering raw data. Purposely, the chosen framework is guided stereo [108] (since it does not explicitly take into account noise, differently from [20]). In Table 3.5 is reported the accuracy yielded by PSMNet-ft-gd-tr – the model provided by the authors – on the 142 split and the percentages of pixels with errors larger than 2, 3, 4 and 5, together with MAE as in [108]. While Surface [185] can slightly improve all metrics by removing less than 5% of the total pixels with available LiDAR value, by filtering a more significant amount of pixels with the proposed method, up to 25%, the proposed method can further improve and achieve the best accuracy. Interestingly, the guided stereo framework has the opposite behavior with respect to depth completion, as it benefits more from strict filtering.

Sensor-Guided Optical Flow. Finally, the framework is exploited to improve the performance of the Sensor-Guided Optical Flow pipeline [103]. It combines flow hints sourced using a LiDAR sensor with a deep optical flow network by filtering LiDAR points before hints computation. Table 3.6 collects both the accuracy of flow hints (a) and the final results achieved by QRAFT weights trained on Chairs, Things and KITTI (CTK), as provided by the authors of [103]. On top, can be noticed how Surface [185] reduces the flow end-point error (EPE) of the computed hints, yet it cannot reduce the number of outliers with an error larger than 3 pixels or 5% (Fl). In contrast, by removing 7% least confident pixels, the proposed method can effectively improve all metrics. At the bottom, are reported the results achieved by guided QRAFT by using filtered hints. The impact of filtering is lower compared to other depth-related tasks. Indeed, in this task, LiDAR points are only one of several sources of errors, among camera pose estimation, flow estimation for dynamic objects and semantic segmentation. However, the proposed method can consistently reduce both EPE and Fl metrics, whereas Surface cannot [185].

Chapter 4

ToF and Multi-View Integration

4.1 Introduction

This chapter studies the relationship between sparse depth data obtained through an active sensor and the depth perception intrinsic in multi-view cues. This study is carried out by analyzing the behaviour of common state-of-the-art multi-view techniques when explicit accurate sparse depth information is injected into their innermost mechanisms.

Multi-view stereo (MVS) is a popular technique to obtain dense 3D reconstructions of real-world objects or scenes from a set of multiple, posed images. It represents the first, pivotal step towards a variety of higher-level applications, such as augmented/virtual reality, robotics, cultural heritage and more. Moreover, it represents one of the fundamental problems in computer vision and it has been studied for years, at first by developing classical algorithms [6, 12, 39, 41, 123], making use of hand-crafted matching functions to measure consistency among the multiple views. However, many challenges keep MVS an open problem, such as occlusions between the views, lack of texture, or non-Lambertian surfaces, to name a few [1, 70, 124]. The advent of deep learning in computer vision, in particular with the introduction of Convolutional Neural Networks (CNNs), allowed for rapid progress even in geometric tasks such as MVS, partially overcoming some of the issues mentioned above. Indeed, deep MVS networks [171, 172, 158, 150, 48, 90] are spreading, thanks to their ever-increasing accuracy on popular benchmarks [62, 124, 70]. Common to most CNNs developed for this purpose is the presence of a 3D cost volume [171], built using plane-sweeping over the source views features and computing their similarity with respect to the reference image features. Such a volume is usually regularized through 3D convolutional layers – or other, more efficient alternatives, such as 2D Long-Short Term Memory (LSTM) layers [172] – before regressing the final depth map. However, despite the more robust feature representation extracted by 2D CNNs and the strong regularization achieved through 3D convolutions, the high-demanding computational requirements still limit the full deployment of such solutions, often requiring some trade-off between accuracy and complexity. For instance, inferring depth at a resolution lower than the one of the input images [171] or implementing coarse-to-fine strategies [48, 168, 17]. Moreover, several challenges mentioned above, such as dealing with untextured regions, thin objects or occlusions,

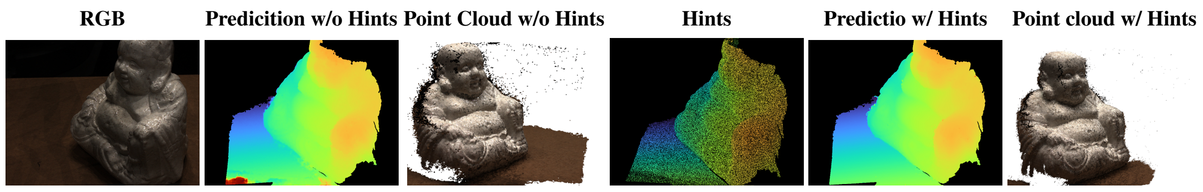


Figure 4.1. **Multi-View Guided Multi-View Stereo in action.** Deep MVS networks struggle to generalize from synthetic to real images, yielding inaccurate depth maps and poor 3D reconstructions. Guiding the network with a set of sparse depth measurements, aggregated over the multiple views can greatly ameliorate the results. Sparse depth hints are dilated by a 2×2 kernel to ease visualization.

remain open.

Most of the challenges mentioned so far are inherent to the image domain itself. Thus, their impact could be significantly softened given the availability of additional information with different modalities, for instance, by having access to a sparse set of depth measurements perceived by an active sensor. Nowadays, such sensors are at hand and readily available as standalone off-the-shelf devices. Moreover, they are always more frequently integrated into consumer products like mobile phones and tablets (e.g., Apple iPhones and iPads). However, despite their ever-increasing diffusion, they often provide only sparse depth data (i.e., at a much lower resolution compared to standard cameras). The recent literature supports such intuition, highlighting the evidence of approaches effectively exploiting the synergy of color images with sparse depth data. For instance, in the case of depth completion [146], fusion with stereo algorithms [100, 101] and networks [108, 20, 152] or, more recently, with optical flow deep architectures [103] as well.

Driven by these facts, in this chapter, it is proposed a framework for guided multi-view stereo depth estimation. Assuming the availability of a sparse set of depth measurements acquired together with images, the cost volume built by any state-of-the-art MVS network [171, 48, 161, 17, 150] is modulated [108] to provide stronger guidance to the architecture towards inferring more accurate depth maps. Moreover, by exploiting the possibility of having multiple sets of sparse depth points acquired from the different viewpoints of the source images, an integration mechanism is introduced to accumulate the multiple depth hints enabling modulating the cost volume inside the deep network with a higher density of guiding points. This allows to boost the performance of an MVS network, allowing it to infer more accurate depth maps, and consequently higher quality 3D reconstructions, for instance when trained on synthetic data and tested on real images, as shown in Fig. 4.1. To validate this claim, an exhaustive set of experiments has been carried out by training a variety of state-of-the-art MVS architectures and their guided counterparts on the BlendedMVG [173] and DTU [62] datasets and assessing their accuracy on them. This proves that the proposed approach consistently boosts the accuracy achievable with any considered deep network in terms of depth map estimation and overall 3D reconstruction when guidance is available. The contributions carried out by this chapter can be summarized as follows:

- Proposal of the Guided Multi-View Stereo framework (gMVS), extending [108] to cope with this chapter's purposes. Then, on top of that, the Multi-View Guided Multi-View Stereo (mvgMVS) is proposed to exploit multiple sets of depth hints acquired from different viewpoints of the multi-view reconstruction task.
- Introduction of coarse-to-fine guidance by applying cost volume modulation multiple times during the forward pass, compliantly to the coarse-to-fine strategy followed by recent MVS networks [48, 17, 150].
- Implementation of the proposed mvgMVS framework within five state-of-the-art deep architectures [171, 48, 161, 17, 150], each one characterized by different regularization and optimization strategies.

4.2 Proposed framework

In this section, the Multi-View Guided Multi-View Stereo (mvgMVS) framework is introduced. First, the background relevant to the proposal is reviewed, specifically concerning deep MVS networks' inner mechanisms. Then, the guided stereo matching framework [108] is applied to the MVS setting and, finally, extended to deal with multi-view depth hints and coarse-to-fine architectures.

4.2.1 Deep Multi-View Stereo background

Most learning-based MVS pipelines follow the same pattern. Given a set of N images, one assumed as the reference and the other $N-1$ as source images, deep MVS networks process them to predict a global dense depth map aligned with the reference one. To this aim, common to most deep networks designed for this purpose is the definition of a cost volume, encoding features similarity between pixels in the reference image and potential matching candidates from the source images. The latter are retrieved along the epipolar lines in the source views, given intrinsic and extrinsic parameters K, E for any camera collecting the N images involved. Specifically, for a particular depth hypothesis $z \in [z_{\text{min}}, z_{\text{max}}]$, features $\mathcal{F}_i$ extracted from a given source view i are projected using a homography-based warping operation π.

$$\mathcal{F}_i^z = \pi(\mathcal{F}_i, z, K_0, E_0, K_i, E_i) \tag{4.1}$$

Then, to encode the similarity between reference features $\mathcal{F}_0$ and $\mathcal{F}_i^z$, a variance-based volume is defined as follows

$$\mathcal{V}(z) = \frac{\sum_{i=0}^{N}(\mathcal{F}_i^z - \mu)^2}{N}, \qquad \mu = \frac{\sum_{i=0}^{N}\mathcal{F}_i^z}{N} \tag{4.2}$$

with $\mathcal{F}_i^z$ consisting of $\mathcal{F}_0$ for $i = 0$. Accordingly, for a given pixel, the lower the variance score, the more similar the features retrieved from the source views are and, thus, the more likely hypothesis z is the correct depth for it.

However, implementing this solution requires high memory and results computationally complex. Consequently, several state-of-the-art networks [48, 161, 17, 150] implement a coarse-to-fine solution. Specifically, a set of variance-based cost volumes is built as

$$\mathcal{V}_s(z) = \frac{\sum_{i=0}^{N}(\hat{\mathcal{F}}^d_{(i,s)} - \mu)^2}{N}, \qquad \mu = \frac{\sum_{i=0}^{N}\hat{\mathcal{F}}^z_{(i,s)}}{N} \tag{4.3}$$

being s a specific resolution or scale at which the cost volume is computed and $\hat{\mathcal{F}}^z_{(i,s)}$ features from image i at resolution s sampled as

$$\hat{\mathcal{F}}^z_{(i,s)} = \pi(\mathcal{F}_{(i,s)}, z_s, K_{(0,s)}, E_0, K_{(i,s)}, E_i) \tag{4.4}$$

with $K_{(i,s)}$ being the intrinsic parameters for camera i adjusted to resolution s and z_s sampled in a range $[z^s_{\text{min}}, z^s_{\text{max}}]$ that differs at any scale.

4.2.2 Guided Multi-View Stereo

By assuming a setup made of a standard camera and a low-resolution depth sensor, for instance a LiDAR, the output of the latter is leveraged to shape the behavior of a deep network estimating depth from a set of color images. When this set is limited to a single frame, a neural network is usually trained to *complete* the sparse depth points [146] guided by the color image [136]. When multiple images are available, the mechanism often reverses, and depth measurements are used as *hints* to guide the image-based estimation process. This strategy is implemented, for instance, by the Guided Stereo framework [108] applied to binocular stereo, by applying a Gaussian modulation to the features volume to peak it in correspondence of a depth hint z.

In analogy, this mechanism can be applied also to multi-view stereo, implementing a Guided Multi-View Stereo pipeline (gMVS). Indeed, the variance volume introduced in Sec. 4.2.1 can be conveniently modulated as well. In this case, since low variance encodes a high likelihood of the corresponding depth hypothesis z to be correct, the Gaussian curve is flipped to force the variance-based cost volume to have a minimum near depth hint z^*

$$\mathcal{V}'(z) = \left[1 - v + v \cdot k \cdot \left(1 - e^{-\frac{(z-z^*)^2}{2c^2}}\right)\right] \cdot \mathcal{V}(z) \tag{4.5}$$

with v being a binary mask equal to 1 for pixels with a valid hint (0 otherwise) and k, c being the amplitude and width of the Gaussian itself. The gMVS formulation outlined so far extends the Guided Stereo framework [108] to MVS. In the remainder, two significant additional contributions are introduced explicitly to deal with the MVS setup and the models designed for it.

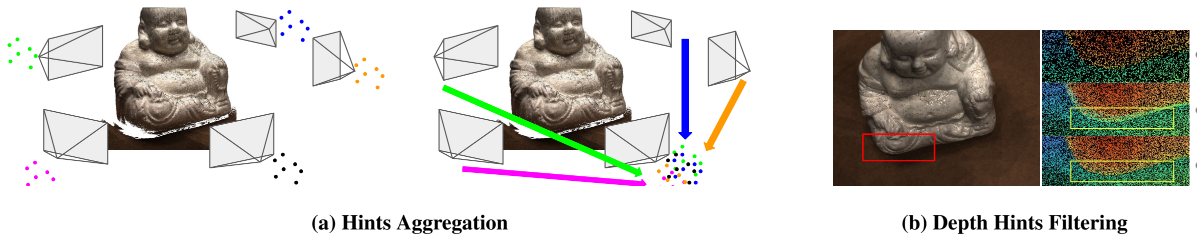


Figure 4.2. **Hints Aggregation and Filtering.** Depth hints from many views can be aggregated on the reference image viewpoint as shown in (a). On the right, is shown the proposed depth filtering approach. From top to bottom, sparse hints over the region inside the red rectangle in the RGB view, respectively from the single viewpoint, aggregated over multiple viewpoints and filtered. Regions in yellow rectangles highlight the effect of filtering. Depth points are densified to ease visualization.

4.2.3 Multi-View Guided Multi-View Stereo

MVS relies on the availability of multiple images acquired from different viewpoints. Moreover, in the proposed use case there is the additional assumption of the availability of sparse depth measurements registered with the color images in the proposed setup. Therefore, a different set of hints is available for each source image. In such a case, aggregating the multiple sets of depth hints from each viewpoint can provide stronger guidance to the network and further improve the results of the baseline gMVS framework. To this aim, two main steps are performed.

Depth hints aggregation. Given a pixel having homogeneous 2D coordinates q_i from any source image $i \in [1, N]$ for which a depth value $d^*_{q_i}$ is available, the 3D coordinates p_0 in the reference image viewpoint are obtained as:

$$p_0 = E_0 E_i^{-1} p_i \qquad \text{with} \qquad p_i = d^*_{q_i} K_i^{-1} q_i \tag{4.6}$$

From p_0, the new depth hint $d^*_{q_0}$ expressed in the reference image viewpoint can be obtained, and projected it on the image plane according to K_0 at coordinates q_0. This allows to aggregate depth hints on the reference view, as shown in Fig. 4.2 (a), and thus obtain a denser depth hints map to modulate the volume in the network with stronger guidance. This extension of the gMVS framework is reported as Multi-View Guided Multi-View Stereo (mvgMVS) in the below sections.

Depth hints filtering. Because of the different viewpoints, some of the depth measurements acquired in one of the source views may belong to occluded regions in the reference view. However, given the sparse nature of the hints, this would cause the aggregation of several wrong values if it were limited to naively projecting them across the views without reasoning about their visibility, as shown in Fig. 4.2 (b). As a consequence, the deep network would be guided by wrong depth hints, harming its accuracy. To detect and remove these outliers, the filtering strategy by Zhao et al. [185] is deployed, defining as outlier any pixels q_0 for which exists at least a pixel s in its neighborhood $S(q_0)$ such that:

- q_0 changes the relative position with respect to s, because occluded. This occurs if the difference between q_0 and s pixels coordinates and angles (in spherical coordinates) have different sign, i.e. if either $(x_{q_0} - x_s)(\theta_{q_0} - \theta_s)$ or $(y_{q_0} - y_s)(\phi_{q_0} - \phi_s)$ are negative

- q_0 distance from the camera is much higher compared to s, i.e. $d_{q_0} > d_s + \varepsilon$, with ε set according to the specific dataset used

Although simple, this strategy allows for removing most of the outliers at a minor computational cost, as shown in Fig. 4.2 (b). For this specific use case, the approach employed in [185] is more suitable than the one proposed in Chapter 3 since this latter i) requires to train a further network for filtering and ii) overfits the camera relative positions and thus it is more suited to a static camera setup, which is not always the case with multi-view stereo.

In the ablation experiments will be shown how this step is necessary to achieve optimal guidance. This final implementation is referred to as filtered mvgMVS (fmvgMVS).

4.2.4 Coarse-to-Fine Guidance

Unlike deep stereo networks, which usually build a single volume processed through stacked 3D convolutions, MVS networks are often designed to embody coarse-to-fine estimation to reduce the computational burden, as introduced previously in Sec. 4.2.1. Any of the multiple cost volumes built by the network represent a possible entry point for guiding the network. Accordingly, any $\mathcal{V}_s$ is modulated during the forward pass

$$\mathcal{V}'_s(z_s) = \left[1 - v_s + v_s \cdot k \cdot \left(1 - e^{-\frac{(z_s - z^*_s)^2}{2c^2}}\right)\right] \cdot \mathcal{V}_s(z_s) \tag{4.7}$$

with v_s and z^*_s being respectively the binary mask v and the depth hints map z^* downsampled to resolution s, with nearest-neighbor interpolation. The following experiments will show how the stronger guidance yielded by these multiple modulations improves the overall network accuracy.

4.3 Experimental Results

4.3.1 Datasets

Since none of the existing MVS data collection provides sparse depth points, in the following experiments the availability of sparse hints is simulated randomly sampling from ground-truth depth maps, similarly to [108, 103]. Consequently, have been selected datasets providing such information only, *e.g.* Tank & Temples [70] can't be used for evaluation.

BlendedMVG. This dataset [173] collects about 110K images sampled from about 500 scenes. It has been created by applying a 3D reconstruction pipeline to recover high-quality textured meshes from images of well-selected scenes. Then, meshes are rendered to color images and depth maps. Following [173] 8 sequences for validation and 7 for testing have been retained, using the rest for training.

DTU [1]. This indoor dataset counts 124 different scenes, all sharing the very same camera trajectory. Images are acquired with a structured light scanner mounted on a robot arm, using one of the cameras in the scanner itself. Training, validation, and testing splits have been selected following existing works [171, 48, 161, 17, 150]. Moreover, evaluation is carried out using both networks trained on BlendedMVG only or also fine-tuning on the DTU training set.

4.3.2 Implementation details

The Guided Multi-View Stereo framework is implemented in PyTorch, starting from existing solutions [150]. Sparse depth hints availability is simulated by randomly sampling 3% of pixels from the ground-truth depth maps. Regarding filtering, $\varepsilon = 3$ is used. Experiments are conducted implementing Guided Multi-View Stereo and variants on top of five state-of-the-art networks.

MVSNet [171]. The very first deep network for MVS: it builds a single variance volume and processes it through 3D convolutions – similarly to 3D stereo networks [67] – and estimates depth at a quarter of the input resolution.

D^{2}HC-RMVSNet [161]. A recurrent architecture, replacing 3D convolutions with 2D convolutional LSTM to reduce memory requirements.

CAS-MVSNet [48]. It implements a cascade cost volume formulation, inferring depth in a coarse-to-fine manner to achieve higher efficiency.

UCSNet [17]. It builds Adaptive Thin Volumes for coarse-to-fine processing. The volumes consist of only a few depth hypotheses selected by modeling uncertainty.

PatchMatchNet [150]. A very efficient model, implementing a differentiable variant of the PatchMatch algorithm [6] within a deep network.

Any network is implemented by integrating the authors' code in the framework and following their default configuration – except for the number of depth hypotheses used by MVSNet and D^{2}HC-RMVSNet, set to 128 due to memory constraints. During both training and evaluation, if the final output of the original network is lower than the input resolution, it is upsampled to the original size through interpolation.

4.3.3 Training and testing protocol

The number of images processed by the networks is set to 5, both during training and testing. Accordingly, depth hints are accumulated from 5 views for mvgMVS.

Training schedule. Each network is trained for 100K iterations on the BlendedMVG dataset on 576×768 images, with a constant learning rate of 10^{-3} – except D^{2}HC-RMVSNet, for which it was set to 10^{-4} to avoid instability. Any training has been carried out on a single Titan Xp

Network	Hints dens.	>1 Px.	>2 Px.	>3 Px.	>4 Px.
MVSNet [171]	-	0.139	0.073	0.046	0.031
MVSNet-g	0.03	0.095	0.046	0.027	0.018
MVSNet-mvg	0.03	0.081	0.040	0.024	0.016
MVSNet-fmvg	0.03	0.076	0.037	0.023	0.015
MVSNet-g	0.15	0.068	0.032	0.020	0.013

(a) Guiding Strategy

Network	Hints dens.	>1 Px.	>2 Px.	>3 Px.	>4 Px.
GuideNet-fmvg	0.03	0.290	0.124	0.080	0.058
MVSNet-fmvg	0.03	0.076	0.037	0.023	0.015

(b) Depth Completion Comparison

Table 4.1. **Ablation study – Guiding Strategy and Depth Completion Comparison.** On the left, is an ablation study on the guiding strategy used. From top to bottom, MVSNet [171] without guiding, guided, guided accumulating points from 5 views without filtering and finally filtering the accumulated depth. Below, is the guided version with sparse depth at a higher density from the target view only (thus, without outliers due to occlusion).

Network	Stages	>1 Px.	>2 Px.	>3 Px.	>4 Px.
CAS-MVSNet [48]	-	0.071	0.036	0.023	0.016
CAS-MVSNet-fmvg	1	0.057	0.024	0.014	0.010
CAS-MVSNet-fmvg	2	0.084	0.042	0.027	0.019
CAS-MVSNet-fmvg	3	0.078	0.041	0.027	0.020
CAS-MVSNet-fmvg	All	0.048	0.018	0.012	0.009

(a) Multi-Stage Guidance

Network	Test Hints	>1 Px.	>2 Px.	>3 Px.	>4 Px.
MVSNet [171]	-	0.139	0.073	0.046	0.031
MVSNet-fmvg	0.00	0.244	0.165	0.126	0.101
MVSNet-fmvg	0.01	0.109	0.054	0.033	0.022
MVSNet-fmvg	0.02	0.087	0.043	0.026	0.017
MVSNet-fmvg	0.03	0.076	0.037	0.023	0.015

(b) Test Time Density Change

Table 4.2. **Ablation Study – Multi-Stage Guidance and Test-Time Hints Density Change.** On the left is the ablation study analyzing the impact of sparse depth multi-stage injection. On the right, the impact of injecting sparse depth at densities different from the one used for training.

GPU, allowing only for a single sample per batch – except for PatchMatchNet, for which batch 2 fits in memory. Moreover, each network is also fine-tuned for 50K further iterations on the DTU training set, processing 512×640 images and using the hyper-parameters as detailed for BlendedMVG.

Testing protocol. Networks are evaluated on the BlendedMVG testing sequences and on the DTU testing split. For each dataset, the percentage of pixels in the estimated depth map having an error larger than τ is reported – respectively in pixels and millimeters on the two datasets, with thresholds set to 1, 2, 3, and 4. Concerning DTU, also the quality of reconstructed point clouds is evaluated: in the former case, *accuracy* and *completeness* metrics defined as in [1] and their average are reported – the lower the better. Fused point clouds are obtained as in [150].

4.3.4 Ablation study

Multi-View Guided MVS. Initially the improvements yielded by multi-view guidance are evaluated. To this aim, experiments with MVSNet are carried out training different variants on the BlendedMVG training split and evaluating on the testing sequences. Tab. 4.1 (a) collects the outcome of this experiment. From top to bottom, the error rates achieved by the original MVSNet architecture, by a variant implementing the baseline guided MVS framework described in Sec. 4.2.2 (-g), followed by mvgMVS versions respectively without (-mvg) and with (-fmvg) filtering.

Network	>1 Px.	>2 Px.	>3 Px.	>4 Px.
MVSNet [171]	0.139	0.073	0.046	0.031
MVSNet-fmvg	0.076	0.037	0.023	0.015
D^{2}HC-RMVSNet [161]	0.174	0.094	0.059	0.040
D^{2}HC-RMVSNet-fmvg	0.081	0.041	0.025	0.017
CAS-MVSNet [48]	0.071	0.036	0.023	0.016
CAS-MVSNet-fmvg	0.048	0.018	0.012	0.009
UCSNet [17]	0.071	0.038	0.024	0.017
UCSNet-fmvg	0.040	0.018	0.011	0.008
PatchMatchNet [150]	0.075	0.039	0.025	0.018
PatchMatchNet-fmvg	0.062	0.033	0.022	0.016

Table 4.3. **Evaluation on BlendedMVG [173] testing scans.** Comparison between original MVS networks[171, 48, 161, 17, 150] and their guided counterparts.

	Depth map evaluation				Point cloud evaluation		
Network	>1 mm	>2 mm	>3 mm	>4 mm	Acc. (mm)	Comp. (mm)	Avg. (mm)
MVSNet [171]	0.658	0.457	0.368	0.326	0.764	0.468	0.616
MVSNet-fmvg	0.393	0.227	0.194	0.180	0.383	0.264	0.324
D^{2}HC-RMVSNet [161]	0.708	0.519	0.423	0.372	0.764	0.586	0.675
D^{2}HC-RMVSNet-fmvg	0.401	0.177	0.134	0.115	0.393	0.234	0.314
CAS-MVSNet [48]	0.558	0.385	0.330	0.303	0.589	0.310	0.450
CAS-MVSNet-fmvg	0.323	0.243	0.220	0.207	0.345	0.286	0.316
UCSNet [17]	0.541	0.402	0.357	0.333	0.561	0.344	0.453
UCSNet-fmvg	0.199	0.174	0.164	0.157	0.290	0.264	0.277
PatchMatchNet [150]	0.627	0.440	0.370	0.335	0.574	0.484	0.529
PatchMatchNet-fmvg	0.446	0.328	0.301	0.287	0.339	0.297	0.318

(a) trained on BlendedMVG

	Depth map evaluation				Point cloud evaluation		
Network	>1 mm	>2 mm	>3 mm	>4 mm	Acc. (mm)	Comp. (mm)	Avg. (mm)
MVSNet [171]	0.555	0.340	0.268	0.237	0.635	0.304	0.470
MVSNet-fmvg	0.219	0.103	0.081	0.072	0.324	0.235	0.280
D^{2}HC-RMVSNet [161]	0.630	0.423	0.329	0.283	0.662	0.342	0.502
D^{2}HC-RMVSNet-fmvg	0.168	0.079	0.061	0.054	0.327	0.240	0.284
CAS-MVSNet [48]	0.480	0.307	0.257	0.233	0.528	0.262	0.395
CAS-MVSNet-fmvg	0.082	0.056	0.047	0.042	0.228	0.279	0.254
UCSNet [17]	0.506	0.332	0.277	0.254	0.551	0.272	0.412
UCSNet-fmvg	0.119	0.105	0.098	0.095	0.319	0.281	0.300
PatchMatchNet [150]	0.475	0.310	0.260	0.236	0.461	0.298	0.380
PatchMatchNet-fmvg	0.336	0.228	0.204	0.193	0.325	0.230	0.278

(b) fine-tuned on DTU

Table 4.4. **Evaluation on DTU [1] testing scans.** Comparison between MVS networks [171, 48, 161, 17, 150] and guided counterparts, trained on BlendedMVG and tested (a) without re-train or (b) after fine-tuning on DTU training split.

Starting from the gMVS baseline, it consistently achieves reduced error rates compared to MVSNet by exploiting the sparse depth guidance. Concerning mvgMVS, there are further improvements thanks to the aggregation of multiple sets of depth hints coming from the 5 different viewpoints. Nonetheless, even if this strategy increases the hints density from 3% up to roughly 15%, the improvement might appear not significant as one might expect with a more extensive set of hints. This fact is due to the several hints in non-visible parts of the source images that are wrongly projected in the reference point of view, as discussed previously. Indeed, by filtering out these outliers and consequently reducing the hints density to about 14%, the performance of MVSNet can be improved further. At the bottom of the table, is also reported the performance achieved by MVSNet when guided by the baseline gMVS implementation and 15% hints density. Not surprisingly, having a higher density of depth hints from the single reference viewpoint is more effective than aggregating them over multiple viewpoints because they are not affected by visibility and possible collisions between projected points. However, fmvgMVS achieves performance close to what is attainable with a depth sensor providing a much denser guide.

To conclude this study, the performance of MVSNet-fmvg is also compared with a depth completion framework. Purposely, GuideNet [136] has been trained to process single, RGB images and multi-view aggregated sparse depth points – the very same used to guide MVSNet – for 100K iterations on BlendedMVG as done for MVSNet. Tab. 4.1 (b) directly compares the error rates achieved by both highlighting how, when multiple sets of depth hints are available, the guided multi-view framework yields better depth maps compared to a depth completion approach.

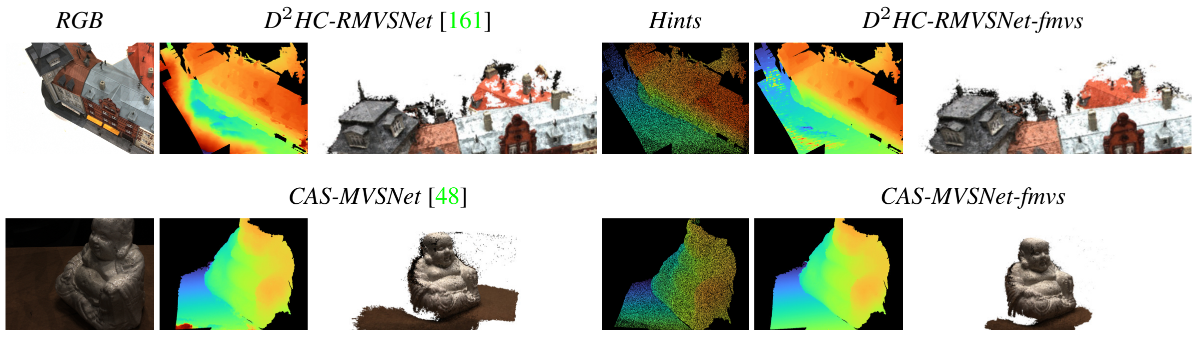


Figure 4.3. **Qualitative results on DTU dataset – *scan9* (top) and *scan114* (bottom).** Depth maps and point clouds yielded by D^{2}HC-RMVSNet (top), CAS-MVSNet (bottom), and guided counterparts trained on BlendedMVG.

Coarse-to-fine strategy. This section provides an ablation study over the coarse-to-fine guidance mechanism introduced in Sec. 4.2.4, by training different variants of CAS-MVSNet. Tab. 4.2 reports results on the BlendedMVG testing split. From top to bottom, error rates achieved by the original CAS-MVSNet without guidance, three models guided during one out of the total three stages implemented by the network (i.e. modulating only one out of the three volumes built during inference), and finally the model guided by modulating any single volume. All guided models implement the filtered mvgMVS formulation. In general, guiding the volume computed only during the first stage already improves the results of the original network. Guiding the second or third stage alone fails at even improving the results by CAS-MVSNet when not guided. Nonetheless, providing a consistent modulation across the three stages allows for the best results.

4.3.5 Multi-View Guided MVS networks

The impact of the mvgMVS framework is now evaluated on the five state-of-the-art networks selected for experimentation. Specifically, both the original networks and their counterpart guided employing filtered mvgMVS have been trained.

Evaluation on BlendedMVG. In Tab. 4.3 are collected the results obtained evaluating all the networks on the BlendedMVG testing split. By looking at the original networks, can notice be noticed that models implementing coarse-to-fine processing [48, 17, 150] result, in general, more accurate compared to MVSNet and D^{2}HC-RMVSNet, achieving about half the error rates with any threshold. This gap is bridged by guiding both with the filtered mvgMVS framework.

Guided counterparts of CAS-MVSNet, UCSNet and PatchMatchNet are further improved too. In particular, CAS-MVSNet-fmvg and UCSNet-fmvg almost halve the error rates at any given threshold, while PatchMatchNet-fmvg benefits from the guidance in minor measure. This latter fact can be ascribed to the random initialization performed at the very first stage of PatchMatch-Net, left unchanged when implementing its guided counterpart.

Generalization to DTU. The impact of the multi-view guided framework on the generalization capacity of the networks to unseen datasets is now assessed. Purposely, the five networks and their guided counterparts are evaluated on the DTU testing split without fine-tuning on the DTU training split. Tab. 4.4 (a) collects the outcome of this experiment, reporting error metrics on both estimated depth maps (left), as well as on 3D point clouds (right).

By focusing on the former, differently from the experiments on BlendedMVG, can be noticed a consistent margin between MVSNet/D^{2}HC-RMVSNet and coarse-to-fine models [48, 17, 150] only concerning the number of pixels with error larger than 1 or 2 mm, with mixed results at the increase of the threshold. By looking at guided counterparts, can be appreciated how they always produce much more accurate depth maps, dramatically reducing the error rates.

Concerning the quality of the reconstructed 3D point cloud, can be observed that coarse-to-fine models achieve both better accuracy and completeness than MVSNet/D^{2}HC-RMVSNet, confirming their effectiveness. Finally, when guided by accumulated depth hints, any network dramatically improves the quality of the fused point clouds, confirming that considerable improvements on single depth maps translate into better 3D reconstructions.

To summarize, this experiment suggests that mvgMVS notably improves the generalization capacity of MVS networks concerning depth map accuracy and 3D reconstruction quality. Fig. 4.3 shows some qualitative examples.

Fine-tuning and evaluation on DTU. To confirm that the effect of the proposed framework on 3D reconstructions is not limited to generalization scenarios, all the previous networks have been also fine-tuned on the DTU training split and evaluated. Tab. 4.4 (b) collects results concerning both estimated depth maps (left) and point clouds (right).

Concerning the original networks, a behavior similar to the one observed in Tab. 4.4 (a) appears, with a margin between coarse-to-fine models and the others, which is consistent only regarding pixels with errors larger than 1 mm. Not surprisingly, any network performs better after being fine-tuned, both in terms of depth map accuracy and point cloud quality. However, by looking at guided networks, can be noticed how their accuracy is further boosted by the fine-tuning phase, with drops of the error rates much higher than those achieved by the original models.

By looking at reconstructed point clouds, for the original networks, can be observed the same trend as in Tab. 4.4, with coarse-to-fine models generally producing higher quality point clouds. Once again, the more accurate depth maps yielded by mvgMVS correspond to better reconstructions.

To summarize, the previous experiments highlight that the mvgMVS framework constantly outperforms the original counterpart concerning generalization capability, as well as when data for fine-tuning is available.

Limitations. Although the previous experiments highlight the potential of the Multi-View Guided Multi-View Stereo framework, effective on both synthetic and real datasets, the proposal suffers from a limitation that may be important in some environments: networks trained with a specific hints density do not generalize to less dense hints inputs. Specifically, once a guided network has been trained with a fixed density of input depth points, if such density is not guaranteed at the testing time, the performance will drop. Table 4.2 investigates this behavior with a further experiment carried out using MVSNet guided with 3% hints aggregated over the views during training and tested with varying density. Can be noticed how, by reducing the number of hints, the network performance lowers as well, although still resulting better than the original MVSNet trained without guidance (first row). However, by neglecting the hints at all (last row), the performance dramatically drops below the original MVSNet. This behavior highlights that the network itself exploits the hints almost *blindly* when trained with them, losing much accuracy when the hints are not available during deployment, consistently with [108].

Chapter 5

ToF and Monocular View Integration

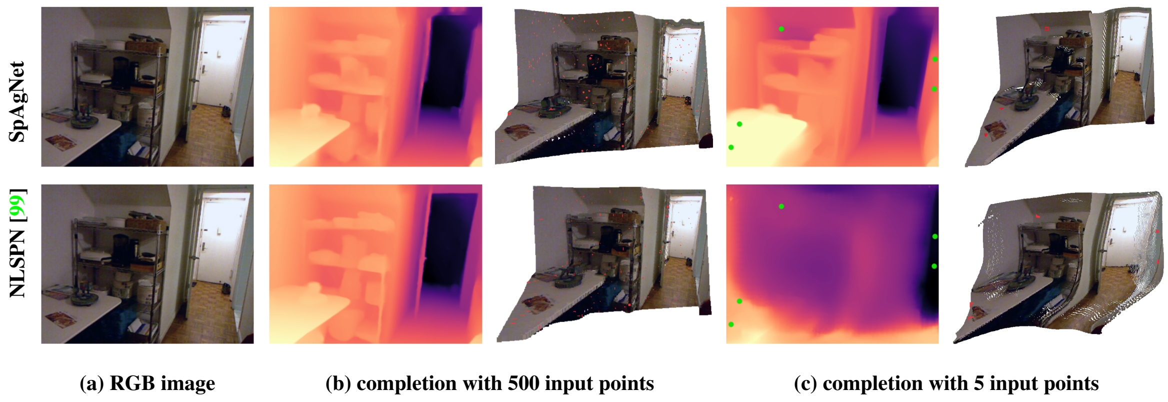


Figure 5.1. **Sparsity-agnostic depth completion.** From left to right: (a) reference image, (b) completed depth and point cloud using 500 depth points, (c) completed depth and point cloud using only 5 depth points. SpAgNet (top) dramatically outperforms NLSPN [99] (bottom) when both are trained with 500 points and tested with much fewer.

5.1 Introduction

In Chapter 2 has been highlighted that to date, accurate depth perception is demanded either to multi-view imaging approaches [171] or to specifically designed sensors such as LiDAR (Light Detection and Ranging) or ToF (Time of Flight) sensors. In Chapter 4 multi-view and active sensors have been integrated. Nonetheless, not in all applicative scenarios ego-motion is available. In this Chapter, the monocular RGB and sparse depth case (depth completion) is deeply studied highlighting the main challenges and proposing a proper solution.

Although more expensive than standard cameras, depth sensors usually allow for higher accurate measurements even though at a lower spatial resolution. On the one hand, ToF sensors are cheap, small, and have been recently integrated into mobile consumer devices [63, 89]. They perturb the scene through coded signals less effective in outdoor daytime environments. To limit power consumption, a sparse emitting pattern is used, yielding meaningful depth measures for only a few points in the scene (∼500 points) [63]. On the other hand, LiDAR sensors employ a moving array of laser emitters scanning the scene and outputting a point cloud [114], which becomes a sparse depth map once projected over the image camera plane due to its much higher resolution. Devices leveraging such technology are expensive and bulky however, being applicable even in daylight outdoor environments, became standard for autonomous driving applications [146]. Since all these depth sensors provide – for different reasons – only sparse information, techniques aimed at recovering a dense depth map from an RGB image and a few measurements have gained much popularity in recent years [92, 99, 18].

Unfortunately, in real scenarios, LiDAR and ToF sensors are affected by additional issues other than sparsity, which may easily lead even to sparser depth points often unevenly distributed. For instance, the noise originating from multi-path interference – when multiple bouncing rays from different scene points collide on the same pixel – might lead the sensor to invalidate the measurement and consequently reduce density. Moreover, low-reflectivity surfaces/materials absorb the whole emitted signal while others reflect it massively, leading to saturation. Despite the two opposite behaviors, depth cannot be reliably measured in both cases, possibly leading to large, unobserved regions.

State-of-the-art depth completion techniques are fragile and fail at reconstructing the structure of the scene for areas where no depth points are available or when the sparsity changes significantly compared to the one used at training time. Indeed, the incapacity to deal with uneven spatial distributions of the sparse depth points – which will be unveiled in this chapter – threatens the possibility of deploying such solutions in different practical contexts. Moreover, this behavior also prevents their seamless deployment when using a different sensor inferring the depth according to a spatial pattern different from the one used while training (e.g., switching from an expensive Velodyne [149] LiDAR system to a cheaper one).

Unfortunately, as reported in this chapter and shown in Figure 5.1, convolutional layers struggle at generalizing when fed with variable sparsity input data. Hence, here is proposed a design strategy that diverges from the literature to overcome this issue by not directly feeding sparse depth points to the convolutional layers. Purposely, the sparse input points are iteratively merged with multiple depth maps predicted by the network. This strategy allows to handle highly variable data sparsity, even training the network with a constant density distribution as done by state-of-the-art methods [99, 18, 35, 49] yet avoiding catastrophic drops in accuracy witnessed by competitors. Such an achievement makes the completion solution a *Sparsity Agnostic* Network, dubbed SpAgNet.

This chapter's contribution can be summarized as follows:

- A novel module designed to incorporate sparse data for depth completion yet being independent by their distribution and density. Such a module plugged into a competitive neural network architecture trained effortlessly can effectively deal with the previously mentioned issues.
- The performance of SpAgNet and state-of-the-art methods is assessed on a set of highly challenging cases using KITTI Depth Completion (DC) and NYU Depth V2 (NYU) datasets highlighting the superior robustness of the proposed solution compared to state-

of-the-art when dealing with uneven input patterns.

5.2 Sparsity Agnostic Depth Completion

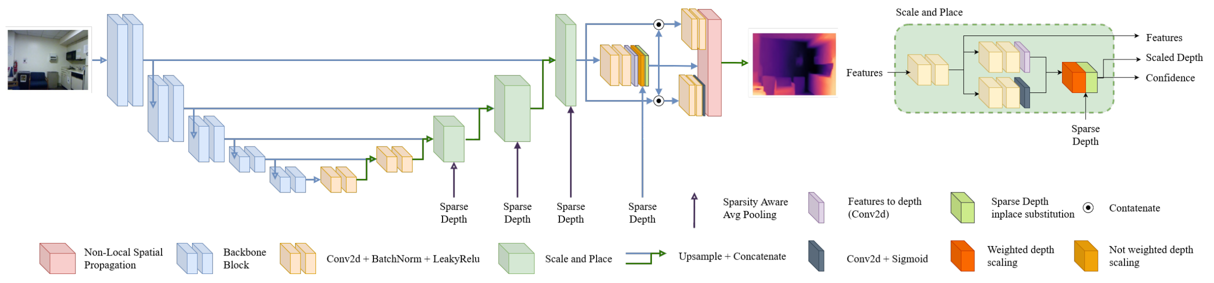
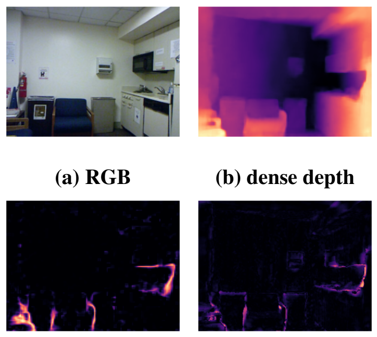


Figure 5.2. **SpAgNet architecture.** The network follows an encoder-decoder design, with a backbone to extract features from the image and a custom decoder to iteratively merge at multiple scales sparse depth hints without directly feeding them as a sparse depth map. Finally, we leverage non-local propagation [99] to improve accuracy further.

As pointed out by [146, 24], 2D convolutions struggle to manipulate sparse information. Additionally, can be further noticed that the density of such input depth data and its spatial distribution – which could be highly uneven – might lead state-of-the-art networks to catastrophic failures, as depicted at the bottom of Figure 5.1. Moreover, these networks mostly rely on the sparse depth input overlooking the image content and substantially ignoring the geometric structure depicted in it.

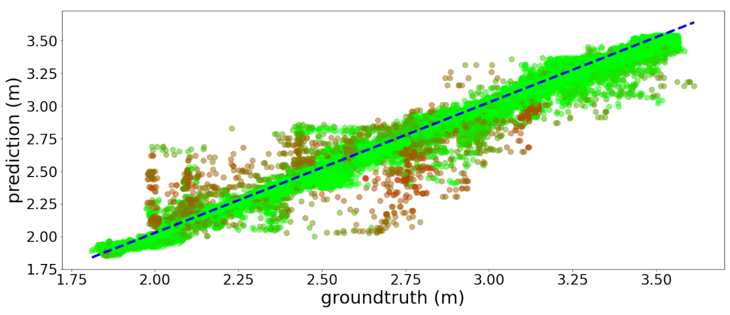
SpAgNet relies on an encoder-decoder structure with skip connections, as depicted in Figure 5.2. However, unlike current depth completion techniques [99, 18, 49, 35], the encoder is not fed with sparse depth information for the reasons previously outlined. Instead, features are extracted from the RGB frame *only* in order to get rid of the sparse input data and, consequently, its density. This strategy allows to constrain the network to exploit the image content fully and, as discussed later, to enforce the network at extracting the geometry of the scene from the RGB.

The decoding step predicts – iteratively and at multiple scales – dense depth from the RGB image and *fuse* it with the sparse input data. The first iterative step takes the input features extracted from the RGB image and generates a lower-scale depth map and a confidence map. Then, the next iterative steps process the same inputs plus the depth map and its confidence, both *augmented* with the sparse input points computed in the previous iteration. Moreover, since each intermediate depth map provides information up to a scale factor, it is scaled according to the sparse input points before each augmenting step. This is required due to the ill-posed nature of monocular depth prediction. Experimental results will corroborate this design choice, especially when dealing with a few sparse input points. At the end of the iterative steps, non-local spatial propagation proposed in [99] is applied to refine the depth map inferred by the network. Figure 5.2 describes the whole framework.



(a) RGB **(b) dense depth**

(c) confidence map **(d) depth error**



(e) depth scaling linear regression

Figure 5.3. **Confidence aware depth scaling.** Example of confidence usage to scale depth. On left, the input image (a), the predicted depth map (b), the estimated confidence (c) and the errors with respect to groundtruth (d). On right, the outcome of the scaling procedure (red means a lower confidence prediction, green a higher one).

5.2.1 Encoder Architecture

Since the proposed framework encodes features from the image only any pre-trained network can be leveraged as an encoding backbone. Such backbone is pre-trained on ImageNet [118]. Among the multiple choices [53, 58, 159] ResNeXt50 [159] has been chosen due to its good trade-off between performance and speed. Specifically, it downsamples the image to scales $\frac{1}{2}$, $\frac{1}{4}$, $\frac{1}{8}$, $\frac{1}{16}$ and $\frac{1}{32}$ and the features are used in the decoding step as input and in the skip connections.

5.2.2 Scale and Place Module

The Scale and Place (S&P) module is in charge of inferring a dense and scaled depth map and its confidence. It takes as input the backbone features, the output of the previous S&P module at a different scale, and the sparse depth points as depicted in Figure 5.2.

Specifically, S&P leverages the input features to jointly generate an initial up-to-scale depth map and its confidence deploying a stem block composed of two convolutional layers and two heads in charge of generating them. Each convolutional layer consists of a 2D convolution, a batch normalization [61] and a Leaky ReLU. Then in the *Scale* step, the S&P module performs a weighted linear regression to scale the depth map according to the available sparse input points, weighted by means of confidence. The parameters of the weighted linear regression can be computed in closed form and in a differentiable way, as described in Eq. 5.1 where p_i is the predicted depth value and c_i its confidence corresponding to an available input sparse point s_i.

$$\beta = \frac{\sum_i c_i(p_i - \hat{p})(s_i - \hat{s})}{\sum_i c_i(p_i - \hat{p})^2} \qquad \alpha = \hat{s} - \beta\hat{p} \tag{5.1}$$
$$\hat{p} = \frac{\sum_i c_i p_i}{\sum_i c_i} \qquad \hat{s} = \frac{\sum_i c_i s_i}{\sum_i c_i}$$

Then, in the *Place* step, for those points where a sparse input depth value is available, the corresponding value in the scaled depth map is replaced with it. Additionally, the same point in the confidence map is updated with the highest score. The latter step can be summarized as follows

$$\hat{D}[x,y] = \begin{cases} D^s[x,y] & \text{if} \quad H[x,y] = 0 \\ H[x,y] & \text{if} \quad H[x,y] \neq 0 \end{cases} \tag{5.2}$$

$$\hat{C}[x,y] = \begin{cases} C^s[x,y] & \text{if} \quad H[x,y] = 0 \\ 1 & \text{if} \quad H[x,y] \neq 0 \end{cases} \tag{5.3}$$

where D^s is the scaled depth map, C^s is the confidence map and H is a sparse depth map containing zeros where an input sparse depth point is not available. The predicted confidence has an empirically chosen range of [0.1 .. 0.9] while confidence 1 is associated with each valid value in H.

The S&P module is applied at scales $\frac{1}{8}$, $\frac{1}{4}$ and $\frac{1}{2}$. The module at $\frac{1}{8}$ computes the initial depth and confidence maps leveraging only the RGB features. The others take in input also the up-sampled dense depth and confidence maps from the previous module to iteratively correct the prediction relying on both the predicted depth and the injected sparse points. Thus, with this strategy, the decoder does not deal directly with sparse data in any of its steps. Nonetheless, the network can locate and effectively leverage reliable sparse information. An example of this mechanism is shown in Figure 5.3, where can be clearly seen how the network learns to locate the most reliable depth values as those closer to the ground truth depth.

It is worth noting that confidence plays a crucial role in the S&P module. At first, in the *Scale* step, it helps to locate outliers in the estimated depth map enabling to soften their impact when performing the scaling procedure. Additionally, in the *Place* step, assigning the highest confidence to the sparse input points enables the network to rely on them effectively. Nonetheless, SpAgNet also exploits the other predicted depth points according to their estimated confidence.

Since the S&P module needs the sparse data at multiple scales, it is downsampled by employing a non-parametric sparsity aware pooling: moving a 3×3 window with stride 2, the mean of the available measures in its neighborhood is assigned to each coordinate. This approach is iteratively applied to reach lower resolutions. This method leads to a densification of the sparse depth map and helps, at all scales, to include even the meager few sparse points available into a large field of view.

5.2.3 Non-Local Spatial Propagation

Spatial propagation concerns the diffusion of information in a localized position to its neighborhoods. This strategy represents a common practice in the depth completion literature [86, 18, 99, 57] and can be achieved by a neural network in charge of learning the affinity among neighbors. Let $X = (x_{m,n}) \in R^{M \times N}$ be a 2D depth map to be refined through propagation, at step t it acts as follows:

$$x^t_{m,n} = w^c_{m,n} x^{t-1}_{m,n} + \sum_{(i,j) \in N_{m,n}} w^{i,j}_{m,n} x^{t-1}_{i,j} \tag{5.4}$$

Where (m, n) is the reference pixel currently being updated, $(i, j) \in N_{m,n}$ the coordinate of the pixels in its neighborhood, $w^{i,j}_{m,n}$ the affinity weights, and $w^c_{m,n}$ the affinity weight of the reference pixel:

$$w^c_{m,n} = 1 - \sum_{(i,j) \in N_{m,n}} w^{i,j}_{m,n} \tag{5.5}$$

The various existing methods differ by the choice of the neighborhood and by the normalization procedure of the affinity weights, the latter is necessary to ensure stability during propagation [86, 18, 99]. Within SpAgNet, the non-local approach [99] is implemented, letting the network dynamically decide the neighborhood using deformable convolutions [30]. Formally:

$$N_{m,n} = \{x_{m+p,n+q} \mid (p, q) \in f_\phi(\mathrm{I}, \mathrm{H}, n, m)\} \\ p, q \in \mathbb{R} \tag{5.6}$$

Where I and H are the RGB image and the sparse depth, and $f_\phi(\cdot)$ is the neural network determining the neighborhood. The non-local propagation module requires in input an initial depth map generated through two convolutional blocks from the last S&P block output, scaled using the full-resolution sparse depth points. However, in this case, the weighted scaling is not used to obtain the best result on the entire frame. Finally, as usual, the sparse depth points override the predicted output. The resulting depth map is then fed along with features to two convolutional blocks to generate the guiding features and confidence required by the propagation module.

5.2.4 Loss Function

At each scale, the network is trained by supervising the depth obtained by the S&P module before *Place* step. The confidence weights the loss of each depth prediction, and a regularization term (controlled by η) enforces the network to maintain the confidence as high as possible. Following [99], both L1 and L2 losses are computed. the loss function, at a specific scale, is described by Eq. 5.7 where C^s and D^s are respectively confidence and depth at a specific scale s. Confidence is not computed for the full-size scale, hence $C^0 = 1$. Finally, it is worth mentioning that lower scales are weighted less through an exponential decay factor γ.

$$L = \sum_{s=0}^{n} \gamma^s \frac{1}{N} \sum_{i}^{m} C^s_i L^{12}_i - \eta \ln C^s_i \quad \text{where} \quad L^{12}_i = |D^s_i - G_i| + |D^s_i - G_i|^2 \tag{5.7}$$

5.3 Experimental Results

SpAgNet has been implemented in PyTorch [102] training with 2 NVIDIA RTX 3090 and using the ADAM optimizer [69] with $\beta_1 = 0.9$ and $\beta_2 = 0.999$. The final model requires 35 milliseconds to perform a prediction on an image of 640×480 resolution employing a single NVIDIA RTX 3090 GPU.

5.3.1 Datasets

NYU Depth V2. The NYU Depth V2 [97] dataset is an indoor dataset containing 464 indoor scenes gathered with a Kinect sensor. The official train/test split is used as previous works, relying on the pre-processed subset by Ma et al. [92] using 249 scenes for training (∼50K samples) and 215 scenes (654 samples) for testing. Each image has been down-sampled to 320×240 and then center-cropped to 304×228. As a common practice on this dataset, 500 random points per image have been extracted to simulate sparse depth. The network is trained for 15 epochs starting with a learning rate 10^{-3} and decreasing it every 3 epochs by 0.1, setting $\gamma = 0.4$ and $\eta = 0.1$. Batch size 24 (12 for each GPU) is used; hence the network is extremely fast to converge since the whole training accounts for less than 30K steps. Color and brightness jittering and horizontal flips are applied while training to limit overfitting.

KITTI Depth Completion (DC). KITTI DC [146] is an outdoor dataset containing over 90K samples, each one providing RGB information and aligned sparse depth information (with a density of about 5%) retrieved by a high-end Velodyne HDL-64E LiDAR sensor. The images have 1216×352 resolution, and the dataset provides a standard split to train (86K samples), validate (7K samples), and test (1K samples). The ground truth has been obtained temporally accumulating multiple LiDAR frames and filtering errors [146], leading to a final density of about 20%. On this dataset 10 training epochs are used with batch size 8 (4 for each GPU), starting with learning rate 10^{-3} and decreasing it every 3 epochs by 0.1, $\gamma = 0.4$ and $\eta = 20.0$. Data augmentation follows the same scheme used for NYU.

5.3.2 Evaluation

In this section, the performance of SpAgNet and state-of-the-art methods are assessed. Following standard practice [99, 18], the following metrics are used:

$$\text{RMSE} = \sqrt{\frac{1}{N}\sum_i |D_i - G_i|^2} \quad \text{MAE} = \frac{1}{N}\sum_i |D_i - G_i| \quad \text{REL} = \frac{1}{N}\sum_i \left|\frac{D_i - G_i}{G_i}\right|$$

For evaluation purposes, in addition to the standard protocol deployed in this field [99, 18], the robustness of the networks is also deeply studied in much more challenging scenarios but always training with the standard procedure (i.e., using 500 points on NYU and 64 LiDAR lines on KITTI). Since KITTI DC is thought for autonomous driving tasks and the sparse depth is

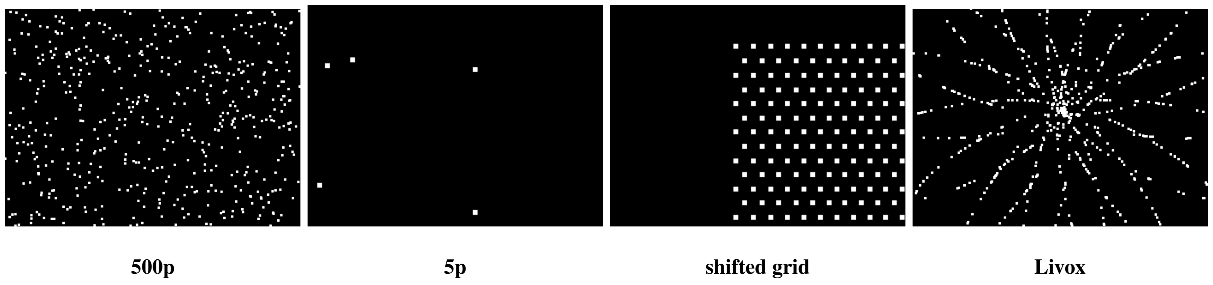


Figure 5.4. **Sparse depth patterns.** Examples of different sparse depth patterns, from left to right: 500 random points, 5 random points, shifted triangular tiling dot pattern and a Livox-like pattern (e.g. Livox Mid-70).

acquired with a high-end 64 Lines Lidar which provides in output always the same pattern, the switch to a cheaper device is simulated providing in output fewer lines. This scenario demonstrates the capability of SpAgNet to generalize over sparse depth density. On NYU Depth V2, sparse depth points are traditionally extracted randomly from the ground truth [92, 99, 18] which is almost dense. Thus, the following cases are tested: i) the extreme case of having only 5 random points, ii) the impact of having large empty areas, and iii) the impact of changing the sparsity pattern. Case ii) is implemented sampling from the ground truth a triangular tiling dot pattern aimed at simulating the output of a commercial VCSEL [89] ToF sensor and then randomly shifting this pattern to leave behind large empty areas where no sparse hints are available while iii) extracting from the ground truth sparse points with the pattern of a Livox Mid-70 [87]. All these patterns are shown in Figure 5.4. The publicly pre-trained state-of-the-art models available either on NYU Depth V2 or KITTI DC are taken into account as competitors. While evaluating, is guaranteed that each architecture sees exactly the *same* sparse points for a fair comparison.

Results on NYU Depth v2. Table 5.1 (a) compares state-of-art methods and SpAgNet on the NYU dataset using different input configurations: in the upper portion by changing the number of samples and in the lower portion by changing the pattern type. From the table, can be noticed that SpAgNet achieves competitive results, being very close to NLSPN [99] and better than other methods when the number of points used is the same as the training phase (i.e., 500). Similar behavior occurs with 200 points. However, when the density of input points decreases further, SpAgNet vastly outperforms the state-of-the-art. The performance gap with other methods gets much higher when decreasing the density further. For instance, with 50 points, the RMSE by SpAgNet is 0.272 m, while the second one (NLSPN [99]) accounts for 0.423 m. Notably, with only 5 points, the same metrics are 0.467 m and 1.033 m (NLSPN [99]), further emphasizing the ability of the proposal to deal even with meager input points, in contrast to competitors. It is worth observing that SpAgNet outperforms competitors with randomly selected input points starting from 100.

The bottom portion of Table 5.1 (a) reports the outcome of the evaluation with different spatial distributions and their average density of depth input points. Specifically, results using the two distributions depicted in the rightmost images of Figure 5.4 are shown. From the table, can be observed that when the spatial distribution covers the whole image, as in the case of the Livox-

Method	Samples	REL ↓	RMSE (m) ↓
pNCNN [35]	500	0.026	0.170
CSPN [18]		0.016	0.118
NLSPN [99]		0.013	0.101
PackNet-SAN [49]		0.019	0.120
SpAgNet		0.015	0.114
pNCNN [35]	200	0.040	0.237
CSPN [18]		0.027	0.177
NLSPN [99]		0.019	0.142
PackNet-SAN [49]		0.027	0.155
SpAgNet		0.024	0.155
pNCNN [35]	100	0.061	0.338
CSPN [18]		0.067	0.388
NLSPN [99]		0.038	0.246
SpAgNet		0.038	0.209
pNCNN [35]	50	0.108	0.568
CSPN [18]		0.185	0.884
NLSPN [99]		0.081	0.423
SpAgNet		0.058	0.272
pNCNN [35]	5	0.722	2.412
CSPN [18]		0.581	2.063
NLSPN [99]		0.262	1.033
SpAgNet		0.131	0.467
pNCNN [35]	shifted grid (∼ 100)	0.519	1.922
CSPN [18]		0.367	1.547
NLSPN[99]		0.175	0.796
SpAgNet		0.110	0.422
pNCNN [35]	livox (∼ 150)	0.061	0.333
CSPN [18]		0.066	0.376
NLSPN [99]		0.037	0.233
SpAgNet		0.039	0.206

(a) NYU Depth v2 Evaluation

Method	Lines	RMSE (mm) ↓	MAE ↓
NLSPN [99]	64	778.00	199.50
pNCNN [35]		1011.86	255.93
PackNet-SAN [49]		1027.32	356.04
PENet [57]		791.62	242.25
SpAgNet		844.79	218.39
NLSPN [99]	32	1217.21	367.49
pNCNN [35]		1766.84	615.93
PackNet-SAN [49]		1836.84	914.33
PENet [57]		1853.06	1025.42
SpAgNet		1164.18	339.22
NLSPN [99]	16	1988.52	693.10
pNCNN [35]		3194.69	1321.74
PackNet-SAN [49]		2841.35	1570.05
PENet [57]		3538.02	2121.46
SpAgNet		1863.25	606.92
NLSPN [99]	8	3234.93	1491.28
pNCNN [35]		5921.94	2999.92
PackNet-SAN [49]		3231.03	1575.41
PENet [57]		6015.02	3812.45
SpAgNet		2691.34	1087.21
NLSPN [99]	4	4834.22	2742.80
pNCNN [35]		9364.58	5362.45
PackNet-SAN [49]		4850.20	2255.08
PENet [57]		9318.86	5819.36
SpAgNet		3533.74	1622.64

(b) KITTI DC Evaluation

Table 5.1. **SpAgNet Evaluation.** Comparison with state-of-the-art methods, trained with 500 random points extracted from the ground truth or a subsampled number of scan lines as input. Each model is then tested with different densities and patterns. The best, second-best and third-best are highlighted.

like pattern, SpAgNet and NLSPN [99] achieve similar performance while other methods fall behind. However, when the input points do not cover significant portions of the scene and the density decreases further, like in the shifted-grid case, SpAgNet dramatically outperforms all competitors by a large margin.

Figure 5.5 shows qualitatively how SpAgNet compares to CSPN [18] and NLSPN [99] on an NYU sample when using 500 random points, 5 points, and the shifted grid. It highlights how only SpAgNet yields meaningful and compelling results with 5 points and the shifted grid, leveraging the image content much better than competitors, thanks to the proposed architectural design. At the same time, it achieves results comparable to competitors with 500 randomly distributed points. This fact further highlights that the robustness of SpAgNet is traded with the capacity of entirely leveraging the sparse depth information when fully available.

Results on KITTI DC. Once assessed the performance on the indoor NYU dataset, Table 5.1 (b) reports the evaluation on KITTI DC. From the table, can be noticed that with 64 lines,

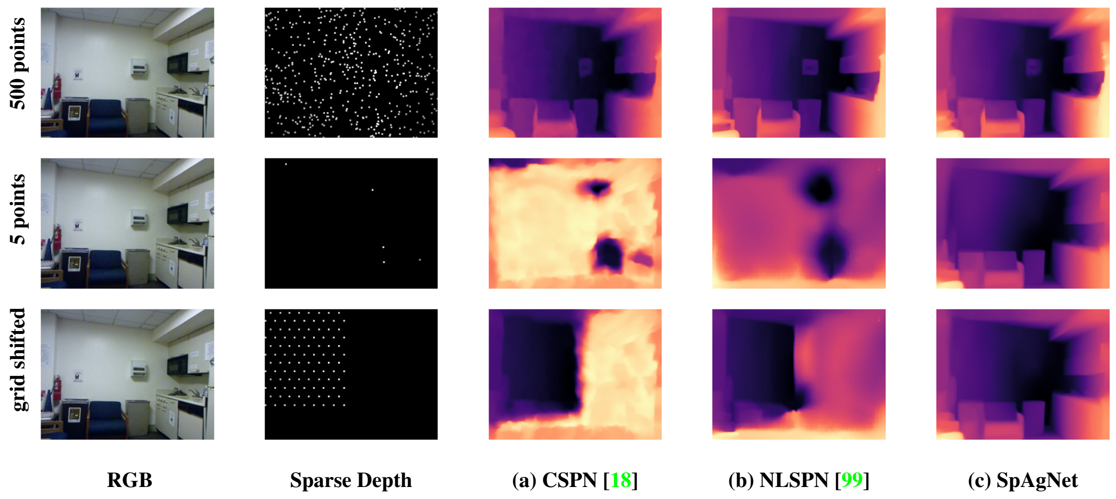


Figure 5.5. **Qualitative results on NYU-Depth v2.** CSPN [18] and NLSPN [99], when processing 5 points or the shifted grid pattern, manifest the complete inability to handle them, while SpAgNet maintains the scene structure.

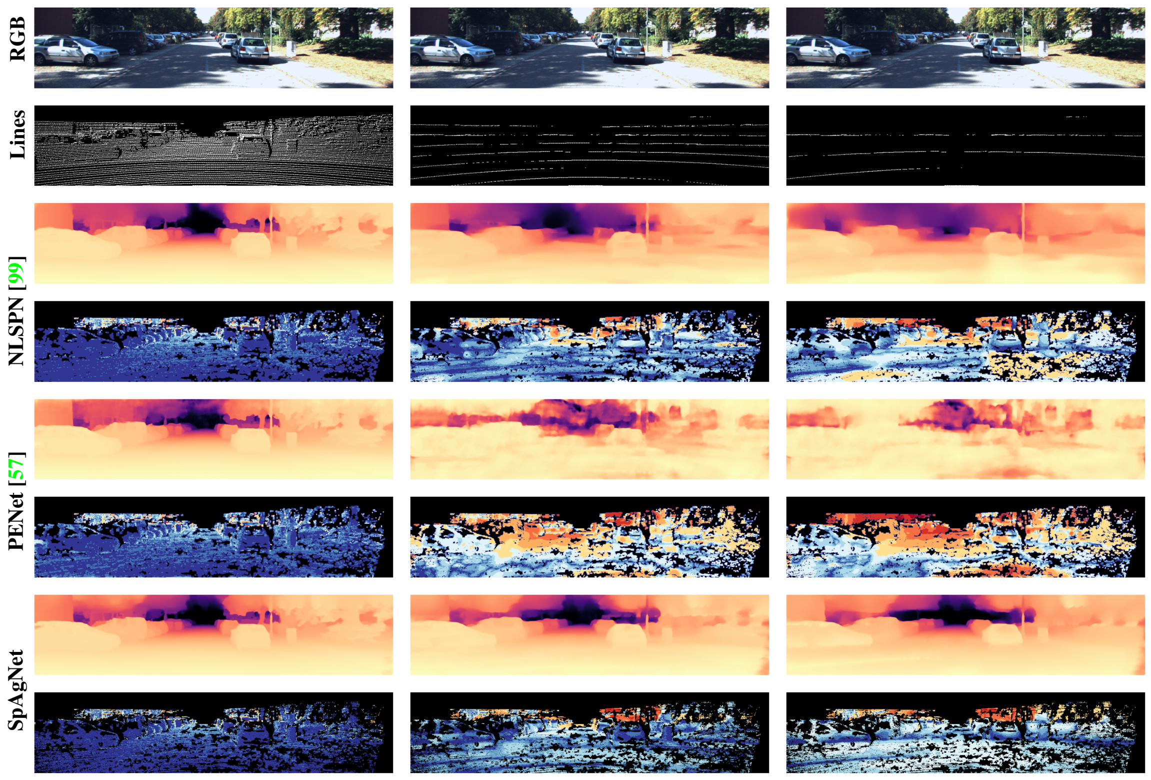


Figure 5.6. **Qualitative results on KITTI DC.** Results are reported using, respectively, from left to right, 64, 8, and 4 lines. From top to bottom the predicted depth and error map of [99], [57], and SpAgNet.

SpAgNet results are almost comparable to the best one, NLSPN. However, by reducing the number of lines from 32 to 4, SpAgNet gets always the best performance with an increasing gap. Interestingly, PackNet-SAN [49], which has been specifically trained to perform well in both depth completion (64 lines) and depth prediction (0 lines) is not able to deal with fewer lines. Indeed, the accuracy it achieves when processing 16, 8, or 4 lines is even lower than the one achieved when performing depth prediction, i.e. with RMSE equal to 2.233 mm. This behavior could be ascribed to the fact that they train an external encoding branch to extract features from sparse data and feed them to the network employing a sum operation. Even though such a branch applies a special and bulky sparse convolution operator [24], it does not seem capable of generalizing to fewer points. On the contrary, the whole network seems to suffer from the same issues of fully convolutional models, resulting effective only when fed with 64 LiDAR lines or none – the only configurations observed during training.

Figure 5.6 shows, on an image of the KITTI DC dataset and for three different numbers of lines, the outcome of NLSPN, PENet, and SpAgNet. In contrast to competitors, SpAgNet consistently infers meaningful depth maps, even when the number of lines decreases. This behavior can be perceived better by looking at the error maps. For instance, it is particularly evident with 4 lines, focusing on the road surface and the far and background objects.

NLSP	Confidence	Scaling	Samples	RMSE (m) ↓
✗	✗	✗	500	0.161
✗	✓	✓		0.127
✓	✗	✓		0.122
✓	✓	✗		0.115
✓	✗	✗		0.132
✗	✓	✗		0.145
✗	✗	✓		0.135
✓	✓	✓		0.114
✗	✗	✗	5	0.770
✗	✓	✓		0.474
✓	✗	✓		0.479
✓	✓	✗		0.526
✓	✗	✗		0.566
✗	✓	✗		0.823
✗	✗	✓		0.484
✓	✓	✓		0.467

(a) Single Components

Backbone	Size	Samples	RMSE (m) ↓
ResNet18	27M	500	0.116
ResNet34	37M		0.121
ResNet50	51M		0.117
ResNeXt50	51M		0.114
DenseNet121	30M		0.118
DenseNet161	61M		0.115
ResNet18	27M	5	0.504
ResNet34	37M		0.474
ResNet50	51M		0.664
ResNeXt50	51M		0.467
DenseNet121	30M		0.678
DenseNet161	61M		0.564

(b) Different Backbones

Table 5.2. **Ablation study on NYU.** Training with 500 points, testing either with 500 or 5 points on the same dataset. The best, second-best and third-best are highlighted.

5.3.3 Ablation Study

Finally, an ablation study concerning the main components of SpAgNet is carried out to measure their effectiveness. Specifically, in Table 5.2, two main studies are conducted, respectively, to evaluate (a) the impact of i) the Scale step of the S&P modules, ii) the usage of confidence and iii) the non-local propagation head, and (b) results achieved with different backbones. From (a), can be noticed that with 500 sparse points, scaling does not significantly improve since the network already learns to generate an output that is almost in scale. However, with only 5 points, applying a global scaling procedure helps retrieve the correct scale even in regions lacking depth measurements. Focusing on confidence, it turns out to be effective with high and low densities of input points. Finally, Non-Local Spatial Propagation further boosts performance in both cases.

In (b), most backbones yield comparable results when tested with 500 points, with ResNeXt50 achieving slightly better results. A significant gap in accuracy emerges when testing the same networks with only 5 points, with ResNeXt50 achieving the best results again.

Chapter 6

Multi-View Cues in Video Depth Estimation

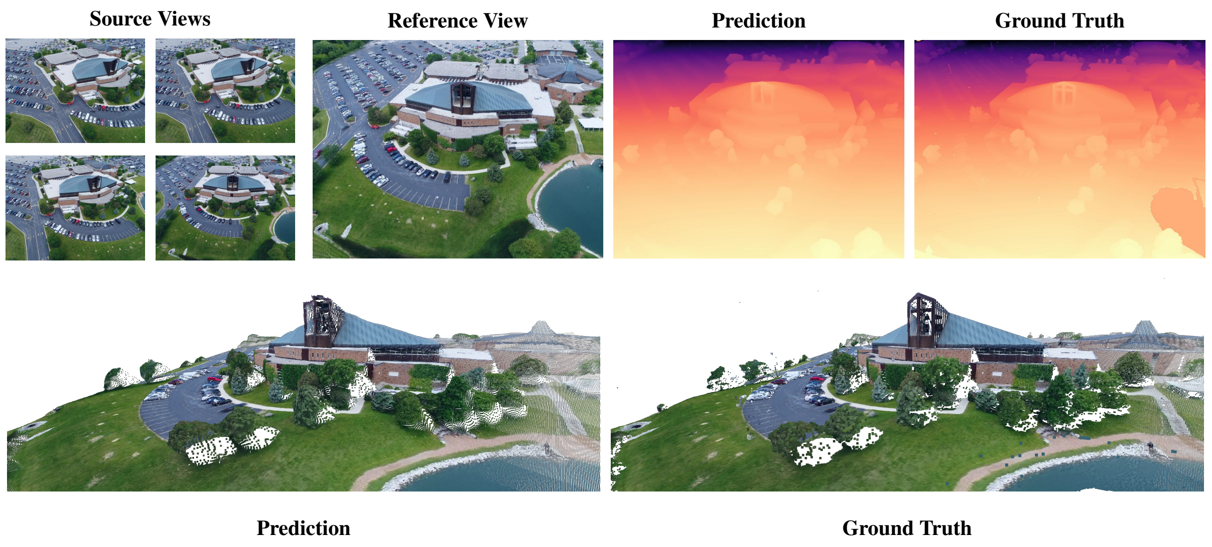


Figure 6.1. **Depth Estimation and 3D reconstruction with RAMDepth on Blended [173].** On top: given five images of the same scene, RAMDepth can estimate accurate depth maps through multi-view geometry without requiring any knowledge about the reference view depth range. At the bottom: the point cloud obtained from the prediction of the network and the respective ground-truth.

6.1 Introduction

In previous Chapters, a deep focus has been placed on the effectiveness and issues of active sensors. Nonetheless, RGB information is always important to ameliorate such issues and provide effective reconstructions. Thus, in this Chapter, the focus shifts to the RGB-only capabilities of performing 3D modeling in the wild from video sequences with the final goal to provide the groundwork for the final framework presented in Chapter 7. Compared with active sensors, passive sensing from standard RGB cameras by triangulation has many advantages. Indeed, RGB cameras are energy efficient, compact in size, and may operate in various conditions. Among passive approaches, stereo vision leverages two calibrated cameras to restrict the matching problem to a 1D search space, yet requires two cameras in a constrained setting – i.e., being nearly coplanar to allow for simpler calibration and rectification. On the other hand, a single monocular RGB camera in motion is the most flexible (as well as challenging) approach.

Traditionally, multi-view 3D reconstruction techniques can be classified in the following broad families: voxel, surface evolution, patch, or depth-based [131, 147, 79, 41, 171]. Despite being tackled with hand-crafted algorithms at first [12, 39], most state-of-the-art methods leverage depth-based deep learning architectures. These frameworks process a set of *source* views and a *reference* view and yield an estimated depth map for the latter. Most deep architectures tackle this task by (i) extracting deep features from the images, (ii) building a cost volume sampled over the epipolar lines through a set of *depth hypotheses* using differentiable homography, and (iii) predicting depth with a typically 3D convolutional module. The depth estimation pipeline sketched so far is effective but affected by some limitations.

First, according to step (ii), prior knowledge of the scene depth range is strictly required to sample depth hypotheses and build a meaningful cost volume [125]. Indeed, on the one hand, sampling hypotheses out of an underestimated range would make the network unable to predict depth values in out-of-range areas. On the other hand, overestimating the depth range will result in sampling coarser hypotheses, thus reducing the fine-grained accuracy of estimated depth maps. Unfortunately, such knowledge cannot be straightforwardly retrieved in real scenarios. When raw images are provided, camera poses can be obtained through traditional Structure-from-Motion (SfM) algorithms [123], possibly estimating the depth range as well. However, such a range might be erroneously estimated due to a number of reasons – e.g., untextured regions, visual occlusion, or poor field of view (FoV) overlap. Can be pointed out that many applications in which camera poses are known by other means exist (e.g., as often occurs in robotic applications [62]) and that modern mobile platforms provide pose information through dedicated inertial sensors.

Second, source frames must be carefully selected to allow proper depth estimation, with a set of requirements such as enough distance between optical centers to allow meaningful displacements, as well as sufficient cross-view overlap, to allow matching. Moreover, the quality of the views must be considered as well: abrupt light or color changes, moving objects, or scene-specific occlusions must be taken into account to maximize matches. Unfortunately, all these aspects cannot be evaluated by simply considering pose similarity since many of them require an analysis of the images themselves. A better approach could be to apply SfM algorithms and analyze the distribution, quality, and amount of keypoint matches across different views, which would require additional offline processing. Can be argued that distinguishing meaningful matches from unreliable ones would ease the depth estimation task – as highlighted by prior works [181, 93] – as well as possibly reduce the computational overhead by limiting the number of source views to those being strictly necessary to estimate accurate depth, although this latter aspect has never been explored.

Prompted by the previous observations, this chapter proposes a novel framework that is (i) *free* from prior knowledge of the depth range from which one samples hypotheses, and (ii) capable of distinguishing the most meaningful source frames among many. It will be showed that the Range Agnostic Multi-View framework (RAMDepth) enjoys the following properties:

- **Scene Depth Range Invariance.** It is completely independent of any input depth range assumption and thus applicable everywhere a set of images along with their pose is provided. Instead of sampling features along epipolar lines according to a fixed set of depth hypotheses and then predicting depth, the reversed mechanism is employed: the framework iteratively updates a depth estimate dynamically moving along epipolar lines according to this latter to compute correlation scores. In this way, fixing an *a priori* set of depth hypotheses is not required.

- **Keyframes Ranking.** The proposed approach not only estimates depth but also provides insights about the match quality of each source view and its contribution to the final prediction, allowing within a single inference step to rank input source views according to their actual matching against the reference view.

To assess the performance of RAMDepth, different challenging benchmarks with heterogeneous specifics have been considered. On Blended [173] and TartanAir [153], it's demonstrated the capability of the proposed framework to seamlessly estimate accurate depth in diverse scenes such as large-scale outdoor environments, top-view buildings, and indoor scenarios. Indeed, on the one hand, Blended [173] is characterized by significant pose changes, occlusions, and large FoV overlap. On the other hand, TartanAir [153] provides video streams characterized by small, unpredictable pose changes, where the depth range of each frame can change abruptly. Moreover, on UnrealStereo4K [144] it is assessed the generalization capability of RAMDepth to video streams and the possibility of applying it to the stereo setup. To conclude, RAMDepth is also validated on DTU [62], where the depth range is fixed. Along with this validation, the peculiar capabilities of this approach are shown through specifically designed experiments. Fig. 6.1 shows the outcome of RAMDepth on Blended [173].

6.2 Proposed Framework

RAMDepth is a deep framework to tackle 3D reconstruction from multiple posed views leveraging 2D convolutional layers only, and an iterative optimization procedure aimed at refining an internal depth map. This design builds upon the following principle: given the reference view, matches over an arbitrary source view can be found given their relative pose and enough visual overlap. Thus, provided an initial depth map, dense matching costs can be computed between the reference and source views. Such information is then fed to a 2D learned module to properly refine the predicted depth map. This way, unlike any other framework that builds a cost volume relying on a set of *a priori* depth hypotheses, RAMDepth can dynamically navigate the matching space, while storing best matches as depth values into an inner state. Epipolar geometry comes into play since updating the stored depth values means moving over the epipolar lines

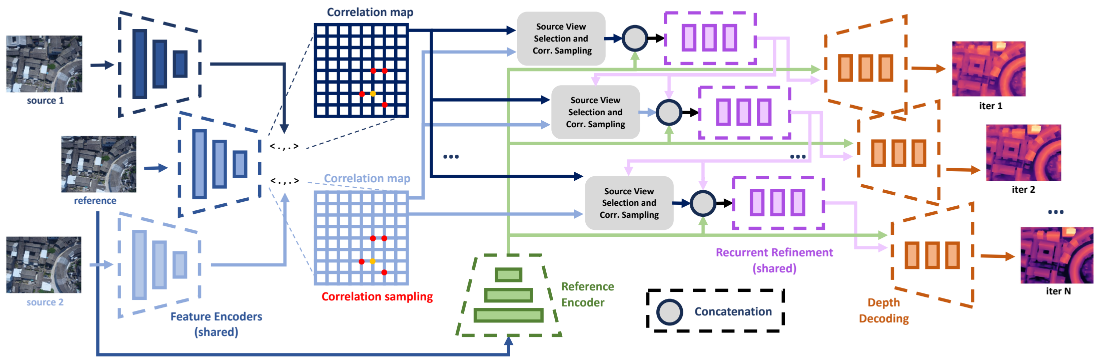

Correlation map
source 1
reference
source 2
Feature Encoders (shared)
Correlation map
Correlation sampling
Source View Selection and Corr. Sampling
Source View Selection and Corr. Sampling
Source View Selection and Corr. Sampling
Reference Encoder
Concatenation
Recurrent Refinement (shared)
Depth Decoding
iter 1
iter 2
iter N

Figure 6.2. **RAMDepth Architecture Description.** RAMDepth instantiates an initial depth map and builds a pair-wise correlation table between the target view and each source image (in dark and light blue). Then, deformable sampling is iteratively performed over it, and the depth state is updated accordingly. Final depth prediction is upsampled through convex upsampling.

defined by pose information. This approach can be thought of as reverting the common pipeline composed of (i) cost volume building and (ii) depth estimation. Moreover, can be pointed out that the dense matching costs computed by this framework, each expressing the relationship between a specific source view and the reference one, can be regarded as a hint of the overall matchability between the views, that takes into account both FoV overlap and overall image quality. Quantitative and qualitative evidence of this will be provided in Section 6.3.

RAMDepth, sketched in Fig. 6.2, can be decomposed into the following modules: (i) image features encoding, (ii) correlation sampling (iii) depth optimization, and (iv) output depth decoding. Steps (ii) and (iii) are performed multiple times for a fixed number of iterations. Thus, the framework outputs a sequence of depth maps $(D^s)_{s\in\mathbb{N}}$ getting progressively more accurate.

6.2.1 Features Encoding

Given a set of views $I^i,\ i \in [0, N]$ I^0 is referred to as the reference view – i.e., the one for which the depth map is predicted – and $I^i,\ i \in [1, N]$ as the source ones. Each view I^i is forwarded to a deep convolutional encoder to extract latent features $\mathcal{F}^i \in \mathbb{R}^{\frac{W}{8}\times\frac{H}{8}\times F}$, that will be used to compute correlation scores in the next step. These are depicted in shades of blue in Fig. 6.2 and share the same weights. Moreover, exclusively for I^0, it's also extracted a disentangled set of feature maps to provide monocular contextual information $\hat{\mathcal{F}} \in \mathbb{R}^{\frac{W}{8}\times\frac{H}{8}\times F}$, depicted in green in Fig. 6.2. Despite the iterative nature of RAMDepth, features are extracted only once, at bootstrap.

6.2.2 Correlation Sampling

Once the reference and source views have been encoded into deep latent features, at any iteration the current depth estimate $D^s_{u_0v_0}$ for pixel $q^0 = [u_0, v_0, 1]^T$ – in homogeneous coordinates – can be used to index a specific pixel $q^i = [u_i, v_i, 1]^T$ of a source view I^i as described in Eq. 6.1, according to camera intrinsic and extrinsic parameters K_0, K_i and E_0, E_i.

$$q^i = K_i E_i E_0^{-1} D_{u_0v_0} K_0^{-1} q^0 \tag{6.1}$$

This procedure leverages epipolar geometry since changing $D^s_{u_0v_0}$ means moving over the corresponding epipolar line while not being bound to *a priori* depth hypotheses. Then, source view features are sampled accordingly to compute a pixel-wise correlation map $C_{u_0v_0u_iv_i}$ – shown in Fig. 6.2 in shades of blue according to the selected source view

$$\mathcal{C}_{u_0v_0u_iv_i} = \sum_{f=1}^{F} \mathcal{F}^0_{u_0v_0f} \mathcal{F}^i_{u_iv_if} \tag{6.2}$$

However, this correlation map does not provide useful information on the direction in which better matches can be found. Thus, to better guide the optimization process correlation scores are computed in a neighborhood $\mathcal{N}(u_i, v_i)$ of q^i. Specifically, such a neighborhood is predicted by a 2D convolutional module Θ, predicting Z index offsets conditioned by the reference features $\hat{\mathcal{F}}$ and each iteration hidden state $\mathcal{H}^s$ – generated by the depth optimization step, see Section 6.2.4. The Z output channels are summed to the u_i, v_i coordinates to obtain the sampling locations.

$$\mathcal{N}(u_i, v_i) = \left[(u_i, v_i) + \Theta(\hat{\mathcal{F}}_{u_0v_0}, \mathcal{H}^s)_z,\ z \in Z\right] \tag{6.3}$$

This mechanism resembles deformable convolutions [30] in that it samples from a dynamic neighborhood, yet it differs since it does not accomplish a proper convolution with the sampled features but instead performs correlation with the features sampled from another view. It is worth observing that since Θ is conditioned with a state that changes at each iteration, these offsets may change at each iteration accordingly. The reference view context potentially allows to adaptively sample correlation scores in a narrower or wider region depending on the ambiguity of the reference image itself, like in the presence of object boundaries or low-textured regions.

The correlation sampling mechanism described so far works on a single source view at a time. This is a problem when multiple source views are available. Following existing approaches, correlation features could be extracted from each source view and then fused together. However, this approach would require developing a merging mechanism independent of the number of source views – e.g., simple concatenation would be unsuitable as it fixes the number of input channels. Many existing models compute feature-level variance to combine the volumes [171]. Instead, in RAMDepth a different source view is used for each update step, following a simple round-robin approach. This methodology is simple and elegant since it exploits the iterative nature of the architecture, does not require hand-crafted fusing modules, and can be extended to any variable number of source views. While different scheduling strategies can be employed, in this chapter only the simplest one is investigated, leaving the in-depth study of possible alternatives to future developments.

6.2.3 Keyframes Ranking

Since RAMDepth exploits a single source view at each iteration, $\mathcal{C}$ is related to a single specific source view, as it contains correlation scores between deep features of the source and reference views. Such correlation grows as the source view features are correctly projected over the reference view, and thus can be regarded as a score about matching quality [83]. It is worth mentioning that such a score is susceptible to the FoV overlap but also moving objects, blurring, or any other factor violating the multi-view geometry assumptions or that the encoding procedure is not robust against. Thus, it is directly linked with the capability of the network to exploit such source views to improve its prediction. Accordingly, each view can be ranked by taking the last correlation map computed for each source view and averaging it over the spatial dimensions. Since the network learns to perform good matches directly from depth supervision, there is no need to directly supervise this output which is a byproduct of the framework.

6.2.4 Depth Optimization

With the components defined so far, RAMDepth estimates a depth map for the reference view iteratively. At any stage s, a shallow recurrent network – in purple in Fig. 6.2 – made of a Gated Recurrent Unit processes the sampled correlation scores $\mathcal{C}$ and reference features $\hat{\mathcal{F}}$ together with the current hidden state $\mathcal{H}^s$ and depth map D^s (i.e., coming from the previous optimization stage) to output an updated hidden state $\mathcal{H}^{s+1}$. Then, two convolutional layers predict a depth update ΔD^s yielding a refined depth map $D^{s+1} := D^s + \Delta D^s$. At bootstrap, D^0 is initialized to zero and then the aforementioned iterative process allows for rapidly updating the depth map state towards a final, accurate prediction. At the first iteration, the correlation scores $\mathcal{C}$ will not be meaningful for depth, thus the network learns to provide a monocular initialization for D^1. Other approaches could consist of either randomly initializing D^0 or inserting a further module to learn an initialization. The former would be inaccurate if no information about the depth range is assumed, the latter is equivalent to zero initialization yet requires an extra component.

6.2.5 Depth Decoding

Since RAMDepth iterates at a lower resolution, a final upsampling of the depth maps to the original input resolution is required. Many approaches leverage either bilinear upsampling [171, 172, 48, 17, 150] or a deep convolutional decoder. Instead, a weighting mask is computed with an upsampling module – in orange in Fig. 6.2 – fed with the latest hidden state $\mathcal{H}^{s+1}$ and the reference view features $\hat{\mathcal{F}}$, then convex upsampling [139] is applied. This approach is faster than employing a decoder and yields much better results compared to using hand-crafted upsampling approaches.

Method	Ground Truth Depth Range							Unique Depth Range						
	MAE	RMSE	>1 m	>2 m	>3 m	>4 m	>8 m	MAE	RMSE	>1 m	>2 m	>3 m	>4 m	>8 m
Yao et al. [171]	0.6168	1.5943	0.1392	0.0731	0.0457	0.0309	0.0103	2.1115	5.3122	0.3021	0.1637	0.1194	0.0964	0.0526
Yao et al. [172]	0.7815	1.7397	0.1864	0.1007	0.0637	0.0433	0.0141	1.2568	2.6033	0.2918	0.1464	0.0933	0.0676	0.0286
Cheng et al. [17]	0.3590	1.3589	0.0704	0.0378	0.0244	0.0171	0.0064	1.6489	4.1094	0.1844	0.1235	0.1046	0.0932	0.0602
Wang et al. [150]	0.3849	1.3581	0.0749	0.0386	0.0247	0.0175	0.0067	22.420	25.026	0.6721	0.5067	0.4989	0.4956	0.4761
Gu et al. [48]	0.3684	1.3449	0.0714	0.0365	0.0234	0.0165	0.0062	1.8978	4.2927	0.2341	0.1427	0.1101	0.0921	0.0597
Zhang et al. [181]	0.3318	1.2396	0.0662	0.0323	0.0197	0.0133	0.0044	1.0536	2.8939	0.1682	0.0913	0.0643	0.0508	0.0285
Sayed et al. [120]	0.5921	1.4340	0.1404	0.0584	0.0308	0.0191	0.0057	0.5921	1.4340	0.1404	0.0584	0.0308	0.0191	0.0057
Ma et al. [94]	2.1666	26.934	0.0752	0.0441	0.0316	0.0247	0.0138	8.2120	55.710	0.5780	0.5400	0.4960	0.3540	1.1150
RAMDepth	0.2982	1.1724	0.0645	0.0285	0.0159	0.0102	0.0033	0.2982	1.1724	0.0645	0.0285	0.0159	0.0102	0.0033

(a) (b)

Table 6.1. **Blended Benchmark**. Comparisons with existing methods under two settings: (a) by providing full knowledge about the scene depth to each method, (b) by assuming a unique depth range to cover the whole test set. Since RAMDepth does not exploit any knowledge about such range, its accuracy is not affected by the setup, unlike others.

6.2.6 Loss Function

RAMDepth is supervised by computing a simple L1 loss between the ground-truth depth D_{gt} and each estimated depth map, with a weight decay γ set to 0.8

$$\mathcal{L} = \sum_{s=1}^{S} \gamma^{S-s} ||D_{\text{gt}} - D^s||_1 \tag{6.4}$$

6.3 Experimental Results

To assess the effectiveness of RAMDepth in the most challenging environments available, experiments are performed on Blended [173], TartanAir [153], UnrealStereo4K [144] and DTU [62]. These datasets cover a wide range of applications of interest – e.g. outdoor multi-view settings, monocular video sequences, stereo perception, and object-centric indoor setups. Specifically, Blended [173] provides large complex aerial views of buildings characterized by high inter-view pose displacements, while TartanAir provides outdoor and indoor monocular video sequences with small but unpredictable pose changes. In both, it is difficult to decide the depth range *a priori* as it is not usually constant within the same scene as well between scenes. On UnrealStereo4K [144], the generalization capability of RAMDepth and the possibility to perform stereo depth perception seamlessly are assessed – to further support its strong matching effectiveness. Finally, DTU [62] provides interesting cues about the performance in a controlled environment, where the depth range can be accurately known *a priori*. RAMDepth consists of 5.9M parameters. In any experiment, it is computed the mean absolute error (MAE), root mean squared error (RMSE), and the percentage of pixels having depth error larger than a given threshold ($> \tau$).

Blended Benchmark. The Blended dataset [173] collects 110K images from about 500 scenes, rendered from meshes obtained through 3D reconstruction pipelines. It features large overhead views where the scene depth range would be hard to be properly recovered in a real use case, but also several object closeups. Following [105], each method is tested with five input images on

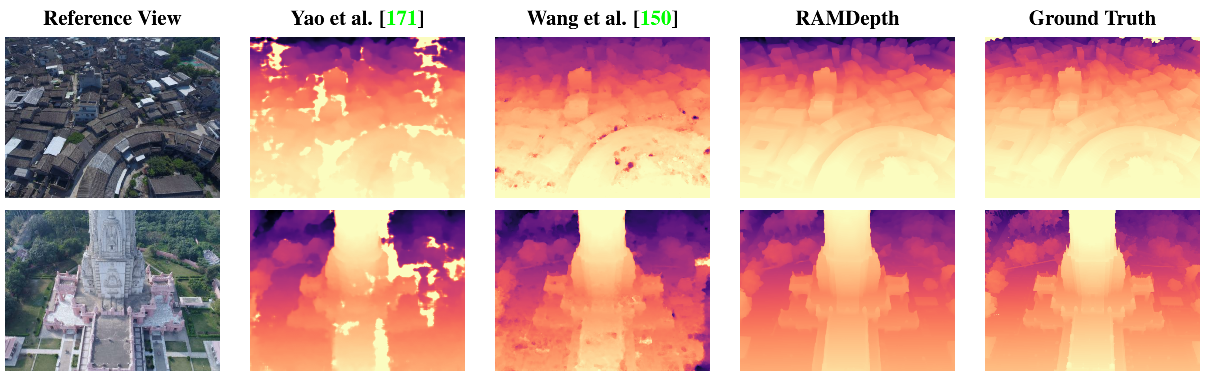
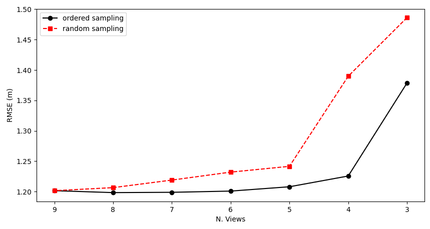


Figure 6.3. **Qualitative results on Blended.** RAMDepth extracts consistent and visually pleasant depth maps, not showing any visible outliers as can be observed in competitor methods.

	Convex Upsampling	Deformable Sampling	MAE	>1 m
Baseline			0.4673	0.1085
Baseline + Deform.		✓	0.4525	0.1046
Baseline + Convex	✓		0.3406	0.0756
Full	✓	✓	0.3197	0.0695
Full (Tuned)	✓	✓	0.2982	0.0645

(a) Ablation Study



(b) Keyframes Ranking

Table 6.2. **Ablation study on RAMDepth.** On the left, the impact of convex upsampling and deformable sampling modules are assessed on Blended [173]. Each ablation has been performed with the same number of training steps, smaller than the total used to train the final model (Tuned). When convex upsampling is not applied bilinear upsampling is used instead. On the right, RMSE is achieved by dropping input views in random order (red) or according to the ranking information provided by RAMDepth (black)

the standard test set, composed of 7 heterogeneous scenes. In this experiment, each method except RAMDepth exploits the reference view ground-truth depth range. Results are collected in Table 6.1 (a). RAMDepth consistently produces more accurate depth maps, despite not exploiting any knowledge about the depth range of any scene. It's worth pointing out how RAMDepth produces much better depth maps than other methods, which show frequent artifacts as shown in Fig. 6.3.

Depth Range Analysis. In this section the focus is on the importance of not depending on prior knowledge about the scene depth range. Purposely, a benchmark tailored to study this specific aspect on the Blended test set has been designed, given the wide set of heterogeneous scenes with depth ranges varying from a few meters up to hundreds. In Table 6.1 (b) each competitor relying on the depth range is fed with a global unique depth range, computed to cover the whole dataset one. To ease the task for competitor methods the following steps are performed: (i) normalize the extrinsic translations between the reference and the source views to have a mean value equal to 1 and compute the corresponding depth scaling factor, (ii) compute the mean depth on the test set using the rescaled ground-truth depth and estimate an appropriate set of depth hypotheses equal for every sample to cover the whole dataset depth range, (iii) the depth predictions by models processing depth hypotheses are scaled back to the original metri-

Method	Monocular Video Benchmark							Stereo Benchmark					
	MAE (m)	RMSE (m)	>1 m	>2 m	>3 m	>4 m	>8 m	MAE (px)	RMSE (px)	>1 px	>2 px	>3 px	>4 px
Yao et al. [171]	5.330	8.638	0.590	0.418	0.329	0.273	0.166	9.1420	16.142	0.685	0.456	0.352	0.304
Yao et al. [172]	4.077	7.106	0.550	0.376	0.278	0.216	0.118	8.9630	16.057	0.663	0.425	0.320	0.270
Cheng et al. [17]	6.511	9.935	0.635	0.468	0.375	0.314	0.196	7.5390	16.096	0.357	0.261	0.230	0.211
Wang et al. [150]	7.883	10.84	0.637	0.495	0.413	0.359	0.244	3.4850	10.462	0.240	0.160	0.129	0.112
Gu et al. [48]	6.364	9.521	0.630	0.469	0.373	0.314	0.197	18.408	68.167	0.424	0.342	0.304	0.278
Zhang et al. [181]	6.287	8.949	0.602	0.454	0.373	0.319	0.208	9.8990	26.357	0.319	0.256	0.226	0.206
Sayed et al. [120]	5.460	7.951	0.743	0.566	0.439	0.350	0.168	22.323	27.022	0.979	0.959	0.944	0.924
Ma et al. [94]	7.344	13.74	0.645	0.474	0.372	0.306	0.187	3.8320	10.156	0.268	0.196	0.161	0.137
RAMDepth	3.773	6.876	0.514	0.353	0.264	0.201	0.101	1.837	5.7930	0.157	0.099	0.076	0.063
Lipson et al † [84]	-	-	-	-	-	-	-	1.646	4.8090	0.139	0.089	0.069	0.057
	(a)							(b)					

Table 6.3. **UnrealStereo4k Benchmark**. Application of RAMDepth and competitor frameworks to UnrealStereo4k either selecting source views from monocular video sequences (a) or using rectified left and right stereo couples as target and source views (b). Images processed at 960×544 resolution.

Method	2D Metrics						
	MAE	RMSE	>1 m	>2 m	>3 m	>4 m	>8 m
Yao et al. [171]	1.887	4.457	0.278	0.183	0.138	0.110	0.056
Yao et al. [172]	2.191	4.729	0.346	0.228	0.170	0.134	0.066
Cheng et al. [17]	1.461	3.860	0.216	0.141	0.106	0.084	0.043
Wang et al. [150]	2.351	4.980	0.331	0.228	0.176	0.144	0.078
Gu et al. [48]	1.582	4.017	0.230	0.150	0.113	0.090	0.047
Sayed et al. [120]	1.561	3.303	0.316	0.167	0.112	0.083	0.036
Ma et al. [94]	3.405	10.20	0.322	0.211	0.163	0.134	0.077
RAMDepth	1.258	3.289	0.203	0.125	0.090	0.070	0.034

(a) TartanAir Benchmark

Method	2D Metrics				3D Metrics		
	>1 mm	>2 mm	>3 mm	>4 mm	acc.	compl.	avg
Yao et al. [171]	0.5550	0.3400	0.2680	0.2370	0.6350	0.3040	0.4695
Yao et al. [172]	0.6300	0.4230	0.3290	0.2830	0.6620	0.3420	0.5020
Cheng et al. [17]	0.5060	0.3320	0.2770	0.2540	0.5510	0.2720	0.4115
Wang et al. [150]	0.4750	0.3100	0.2600	0.2360	0.4610	0.2980	0.3795
Gu et al. [48]	0.4800	0.3070	0.2570	0.2330	0.5280	0.2620	0.3950
Ma et al. [94]	0.4126	0.2556	0.2029	0.1770	0.4966	0.2581	0.3773
RAMDepth	0.3683	0.2439	0.2063	0.1884	0.4466	0.2775	0.3620

(b) DTU Benchmark

Table 6.4. **TartanAir & DTU Benchmark.** Results achieved by existing multi-view frameworks and RAMDepth on TartanAir [153] and DTU [62]. On TartanAir the proposed method consistently demonstrates better performance. On DTU RAMDepth is comparable in performance despite the advantage of the other methods, knowing the fixed depth range of the dataset.

cal range. This procedure acts on the scene scale only, not affecting the performance of trained networks, except for changing the set of depth hypotheses used to build cost volumes. This procedure is a precaution adopted to have closer values for minimum and maximum depth in large-scale scenes, to ease numerical stability. Nonetheless, results reported in Table 6.1 (b) emphasize how the lack of precise knowledge about the depth range of the scene heavily penalizes existing methods, whereas RAMDepth remains unaffected.

Keyframes Ranking. To assess the quality of the source views ranking produced by RAMDepth, a peculiar experiment is proposed: for each sample in the Blended test set, its source frames are ranked according to the method described in Section 6.2. Then, the number of source frames provided to the framework is progressively decreased by selecting them either randomly or according to the generated ranking. Fig. 6.2 (b) shows the results of this experiment. Selecting frames according to the proposed ranking approach yields an overall error that diverges much more slowly. Despite not being the direct goal of this chapter, this experiment lays the groundwork for interesting potential applications like automatically removing blurred, out-of-view, or non-static frames from the set of source views, as may happen on video sequences.

UnrealStereo4k Benchmark. The UnrealStereo4K dataset [144] provides synthetic stereo videos in different challenging scenarios. On this dataset, it's assessed the generalization capabilities of RAMDepth on monocular video sequences and, peculiarly, at dealing with the

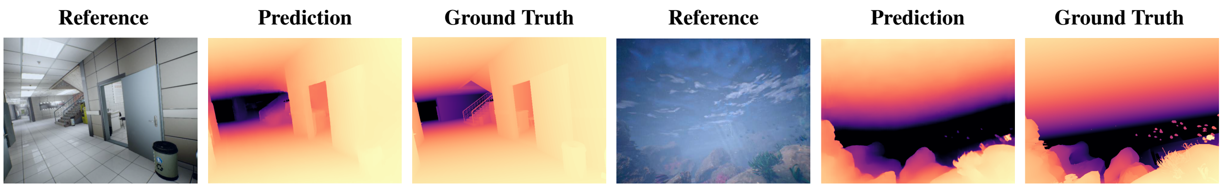


Figure 6.4. **TartanAir Qualitatives.** TartanAir provides a wide range of complex environments, in figure a few examples along with the predictions by RAMDepth.

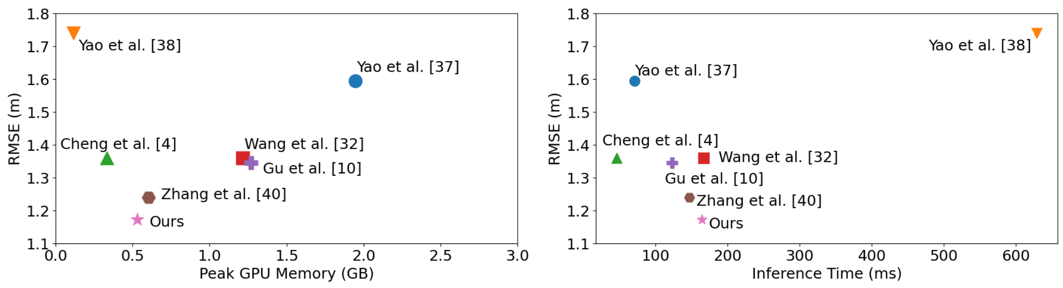


Figure 6.5. **Benchmark on Memory and Time Requirements.** Each model is tested in evaluation mode on a single NVIDIA RTX 3090 in 32FP precision, with input size 768×576 and 5 input views. peak memory is measured since it is the minimum memory needed to run a model in evaluation, time in milliseconds and RMSE on Blended [173].

rectified stereo use case. Thus, the Blended pre-trained models are used without any kind of fine-tuning. Concerning the stereo perception application, the right view is used as the reference view and the left as the source one. Even in this case, the ground-truth depth range is provided in input to all the methods requiring *a priori* depth hypotheses. However, it is worth mentioning that from a practical point of view, this is an unrealistic assumption when dealing with left-right stereo pairs, yet necessary to deploy multi-view networks relying on depth hypotheses in this setting – except for RAMDepth. In Table 6.3 five consecutive frames are leveraged (a) or a single stereo pair (b). In both cases, it achieved substantial improvements over existing models, highlighting a dramatic margin by RAMDepth over other solutions. As a reference, it is also reported the performance achieved by [84], a state-of-the-art stereo network trained on a variety of stereo datasets, to highlight how close the proposed solution gets to it, despite not being trained explicitly to deal with this specific setting – since Blended is not even a stereo dataset. This evidence further supports the great flexibility of RAMDepth.

TartanAir Benchmark. The TartanAir dataset [153] is a large synthetic dataset composed of a wide spectrum of indoor, outdoor, aerial, and underwater scenarios recorded by a monocular camera, with different moving patterns of variable toughness. It also contains a few moving objects like fishes, steam, and industrial machines as well as high-frequency details like tree leaves. In this scenario, the depth range of each single view is hard to define since it can embrace hundreds of meters in a landscape view or a few meters when the camera moves around a wall, and this can happen within the same scene as well. Thus, this environment is a perfect benchmark for RAMDepth. Table 6.4 (a) shows the performance of the proposed approach and existing multi-view methods, where each competitor is fed with the depth range from the ground-truth depth. Even though this is unfair to RAMDepth, not knowing anything about the prediction range, still it exhibits the best performance. A few qualitative examples are shown in Fig. 6.4.

DTU Benchmark. DTU [62] is a dataset composed of small objects whose 3D structure is captured by means of a robotic arm and a structured light sensor. Due to these specifics, it exhibits a really small and fixed depth range. In this context, methods relying on the scene depth range are advantaged since they can make use of robust and precise information which limits outliers, especially in texture-less areas. Following [105] each method is pretrained on [173]. In Table 6.4 (b) are showed both 2D depth metrics and standard 3D metrics obtained with the same reconstruction pipeline from [105] on [62]. RAMDepth is still competitive in both 3D point cloud reconstruction and depth estimation, despite being disadvantaged in this context.

Ablation study. A simple ablation study about neighborhood sampling and depth decoding components is shown in Table 6.2 (a). Such an experiment is performed with a slightly smaller number of training steps on Blended [173] with respect to the final tuned model, thus are reported also the results of the final model for a better comparison. RAMDepth greatly benefits from both of these modules.

Memory and Time Analysis. In this section, an analysis of the time and memory requirements of the proposed method is provided, compared with existing approaches in Fig. 6.5. Peak memory usage, runtime, and RMSE error are measured using 5 input views of size 768×576, on a single NVIDIA RTX 3090. The choice to measure peak memory is justified by the fact that this latter is the minimum memory required when deploying these models in a real application and thus it is the most significant metric in this sense. In Fig. 6.5 can be clearly observed that despite being neither the fastest nor the lighter approach, RAMDepth provides a good balance in memory usage and inference time, while still being the best one in performance.

Qualitative Results. In this section are reported a few sample scenes from the datasets used in this chapter to show the network's capability to extract fine details. In Figure 6.6 4 out of 5 views from TartanAir [153] are plotted along with the prediction and the ground truth (dark black represents missing values in the ground truth).

In Figure 6.8 a few samples from the available sequences of UnrealStereo4K are shown. UnrealStereo4K is a very challenging dataset containing heterogeneous indoor and outdoor scenes. RAMDepth is not fine-tuned on the dataset itself – *i.e.* the model trained on Blended is used to assess the generalization capabilities of the proposed approach.

In Figure 6.9, qualitative results about 3D reconstruction on DTU are reported. A depth map for each view available for a single scene is generated, leveraging a total of 5 views for each prediction. Then, such depth maps are assembled together by applying geometric and photometric filtering common in the literature [62].

Keyframes Ranking Qualitatives. In this section, a qualitative example of the effectiveness of keyframes ranking enabled by RAMDepth is provided. Figure 6.10 showed a scene from Blended composed of 6 frames: the first one is the reference view, then source views follow in random order. Correlation scores are extracted with the procedure described in Section 6.2.2

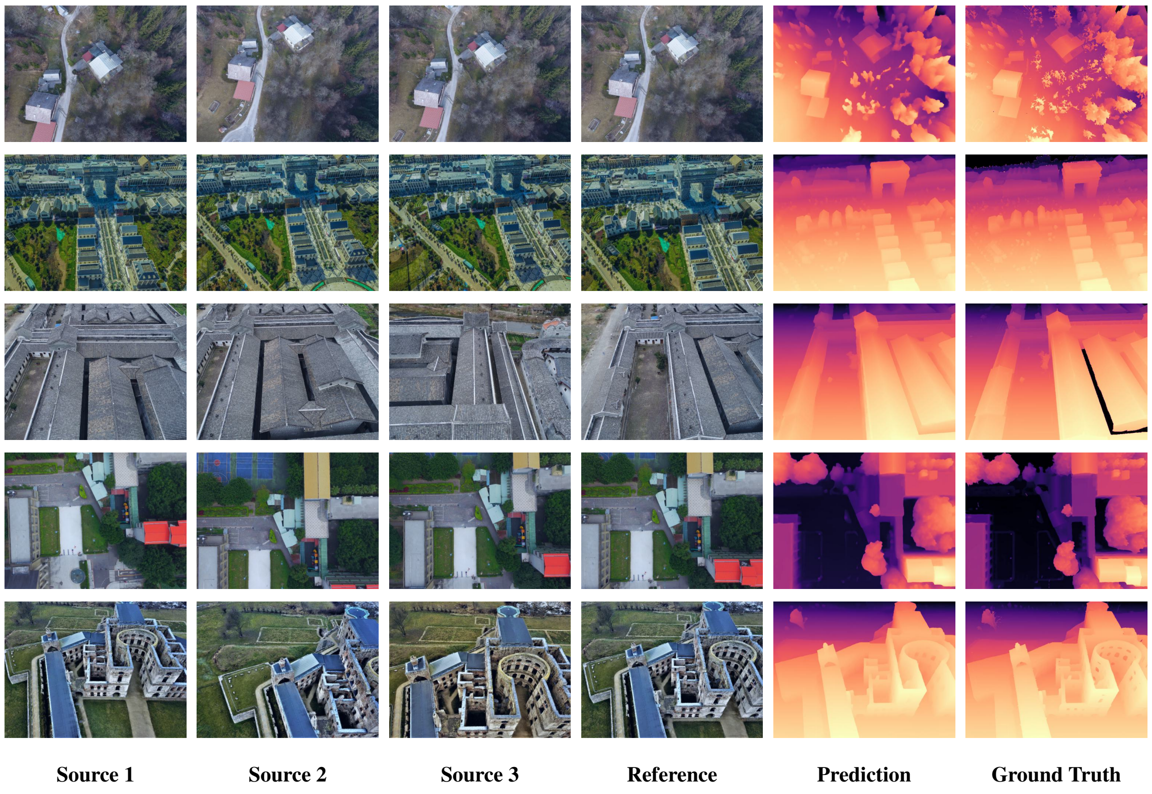


Figure 6.6. **Qualitative results on Blended.** Predictions obtained by using 5 views as input (only 4 are showed for representative purpose).

and order views accordingly in the second row. Can be noticed that higher scores are assigned to views with a higher visual overlap, *e.g.* the first source view in the ordered row is the one that maximizes matches with the building highlighted in red, the street on its left, and the garden between buildings, which are largely occluded in the other views. Finally, in the last row, the first 2 most correlated views according to the framework output are corrupted with a simple Gaussian blur to simulate out-of-focus images and rank once more. Can be observed that the proposed framework now assigns the lowest score to the out-of-focus views, although these were the best before. These experiments qualitatively demonstrate that RAMDepth approach takes deeply into account not only the relative position between views but also the 3D structure of the scene and the quality of matches it can recover from the available views. Thus, the proposed framework provides a view-centric methodology to discard poorly correlated views (e.g. out-of-focus, blurred, with moving objects), which cannot be achieved by reasoning only about relative pose.

Source Views Scheduling Analysis. As already previously detailed, a simple round-robin schedule is applied to sample the source view used to sample correlation matches at each network iteration. This approach does not cause any particular problem. Indeed, even though the source view is changed at each iteration the depth state is independent of the latter, thus enforcing consistency. To assess that the proposed approach is not significantly affected by source views ordering, a simple experiment is performed: on the Blended [173] test set, metrics can be computed for each permutation of the source views, and then the standard deviation can be

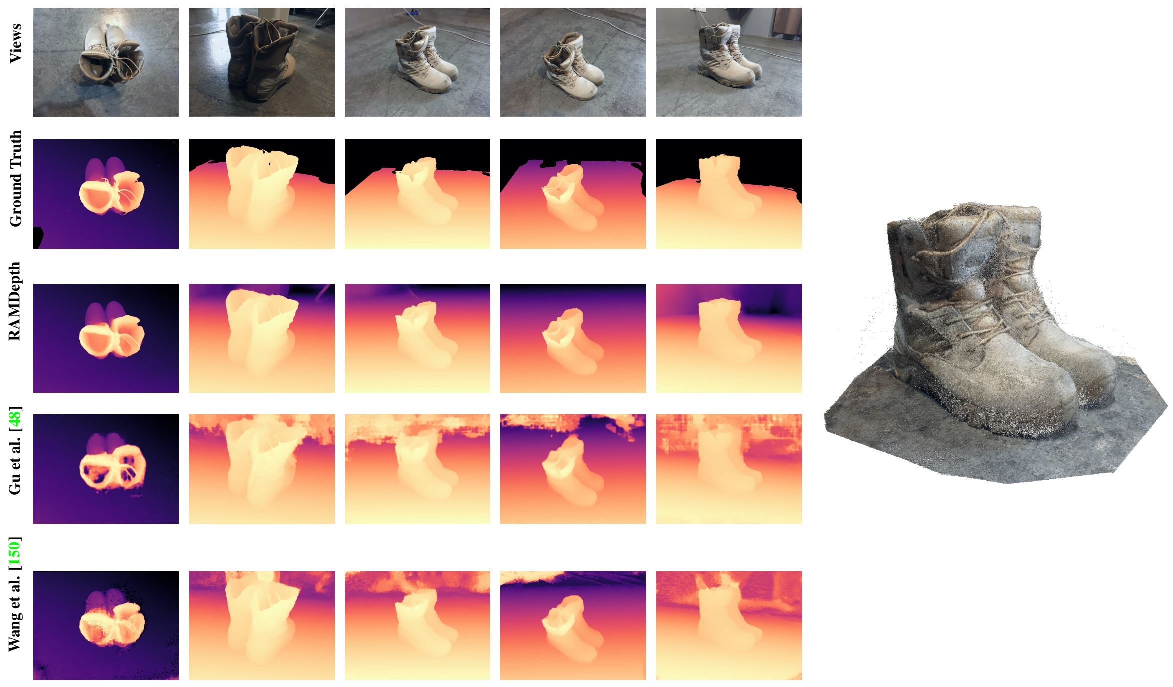


Figure 6.7. **Wrong Depth Range Effects Example on Blended** On left: five views from the Blended [173] scene and their ground-truth depth, followed by depth maps estimated by RAMDepth, [48] and [150]. RAMDepth does not require any knowledge about the depth range and thus provides consistently smoother depth maps on the entire scene, even out of the pre-defined range where [48] and [150] struggle. On the right: 3D reconstruction obtained by merging the predictions, the floor reconstruction is limited to better highlight the object details, despite the fact that RAMDepth is able to reconstruct the whole area.

evaluated. In Table 6.5 are reported the results of such experiment. The very low variance reported at the very bottom confirms how the ordering used to iterate over the source views has a negligible impact on the final quality of the predicted depth map.

Impact of the Depth Range. An example showing the negative impact that an inaccurate depth range can produce on the predictions of existing frameworks is reported in this section. Figure 6.7 shows, from top to bottom, five images taken from Blended, their corresponding ground-truth depth, and the predictions by RAMDepth and two previous works [48, 6]. These latter expose large artifacts in the farthest regions of the images, caused by the inaccurate depth range over which they operate. Indeed, as can be noticed in the second row, ground-truth depth is not provided for those regions, and thus the depth range used for computing the depth map does not contain them – since it is estimated directly from ground-truth. On the contrary, RAMDepth produces clean and detailed depth maps even in these portions of the images.

Architecture Details. In this section, the core components of RAMDepth are described in detail. In Table 6.6, each module is detailed in terms of layers along with their parameters, inputs (in red), and outputs (in blue). Input source and target views are encoded through the Feature Encoder, then disentangled information is extracted from the reference view through the Reference Encoder (called context in Table 6.6 and accounting 128 channels). Depth, hidden

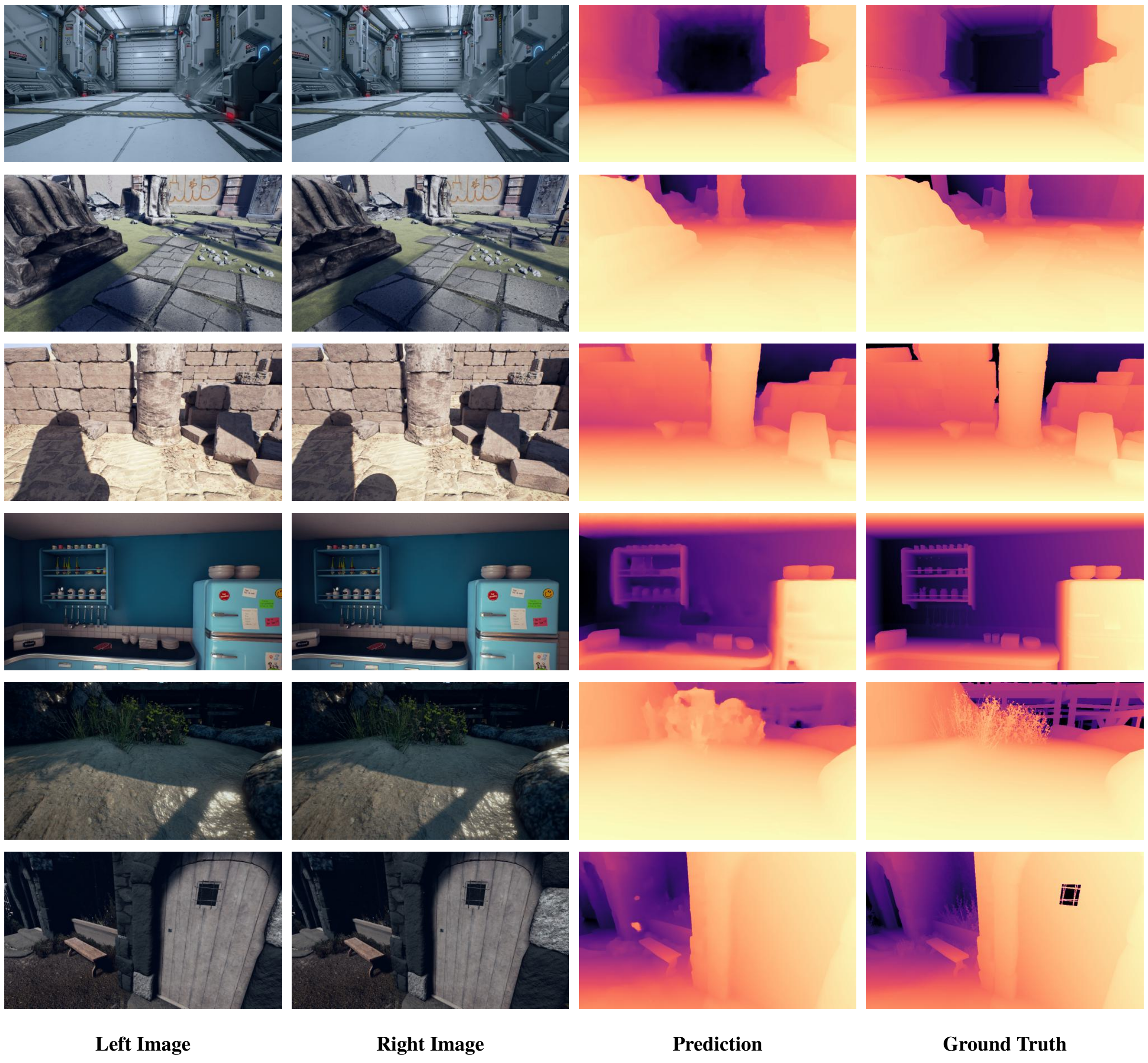


Figure 6.8. **Qualitative results on UnrealStereo4K Stereo.** The pre-trained model on Blended are used to show the capability of the proposed method to generalize across datasets.

state, and reference features are used to predict sampling offsets, correlation scores are sampled according to the methodology in previous sections and the recurrent block predicts a new hidden state and a Δdepth update. Finally, a shallow module predicts upsampling weights from the hidden state and reference information and performs convex upsampling. RAMDepth is trained on Blended, TartanAir, and DTU with AdamW, learning rate 10^{-4} and weight decay 10^{-5}. Gradients are always clipped with global norm 1 to stabilize the behavior of Gated Recurrent Units. On Blended, the relative pose translation (between the reference and source views) is normalized to have a mean value of 1 for better numerical stability. On Blended, 200K steps are performed, and then 100K steps with a learning rate of 10^{-5} are applied for fine-tuning. On DTU, the 200K Blended checkpoint is fine-tuned for 100K steps with a learning rate of 10^{-4}. Training is always with batch size 1 on 2 RTX 3090 in mixed precision. During training and evaluation, 10 cycles over the input source views are always used, that is 40 total steps with 4 source views, except for the UnrealStereo4K stereo benchmark where 40 total updating steps on

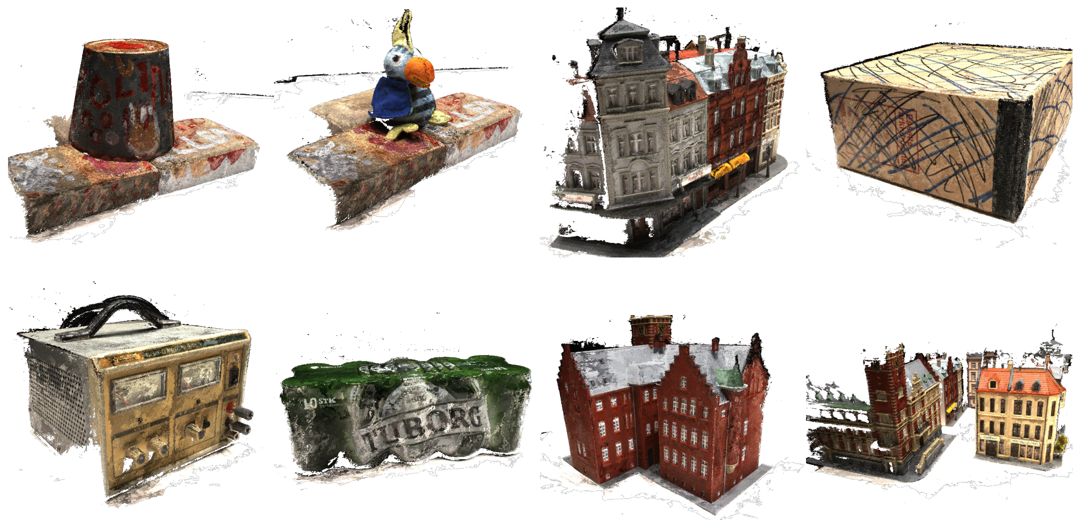


Figure 6.9. **Qualitative results on DTU.** 3D reconstruction of different objects and scenes provided by DTU to assess the capability of the proposed approach to generate accurate point cloud reconstructions even though RAMDepth is focused on highly detailed depth map estimation.

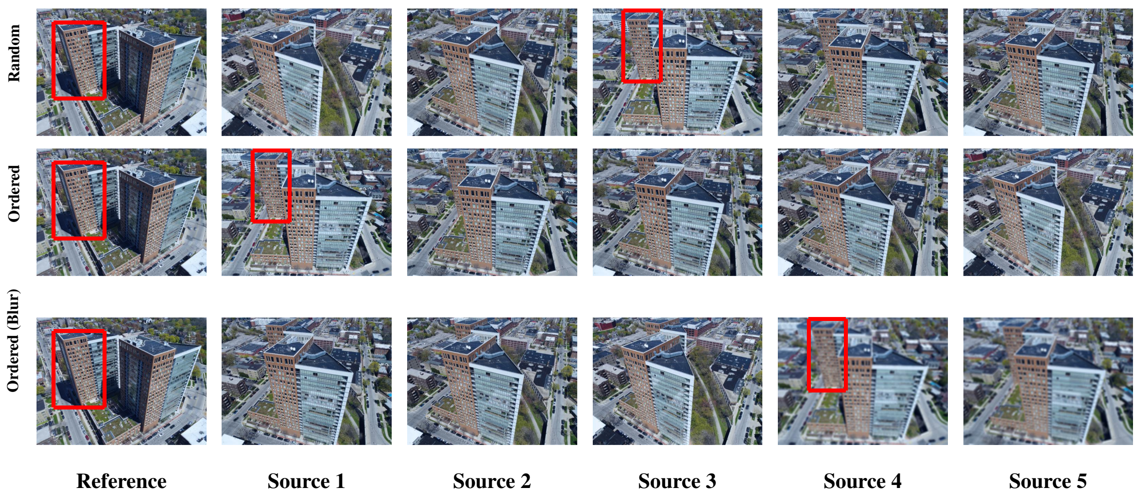


Figure 6.10. **Keyframes Ranking Example.** In the first row, a scene from Blended containing 5 source views in random order is showed. In the second row, the framework reordering. If Gaussian blur is applied to the images with the best score and apply again reordering (last row), RAMDepth assigns to them the worst score this time.

the unique source view available are used. In all the experiments, dynamic offsets are computed in a neighborhood of size $||\mathcal{N}|| = 9 \times 9$ for a total of 81 sampling coordinates.

Permutation	MAE	RMSE	>1 m	>2 m	>3 m	>4 m	>8 m
N. 1	0.316181	1.186300	0.069861	0.031390	0.017675	0.011238	0.003503
N. 2	0.317703	1.188342	0.070145	0.031508	0.017682	0.011203	0.003491
N. 3	0.318048	1.187708	0.070530	0.031465	0.017649	0.011226	0.003501
N. 4	0.319271	1.187811	0.070630	0.031412	0.017647	0.011224	0.003536
N. 5	0.315665	1.186850	0.069935	0.031355	0.017527	0.011122	0.003460
N. 6	0.316765	1.187873	0.070035	0.031531	0.017691	0.011216	0.003522
Std.	0.001328	0.000755	0.000319	0.000069	0.000061	0.000042	0.000026

Table 6.5. **Source Views Scheduling Analysis.** For each sample in the test set of Blended each source views permutation is evaluated the standard deviation of the metrics is computed. 3 source views are used to limit the number of permutations.

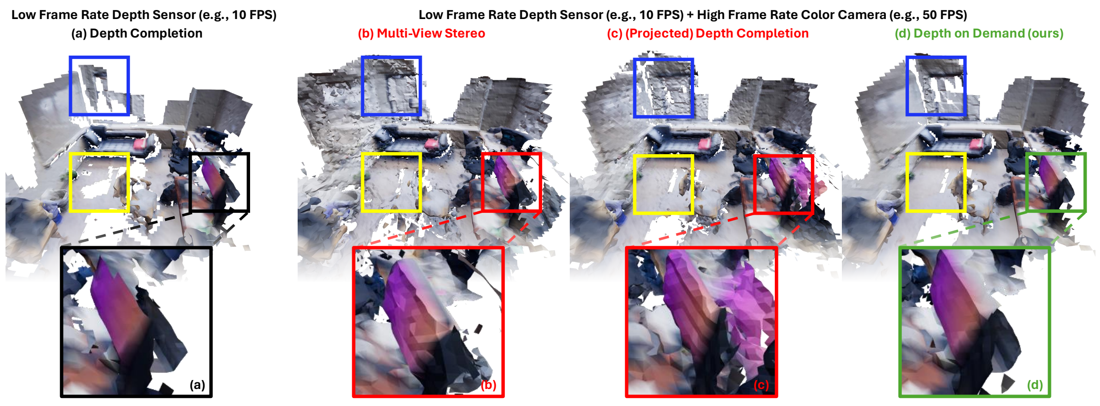
Name	Layer	K	S	In/Out	Input
Residual Block Stride 2					
conv0	Conv2D + BatchNorm2D + ReLU	3	2	In/Out	**input**
conv1	Conv2D + BatchNorm2D + ReLU	3	1	Out/Out	conv0
downs	Conv2D + BatchNorm2D + ReLU	1	2	In/Out	**input**
out	ReLU	-	-	Out/Out	downs + conv1
Residual Block Stride 1					
conv0	Conv2D + BatchNorm2D + ReLU	3	2	In/Out	**input**
conv1	Conv2D + BatchNorm2D + ReLU	3	1	Out/Out	conv0
out	ReLU	-	-	Out/Out	conv1 + input
Feature Encoder & Reference Encoder					
conv0	Conv2D + BatchNorm2D + ReLU	7	2	3/64	**image**
conv1	Residual Block Stride 2	-	-	64/64	conv0
conv2	Residual Block Stride 1	-	-	64/64	conv1
conv3	Residual Block Stride 2	-	-	64/96	conv2
conv4	Residual Block Stride 1	-	-	96/96	conv3
conv5	Residual Block Stride 2	-	-	96/128	conv4
conv6	Residual Block Stride 1	-	-	128/128	conv5
conv7	Residual Block Stride 2	-	-	96/128	conv6
conv8	Residual Block Stride 1	-	-	128/128	conv7
feats	Conv2D	1	1	128/256	conv8

Name	Layer	K	S	In/Out	Input
Offsets Computation					
conv0	Conv2D + BatchNorm2D + ReLU	3	1	128+128+1/256	**context**, **hidden**$_{s-1}$, **depth**$_{s-1}$
offsets	Conv2D	1	1	256/9×9×2	conv0
Recurrent Block					
corr0	Conv2D + ReLU	1	1	9×9/256	**corrfeats**
corr1	Conv2D + ReLU	3	1	256/192	corr0
dfeats0	Conv2D + ReLU	7	1	1/128	**depth**$_{s-1}$
dfeats1	Conv2D + ReLU	3	1	128/64	dfeats0
conv0	Conv2D + ReLU	3	1	192+64/128-1	dfeats1, corr1
hidden0	ConvGRU2D	(1, 5)	1	128+1+128+128/128	**context**, conv0, **depth**$_{s-1}$, **hidden**$_{s-1}$
hidden$_s$	ConvGRU2D	(5, 1)	1	128/128	hidden$_0$
conv1	Conv2D + ReLU	3	1	128/64	**hidden**$_s$
Δdepth	Conv2D + ReLU	3	1	64/1	conv1
Convex Upsampling					
conv0	Conv2D + ReLU	3	1	128+256/128+256	**hidden**$_s$, **context**
upmask	Conv2D	1	1	128+256/8×8×9	conv0

Table 6.6. **Framework Modules Description.** Each learned component of RAMDepth. Each module inputs and outputs are shown in red and blue, respectively.

Chapter 7

ToF and RGB Video Streams Integration



Low Frame Rate Depth Sensor (e.g., 10 FPS)
(a) Depth Completion
Low Frame Rate Depth Sensor (e.g., 10 FPS) + High Frame Rate Color Camera (e.g., 50 FPS)
(b) Multi-View Stereo
(c) (Projected) Depth Completion
(d) Depth on Demand (ours)
(a)
(b)
(c)
(d)

Figure 7.1. **3D reconstruction with a low frame rate, sparse depth sensor.** Running depth completion (a) on low FPS, sparse depth maps generate holes in the final reconstruction. Adding a higher FPS color camera allows for obtaining depth from Multi-View Stereo (b) or projecting depth to nearby color views and running completion (c), with unsatisfactory results. DoD (d) performs *temporal* completion using two views and one sparse depth frame, yielding denser and more accurate meshes.

7.1 Introduction

High frame rate and accurate depth estimation play an important role in several tasks crucial to robotics and automotive perception. To date, this can be achieved through ToF and LiDAR devices for indoor and outdoor applications, respectively. However, their applicability is limited by low frame rate, energy consumption, and spatial sparsity. In this Chapter is proposed Depth on Demand (DoD) – a framework that allows for accurate temporal and spatial depth densification achieved by exploiting a high frame rate RGB sensor coupled with a potentially lower frame rate and sparse active depth sensor. Such a framework jointly enables lower energy consumption and denser shape reconstruction, by significantly reducing the streaming requirements on the depth sensor thanks to its three core stages: i) multi-modal encoding, ii) iterative multi-modal integration, and iii) depth decoding. Extended evidence is reported assessing the effectiveness of DoD on indoor and outdoor video datasets, covering both environment scanning and automotive perception use cases. In the following, the intrinsic issues related to active depth sensing are deeply and how DoD addresses them is deeply described.

In the last decade, RGB-D camera systems have become prominent in fields such as robotics, automotive, and augmented reality, and have scaled down from Kinect v1 to mobile handheld devices such as the Apple iPad. In such systems, one or more conventional RGB cameras are coupled with an *active* depth sensor, *i.e.*, a device that leverages active illumination to infer the 3D structure of the framed scene [113]. Among these sensors, Time-of-Flight (ToF) cameras infer the distance by emitting modulated infrared light into the scene and measuring its return time [113, 8, 5]. On the other hand, Light Detection And Ranging (LiDAR) sensors allow for long-range measurements up to hundreds of meters with or without sunlight at much higher energy consumption and footprint.

Despite the reconstruction accuracy of active depth sensors, their inherent structure places limits on their usability. ToF sensors are mainly used for mobile devices and can achieve high frame rates, but imply high energy consumption compared to the limited available battery and overheating. Usually, a drastic reduction of their frame rate is required since it corresponds to a drastic reduction in energy consumption. On the other hand, LiDAR sensors are bulky devices mainly used for autonomous driving, and their moving mechanical components (scanning mirrors) limit the frame rate. Finally, both these technologies manifest spatial sparsity, generating meaningful predictions only for specific spatial locations. Such sparsity can intentionally be induced to minimize acquisition time (in scanned LiDAR [91, 7]) or energy consumption (in ToF [63, 89]). Due to these constraints, the adoption of RGB-D camera systems is difficult in various scenarios. Indeed, Augmented Reality (AR) requires extremely low-power camera systems to fit severely constrained energy budgets. Autonomous driving requires high frame rate depth perception to allow reactive safety-critical applications. 3D shape reconstruction from video streams benefits high frame rate reconstruction to achieve dense meshes without requiring very slow movements from the operator.

This chapter proposes Depth on Demand (DoD), a framework addressing the three major issues related to active depth sensors in streaming dense depth maps – *i.e.* spatial sparsity, energy consumption, and limited frame rate. The spatial resolution problem represents a well-known issue deeply investigated by the research community through depth completion [56]. On the other hand, energy footprint and low frame rate issues have often been ignored in the literature, although prominent in the deployment of RGB-D systems. It's worth observing that reducing the active sensor temporal resolution – *i.e.* its frame rate – power consumption can be modulated accordingly. Indeed, ToF sensors' energy consumption scales almost linearly with frame rate [16]. DoD allows coupling together an active depth sensor and an RGB camera to stream dense depth at the RGB camera frame rate, which may be much higher than the former one. The benefit is twofold. On the one hand, it allows the adaptation of active depth sensor energy consumption to the task specifications, thus meeting the energy constraints of the ToF use case. On the other, it unlocks frame rates higher than the maximum attainable by the active depth sensor itself – this effectively tackles the LiDAR use case, where acquisition is limited to 10

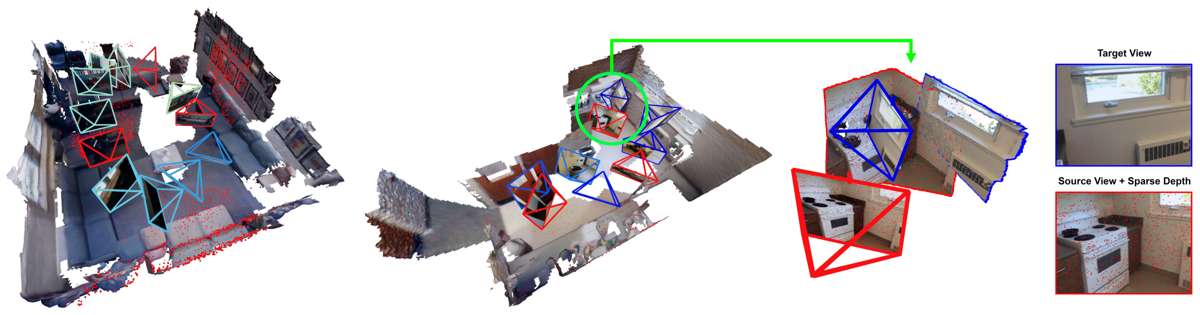


Figure 7.2. **Temporal Depth Stream Densification Setup.** On the left, is an example of DoD applied to an indoor video sequence where only a few frames (red views) are associated with sparse depth data. On the right, is a close-up example of the supposed setup. Using an RGB-D video stream with only a few sparse depth frames requires the integration of monocular, multi-view, and sparse depth cues. DoD smoothly enables the recovery of temporal and spatial depth resolution in such a scenario.

Hz while RGB cameras can easily attain 30 Hz or more. Increasing the depth perception frame rate is of great interest in safety-critical applications such as autonomous driving. However, decoupling frame rates benefits also 3D scene reconstruction as it reduces energy consumption and allows for denser reconstructions. This is showcased in Figure 7.1: by performing depth completion only at a low frame rate (a) several holes appear in the mesh. Integrating information from a higher frame rate RGB camera (b-d) produces denser meshes. A simple solution to achieve the latter would be relying on Multi-View Stereo algorithms without using depth sensor data (b), or performing depth completion by projecting previous sparse depth points (c). Both these approaches introduce several artifacts as in the highlighted boxes. DoD produces denser and more accurate reconstructions (d). To summarize, this chapter's main contributions are as follows:

- Introduction of the temporal depth stream densification task, in which the aim concernings matching the spatial and temporal resolution of a sparse active sensor with one of a higher frame rate RGB camera.
- Design of a deep architecture devoted to this purpose, exploiting sparse depth measurements and RGB data collected at time $t - n$ to obtain a dense depth map aligned with the RGB frame at time t.
- Evaluation of the proposed framework on several datasets featuring RGB-D video streams and comparison with existing approaches compatible with the outlined setting, proving the superiority of DoD.

7.2 Proposed Framework

DoD consists of exploiting the higher framerate of an RGB camera to increase the temporal resolution of an active depth sensor. This is carried out by leveraging multi-view geometry on the RGB video stream and estimating depth for any RGB frame, both those for which measurements from the active sensor are available and those for which are not. To this aim, the minimum amount of information needed to exploit geometry is leveraged – *i.e.* for each RGB view on which computing depth is required (the *target* view) it is retained a previously collected RGB frame (*source* view) and sparse depth points (*source* depth). The supposed setup is illustrated in Figure 7.2. The framework can be conceptually divided into a set of three sequential steps: i) multi-modal encoding, ii) iterative multi-modal integration, and iii) depth decoding.

7.2.1 Multi-Modal Encoding

DoD exploits information from different *modalities* to perceive 3D structures – *i.e.*, multi-view geometry, monocular cues, and sparse depth measurements. To this extent, the framework can be defined as multi-modal. Accordingly, it is important to properly extract useful cues for each of such information sources. Instead of performing early fusion [40], domain-appropriate features are separately computed and a fusion is delegated to a specific module, detailed in Section 7.2.2, to properly integrate them in a common representation. In this section, the encoding of each domain is specified, while Figure 7.3 depicts an overall overview of DoD.

Geometry Encoding. Multi-view geometry cues stem from the capability to perform matching. The first layers of a ResNet18 [53] are employed to design a shared encoder, used to extract features $\mathcal{F}^t$,$\mathcal{F}^s$ at $\frac{1}{8}$ spatial resolution, from target and source views respectively. Such features are exploited to compute correlation scores between pixels of the target view and those of the source view. Given the predicted depth at a specific coordinate of the target view D_{u_t,v_t}, the matching coordinates of the same point in the source view can be obtained as

$$q^s = KPD_{u_t,v_t}K^{-1}q^t \tag{7.1}$$

where K and P are the camera intrinsic parameters and the relative pose between target and source views, while $q^s = [u_s\ v_s\ 1]^T$ and $q^t = [u_t\ v_t\ 1]^T$ are homogeneous point coordinates in the two frames respectively. These latter coordinates allow for sampling features from $\mathcal{F}^s$ and $\mathcal{F}^t$ and compute per-point correlation cues as

$$\mathcal{C} = \frac{1}{\sqrt{F}} \sum_{f=1}^{F} \mathcal{F}^t_{u_t v_t f} \mathcal{F}^s_{u_s v_s f} \tag{7.2}$$

However, single pixel-wise correlation scores are not sufficient to guide the depth update process effectively. According to multi-view geometry, as the depth value of a pixel in the target image changes, its corresponding matching pixel in the source view is supposed to move along the epipolar line, moving away or approaching the epipole. Meaningful multi-view cues are guaranteed only if the correct updating direction for estimated depth can be inferred. Thus, for each $D_{u_t v_t}$ a set of depth hypotheses are sampled relative to the former, moving along the epipolar line. Then, for each of them, a patch of correlation values is computed to increase the distinctiveness of each sampling. Such procedure generates the “Epipolar Correlation Features”

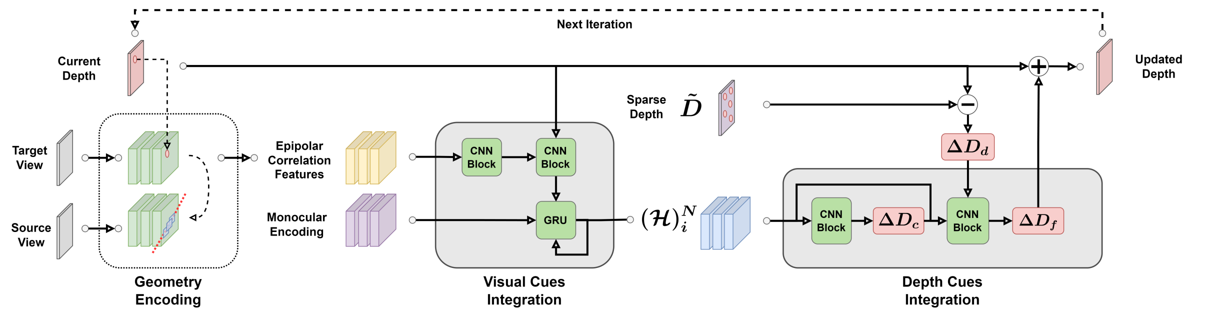


Figure 7.3. **Depth on Demand Framework Overview.** DoD embeds multi-view cues and monocular features in the Visual Cues Integration, then integrates sparse depth updates in the Depth Cues Integration. To properly exploit both these information these stages are applied iteratively in the form of depth updates.

represented in the early stages of Figure 7.3. If not otherwise specified, 41 3×3 patches are linearly sampled within a range of 2 meters, which corresponds to sampling at intervals of 0.1 meters.

Monocular Encoding. Sparse depth information and multi-view correlation data are crucial to deliver accurate 3D reconstructions. Nonetheless, such information fails in the case of moving objects or is not available when large camera pose changes happen, potentially leaving large areas of the field of view empty of information. Thus, it is important to provide a fallback monocular source of information to smoothly complete the not-covered areas. Purposely, a monocular encoder exploiting the first layers of a ResNet34 [53] is introduced to output multi-scale feature maps $\tilde{\mathcal{F}}_2^t$, $\tilde{\mathcal{F}}_4^t$ and $\tilde{\mathcal{F}}_8^t$ at respectively $\frac{1}{2}$, $\frac{1}{4}$, $\frac{1}{8}$ resolution out of the target view alone.

Sparse Depth Encoding. Finally, the availability of sparse depth data obtained from an active sensor captured at a previous time instant is assumed. Such sparse depth points are projected onto the target view by means of pose information, obtaining a coarse depth map $\tilde{D}$ that will be exploited for both initialization and iterative multi-modal fusion. Since projecting at a lower resolution may lead to inaccurate positioning when building the sparse depth map, sub-pixel projection coordinates are also propagated too. Unlike the depth completion task, projected sparse depth is characterized by errors on moving objects and occlusion, making it more difficult to exploit as a source. Their management is delegated to the integration phase, where the exploitation of multiple modalities ameliorates such issues.

7.2.2 Multi-Modal Integration

Once features have been extracted for each input modality, a fusion module is employed to combine such information in the common representation of a target-aligned depth map, that is iteratively refined for a fixed number of steps N. The integration module is depicted in Figure 7.3 and can be logically divided into two sequential components.

Visual Cues Integration. The first stage of the fusing module is in charge of extracting depth-related features by visual cues only. Features extracted in the Monocular Encoding step are integrated with the geometric information. This latter consists of a set of correlation features extracted by sampling over the epipolar lines in a relative range with respect to the current depth estimate, as detailed in Section 7.2.1. All these cues are embedded in the hidden state $(\mathcal{H})_i^N$ of a Gated Recurrent Unit, where $(\cdot)_i^N$ indicates a sequence of tensors across a set of iterations from $i = 0$ to $i = N - 1$. $(\mathcal{H})_{i=0}^N$ is initialized with a deep convolutional module fed with monocular features.

Depth Cues Integration. The second stage of the fusing module takes into account sparse depth data availability. First, a branch predicts a depth update ΔD_c from visual depth-related cues only. Then, a sparse depth update ΔD_d is computed pixel-wise versus the current prediction as

$$\Delta D_d = \begin{cases} \tilde{D}_{i,j} - D_{i,j} & \text{if } \tilde{D}_{i,j} > 0 \\ 0 & \text{otherwise} \end{cases} \tag{7.3}$$

It is worth observing that this latter step generates an update that seeks to refine the current prediction by injecting the exact sparse points, for which zero means either that the depth is correct versus *a priori* information or that the depth measurement is missing for a specific pixel. Finally, the fusion of these updates is carried out by a further branch predicting ΔD_f, which is used to update the current depth prediction. This integration procedure allows for filtering the sparse depth which is likely to contain several outliers due to reprojection – *e.g.*, as in the case of background points being blended with foreground points at occlusions [27]. Moreover, since the sparse depth data is fused in the update space, missing values can be integrated as zero updates, effectively dealing with the varying sparsity problem often affecting depth completion methods [28].

Iterations and Depth Initialization. The previously described multi-modal updating strategy is applied multiple times, generating at each iteration a refined depth map that is then used at the subsequent iteration to improve the multi-view correlation samples and the sparse depth update. Accordingly, an initial depth state is required: the sparse depth data are used to initialize the depth for the first iteration, filling the missing coordinates with the mean value of the valid ones. In case no projected sparse depth points are available in the target view a reasonable depth value of 3 meters is used as initialization.

7.2.3 Depth Decoding

The multi-modal integration module outputs a sequence of incrementally refined depth maps $(D)_i^N$ at $\frac{1}{8}$ resolution. While working at a lower resolution is beneficial in terms of memory and computational time, a method to perform effective upsampling is required. A learned procedure inspired by convex upsampling [139] is exploited, given the depth map at $\frac{1}{8}$ resolution, it's employed a set of three modules $\theta_s(\cdot)\ \ s \in \{2, 4, 8\}$ performing a $2\times$ resolution upsampling composed of two convolutional layers. Each module takes in input the depth map to be upsampled, monocular context information $\tilde{\mathcal{F}}_s\ \ s \in \{2, 4, 8\}$ and a set of features from the previous step, then it outputs $\frac{H}{s} \times \frac{W}{s} \times M$ feature channels and an upsampling mask $\mathcal{W}_s$ of shape

	Method	Views	2D Metrics					3D Metrics					
			MAE↓	RMSE↓	Abs Rel↓	Sq Rel↓	$\sigma < 1.05$↑	Comp↓	Acc↓	Chamfer↓	Prec↑	Recall↑	F-Score↑
MVS	Sayed et al. [120]	8	0.093	0.151	0.047	0.016	0.717	0.062	0.056	0.059	0.702	0.646	0.671
	Wang et al. [150]	8	0.184	0.270	0.102	0.048	0.437	0.106	0.086	0.096	0.511	0.433	0.467
	Gu et al. [48]	8	0.170	0.254	0.091	0.044	0.507	0.086	0.082	0.084	0.545	0.498	0.519
	Cheng et al. [17]	8	0.167	0.252	0.088	0.042	0.512	0.084	0.082	0.083	0.547	0.502	0.522
MVS + Depth	Guided PatchMatch-Net [150]+[105]	8	0.183	0.267	0.102	0.048	0.437	0.106	0.085	0.095	0.512	0.432	0.467
	Guided CAS-MVSNet [48]+[105]	8	0.124	0.203	0.068	0.029	0.635	0.064	0.061	0.062	0.667	0.634	0.649
	Guided UCS-Net [17]+[105]	8	0.133	0.210	0.074	0.030	0.576	0.070	0.065	0.068	0.616	0.578	0.595
	Guided PatchMatch-Net [150]+[105]	2	0.291	0.384	0.160	0.096	0.284	0.135	0.125	0.130	0.406	0.315	0.353
	Guided CAS-MVSNet [48]+[105]	2	0.286	0.388	0.154	0.094	0.304	0.099	0.109	0.104	0.447	0.419	0.431
	Guided UCS-Net [17]+[105]	2	0.258	0.353	0.148	0.093	0.328	0.103	0.099	0.101	0.451	0.385	0.414
Depth	SpAgNet [28]	1	0.069	0.138	0.039	0.016	0.824	0.046	0.037	0.042	0.836	0.789	0.810
	NLSPN [99]	1	0.067	0.137	0.037	0.017	0.847	0.046	0.035	0.041	0.851	0.799	0.822
	CompletionFormer [182]	1	0.075	0.149	0.041	0.019	0.829	0.047	0.037	0.042	0.846	0.795	0.818
	DoD	2	0.041	0.103	0.022	0.008	0.899	0.039	0.025	0.032	0.904	0.845	0.871

(a) **(b)**

Method	3D Metrics					
	Comp↓	Acc↓	Chamfer↓	Prec↑	Recall↑	F-Score↑
DoD – Low Temporal Resolution	0.064	0.014	0.039	0.961	0.778	0.856
DoD – High Temporal Resolution	0.039	0.025	0.032	0.904	0.845	0.871

(c)

Table 7.1. **Results on ScanNetV2.** On top, (a) 2D and (b) 3D performance by DoD and competing approaches. At the bottom, (c) 3D performance by DoD with low/high temporal resolution. The best, second-best and third-best are highlighted.

$\frac{H}{s} \times \frac{W}{s} \times (2 \times 2 \times 9)$. This latter is used to perform a weighted combination over the 3×3 neighborhood of each depth value normalized by a softmax operation and yields a $2\times$ upsampled depth map. Features are upsampled by nearest neighbor interpolation. The first module $\theta_8(\cdot)$ takes in input the last hidden state $(\mathcal{H})_{i=N-1}^{N}$. This approach enables both embedding fine-grain monocular contextual information and enforcing locally smooth and consistent depth propagation. Moreover, it allows for a drastic reduction of the upsampling module weights number with respect to conventional convex upsampling. While optimizing, the upsampling module is applied at each iteration for supervision; however, at deploying time it can be used only once for the final prediction, to maximize efficiency.

7.3 Experimental Results

DoD is evaluated in a wide range of scenarios – *i.e.* indoor video sequences, aerial scenes, automotive environments – to evaluate its accuracy within single domains, as well as its generalization capabilities. Since each setting manifests its own challenges, experiments are classified by scenario and highlight the main difficulties faced on a per-dataset basis.

Training Protocol. For each target frame I_i is associated a buffer of previous N frames $\{(I_j,\ \tilde{D}_j)\ :\ j \in [i-N, i-1]\}$. Then, at each iteration, a frame from such buffer is selected randomly as the source one. This is done to augment as much as possible the number of relative poses observed between the source and target view. Random color jitter and horizontal flips are applied, adjusting the pose accordingly. For each sample, progressively refined depth maps yielded by the multi-modal integration unit are recorded and upsampled to full resolution with the depth decoding approach described in Section 7.2.3. Supervision of such a sequence of depth maps $(D)_i^N$ exploits an exponentially decayed ℓ_1-loss, as described in Equation 7.4 with

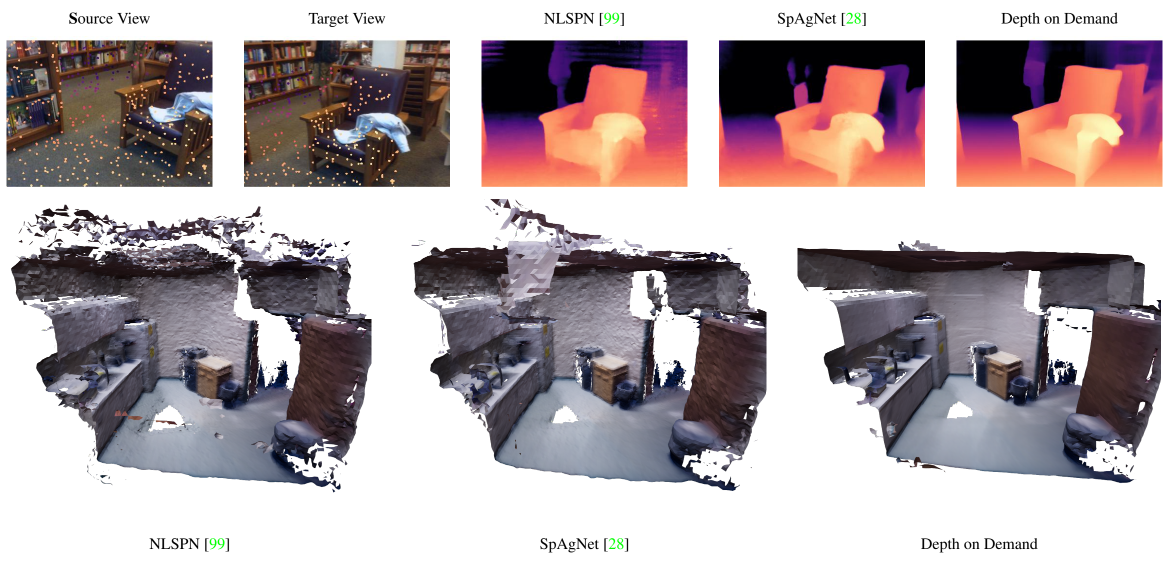


Figure 7.4. **Qualitative results on ScanNetV2.** On top: from left to right the source view with sparse depth points, the target view with projected sparse depth points, and predictions by competitors and DoD. At the bottom: reconstructed meshes by competitors and DoD, respectively at low and high temporal resolution.

decaying factor $\nu = 0.8$.

$$L = \sum_{i=1}^{N} \nu^{N-i} ||(D)_i^N - D_{\text{gt}}||_1 \tag{7.4}$$

Testing Protocol. While testing, an RGB video stream with a higher frame rate than the active depth sensor is assumed. Thus, given a sequence of RGB frames $[I_0, \dots, I_n]$ only a few of them will be associated with a depth frame $\{\tilde{D}_0, \dots, \tilde{D}_m\}$. In each testing video sequence, each RGB view I_i is linked with the immediately preceding RGB image coupled with a depth frame $(I_j,\ \tilde{D}_j)$, $j \leq i$. DoD and competing methods are fed with such data to predict a dense depth map. By modulating the temporal sparsification ratio between the depth and RGB frames $\tau = f_{\text{D}}/f_{\text{RGB}}$ the frequency of sparse depth frames can be controlled; with ratio 1 the task is equivalent to depth completion as depth would be given at every frame.

Competitors. To fairly compare with existing approaches, each one is retrained following the authors' guidelines but applying the previously described training protocol to increase their robustness to the peculiarities of the proposed task since the standard original training protocol for depth completion struggles at dealing with the considered setup. Concerning depth completion, state-of-the-art frameworks [27, 99, 182] are used for comparison projecting sparse depth points from the source frame onto the target view I_i. Concerning Multi-View Stereo methods, [105] stems as a natural competitor since it enables standard MVS frameworks [150, 48, 17] to exploit both sparse depth and multi-view data natively. Also in this latter case, training happens with the aforementioned scheme. Since Multi-View Stereo methods usually exploit a large number of views, training, and evaluation is performed with both 2 and 8 input views.

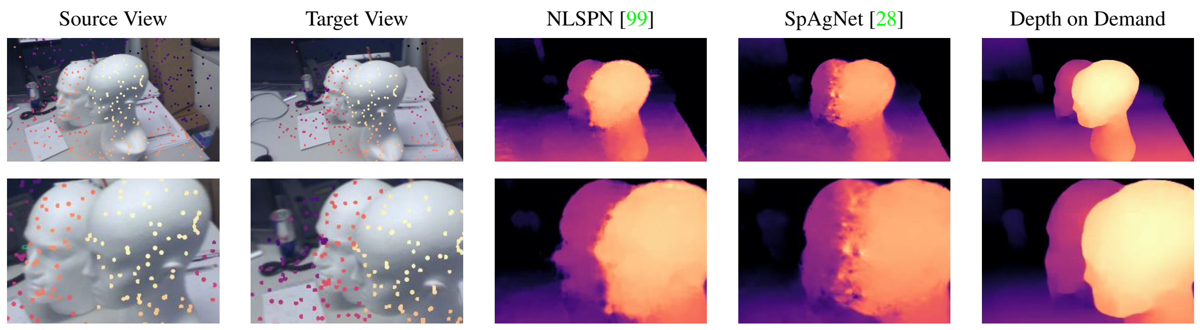


Figure 7.5. **Qualitative results on 7Scenes.** From left to right: source view with sparse depth points, the target view with projected sparse depth points, and predictions by DoD and existing methods.

7.3.1 Indoor Scenario

For indoor applications, ToF sensors are a popular solution to perceive depth, even though they are characterized by relatively low spatial resolution and a short working range. In this setting, the temporal resolution of the ToF sensor is limited as well, in order to reduce power consumption and overheating. The main challenge in indoor environments is the FoV overlap across consecutive views, which may vary from being almost complete – *e.g.*, the camera is not moving – to completely absent. DoD is trained on ScanNetV2 [29] and tested on both this latter and 7Scenes [45], following the protocols described in Section 7.3 by randomly sampling 500 sparse depth points consistently with the depth completion literature [18, 99, 82]. For testing, depth is sparsified over time according to $\tau = 0.2$.

ScanNetV2. ScanNetV2 [29] is an RGB-D video dataset containing more than 1500 scans of indoor environments. Table 7.1 (a) shows the 2D performance of DoD on standard metrics for depth map evaluation. At the top, the performance of RGB-only methods [120, 150, 48, 17] is reported with 8 views in input. Below, is the performance of [105] with either 8 or 2 input views and projected sparse depth. In such methods, integrating sparse depth in the proposed applicative scenario provides a small improvement, nullified by using only 2 views, *i.e.* the target and a single source view. this may be ascribed to their specific design, poor at processing sequential frames. On the contrary, depth completion methods [28, 99, 182] relying only on the target view and projected sparse depth from the source view – showed at the bottom of Table 7.1 – result in being the most competitive solution for temporal depth stream densification among those existing in the literature. Eventually, DoD indisputably outperforms completion models on any metric, thanks to the joint use of monocular, multi-view, and sparse depth cues. This superior accuracy of the predicted depth maps also translates into more accurate, dense 3D reconstructions: Table 7.1 (b) shows how even a few temporally sparse depth measurements largely improve performance versus RGB-only reconstruction carried out by state-of-the-art methods. Again, DoD outperforms existing depth completion solutions by an evident margin. Table 7.1 (c), instead, highlights the effect of increasing the temporal resolution at which depth is estimated. Can be noticed how keeping a low temporal resolution – i.e., the same as the depth sensors – yields slightly accurate reconstructed meshes, while a higher temporal resolution

	Method	Views	2D Metrics				
			MAE↓	RMSE↓	Abs Rel↓	Sq Rel↓	$\sigma < 1.05$↑
MVS	Sayed et al. [120]	8	0.121	0.169	0.068	0.021	0.536
	Wang et al. [150]	8	0.193	0.268	0.112	0.048	0.390
	Gu et al. [48]	8	0.177	0.251	0.101	0.041	0.421
	Cheng et al. [17]	8	0.176	0.250	0.099	0.040	0.428
MVS + Depth	Guided PatchMatch-Net [150]+[105]	8	0.191	0.264	0.112	0.047	0.391
	Guided CAS-MVSNet [48]+[105]	8	0.120	0.192	0.071	0.024	0.587
	Guided UCS-Net [17]+[105]	8	0.141	0.209	0.083	0.028	0.484
	Guided PatchMatch-Net [150]+[105]	2	0.267	0.345	0.158	0.080	0.268
	Guided CAS-MVSNet [48]+[105]	2	0.250	0.338	0.141	0.069	0.303
	Guided UCS-Net [17]+[105]	2	0.228	0.306	0.139	0.063	0.303
Depth	SpAgNet [28]	1	0.068	0.139	0.040	0.014	0.806
	NLSPN [99]	1	0.061	0.134	0.037	0.014	0.842
	CompletionFormer [182]	1	0.067	0.144	0.039	0.015	0.827
	DoD	2	0.043	0.106	0.025	0.008	0.896

Table 7.2. **Results on 7Scenes.** 2D performance by DoD and competing approaches in generalization on 7Scenes. The best, second-best and third-best are highlighted.

	Method	Views	2D Metrics				
			MAE↓	RMSE↓	Abs Rel↓	Sq Rel↓	$\sigma < 1.05$↑
MVS + Depth	Guided PatchMatch-Net [150]+[105]	8	2.353	5.259	0.234	2.285	0.470
	Guided CAS-MVSNet [48]+[105]	8	1.296	3.753	0.126	1.270	0.647
	Guided UCS-Net [17]+[105]	8	1.231	3.624	0.115	1.106	0.675
	Guided PatchMatch-Net [150]+[105]	2	3.629	6.564	0.438	3.669	0.230
	Guided CAS-MVSNet [48]+[105]	2	1.985	4.845	0.185	1.794	0.492
	Guided UCS-Net [17]+[105]	2	1.804	4.513	0.177	1.526	0.486
Depth	SpAgNet [28]	1	0.841	2.273	0.090	0.561	0.718
	NLSPN [99]	1	0.941	2.327	0.113	0.623	0.613
	CompletionFormer [182]	1	0.961	2.411	0.106	0.608	0.625
	DoD	2	0.648	2.230	0.056	0.490	0.832

Table 7.3. **Results on TartanAir.** 2D performance of DoD and competing approaches on TartanAir [153]. The best, second-best and third-best are highlighted.

trades accuracy to increase completeness. Nonetheless, maintaining a high temporal resolution yields better F-Scores overall. Finally, Figure 7.4 shows some qualitative results.

7Scenes. Generalization capabilities of DoD and the existing alternatives are assessed in different indoor environments on the 7Scenes dataset [130], by testing the models trained on ScanNetV2 [29] without any fine-tuning. Results are collected in Table 7.2, where can be noticed a trend consistent with what was observed on ScanNetV2: DoD shows remarkable capabilities concerning generalization in the indoor scenario, staying in the lead of the competing approaches. In Figure 7.5 a comparison of handling erroneous sparse depth points due to occlusion is provided, DoD is able to disregard outliers by exploiting multi-view cues.

7.3.2 Outdoor Scenario

3D reconstruction in outdoor environments poses significantly different challenges compared to indoor – *e.g.*, it features much larger depth ranges and, possibly, scattering of the depth measurements. To study temporal depth completion in this context, two datasets are leveraged: TartanAir [153] and KITTI [44].

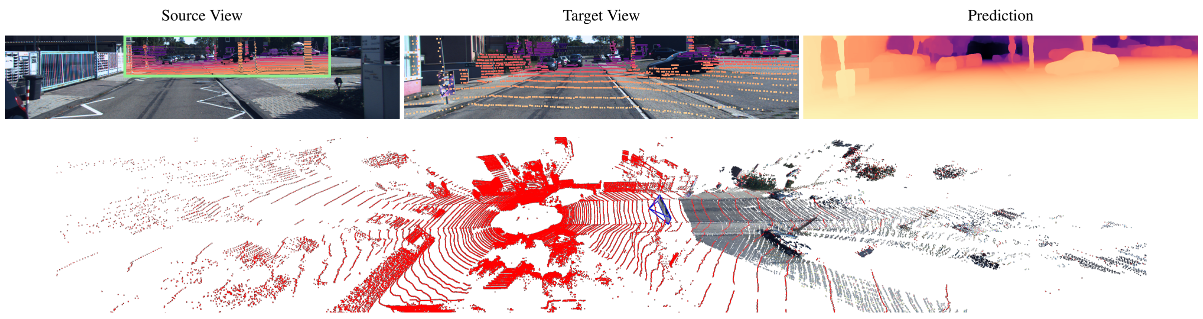


Figure 7.6. **KITTI Setup.** On KITTI, the 360° LiDAR point cloud is projected over the target point of view. If the camera is moving forward – as usually happens – the furthest scan lines are used only, leading to noisy and spaced depth values on the target view. However, the FoV of the target image is usually fully covered.

	Method	Views	2D Metrics – **LiDAR 1Hz**					2D Metrics – **LiDAR 0.5Hz**				
			MAE↓	RMSE↓	Abs Rel↓	Sq Rel↓	$\sigma < 1.05$↑	MAE↓	RMSE↓	Abs Rel↓	Sq Rel↓	$\sigma < 1.05$↑
MVS + Depth	Guided PatchMatch-Net [150]+[105]	8	2.649	5.149	0.216	3.090	0.453	2.809	5.353	0.232	3.330	0.416
	Guided CAS-MVSNet [48]+[105]	8	0.608	2.126	0.034	0.229	0.888	0.873	2.501	0.052	0.350	0.786
	Guided UCS-Net [17]+[105]	8	0.575	1.930	0.034	0.229	0.881	0.828	2.321	0.050	0.303	0.789
	Guided PatchMatch-Net [150]+[105]	2	1.898	4.165	0.117	0.863	0.496	2.282	4.564	0.145	1.154	0.404
	Guided CAS-MVSNet [48]+[105]	2	0.676	2.203	0.035	0.225	0.872	0.916	2.562	0.052	0.328	0.773
	Guided UCS-Net [17]+[105]	2	0.545	1.859	0.030	0.146	0.885	0.837	2.330	0.049	0.277	0.779
Depth	SpAgNet [28]	1	0.532	1.626	0.027	0.095	0.879	0.687	1.865	0.037	0.133	0.808
	NLSPN [99]	1	0.426	1.282	0.023	0.069	0.902	0.614	1.591	0.035	0.121	0.827
	CompletionFormer [182]	1	0.348	1.299	0.019	0.085	0.939	0.555	1.695	0.031	0.150	0.868
	DoD	2	0.347	1.288	0.017	0.061	0.944	0.492	1.544	0.025	0.094	0.890

Table 7.4. **Results on KITTI.** 2D performance by DoD and competing approaches. The best, second-best and third-best are highlighted.

TartanAir. TartanAir [153] is a large synthetic dataset featuring photo-realistic environments with different weather and light conditions. It provides a drone-like point of view in a wide set of scenarios featuring high-frequency details and fast camera motion. Table 7.3 collects the results achieved by existing methods combining multi-view geometry and sparse depth measurements [105] or performing depth completion [99, 182, 28] and DoD. As for the indoor case, completion models largely outperform competitor networks, confirming the limitations of these latter at dealing with the considered problem. Again, DoD shines in accuracy, achieving the lowest errors by a notable margin.

KITTI. The KITTI [44] dataset is a well-known outdoor benchmark with LiDAR data, widely used for visual odometry, monocular depth prediction, and depth completion. For automotive applications, 360° LiDAR sensors are usually employed, providing long-range depth at a frequency limited by the revolution time required by the rotating laser beams. Thus, despite the color cameras can acquire frames at a much higher rate, this is usually constrained to the LiDAR frame rate when performing tasks exploiting both – *e.g.*, depth completion. Nonetheless, when the LiDAR scans are projected from a previous frame over a consequent one as it is done to perform temporal depth completion, the target FoV is almost always fully covered with sparse depth points, yet with higher spatial sparsity. Figure 7.6 shows an example where a LiDAR point cloud collected at a certain time frame is projected over an RGB image collected thereafter, with the camera having moved forward in between the two acquisitions. This causes only

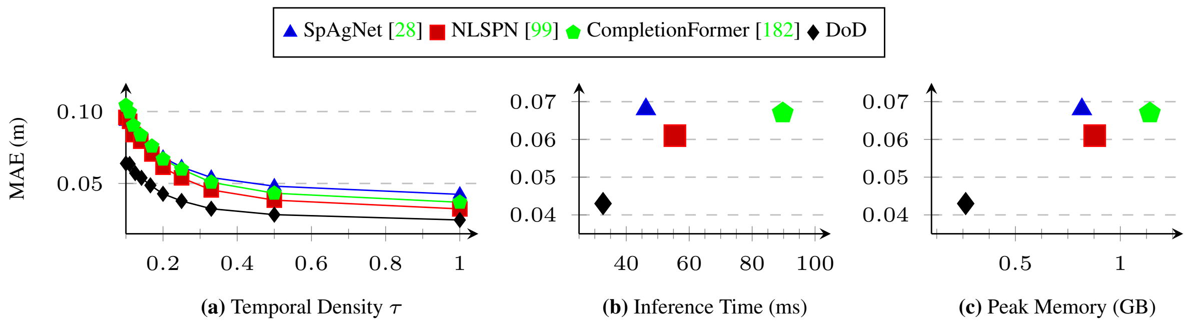


Figure 7.7. **Memory and Time Study.** Time and memory footprint in evaluation on a single RTX 3090 GPU of DoD and competing methods 7Scenes.

the furthest scan lines to be projected over the target view, looking sparser, noisier, and manifesting errors due to occlusions or moving objects. Large areas missing any depth measure may occur in case of occlusion caused by objects in the source view, but still, the FoV is usually fully covered. In this scenario, DoD is at a disadvantage compared to other methods cause i) the reduced multi-view visual overlap on long distances does not provide large benefits and ii) the spatial distribution of the depth measurements is more steady.

Table 7.4 shows the results achieved by existing methods and DoD on this dataset, by simulating different temporal sparsification levels – *i.e.* RGB camera at 10Hz and the LiDAR sensor at respectively 1Hz and 0.5Hz. An off-the-shelf keypoint matcher [83], perspective-n-points [74], and locally-optimized RANSAC [26] are exploited to estimate accurate pose, as already done in the depth completion literature [91]. Despite the more challenging setting, DoD still outperforms any existing alternative.

7.3.3 Temporal Sparsification Study

The sensitivity of DoD to different *temporal* densities – *i.e.* frame rate imbalances between the RGB and depth sensor (that is, $\tau = f_{\mathrm{D}}/f_{\mathrm{RGB}}$) – is deeply studied. In Figure 7.7(a) it is reported the Mean Absolute Error (MAE) on the 7Scenes test split with the testing protocol described in Section 7.3, while varying the temporal sparsification τ from 0.1 – *i.e.* one out of ten frames – to 1. Actually, when $\tau = 1$ the source view always matches with the target view, and sparse points are aligned with it. Thus, $\tau = 1$ is equivalent to the well-studied depth completion case. This may also occur in real use cases where the camera is static.

7.3.4 Memory and Time Analysis

Figures 7.7 (b) and 7.7 (c) report the memory footprint and execution time of the main methods involved in the experiments. The peak memory and computation time the model requires for inference when processing 640×480 inputs are measured. All measurements are based on a single RTX 3090 GPU and 32-bit floating-point precision. DoD approach excels in terms of both.

MVS	Depth	2D Metrics		3D Metrics	
		MAE↓	RMSE↓	Chamfer↓	F-Score↑
✓		0.187	0.257	0.084	0.530
	✓	0.049	0.115	0.033	0.870
✓	✓	0.041	0.103	0.032	0.871

(a) Components Ablation

Module	Single Inf. Time	Calls	Tot. Time
	(ms)	(nr.)	(ms)
Geometry Encoding	2.060 ± 0.182	$1\times$	2.060 ± 0.182
Monocular Encoding	2.175 ± 0.031	$1\times$	2.175 ± 0.031
Correlation Features	0.373 ± 0.001	$10\times$	3.733 ± 0.015
Visual Cues Integration	1.748 ± 0.008	$10\times$	17.480 ± 0.082
Depth Cues Integration	0.274 ± 0.005	$10\times$	2.724 ± 0.053
Depth Decoding	1.444 ± 0.040	$1\times$	1.444 ± 0.040
Total Time			30.196 ± 0.260

(b) Detailed Runtime

Table 7.5. **Ablation studies.** Experiments on ScanNetV2 aimed at highlighting (a) the impact of multi-view stereo and depth cues, and (b) runtime for each component.

7.3.5 Ablation study

This section describes an ablation study to assess the impact of each module composing DoD. Table 7.5 reports results on ScannetV2 concerning two main studies. In (a), is shown how processing multi-view stereo cues and sparse depth impacts the overall accuracy achieved by DoD. Not surprisingly, depth points play a prevalent role, yet alone are insufficient to achieve the best results. In (b), it is reported the detailed runtime required by any single component in DoD.

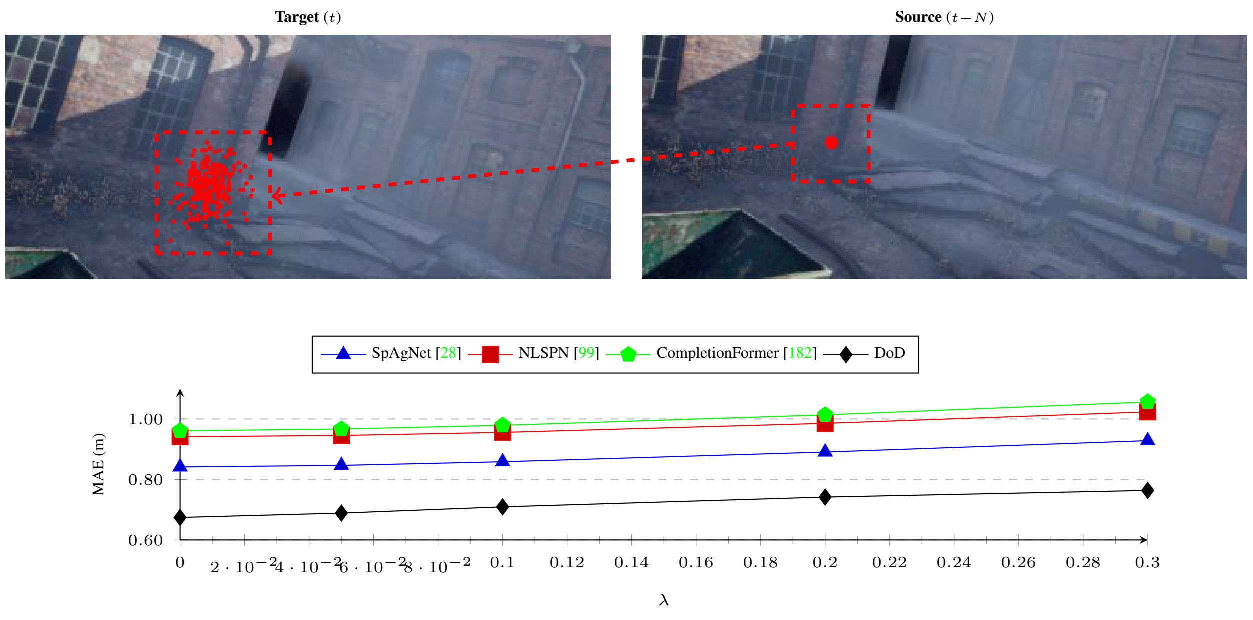


Figure 7.8. **Pose Noise Sensitivity Study.** Impact of noisy pose in DoD and competitor frameworks perturbing pose information with Gaussian noise. At the top, is shown qualitatively how such a noise affects depth point projection. At the bottom, methods performance versus noise intensity on TartanAir [153]. Each model tested is trained with the noise-free pose.

7.3.6 Number of Iterations Study

DoD is characterized by an iterative module for depth refinement and multi-modal integration, in this section is studied the impact of applying a different number of iterations at testing time. During training, 10 iterations are always performed. Figure 7.9 shows the mean absolute error in meters on the test split of 7Scenes [130] performing a different number of iterations. As can be clearly seen the network stabilizes its performance starting from 8 iterations, demonstrating its capability to reach a point of convergence. The number of iterations applied affects the time-accuracy trade-off of the approach. Thus, this latter can be adapted to the deploying requirements modulating such a parameter.

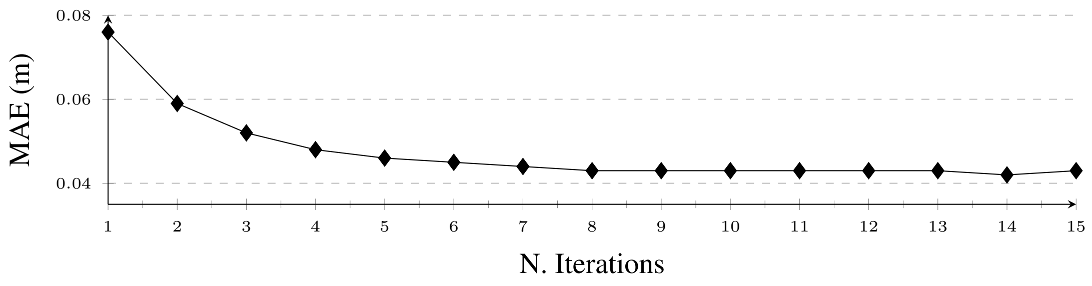


Figure 7.9. **Number of Iterations.** Performance using a different number of iterations on 7Scenes [130]. DoD allows to change the number of iterations to adapt the time-accuracy trade-off required by the deploying environment.

7.3.7 Pose Noise Sensitivity

In this section an additional experiment against test-time pose noise sensitivity is proposed, in the event that the pose estimator – *e.g.*, any visual-inertial odometry or SLAM pipeline – may introduce an error in the pose estimate. Let us represent a pose by a six degrees of freedom vector $\mathbf{q} := (\mathbf{t}, \mathbf{r}) \in \mathbb{R}^6$ with translation $\mathbf{t}$ and rotation $\mathbf{r}$ (in Euler angles). Given each ground truth pose $\mathbf{q}$, random Gaussian noise is drawn on it, yielding the perturbed pose $\hat{\mathbf{q}} \sim \mathcal{N}(\mathbf{q}, \lambda^2 \operatorname{diag}(\mathbf{q})^2)$ where λ is the pose noise factor. $\hat{\mathbf{q}}$ is fed everywhere $\mathbf{q}$ would be used in the pipeline. At the top of Figure 7.8, it is reported a visual example where a known sparse depth point is projected with a set of 100 noisy poses drawn from the aforementioned distribution with pose noise factor $\lambda = 0.3$ for a given $\mathbf{q}$. Such noise not only affects DoD but any other depth completion method as well, in the assumption that the pose is used to reproject the sparse depth points since it corresponds to a significant uncertainty in the depth hints localization. In the DoD case, the impact of pose errors is more subtle; first off, it affects the geometry cues, in that the epipolar correlation block samples along perturbed epipolar lines. Secondly, it affects the reprojected sparse depth points on the target view, as per the completion case. At the bottom of Figure 7.8 quantitative results are reported sweeping $\lambda \in [0, 0.3]$ (where 0 is the noiseless case). Each model in this evaluation is trained with the noise-free pose on TartanAir [153]. As it would seem that DoD could be more affected by pose errors, by this test can be assessed that the gap in performance between DoD and depth completion methods remains fixed w.r.t. λ, *i.e.*, in a fair evaluation pose noise causes a degradation that increases gracefully with λ whilst keeping an almost fixed quality gap between all methods.

7.3.8 Moving Objects

Furthermore, the behavior of the proposed approach on scenes with moving objects is also evaluated. Traditionally, multi-view methods work under the assumption of a static environment, to enable the use of multi-view geometry cues. Nonetheless, moving objects can occur in real use cases. In the automotive scenario, other vehicles move [44]; in the indoor scenario, people or objects can move. In this chapter, we didn't focus on the challenge of dealing with scene motion. Nonetheless, the existence of this issue must be acknowledged and thus a qualitative study of how DoD behaves in moving areas is provided. Figure 7.10 provides an example scene from TartanAir [153] where a robotic arm moves on an assembly line. When sparse depth points are projected from the source view I_{t-N} to the target view I_t sparse depth points gathered on the arm are wrongly projected. Moreover, multi-view cues are not useful in this case – *i.e.* even if the network predicts the correct depth on the target view for a moving object the projection on the source view leads to a wrong position. Thus, the monocular features are the unique useful information to estimate depth in the dashed red box. DoD demonstrates to better exploit such information than NLSPN [99], effectively ignoring misleading multi-view and sparse depth information. Nonetheless, monocular depth estimated from a non-specialized approach is far from being fully accurate, as can be observed in the point cloud at the bottom of Figure 7.10.

7.3.9 Architectural Details

In this section, the core components of DoD are deeply detailed – described in Section 3. Table 7.6 provides implementation details of the main components of the architecture. The visual cues integration module encodes separately the epipolar correlation features and the current depth estimate $(D)_i^N$, then it fuses such data with monocular data $\tilde{\mathcal{F}}_8^t$ using two Gated Recurrent Units with kernel size of 1×5 and 5×1; this latter choice is done as it leads to a lighter model than using a single 5×5 kernel. The depth cues integration module, composed of only four convolutional layers, fuses different depth representations obtained by different sources, as described in Figure 2. The depth decoding module computes multi-scale depth maps. At each iteration a set of upsampling weights and features are computed, then the upsampling is performed using convex upsampling. Finally, the hidden state $(\mathcal{H})_{i=0}^N$ initialization is performed by means of a simple convolutional layer. Our model is trained on ScanNetV2 [29], TartanAir [153], and KITTI [44] with AdamW, 10^{-4} learning rate and 10^{-5} weight decay. 100K training steps are always performed, dividing the learning rate by 10 at 60K and 90K steps. Trainings are performed on 2 RTX 3090 in mixed precision with (total) batch size 8. Moreover, gradients are clipped with global norm 1 to stabilize the behavior of Gated Recurrent Units and the same random seed is enforced in each training to increase reproducibility. On ScanNetV2 [29] the training is performed on the same split defined by [120] with a buffer of 7 source frames to enable consistent comparisons. However, evaluation happens on the whole test video sequences

Visual Cues Integration

Input Tensor	Layer	K	S	In	Out	Output Tensor
$\mathcal{C}$	Conv2D + ReLU	1	1	$3^2 \times 41$	256	corr0
corr0	Conv2D + ReLU	3	1	256	192	corr1
$(D)_i^N$	Conv2D + ReLU	7	1	1	128	depth0
depth0	Conv2D + ReLU	3	1	128	64	depth1
depth1, corr1	Conv2D + ReLU	3	1	192+64	128-1	conv0
conv0, $\tilde{\mathcal{F}}_8^t$, $(\mathcal{H})_i^N$, $(D)_i^N$	ConvGRU2D	(1, 5)	1	128+128+128	128	hidden0
hidden0	ConvGRU2D	(5, 1)	1	128	128	$(\mathcal{H})_{i+1}^N$

Depth Cues Integration

Input Tensor	Layer	K	S	In	Out	Output Tensor
$(\mathcal{H})_{i+1}^N$	Conv2D + ReLU	3	1	128	64	conv0
conv0	Conv2D	3	1	64	1	ΔD_c
$(\mathcal{H})_{i+1}^N$, ΔD_c, ΔD_d	Conv2D + ReLU	3	1	128+1+1	64	conv1
conv1	Conv2D	3	1	64	1	ΔD_f

Depth Decoding

Input Tensor	Layer	K	S	In	Out	Output Tensor
$(\mathcal{H})_{i=N-1}^N$, $\tilde{\mathcal{F}}_8^t$, $(D)_{i=N-1}^N$	Conv2D + ReLU	3	1	$128+128+1$	$3^2 \times 4 + 64$	conv0
conv0	Conv2D	3	1	$3^2 \times 4 + 64$	$3^2 \times 4 + 64$	upweights8,feats8
upweights8, $(D)_{i=N-1}^N$	ConvexUpsample	-	-	$3^2 \times 4 + 1$	1	D^4
$\tilde{\mathcal{F}}_4^t$, D_4, feats8	Conv2D + ReLU	3	1	$64+64+1$	$3^2 \times 4 + 32$	conv1
conv1	Conv2D	3	1	$3^2 \times 4 + 32$	$3^2 \times 4 + 32$	upweights4,feats4
upweights4, D_4	ConvexUpsample	-	-	$3^2 \times 4 + 1$	1	D^2
$\tilde{\mathcal{F}}_2^t$, D_2, feats4	Conv2D + ReLU	3	1	$64+32+1$	$3^2 \times 4$	conv2
conv2	Conv2D	3	1	$3^2 \times 4$	$3^2 \times 4$	upweights4
upweights4, D_2	ConvexUpsample	-	-	$3^2 \times 4$	1	D^1

Hidden State Initialization

Input Tensor	Layer	K	S	In	Out	Output Tensor
$\tilde{\mathcal{F}}_8^t$	Conv2D + Tanh	3	1	128	128	$(\mathcal{H})_{i=0}^N$

Table 7.6. **Architecture Modules Description.** Description of the main components of DoD in terms of layers, input, and output dimensions. Each line represents a layer of a module where "Input Tensor" is the name of the input, "Layer" the type of layer used, "K" the kernel size if the layer is convolutional, "S" the stride if the layer is convolutional, "In" the number of input channels, and "Out" the number of output channels. Finally, "Output Tensor" is the name associated to the output tensor.

subsampled by a factor of ten to mimic a fast-moving camera in an indoor environment; since the camera moves really slow with respect to its high frame rate. 7Scenes [130] is used in the same way. On TartanAir [153] training and testing are performed on the whole video sequences without any frames subsampling using a buffer of 7 source frames while training. Finally, on KITTI [44] trainings are performed with a buffer of 3 source frames sampled with a frequency of 2Hz.

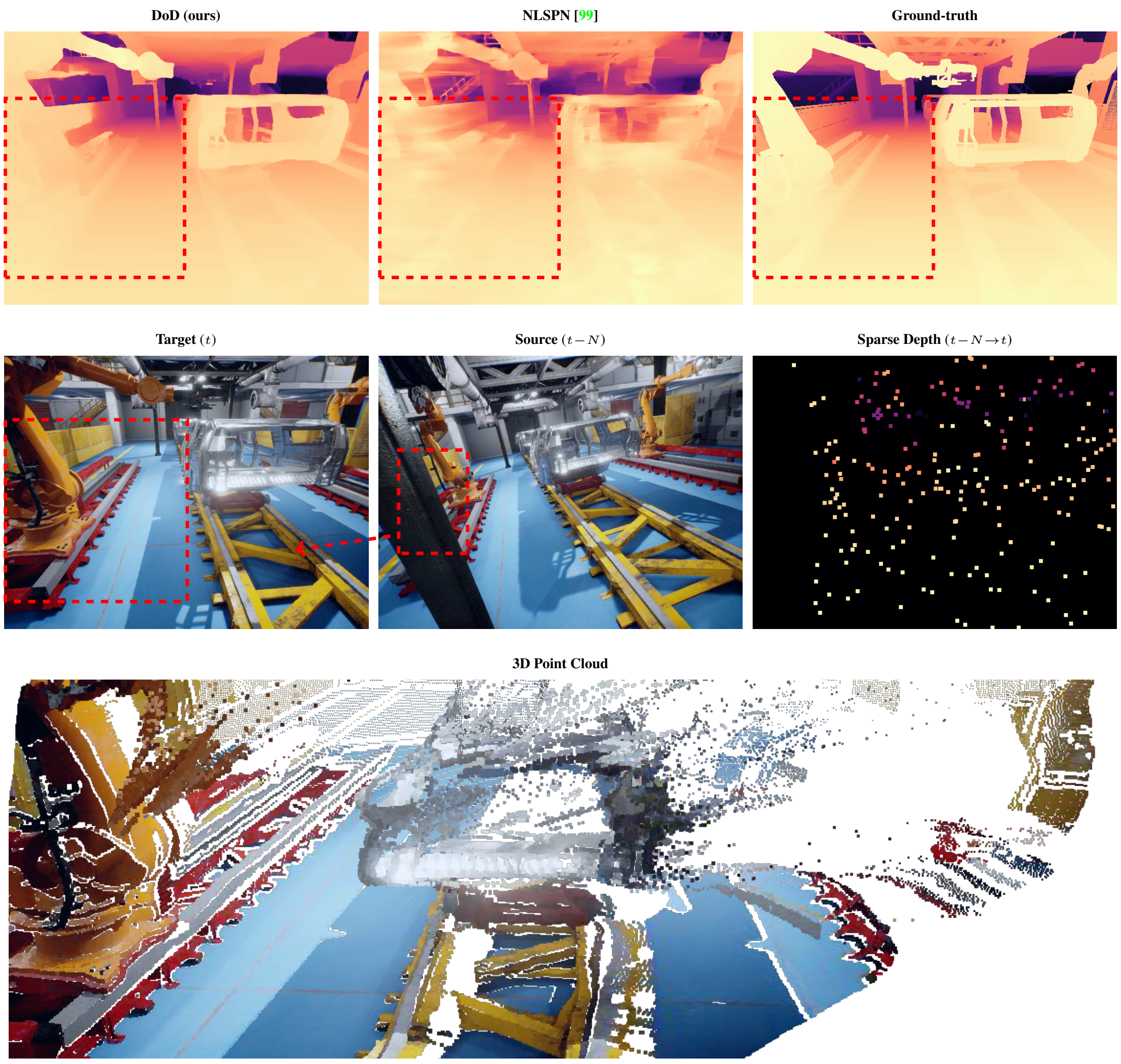


Figure 7.10. **Moving Objects.** Example of DoD behavior on moving objects in a video sequence from TartanAir [153]. On top, ours and [99] depth estimation. Below, are the target, source, and depth points used. In the dashed red bounding box is highlighted a robotic arm moving regardless of the camera. Last, is the 3D projection of the depth estimation. DoD provides consistent monocular depth estimation where multi-view and depth cues are wrong.

Chapter 8

Conclusions

8.1 Summary of Thesis Achievements

In this PhD thesis, the foundational concepts and applications of depth perception have been introduced. Depth sensing is a crucial component in a wide range of downstream tasks, such as robotics, autonomous systems, and augmented reality. Given its broad applicability, various approaches have been developed, each tailored to different operational contexts and input modalities. For example, depth perception can be derived from monocular RGB frames, stereo image pairs, video streams, or active sensing technologies such as Time-of-Flight (ToF) cameras and LiDAR systems.

Despite the diversity of these techniques, many of these technologies are often available concurrently, suggesting the potential to combine and integrate multiple information sources into a unified depth perception framework. This thesis investigates the performance and limitations of different depth-sensing modalities in several key chapters. In Chapter 3, Chapter 6, and Chapter 5, each modality is studied independently, with each analysis contributing novel insights and advancements to its respective sub-field.

Finally, this accumulated expertise is fully harnessed in Chapter 7, where active and multi-modal passive sensors are jointly leveraged. A novel framework is proposed that integrates these different sensing technologies, offering an innovative solution to enhance depth perception accuracy and robustness in complex environments.

8.2 Future Work

While this thesis has laid the groundwork for understanding and advancing depth perception across various sensing modalities, many promising paths may be investigated for further exploration. Future research could focus on the fusion of additional sensing modalities such as event cameras or hyper-spectral imaging, and adaptation to diverse environmental conditions such as low light or scenarios with fast and large moving objects. Finally, there is significant potential to apply the findings of this thesis in emerging fields such as augmented reality where accurate depth perception is crucial for seamless user experiences. Further work could explore the adaptation of the proposed multi-modal frameworks to these domains, considering the unique challenges they present like improving real-time processing and efficiency on embedded platforms. By addressing these areas, future research can continue to push the boundaries of depth perception technologies, ultimately improving their performance and broadening their applicability across both established and emerging fields.

Bibliography

[1] Henrik Aanæs, Rasmus Ramsbøl Jensen, George Vogiatzis, Engin Tola, and Anders Bjorholm Dahl. Large-scale data for multiple-view stereopsis. *International Journal of Computer Vision*, 120(2):153–168, 2016. xi, 21, 27, 28, 29

[2] Shubhra Aich, Jean Marie Uwabeza Vianney, Md Amirul Islam, and Mannat Kaur Bingbing Liu. Bidirectional attention network for monocular depth estimation. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 11746–11752, 2021. 5

[3] Filippo Aleotti, Fabio Tosi, Pierluigi Zama Ramirez, Matteo Poggi, Samuele Salti, Luigi Di Stefano, and Stefano Mattoccia. Neural disparity refinement for arbitrary resolution stereo. In *International Conference on 3D Vision (3DV)*, 2021. 7

[4] Filippo Aleotti, Fabio Tosi, Li Zhang, Matteo Poggi, and Stefano Mattoccia. Reversing the cycle: self-supervised deep stereo through enhanced monocular distillation. In *European Conference on Computer Vision*, pages 614–632. Springer, 2020. 7, 16, 17, 18

[5] Cyrus Bamji, John Godbaz, Minseok Oh, Swati Mehta, Andrew Payne, Sergio Ortiz, Satyadev Nagaraja, Travis Perry, and Barry Thompson. A review of indirect time-of-flight technologies. *IEEE Transactions on Electron Devices*, 69(6):2779–2793, 2022. 62

[6] Connelly Barnes, Eli Shechtman, Adam Finkelstein, and Dan B Goldman. Patchmatch: A randomized correspondence algorithm for structural image editing. *ACM Trans. Graph.*, 28(3):24, 2009. 21, 27, 57

[7] Florent Bartoccioni, Éloi Zablocki, Patrick Pérez, Matthieu Cord, and Karteek Alahari. Lidartouch: Monocular metric depth estimation with a few-beam lidar. *Computer Vision and Image Understanding*, 227:103601, 2023. 62

[8] Ayush Bhandari and Ramesh Raskar. Signal processing for time-of-flight imaging sensors: An introduction to inverse problems in computational 3-d imaging. *IEEE Signal Processing Magazine*, 33(5):45–58, 2016. 62

[9] Yuri Boykov, Olga Veksler, and Ramin Zabih. Fast approximate energy minimization via graph cuts. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(11):1222–1239, 2001. 6

[10] Aljaz Bozic, Pablo Palafox, Justus Thies, Angela Dai, and Matthias Nießner. TransformerFusion: Monocular RGB scene reconstruction using transformers. *NeurIPS*, 2021. 8

[11] Changjiang Cai, Matteo Poggi, Stefano Mattoccia, and Philippos Mordohai. Matching-space stereo networks for cross-domain generalization. In *2020 International Conference on 3D Vision (3DV)*, pages 364–373, 2020. 7

[12] Neill DF Campbell, George Vogiatzis, Carlos Hernández, and Roberto Cipolla. Using multiple hypotheses to improve depth-maps for multi-view stereo. In *European Conference on Computer Vision*, pages 766–779. Springer, 2008. 7, 21, 46

[13] Jia-Ren Chang and Yong-Sheng Chen. Pyramid stereo matching network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018. 7

[14] Ming-Fang Chang, John W Lambert, Patsorn Sangkloy, Jagjeet Singh, Slawomir Bak, Andrew Hartnett, De Wang, Peter Carr, Simon Lucey, Deva Ramanan, and James Hays. Argoverse: 3d tracking and forecasting with rich maps. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 10

[15] Anpei Chen, Zexiang Xu, Fuqiang Zhao, Xiaoshuai Zhang, Fanbo Xiang, Jingyi Yu, and Hao Su. Mvsnerf: Fast generalizable radiance field reconstruction from multi-view stereo. *arXiv preprint arXiv:2103.15595*, 2021. 8

[16] Yan Chen, Jimmy S. J. Ren, Xuanye Cheng, Keyuan Qian, and Jinwei Gu. Very power efficient neural time-of-flight. *2020 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 2246–2255, 2018. 62

[17] Shuo Cheng, Zexiang Xu, Shilin Zhu, Zhuwen Li, Li Erran Li, Ravi Ramamoorthi, and Hao Su. Deep stereo using adaptive thin volume representation with uncertainty awareness. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2524–2534, 2020. xi, 8, 21, 22, 23, 24, 27, 29, 30, 31, 50, 51, 53, 67, 68, 69, 70, 71

[18] Xinjing Cheng, Peng Wang, and Ruigang Yang. Depth estimation via affinity learned with convolutional spatial propagation network. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 103–119, 2018. vii, 6, 34, 35, 37, 38, 39, 40, 41, 42, 69

[19] Xinjing Cheng, Peng Wang, and Ruigang Yang. Learning depth with convolutional spatial propagation network. *IEEE transactions on pattern analysis and machine intelligence*, 2019. 6, 7

[20] Xuelian Cheng, Yiran Zhong, Yuchao Dai, Pan Ji, and Hongdong Li. Noise-aware unsupervised deep lidar-stereo fusion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019. 16, 19, 22

[21] Xuelian Cheng, Yiran Zhong, Mehrtash Harandi, Yuchao Dai, Xiaojun Chang, Tom Drummond, Hongdong Li, and Zongyuan Ge. Hierarchical neural architecture search for deep stereo matching. *CoRR*, abs/2010.13501, 2020. 7

[22] Cheng Chi, Qingjie Wang, Tianyu Hao, Peng Guo, and Xin Yang. Feature-level collaboration: Joint unsupervised learning of optical flow, stereo depth and camera motion. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2463–2473, 2021. 7

[23] Jaesung Choe, Sunghoon Im, Francois Rameau, Minjun Kang, and In So Kweon. VolumeFusion: Deep depth fusion for 3D scene reconstruction. In *ICCV*, 2021. 8

[24] Christopher Choy, Junyoung Gwak, and Silvio Savarese. 4d spatio-temporal convnets: Minkowski convolutional neural networks. 2019. 35, 43

[25] WeiQin Chuah, Ruwan Tennakoon, Reza Hoseinnezhad, Alireza Bab-Hadiashar, and David Suter. Itsa: An information-theoretic approach to automatic shortcut avoidance and domain generalization in stereo matching networks. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 13022–13032, 2022. 7

[26] Ondřej Chum, Jiri Matas, and Josef Kittler. Locally optimized ransac. In *DAGM-Symposium*, 2003. 72

[27] Andrea Conti, Matteo Poggi, Filippo Aleotti, and Stefano Mattoccia. Unsupervised confidence for lidar depth maps and applications. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2022. IROS. 66, 68

[28] Andrea Conti, Matteo Poggi, and Stefano Mattoccia. Sparsity agnostic depth completion. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pages 5871–5880, January 2023. 66, 67, 68, 69, 70, 71, 72, 73

[29] Angela Dai, Angel X. Chang, Manolis Savva, Maciej Halber, Thomas Funkhouser, and Matthias Nießner. Scannet: Richly-annotated 3d reconstructions of indoor scenes. In *Proc. Computer Vision and Pattern Recognition (CVPR), IEEE*, 2017. 69, 70, 75

[30] Jifeng Dai, Haozhi Qi, Yuwen Xiong, Yi Li, Guodong Zhang, Han Hu, and Yichen Wei. Deformable convolutional networks. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, Oct 2017. 38, 49

[31] Kangle Deng, Andrew Liu, Jun-Yan Zhu, and Deva Ramanan. Depth-supervised nerf: Fewer views and faster training for free. *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 12872–12881, 2021. 8

[32] Martin D. Dimitrievski, Peter Veelaert, and Wilfried Philips. Learning morphological operators for depth completion. In *Advanced Concepts for Intelligent Vision Systems Conference*, 2018. 6

[33] Shivam Duggal, Shenlong Wang, Wei-Chiu Ma, Rui Hu, and Raquel Urtasun. Deeppruner: Learning efficient stereo matching via differentiable patchmatch. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 4384–4393, 2019. 7

[34] David Eigen, Christian Puhrsch, and Rob Fergus. Depth map prediction from a single image using a multi-scale deep network. In Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 27. Curran Associates, Inc., 2014. 5

[35] Abdelrahman Eldesokey, Michael Felsberg, Karl Holmquist, and Michael Persson. Uncertainty-aware cnns for depth completion: Uncertainty from beginning to end. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020. 10, 15, 16, 17, 34, 35, 41

[36] Abdelrahman Eldesokey, Michael Felsberg, and Fahad Shahbaz Khan. Propagating confidences through cnns for sparse data regression. In *British Machine Vision Conference*, 2018. 5

[37] Rizhao Fan, Zhigen Li, Matteo Poggi, and S. Mattoccia. A cascade dense connection fusion network for depth completion. In *British Machine Vision Conference*, 2022. 6

[38] Huan Fu, Mingming Gong, Chaohui Wang, Kayhan Batmanghelich, and Dacheng Tao. Deep ordinal regression network for monocular depth estimation. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2002–2011, 2018. 5

[39] Yasutaka Furukawa and Jean Ponce. Accurate, dense, and robust multiview stereopsis. *IEEE transactions on pattern analysis and machine intelligence*, 32(8):1362–1376, 2009. 7, 21, 46

[40] Konrad Gadzicki, Razieh Khamsehashari, and Christoph Zetzsche. Early vs late fusion in multimodal convolutional neural networks. In *2020 IEEE 23rd international conference on information fusion (FUSION)*, pages 1–6. IEEE, 2020. 64

[41] Silvano Galliani, Katrin Lasinger, and Konrad Schindler. Massively parallel multiview stereopsis by surface normal diffusion. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 873–881, 2015. 7, 21, 46

[42] Mathias Gehrig, Willem Aarents, Daniel Gehirg, and Davide Scaramuzza. Dsec: A stereo event camera dataset for driving scenarios. *IEEE Robotics and Automation Letters*, 2021. 10

[43] Andreas Geiger, Philip Lenz, Christoph Stiller, and Raquel Urtasun. Vision meets robotics: The kitti dataset. *International Journal of Robotics Research (IJRR)*, 2013. 10

[44] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 3354–3361. IEEE, 2012. 10, 14, 70, 71, 75, 76

[45] Ben Glocker, Shahram Izadi, Jamie Shotton, and Antonio Criminisi. Real-time rgb-d camera relocalization. In *International Symposium on Mixed and Augmented Reality (ISMAR)*. IEEE, October 2013. 69

[46] Clement Godard, Oisin Mac Aodha, Michael Firman, and Gabriel J. Brostow. Digging into self-supervised monocular depth estimation. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 3827–3837, United States, February 2020. Institute of Electrical and Electronics Engineers (IEEE). International Conference on Computer Vision 2019, ICCV 2019 ; Conference date: 27-10-2019 Through 02-11-2019. 5

[47] Jiaqi Gu, Zhiyu Xiang, Yuwen Ye, and Lingxuan Wang. Denselidar: A real-time pseudo dense depth guided depth completion network. *IEEE Robotics and Automation Letters*, 6:1808–1815, 2021. 6

[48] Xiaodong Gu, Zhiwen Fan, Siyu Zhu, Zuozhuo Dai, Feitong Tan, and Ping Tan. Cascade cost volume for high-resolution multi-view stereo and stereo matching. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2495–2504, 2020. viii, xi, 8, 21, 22, 23, 24, 27, 28, 29, 30, 31, 50, 51, 53, 57, 67, 68, 69, 70, 71

[49] Vitor Guizilini, Rares Ambrus, Wolfram Burgard, and Adrien Gaidon. Sparse auxiliary networks for unified monocular depth prediction and completion. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021. 34, 35, 41, 42

[50] Weiyu Guo, Zhaoshuo Li, Yongkui Yang, Zheng Wang, Russell H Taylor, Mathias Unberath, Alan Yuille, and Yingwei Li. Context-enhanced stereo transformer. In *European Conference on Computer Vision (ECCV)*, 2022. 7

[51] Xiaoyang Guo, Kai Yang, Wukui Yang, Xiaogang Wang, and Hongsheng Li. Group-wise correlation stereo network. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3273–3282, 2019. 7

[52] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 2961–2969, 2017. 19

[53] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016. 36, 64, 65

[54] H. Hirschmuller. Accurate and efficient stereo processing by semi-global matching and mutual information. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, volume 2, pages 807–814 vol. 2, 2005. 6

[55] Heiko Hirschmuller. Stereo processing by semiglobal matching and mutual information. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(2):328–341, 2008. 6, 10, 14, 16, 17, 18

[56] Junjie Hu, Chenyu Bao, Mete Ozay, Chenyou Fan, Qing Gao, Honghai Liu, and Tin Lun Lam. Deep depth completion from extremely sparse data: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 1–20, 2022. 62

[57] Mu Hu, Shuling Wang, Bin Li, Shiyu Ning, Li Fan, and Xiaojin Gong. Penet: Towards precise and efficient image guided depth completion. In *ICRA*, 2021. vii, 6, 18, 19, 37, 41, 42

[58] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4700–4708, 2017. 36

[59] Eddy Ilg, Ozgun Cicek, Silvio Galesso, Aaron Klein, Osama Makansi, Frank Hutter, and Thomas Brox. Uncertainty estimates and multi-hypotheses networks for optical flow. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 652–667, 2018. 15

[60] Saif Muhammad Imran, Yunfei Long, Xiaoming Liu, and Daniel Morris. Depth coefficients for depth completion. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 12438–12447, 2019. 6

[61] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In Francis Bach and David Blei, editors, *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pages 448–456, Lille, France, 07–09 Jul 2015. PMLR. 36

[62] Rasmus Jensen, Anders Dahl, George Vogiatzis, Engil Tola, and Henrik Aanæs. Large scale multi-view stereopsis evaluation. In *2014 IEEE Conference on Computer Vision and Pattern Recognition*, pages 406–413. IEEE, 2014. xi, 21, 22, 46, 47, 51, 53, 55

[63] Xiaowen Jiang, Valerio Cambareri, Gianluca Agresti, Cynthia Ifeyinwa Ugwu, Adriano Simonetto, Fabien Cardinaux, and Pietro Zanuttigh. A low memory footprint quantized neural network for depth completion of very sparse time-of-flight depth maps. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, pages 2687–2696, June 2022. 33, 62

[64] Mohammad Mahdi Johari, Yann Lepoittevin, and Franccois Fleuret. Geonerf: Generalizing nerf with geometry priors. *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 18344–18347, 2021. 8

[65] Kevin Karsch, Ce Liu, and Sing Bing Kang. Depth transfer: Depth extraction from video using non-parametric sampling. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(11):2144–2158, 2014. 5

[66] Alex Kendall and Yarin Gal. What uncertainties do we need in bayesian deep learning for computer vision? In *Advances in neural information processing systems*, pages 5574–5584, 2017. 12, 16

[67] Alex Kendall, Hayk Martirosyan, Saumitro Dasgupta, Peter Henry, Ryan Kennedy, Abraham Bachrach, and Adam Bry. End-to-end learning of geometry and context for deep stereo regression. *CoRR*, abs/1703.04309, 2017. 7, 27

[68] Sameh Khamis, Sean Fanello, Christoph Rhemann, Adarsh Kowdle, Julien Valentin, and Shahram Izadi. Stereonet: Guided hierarchical refinement for real-time edge-aware depth prediction. In *European Conference on Computer Vision (ECCV)*, pages 573–590, 2018. 7

[69] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 39

[70] Arno Knapitsch, Jaesik Park, Qian-Yi Zhou, and Vladlen Koltun. Tanks and temples: Benchmarking large-scale scene reconstruction. *ACM Transactions on Graphics (ToG)*, 36(4):1–13, 2017. 21, 26

[71] Vladimir Kolmogorov and Ramin Zabin. What energy functions can be minimized via graph cuts? *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(2):147–159, 2004. 6

[72] Hsueh-Ying Lai, Yi-Hsuan Tsai, and Wei-Chen Chiu. Bridging stereo matching and optical flow via spatiotemporal correspondence. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1890–1899, 2019. 7

[73] Jin Han Lee, Myung-Kyu Han, Dong Wook Ko, and Il Hong Suh. From big to small: Multi-scale local planar guidance for monocular depth estimation, 2019. 5

[74] Vincent Lepetit, Francesc Moreno-Noguer, and P. Fua. Epnp: An accurate o(n) solution to the pnp problem. *International Journal of Computer Vision*, 81:155–166, 2009. 72

[75] M. Lhuillier and L. Quan. A quasi-dense approach to surface reconstruction from uncalibrated images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(3):418–433, 2005. 7

[76] Ang Li, Zejian Yuan, Yonggen Ling, Wanchao Chi, Shenghao Zhang, and Chong Zhang. A multi-scale guided cascade hourglass network for depth completion. *2020 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 32–40, 2020. 6

[77] Jiankun Li, Peisen Wang, Pengfei Xiong, Tao Cai, Ziwei Yan, Lei Yang, Jiangyu Liu, Haoqiang Fan, and Shuaicheng Liu. Practical stereo matching via cascaded recurrent network with adaptive correlation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 16263–16272, 2022. 7

[78] Zhaoshuo Li, Xingtong Liu, Nathan Drenkow, Andy Ding, Francis X Creighton, Russell H Taylor, and Mathias Unberath. Revisiting stereo depth estimation from a sequence-to-sequence perspective with transformers. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 6197–6206, 2021. 7

[79] Zhaoxin Li, Kuanquan Wang, Wangmeng Zuo, Deyu Meng, and Lei Zhang. Detail-preserving and content-aware variational multi-view stereo reconstruction. *IEEE Transactions on Image Processing*, 25, 05 2015. 46

[80] Chia-Kai Liang, Chao-Chung Cheng, Yen-Chieh Lai, Liang-Gee Chen, and Homer H Chen. Hardware-efficient belief propagation. *IEEE Transactions on Circuits and Systems for Video Technology*, 21(5):525–537, 2011. 6

[81] Zhengfa Liang, Yiliu Feng, Yulan Guo, Hengzhu Liu, Wei Chen, Linbo Qiao, Li Zhou, and Jianfeng Zhang. Learning for disparity estimation through feature constancy. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018. 7

[82] Yuan Qin. Lin, Tao Cheng, Qianglong Zhong, Wending Zhou, and Huanhuan Yang. Dynamic spatial propagation network for depth completion. In *AAAI Conference on Artificial Intelligence*, 2022. 6, 69

[83] Philipp Lindenberger, Paul-Edouard Sarlin, and Marc Pollefeys. LightGlue: Local Feature Matching at Light Speed. In *ICCV*, 2023. 50, 72

[84] Lahav Lipson, Zachary Teed, and Jia Deng. Raft-stereo: Multilevel recurrent field transforms for stereo matching. In *International Conference on 3D Vision (3DV)*, 2021. 7, 53, 54

[85] Biyang Liu, Huimin Yu, and Guodong Qi. Graftnet: Towards domain generalized stereo matching with a broad-spectrum and task-oriented feature. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 13012–13021, 2022. 7

[86] Sifei Liu, Shalini De Mello, Jinwei Gu, Guangyu Zhong, Ming-Hsuan Yang, and Jan Kautz. Learning affinity via spatial propagation networks. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. 37, 38

[87] Livox Technology Company Limited. `https://www.livoxtech.com/mid-70`. 40

[88] Kaiyue Lu, Nick Barnes, Saeed Anwar, and Liang Zheng. From depth what can you see? depth completion via auxiliary image reconstruction. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11303–11312, 2020. 5

[89] Gregor Luetzenburg, Aart Kroon, and Anders A. Bjørk. Evaluation of the Apple iPhone 12 Pro LiDAR for an Application in Geosciences. *Scientific Reports*, 11(1), November 2021. 33, 40, 62

[90] Keyang Luo, Tao Guan, Lili Ju, Haipeng Huang, and Yawei Luo. P-mvsnet: Learning patch-wise matching confidence aggregation for multi-view stereo. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10452–10461, 2019. 21

[91] Fangchang Ma, Guilherme Venturelli Cavalheiro, and Sertac Karaman. Self-supervised sparse-to-dense: Self-supervised depth completion from lidar and monocular camera. *2019 International Conference on Robotics and Automation (ICRA)*, pages 3288–3295, 2018. 10, 18, 62, 72

[92] Fangchang Ma and Sertac Karaman. Sparse-to-dense: Depth prediction from sparse depth samples and a single image. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4796–4803, 2018. 6, 10, 18, 34, 39, 40

[93] Xinjun Ma, Yue Gong, Qirui Wang, Jingwei Huang, Lei Chen, and Fan Yu. Epp-mvsnet: Epipolar-assembling based depth prediction for multi-view stereo. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5732–5740, 2021. 46

[94] Zeyu Ma, Zachary Teed, and Jia Deng. Multiview stereo with cascaded epipolar raft. In *Proceedings of the European conference on computer vision (ECCV)*, 2022. 51, 53

[95] Nikolaus Mayer, Eddy Ilg, Philip Hausser, Philipp Fischer, Daniel Cremers, Alexey Dosovitskiy, and Thomas Brox. A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun 2016. 7

[96] Moritz Menze, Christian Heipke, and Andreas Geiger. Object scene flow. *ISPRS Journal of Photogrammetry and Remote Sensing (JPRS)*, 2018. x, 14, 15

[97] Pushmeet Kohli Nathan Silberman, Derek Hoiem and Rob Fergus. Indoor segmentation and support inference from rgbd images. In *ECCV*, 2012. 39

[98] Jiahao Pang, Wenxiu Sun, Jimmy SJ. Ren, Chengxi Yang, and Qiong Yan. Cascade residual learning: A two-stage convolutional neural network for stereo matching. In *IEEE International Conference on Computer Vision (ICCV)*, Oct 2017. 7

[99] Jinsun Park, Kyungdon Joo, Zhe Hu, Chi-Kuei Liu, and In So Kweon. Non-local spatial propagation network for depth completion. In *European Conference on Computer Vision (ECCV)*, 2020. vii, ix, 6, 33, 34, 35, 37, 38, 39, 40, 41, 42, 67, 68, 69, 70, 71, 72, 73, 75, 77

[100] Kihong Park, Seungryong Kim, and Kwanghoon Sohn. High-precision depth estimation with the 3d lidar and stereo fusion. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2156–2163. IEEE, 2018. 22

[101] Kihong Park, Seungryong Kim, and Kwanghoon Sohn. High-precision depth estimation using uncalibrated lidar and stereo fusion. *Ieee transactions on intelligent transportation systems*, 21(1):321–335, 2019. 22

[102] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. In *NIPS 2017 Workshop on Autodiff*, 2017. 39

[103] Matteo Poggi, Filippo Aleotti, and Stefano Mattoccia. Sensor-guided optical flow. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021. x, 10, 19, 22, 26

[104] Matteo Poggi, Filippo Aleotti, Fabio Tosi, and Stefano Mattoccia. On the uncertainty of self-supervised monocular depth estimation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. 15

[105] Matteo Poggi, Andrea Conti, and Stefano Mattoccia. Multi-view guided multi-view stereo. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2022. IROS. 51, 55, 67, 68, 69, 70, 71

[106] Matteo Poggi, Seungryong Kim, Fabio Tosi, Sunok Kim, Filippo Aleotti, Dongbo Min, Kwanghoon Sohn, and Stefano Mattoccia. On the confidence of stereo matching in a deep-learning era: a quantitative evaluation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2021. 7, 15

[107] Matteo Poggi and Stefano Mattoccia. Learning a general-purpose confidence measure based on o(1) features and a smarter aggregation strategy for semi global matching. In *3DV*, pages 509–518. IEEE, 2016. 12

[108] Matteo Poggi, Davide Pallotti, Fabio Tosi, and Stefano Mattoccia. Guided stereo matching. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019. x, 10, 19, 22, 23, 24, 26, 32

[109] Matteo Poggi, Alessio Tonioni, Fabio Tosi, Stefano Mattoccia, and Luigi Di Stefano. Continual adaptation for deep stereo. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2021. 7, 10

[110] Matteo Poggi and Fabio Tosi. Federated online adaptation for deep stereo. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024. 7

[111] Matteo Poggi, Fabio Tosi, Konstantinos Batsos, Philippos Mordohai, and Stefano Mattoccia. On the synergies between machine learning and binocular stereo for depth estimation from images: a survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 1–1, 2021. 7, 9

[112] Siyuan Qiao, Yukun Zhu, Hartwig Adam, Alan Yuille, and Liang-Chieh Chen. Vip-deeplab: Learning visual perception with depth-aware video panoptic segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3997–4008, June 2021. 5

[113] Xin Qiao, Matteo Poggi, Pengchao Deng, Hao Wei, Chenyang Ge, and Stefano Mattoccia. Rgb guided tof imaging system: A survey of deep learning-based methods. *Int. J. Comput. Vis.*, 2024. 62

[114] Thinal Raj, Fazida Hanim Hashim, Aqilah Baseri Huddin, Mohd Faisal Ibrahim, and Aini Hussain. A survey on lidar scanning mechanisms. *Electronics*, 9(5), 2020. 34

[115] René Ranftl, Katrin Lasinger, David Hafner, Konrad Schindler, and Vladen Koltun. Towards robust monocular depth estimation: Mixing datasets for zero-shot cross-dataset transfer. In *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, Agost 2020. 5

[116] Alexander Rich, Noah Stier, Pradeep Sen, and Tobias Höllerer. 3dvnet: Multi-view depth prediction and volumetric refinement. In *International Conference on 3D Vision (3DV)*, 2021. 8

[117] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *Medical Image Computing and Computer-Assisted Intervention (MICCAI)*, pages 234–241, 2015. 7

[118] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115(3):211–252, 2015. 36

[119] Tonmoy Saikia, Yassine Marrakchi, Arber Zela, Frank Hutter, and Thomas Brox. Autodispnet: Improving disparity estimation with automl. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 1812–1823, 2019. 7

[120] Mohamed Sayed, John Gibson, Jamie Watson, Victor Prisacariu, Michael Firman, and Clément Godard. Simplerecon: 3d reconstruction without 3d convolutions. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2022. 51, 53, 67, 69, 70, 75

[121] Daniel Scharstein and Richard Szeliski. A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. *International journal of computer vision*, 47(1):7–42, 2002. 6, 9

[122] Johannes L Schonberger and Jan-Michael Frahm. Structure-from-motion revisited. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4104–4113, 2016. 9

[123] Johannes L Schönberger, Enliang Zheng, Jan-Michael Frahm, and Marc Pollefeys. Pixelwise view selection for unstructured multi-view stereo. In *European Conference on Computer Vision*, pages 501–518. Springer, 2016. 7, 21, 46

[124] Thomas Schops, Johannes L Schonberger, Silvano Galliani, Torsten Sattler, Konrad Schindler, Marc Pollefeys, and Andreas Geiger. A multi-view stereo benchmark with high-resolution images and multi-camera videos. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3260–3269, 2017. 21

[125] Philipp Schröppel, Jan Bechtold, Artemij Amiranashvili, and Thomas Brox. A benchmark and a baseline for robust multi-view depth estimation. In *Proceedings of the International Conference on 3D Vision (3DV)*, 2022. 46

[126] S.M. Seitz and C.R. Dyer. Photorealistic scene reconstruction by voxel coloring. In *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 1067–1073, 1997. 7

[127] Steven M Seitz, Brian Curless, James Diebel, Daniel Scharstein, and Richard Szeliski. A comparison and evaluation of multi-view stereo reconstruction algorithms. In *2006 IEEE computer society conference on computer vision and pattern recognition (CVPR'06)*, volume 1, pages 519–528. IEEE, 2006. 9

[128] Dmitry Senushkin, Ilia Belikov, and Anton Konushin. Decoder modulation for indoor depth completion. *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2181–2188, 2021. 6

[129] Zhelun Shen, Yuchao Dai, and Zhibo Rao. Cfnet: Cascade and fused cost volume for robust stereo matching. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 13906–13915, June 2021. 7

[130] Jamie Shotton, Ben Glocker, Christopher Zach, Shahram Izadi, Antonio Criminisi, and Andrew William Fitzgibbon. Scene coordinate regression forests for camera relocalization in rgb-d images. *2013 IEEE Conference on Computer Vision and Pattern Recognition*, pages 2930–2937, 2013. ix, 70, 74, 76

[131] Sudipta N. Sinha, Philippos Mordohai, and Marc Pollefeys. Multi-view stereo via graph cuts on the dual of an adaptive tetrahedral mesh. In *2007 IEEE 11th International Conference on Computer Vision*, pages 1–8, 2007. 7, 46

[132] Xiao Song, Xu Zhao, Hanwen Hu, and Liangji Fang. Edgestereo: A context integrated residual pyramid network for stereo matching. In *Asian Conference on Computer Vision (ACCV)*, 2018. 7

[133] Noah Stier, Alexander Rich, Pradeep Sen, and Tobias Höllerer. Vortx: Volumetric 3d reconstruction with transformers for voxelwise view selection and fusion. In *International Conference on 3D Vision (3DV)*, 2021. 8

[134] Jiaming Sun, Yiming Xie, Linghao Chen, Xiaowei Zhou, and Hujun Bao. NeuralRecon: Real-time coherent 3D reconstruction from monocular video. In *CVPR*, 2021. 8

[135] Richard Szeliski, Ramin Zabih, Daniel Scharstein, Olga Veksler, Vladimir Kolmogorov, Aseem Agarwala, Marshall Tappen, and Carsten Rother. A comparative study of energy minimization methods for markov random fields with smoothness-based priors. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(6):1068–1080, 2008. 6

[136] Jie Tang, Fei-Peng Tian, Wei Feng, Jian Li, and Ping Tan. Learning guided convolutional network for depth completion. *IEEE Transactions on Image Processing*, 30:1116–1129, 2020. 6, 24, 29

[137] Tatsunori Taniai, Yasuyuki Matsushita, and Takeshi Naemura. Graph cut based continuous stereo matching using locally shared labels. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1613–1620, 2014. 6

[138] Vladimir Tankovich, Christian Hane, Yinda Zhang, Adarsh Kowdle, Sean Fanello, and Sofien Bouaziz. Hitnet: Hierarchical iterative tile refinement network for real-time stereo matching. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 14362–14372, June 2021. 7

[139] Zachary Teed and Jia Deng. Raft: Recurrent all-pairs field transforms for optical flow. In Andrea Vedaldi, Horst Bischof, Thomas Brox, and Jan-Michael Frahm, editors, *Computer Vision – ECCV 2020*, pages 402–419, Cham, 2020. Springer International Publishing. 7, 50, 66

[140] Federico Tombari, Stefano Mattoccia, Luigi Di Stefano, and Elisa Addimanda. Classification and evaluation of cost aggregation methods for stereo correspondence. In *2008 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8, 2008. 6

[141] Alessio Tonioni, Matteo Poggi, Stefano Mattoccia, and Luigi Di Stefano. Unsupervised adaptation for deep stereo. In *International Conference on Computer Vision (ICCV)*, Oct 2017. 7

[142] Alessio Tonioni, Oscar Rahnama, Tom Joy, Luigi Di Stefano, Ajanthan Thalaiyasingam, and Philip Torr. Learning to adapt for stereo. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019. 7

[143] Alessio Tonioni, Fabio Tosi, Matteo Poggi, Stefano Mattoccia, and Luigi Di Stefano. Real-time self-adaptive deep stereo. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019. 7

[144] Fabio Tosi, Yiyi Liao, Carolin Schmitt, and Andreas Geiger. Smd-nets: Stereo mixture density networks. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021. 47, 51, 53

[145] Fabio Tosi, Alessio Tonioni, Daniele De Gregorio, and Matteo Poggi. Nerf-supervised deep stereo. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 855–866, June 2023. 7

[146] Jonas Uhrig, Nick Schneider, Lukas Schneider, Uwe Franke, Thomas Brox, and Andreas Geiger. Sparsity invariant cnns. In *2017 International Conference on 3D Vision (3DV)*, pages 11–20, 2017. x, 5, 10, 11, 14, 15, 17, 22, 24, 34, 35, 39

[147] Ali Osman Ulusoy, Michael J. Black, and Andreas Geiger. Semantic multi-view stereo: Jointly estimating objects and voxels. In *Proceedings IEEE Conference on Computer Vision and Pattern Recognition (CVPR) 2017*, pages 4531–4540, Piscataway, NJ, USA, July 2017. IEEE. 7, 46

[148] Olga Veksler. Stereo correspondence by dynamic programming on a tree. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 2, pages 384–390, 2005. 6

[149] Velodyne Lidar. `https://velodynelidar.com/products/puck`. 34

[150] Fangjinhua Wang, Silvano Galliani, Christoph Vogel, Pablo Speciale, and Marc Pollefeys. Patchmatchnet: Learned multi-view patchmatch stereo. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14194–14203, 2021. viii, xi, 21, 22, 23, 24, 27, 28, 29, 30, 31, 50, 51, 52, 53, 57, 67, 68, 69, 70, 71

[151] Peng Wang, Xinyu Huang, Xinjing Cheng, Dingfu Zhou, Qichuan Geng, and Ruigang Yang. The apolloscape open dataset for autonomous driving and its application. *IEEE transactions on pattern analysis and machine intelligence*, 2019. 10

[152] Tsun-Hsuan Wang, Hou-Ning Hu, Chieh Hubert Lin, Yi-Hsuan Tsai, Wei-Chen Chiu, and Min Sun. 3d lidar and stereo fusion using stereo matching network with conditional cost volume normalization. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5895–5902. IEEE, 2019. 17, 22

[153] Wenshan Wang, Delong Zhu, Xiangwei Wang, Yaoyu Hu, Yuheng Qiu, Chen Wang, Yafei Hu, Ashish Kapoor, and Sebastian Scherer. Tartanair: A dataset to push the limits of visual slam. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2020. ix, xi, xii, 47, 51, 53, 54, 55, 70, 71, 73, 74, 75, 76, 77

[154] Xianqi Wang, Gangwei Xu, Hao Jia, and Xin Yang. Selective-stereo: Adaptive frequency information selection for stereo matching. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024. 7

[155] Yan Wang, Zihang Lai, Gao Huang, Brian H Wang, Laurens Van Der Maaten, Mark Campbell, and Kilian Q Weinberger. Anytime stereo image depth estimation on mobile devices. In *International Conference on Robotics and Automation (ICRA)*, pages 5893–5900, 2019. 7

[156] Yang Wang, Peng Wang, Zhenheng Yang, Chenxu Luo, Yi Yang, and Wei Xu. Unos: Unified unsupervised optical-flow and stereo-depth estimation by watching videos. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8071–8081, 2019. 7

[157] Jamie Watson, Oisin Mac Aodha, Daniyar Turmukhambetov, Gabriel J. Brostow, and Michael Firman. Learning stereo from single images. In *European Conference on Computer Vision (ECCV)*, 2020. 7

[158] Zizhuang Wei, Qingtian Zhu, Chen Min, Yisong Chen, and Guoping Wang. Aa-rmvsnet: Adaptive aggregation recurrent multi-view stereo network. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6187–6196, 2021. 8, 21

[159] Saining Xie, Ross Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. Aggregated residual transformations for deep neural networks. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5987–5995, 2017. 36

[160] Gangwei Xu, Xianqi Wang, Xiaohuan Ding, and Xin Yang. Iterative geometry encoding volume for stereo matching. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 21919–21928, 2023. 7

[161] Jianfeng Yan, Zizhuang Wei, Hongwei Yi, Mingyu Ding, Runze Zhang, Yisong Chen, Guoping Wang, and Yu-Wing Tai. Dense hybrid recurrent multi-view stereo net with dynamic consistency checking. In *ECCV*, 2020. xi, 8, 22, 23, 24, 27, 29, 30

[162] Zhiqiang Yan, Yuankai Lin, Kun Wang, Yupeng Zheng, Yufei Wang, Zhenyu Zhang, Jun Li, and Jian Yang. Tri-perspective view decomposition for geometry-aware depth completion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4874–4884, 2024. 6

[163] Zhiqiang Yan, Kun Wang, Xiang Li, Zhenyu Zhang, Jun Li, and Jian Yang. Rignet: Repetitive image guided network for depth completion. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2022. 6

[164] Zhiqiang Yan, Kun Wang, Xiang Li, Zhenyu Zhang, Jun Li, and Jian Yang. Desnet: Decomposed scale-consistent network for unsupervised depth completion. In *Proceedings of the AAAI conference on artificial intelligence*, volume 37, pages 3109–3117, 2023. 6

[165] Gengshan Yang, Joshua Manela, Michael Happold, and Deva Ramanan. Hierarchical deep stereo matching on high-resolution images. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5515–5524, 2019. 7

[166] Guorun Yang, Xiao Song, Chaoqin Huang, Zhidong Deng, Jianping Shi, and Bolei Zhou. Drivingstereo: A large-scale dataset for stereo matching in autonomous driving scenarios. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 10

[167] Guorun Yang, Hengshuang Zhao, Jianping Shi, Zhidong Deng, and Jiaya Jia. Segstereo: Exploiting semantic information for disparity estimation. In *European Conference on Computer Vision (ECCV)*, pages 636–651, 2018. 7

[168] Jiayu Yang, Wei Mao, Jose M Alvarez, and Miaomiao Liu. Cost volume pyramid based depth inference for multi-view stereo. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4877–4886, 2020. 21

[169] Qingxiong Yang, Liang Wang, and Narendra Ahuja. A constant-space belief propagation algorithm for stereo matching. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1458–1465, 2010. 6

[170] Qingxiong Yang, Liang Wang, Ruigang Yang, Henrik Stewénius, and David Nistér. Stereo matching with color-weighted correlation, hierarchical belief propagation, and occlusion handling. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(3):492–504, 2008. 6

[171] Yao Yao, Zixin Luo, Shiwei Li, Tian Fang, and Long Quan. Mvsnet: Depth inference for unstructured multi-view stereo. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 767–783, 2018. x, xi, 8, 21, 22, 23, 27, 28, 29, 33, 46, 49, 50, 51, 52, 53

[172] Yao Yao, Zixin Luo, Shiwei Li, Tianwei Shen, Tian Fang, and Long Quan. Recurrent mvsnet for high-resolution multi-view stereo depth inference. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5525–5534, 2019. 8, 21, 50, 51, 53

[173] Yao Yao, Zixin Luo, Shiwei Li, Jingyang Zhang, Yufan Ren, Lei Zhou, Tian Fang, and Long Quan. Blendedmvs: A large-scale dataset for generalized multi-view stereo networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1790–1799, 2020. vii, viii, xi, 22, 26, 27, 29, 45, 47, 51, 52, 54, 55, 56, 57

[174] Zhichao Yin, Trevor Darrell, and Fisher Yu. Hierarchical discrete distribution decomposition for match density estimation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6044–6053, 2019. 7

[175] Ramin Zabih and John Woodfill. Non-parametric local transforms for computing visual correspondence. In *European Conference on Computer Vision (ECCV)*, pages 151–158, 1994. 6

[176] Jure Zbontar, Yann LeCun, et al. Stereo matching by training a convolutional neural network to compare image patches. *J. Mach. Learn. Res.*, 17(1):2287–2318, 2016. 6

[177] Jiaxi Zeng, Chengtang Yao, Lidong Yu, Yuwei Wu, and Yunde Jia. Parameterized cost volume for stereo matching. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 18347–18357, 2023. 7

[178] Feihu Zhang, Victor Prisacariu, Ruigang Yang, and Philip HS Torr. Ga-net: Guided aggregation net for end-to-end stereo matching. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 185–194, 2019. 7

[179] Feihu Zhang, Xiaojuan Qi, Ruigang Yang, Victor Prisacariu, Benjamin Wah, and Philip Torr. Domain-invariant stereo matching networks. In *European Conference on Computer Vision (ECCV)*, 2020. 7

[180] Jiawei Zhang, Xiang Wang, Xiao Bai, Chen Wang, Lei Huang, Yimin Chen, Lin Gu, Jun Zhou, Tatsuya Harada, and Edwin R Hancock. Revisiting domain generalized stereo matching networks from a feature consistency perspective. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 13001–13011, 2022. 7

[181] Jingyang Zhang, Yao Yao, Shiwei Li, Zixin Luo, and Tian Fang. Visibility-aware multi-view stereo network. In *British Machine Vision Conference (BMVC)*, 2020. 46, 51, 53

[182] Youmin Zhang, Xianda Guo, Matteo Poggi, Zheng Zhu, Guan Huang, and Stefano Mattoccia. Completionformer: Depth completion with convolutions and vision transformers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 18527–18536, June 2023. 67, 68, 69, 70, 71, 72, 73

[183] Haoliang Zhao, Huizhou Zhou, Yongjun Zhang, Jie Chen, Yitong Yang, and Yong Zhao. High-frequency stereo matching network. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1327–1336, 2023. 7

[184] Shanshan Zhao, Mingming Gong, Huan Fu, and Dacheng Tao. Adaptive context-aware multi-modal network for depth completion. *IEEE Transactions on Image Processing*, 30:5264–5276, 2020. 6

[185] Yiming Zhao, Lin Bai, Ziming Zhang, and Xinming Huang. A surface geometry model for lidar depth completion. *IEEE Robotics and Automation Letters*, 6(3):4457–4464, 2021. 10, 15, 16, 17, 18, 19, 20, 25, 26

[186] Yiran Zhong, Yuchao Dai, and Hongdong Li. Self-supervised learning for stereo matching with self-improving ability. *arXiv:1709.00930*, 2017. 7