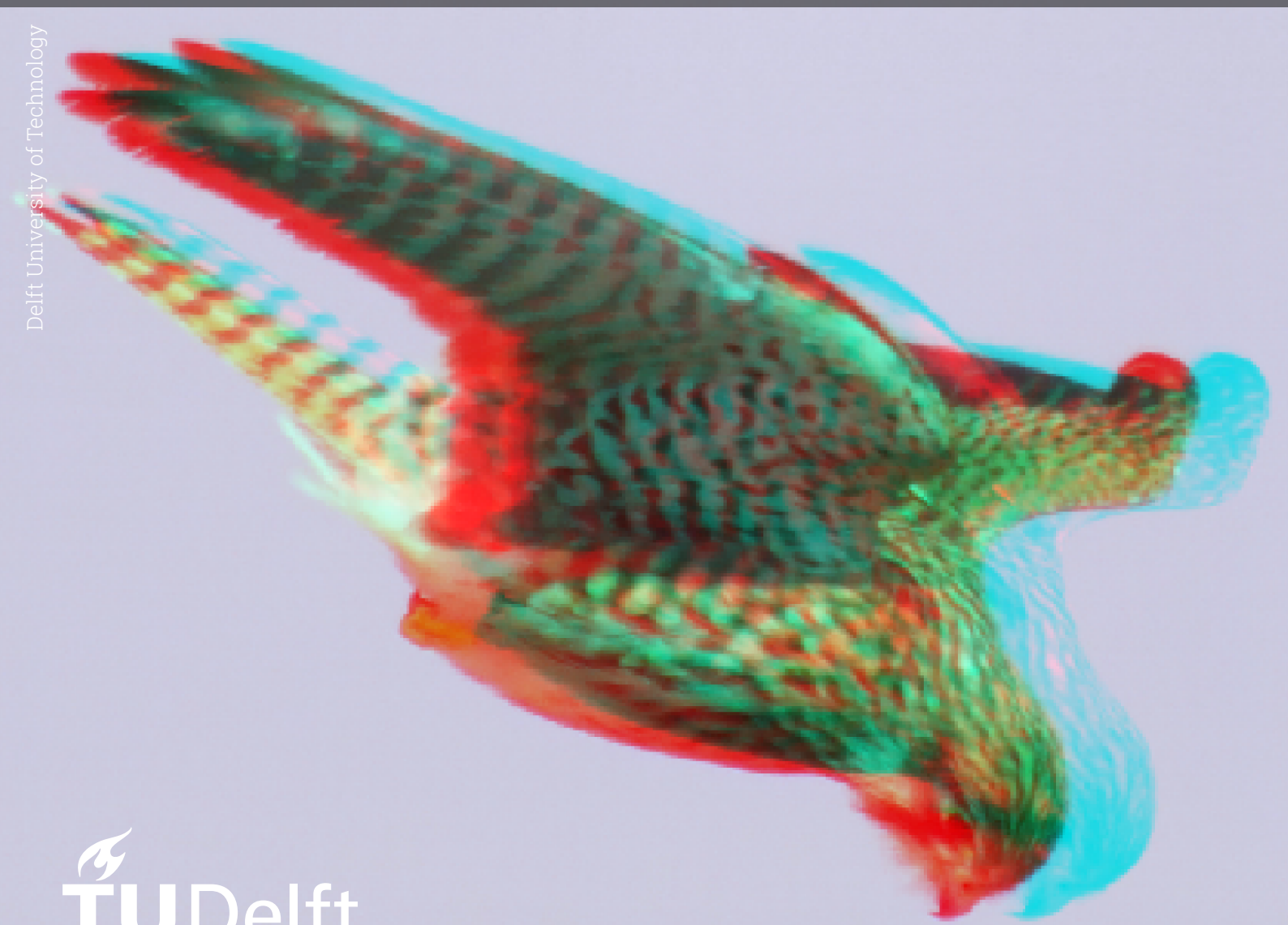
Deep Learning Fusion of Monocular and Stereo Depth Maps Using Convolutional Neural Networks

Dániel Tóth

Delft University of Technology



TU Delft

Deep Learning Fusion of Monocular and Stereo Depth Maps Using Convolutional Neural Networks

by

Dániel Tóth

to obtain the degree of Master of Science
at the Delft University of Technology,
to be defended publicly on Tuesday, June 25, 2024 at 13:00 PM.

Student number:	4460375	
Project duration:	May 1, 2022 – June 25, 2024	
Thesis committee:	Ast. Prof. dr. ir. C. de Wagter,	TU Delft, Chair
	Ast. Prof. dr. ir. N. Eleftheroglou,	TU Delft, Examiner
	Prof. dr. ir. G. C. H. E. de Croon,	TU Delft, supervisor

Cover: Kestrel hovering in sky [anaglyph 3D image] by Alwayswonder under CC BY-SA 4.0 (Modified)
Style: TU Delft Report Style, with modifications by Daan Zwaneveld

An electronic version of this thesis is available at `http://repository.tudelft.nl/`.

TU Delft

Preface

This thesis not only serves as the fulfillment of my graduation requirements, but also represents the culmination of my Control & Operations Master program at TU Delft. It has been a long journey, some would say too long, even, with many challenges, all of them leading to personal growth and academic development. This thesis could not have been completed without the help of my supervisor, Prof. Guido de Croon, and my mentor, Tom van Dijk. I am grateful for their continued support and patience, which was both essential and motivational. Furthermore, I appreciate the efforts of all TU Delft staff members who participated in my education. Finally, I would like to thank my family, friends, and all the people close to me that pushed me when I had to be pushed, and believed in me when I was in doubt.

Dániel Tóth
Delft, June 2024

Introduction

Although only a subject of science fiction a few decades ago, unmanned aerial vehicles, or more commonly drones, have become a widely known commonplace by the 2020s. From toys on the consumer market to sensor platforms in professional applications, they are more present in all aspects of life each year. Originally remotely piloted, recent advances in sensor and computing technology have led these flying robots down the path of autonomy. The proliferation of drones must be accompanied with improved safety measures, via providing the remote pilot with increased situational awareness, or to enable the robot to navigate our world in a safe manner.

The solution is better sensors on the drones and smarter processing of the acquired data. Miniature cameras are a popular payload for small drones. They are light weight, commercially available, and their video feed has a wide range of applications. One of these applications is depth perception, which may be used for obstacle detection and collision avoidance. The goal of this research is to further improve the capability of robots, and more specifically small autonomous aircraft, to safely travel and interact with the world around us. In this thesis article, we investigate the strengths and weaknesses of vision-based depth estimation techniques and propose a novel way to enhance current solutions. Specifically, the following research question will be explored and answered in detail:

Can the fusion of monocular and stereo depth cues through a learning-based approach enhance the performance of its input monocular and stereo networks?

This document is structured as follows: Part I includes the thesis article that outlines our proposed depth fusion solution, details the experiments conducted, and presents our conclusions. In part II, a review of the relevant literature is provided. Chapter 1 highlights the importance of Micro Aerial Vehicle (MAV) technology in today's world. Chapter 2 discusses the challenges related to collision avoidance in MAVs. The development of monocular and stereo depth estimation, along with the current State-of-the-Art, are examined in chapter 3. Finally, chapter 4 deconstructs the research question and summarizes the related research objectives.

Contents

Preface i

Introduction ii

I Research Article 1

II Literature Study 22

1 Micro Aerial Vehicles around us 23

2 Collision avoidance in the air 25

3 Vision-based depth perception 27
3.1 Stereo depth estimation . . . 27
3.2 Model-based stereo matching . . . 28
3.2.1 Convolutional Networks in stereo matching . . . 29
3.3 Monocular depth estimation . . . 30
3.4 Optical flow . . . 35
3.5 Accuracy-runtime Pareto optimal depth estimation . . . 36
3.5.1 Monocular Depth Estimation . . . 36
3.5.2 Stereo Depth Estimation . . . 37
3.6 Fusion . . . 37
3.7 Self-supervised learning for depth estimation . . . 38

4 Research Questions and Objectives 41
4.1 Research Question . . . 41
4.2 Research Objective . . . 41

References 42

Part I

Research Article

Convolutional Neural Network for Stereo and Monocular Depth Estimation

Dániel Tóth, Tom van Dijk and Guido de Croon

***Abstract*—This paper presents an encoder-decoder-style convolutional neural network (CNN) for the purpose of improving monocular and stereo depth estimation (SDE) estimates, by combining them with the corresponding monocular estimates through a fusion network, assisted by prior information to provide context for the fusion. Video cameras are commonly used for depth perception in robotics, especially weight-sensitive applications, such as on Micro Aerial Vehicles (MAV). The two primary paradigms for vision-based depth perception are monocular and stereo depth or disparity estimation, each having their own strengths and weaknesses. These strengths and weaknesses seem to be complementary, and thus a fusion of the two may result in more accurate predictions. In this paper, we investigate this fusion by training a CNN that combines stereo and monocular depth or disparity estimates. The fusion network is agnostic to the choice of the input networks, providing great flexibility. It was found that such a fusion network, while increasing the computational complexity of the depth perception pipeline, indeed improves the accuracy of the estimates. The number of outlier predictions has been significantly decreased, while also limiting some fundamental limitations of both stereo and monocular methods, such as errors arising from occluded regions.**

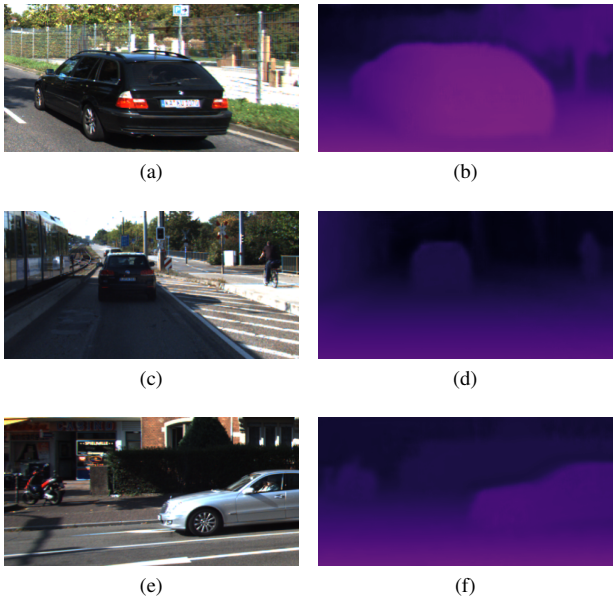


Fig. 1. Scenes from KITTI2015 evaluation dataset, and the corresponding disparity estimates of the triple encoder - triple decoder fusion network with left-view image and confidence map assistance.

I. Introduction

Depth perception is a crucial ability in all autonomous vehicles for navigation and collision avoidance. It has further uses, such as 3D reconstruction and mapping. LIDAR has been a popular sensor for depth perception, but its weight, price, and power requirement limit its practicality on smaller robots, especially micro aerial vehicles (MAV)[1]. Cameras, on the other hand, seem like attractive alternatives due to their light weight, low cost, and the possibility of denser depth maps (100% density in modern, learning-based solutions)[1]. However, retrieving accurate depth information from color images is a non-trivial task. The two primary methods for generating depth maps from still images are monocular and stereo depth estimators.

Depth may also be estimated from motion cues, such as optical flow (i.e. the disparity between two images with a temporal offset)[2]. However, such methods suffer from numerous key limitations that render them impractical for use in autonomous vehicles. Most importantly, optical flow-based depth perception relies on good knowledge (accurate and frequent measurements) of the vehicle velocity vector at any given time, which is not a given due to inaccuracies and biases of the accelerometer[3], [4]. Moreover, optical flow-based methods are not instantaneous; they rely on both a temporal offset and translational motion of the camera. Furthermore, the flow converges to zero in the direction of ego-motion, leading to larger errors in the area of highest interest[5]. For these reasons, optical flow is not given any further consideration in this paper.

The goal of this research is to show the feasibility of learning-based fusion of monocular and stereo depth maps to improve their accuracy. Although fusion has been widely applied to the problem of depth completion, in the field of depth estimation[6], only rule-based fusion has been investigated [7], [8]. To the best of our knowledge, this paper is the first to investigate deep learning based fusion of monocular and stereo depth cues, in order to estimate depth. Due to the proof-of-concept nature of this research, the pursuit of the highest-quality results was not the priority. Thus, the proposed algorithm often features design elements motivated by ease of implementation, rather than performance.

Stereo depth estimation (SDE) attempts to match pixels between the images of a stereo pair to calculate the corresponding disparities. The disparities can then be easily converted to depth values with information on the focal lengths and the baseline between the two cameras that produced the image pair. When the pixels are matched correctly, stereo methods are likely to produce high-quality depth estimates. However, matching is a difficult task in areas with low texture or

repeating patterns [9]. Additionally, it is an outright impossible task at areas of occlusions, i.e. parts of the scene that are only visible to one of the cameras, but not both. Such regions are referred to as "occluded" regions.

Monocular depth estimation (MDE) is the mathematically ill-defined problem of retrieving depth from a single still image, relying on monocular visual cues. The difficulties arise from the fact that an infinite number of 3D scenes can produce the same 2D projection, leaving monocular estimators prone to error[10].

As pixel matching is not part of the underlying process, estimates from monocular methods are not fundamentally hampered in textureless regions or areas featuring repetitive patterns. At the same time, stereo methods have no problems retrieving accurate depth values when correct pixel matching can be achieved. Thus, the two approaches may be able to make up for each other's shortcomings when combined adequately. Despite these potential synergies, stereo-monocular depth fusion has not been studied extensively. The works of Martins et al.[7] and Facil et al.[8] achieved good results in the combination of monocular and stereo depth cues, their fusion strategies were based on hand-crafted rules. As the main contribution, this research shows that even state-of-the-art SDE predictions can be improved when they are properly fused with MDE predictions. Some examples of output depth maps are presented in Figure 1. Our fusion network (see Figure 2 for the high-level design), featuring an encoder-decoder CNN architecture with skip connections (a.k.a. U-net), was assisted with additional contextual information, namely the gray-scale left-view image, and reconstruction error-based confidence maps corresponding to the SDE and MDE input estimates. As part of the ablation study, multiple versions of the algorithm (combinations of different depths and inputs) were evaluated, and their results relative to each other were analyzed, in order to provide information about the effect and impact of each individual design feature.

II. Related work

A. Stereo depth estimation

Finding the disparity at distinct points, such as edges and corners, is quite straightforward, and non-learning-based computational methods were developed before the popularization of deep learning techniques. Such model-based algorithms, like Semi-Global Matching (SGM) [11], are simple to implement, do not need training (and consequently can be deployed in any environment) and require low computing power. Substantial improvement in terms of precision was achieved with the adoption of deep convolutional networks, DispNetCorr [12] being one of the pioneering works. It used a fairly large network, reminiscent of the encoder-decoder architecture, where the encoder and decoder layers are connected with skip connections. Such an architecture is referred to as U-net and has become a proven method in dense prediction tasks. GCNet of Kendall et al.[13] makes use of a much more efficient pipeline, building on traditional stereo matching algorithms: first calculate the matching costs between pixels and then optimize for the lowest matching cost. In GCNet, the stereo pair is fed into a feature extractor. Each feature extracted from one image is concatenated with the corresponding feature of the other image, forming a 4D tensor, called cost volume. The matching cost is then computed for each feature vector. Kendall et al. claim that the use of such cost volumes leads to better performance compared to direct distance or correlation computation between the features. Additionally, the matching of extracted features leads to better results compared to matching the image pixels directly. For regularization, aimed at refining the disparity estimates by learning the context, GCNet makes use of a 3D encoder-decoder network. This pipeline has been commonly used by more modern SDEs, including the state-of-the-art LEASterero [14]. CRD-Fusion [15] is a self-supervised deep stereo matching network, which uses an initial disparity map from a traditional stereo matching algorithm for guidance. A confidence map is generated from the initial disparity to identify incorrect pixels. This confidence is calculated from the intensity differences between the right-view image and the shifted left-view image. Using a stereo matching algorithm reminiscent of GCNet, a cost volume is created from the extracted features of the stereo pair, which is then aggregated through a series of 3D convolutional layers, leading to the fused disparity estimate. The final fused disparity is computed as the weighted average of the learning-based and model-based disparities, with the corresponding confidence values serving as the weights. Although learning-based stereo algorithms have improved a lot compared to previous traditional methods, the fundamental limitations of stereo matching still remain. Li et al. [16] identified the three primary challenges of learning-based stereo matching, such as dealing with imperfect rectification; accurately estimating the disparity in areas with repeating or no texture; and accurately estimating the disparity in fine details, such as nets.

It is clear that even the state-of-the-art stereo depth algorithms struggle with areas of low texture, or partially occluded regions, as accurate matches are very difficult or outright impossible to make. The proposed fusion network aims to improve the stereo estimates on these aspects.

B. Monocular depth estimation

Although there are a multitude of ingenious model-based solutions to retrieve some sort of depth information from a single still image, such as convolving with hand-crafted feature filters [17] or sky segmentation [18], they are limited in practice. The true breakthrough in monocular depth estimation came with the proliferation of convolutional neural networks. The two-scale network of Eigen et al. [19] fuse their two convolution stacks by refining a crude, but global, initial estimate with a finer one. The pioneering works of Garg et al. [10] and Laina et al. [20] adapted the encoder-decoder architecture of autoencoder CNNs for the task of depth estimation from a single still image. Select layers of the encoder and decoder are connected via skip connections, leading to U-net-style networks. Encoders

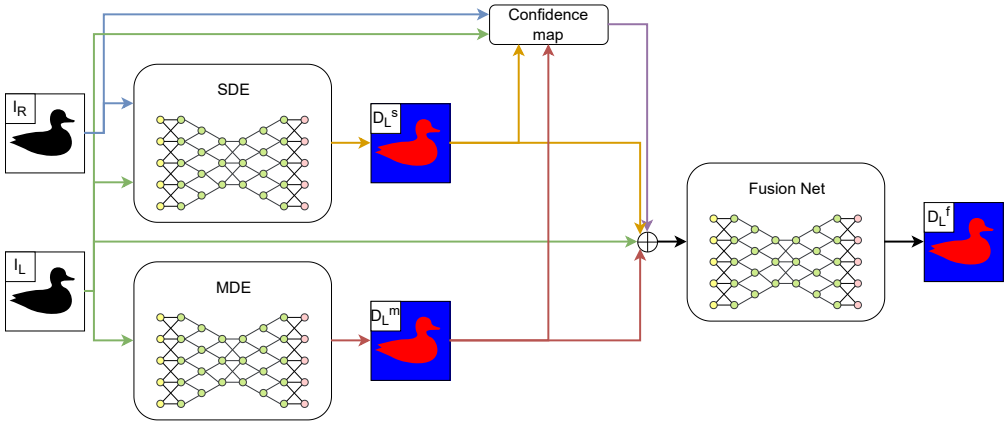


Fig. 2. Outline of the proposed fusion network: The stereo image pair is used to generate a pair of preliminary disparity map estimates via monocular and stereo networks. These are then concatenated along with the reconstruction error-based confidence maps and the left-view image, which are fused into the refined disparity map through the fusion network.

incrementally increase the receptive field of the convolutional operator by reducing the spatial resolution of the feature space via downsampling (spatial pooling). The original spatial resolution is then incrementally retrieved through the decoder layers via deconvolutional or up-convolutional operations. In this way, the network can work with long-range (spatial) information, such as perspective or relative size [21]. The primary limitation of such architectures is the loss of detail in the deep encoder layers. To mitigate this effect, skip connections are commonly utilized, fusing early layers back into the decoder [10], [20], [22], [23]. An alternative way to address the loss of granularity at the deep encoder levels is the use of larger convolutional kernels [24]. Although such large kernels greatly increase model size and computational complexity, this can be somewhat mitigated through dilated convolution, that is, convolution with a sparse kernel [25].

Although more recent research involving visual transformers (such as dense prediction transformers[26]) promises a solution to the problem of loss of detail of pure CNNs, accurate prediction of the global scale remains a fundamental challenge of monocular methods. The proposed fusion network aims to fix this shortcoming by introducing information about the global scale from stereo disparity estimates, which infer this information from the baseline distance between the cameras.

C. Depth prediction via fusion

Just as in the case of stereo and monocular methods, the transition from model-based approaches to learning-based approaches may lead to substantial improvements in the fusion of SDE and MDE depth or disparity maps. Despite the promising synergy between monocular and stereo methods, their fusion has received relatively little scientific attention from the scientific community.

Facil et al. [8] fuses a semi-dense depth map of a traditional stereo matching algorithm with a dense depth map of a monocular CNN. Each pixel in the output depth map is the weighted interpolation of the depths over the available stereo estimates. The weight for each monocular depth pixel attempts to capture the likelihood of it belonging to the same local structure as any of the stereo estimates. It is made up of the relative distance and similarity in gradients between nearby monocular depth estimates. The justification is that a monocular estimate near a stereo one, both of them having similar depth gradients, is likely to be part of the same local structure, and thus their depth values should be similar. Martins et al. [7] proposed a rule-based fusion, where the expected precision of dense stereo and monocular depth estimates in given regions of the input scene is determined through a set of hand-crafted functions. These functions attempt to capture global scale, contrast, occlusion, and texture. These are then used as weights to average the monocular and stereo depth maps. In addition to fusion, stereo estimates are also used to compute the loss while training their monocular estimator in a self-supervised manner.

Depth completion is the task of creating a dense depth map from a single still image and a corresponding sparse but reliable input depth map (such as a LIDAR point cloud). As this task possesses similarities with MDE-SDE fusion (especially when additional guidance, such as color image, is used), it is worth mentioning in this context. One of the simplest approaches, termed early fusion by Jaritz et al. [6] is the method of concatenating the sparse map and the RGB image before feeding it into a learning network. Mat et al. [27] did just that, with the deep network following an encoder-decoder architecture. Late fusion is the process of applying feature extractors with independent weights to each of the two inputs to translate them into a joint representation and to upscale the fused feature map into the desired dense depth map, as was done by Jaritz et al. [6]. As the extracted feature spaces share modalities, they are simply summed up elementwise before being fed into the decoder. Similar multi-branch architectures are preferred to deal with the differing modalities between inputs. The feature extractor branches were expanded to complete encoder-decoder networks in PENet[28], each processing one of the two modalities (color and depth).

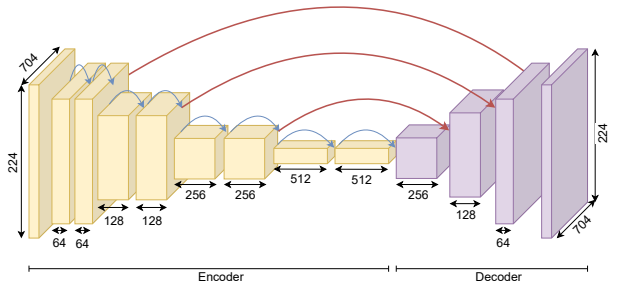


Fig. 3. Outline of the basic version of the fusion network, with feature depths (channel dimension) indicated. The blue arrows represent the shortcuts between residual blocks, and the red arrows represent the skip connections between the encoder and the decoder. Note that two separate confidence maps are generated, corresponding to the SDE and MDE inputs.

III. Methodology

The high-level framework of the fusion augmented depth estimator is shown in Figure 2. The SDE and MDE are pre-trained networks with either scaled depth or raw disparity maps as outputs. These outputs are converted appropriately to match the label or the desired output type (i.e., depth or disparity). For this research, the stereo and monocular estimators were chosen as LEAStereo [14] and BTS [29], respectively. They both have publicly accessible PyTorch implementations and are also performing well on the KITTI benchmark. Both models are pre-trained on KITTI2015 stereo [30], and the much larger KITTI mono depth[31] respectively. Although multiple BTS models have been published, the one that uses DenseNet161 was chosen to be used in this investigation, as it showed the best performance during their ablation study. Note that the proposed algorithm may be used with any other depth or disparity estimator, as they only serve to produce the input estimates. The input models are pre-trained, and their parameters are not modified any further during the training process of the fusion network.

The fusion network takes the SDE and MDE preliminary estimates, and either or both the grayscale left-view image (shortened as LV) and confidence maps (refered to as SSIM) corresponding to the stereo and monocular estimates. SSIM stands for structural similarity index measurement and will be further explained in subsection III-A. Unless specified otherwise, all presented results and examples are produced via the deepest network (3x encoder, 3x decoder), with both additional inputs, as this set-up showed the best performance during experiments. To help the fusion algorithm successfully identify regions to be improved on both input estimates, further contextual information can be provided. To fulfill this purpose, we chose the left-view image from the input stereo pair. The grayscale image is used instead of the original RGB image to keep the number of input channels low. It is assumed that the primary information in an image for depth/disparity estimation is the intensity and texture, both of which are preserved in the grayscale version. In fact, the use of grayscale input instead of RGB has been shown to lead to only a minor accuracy degradation [32]. This additional input will be referred to as LV for left view. The confidence map is yet another optional pair of inputs: These are intended to provide a rough estimate of the correctness corresponding to the monocular and stereo depth maps. The details of how these are obtained are given in subsection III-A. This additional input will be referred to as SSIM, for structural similarity index measure. All fusion inputs (SDE and MDE estimates, the corresponding confidence maps, and the left-view image) are all concatenated along the channel dimension before being passed to the initial layer of the fusion algorithm. Note that these inputs have different modalities: depth for the SDE and MDE estimates, confidence scaled from 0 to 1, and the 8 bit grayscale image. Concatenation has been successfully used in depth completion tasks for multi-modal fusion before, such as in Ma et al. [33].

Although there are more sophisticated methods for multimodal fusion, such as a two-branch method [28], [34], the simple concatenation, agnostic to the modality of the inputs, allows easy implementation of any kind of additional information as input channels, improving the modularity of the proposed algorithm. Additionally, multi-branch methods have significantly more learning parameters, leading to much larger models.

Fig. 4. Double-mapping artifact from image reconstruction, and the masked out areas after validity check

A. Confidence Map

Supplying confidence maps to the fusion algorithm is assumed to help the fusion by providing insight into which areas of the input disparity maps might be prone to error. Such a confidence map is presented in Figure 5. The intended behavior of the fusion is to be less prone to make large corrections at areas of high confidence. Additionally, the algorithm may better fuse areas where one source has considerably higher confidence than the other. The confidence maps are based on the reconstruction error introduced by Garg et al. [10]. Reconstruction error is a convenient confidence metric, as its computation does not rely on true depth. The initial disparity map d_L is used to create a synthetic left-view image I'_L by shifting the pixels of the right-view image I_R along the horizontal, x axis. This transformation is shown in Equation 1.

$$I'_L(x, y) = I_R(x + d_L(x, y), y) \tag{1}$$

Note that all depth and disparity estimates in this work correspond to the left-view image. The structural similarity

map is then calculated between the synthetic and true left-view images via Equation 2 [35], where x and y are the two images, and μ, σ^2, and σ_{xy} are the mean, variance, and cross-correlation of pixel values. These are evaluated by sliding a 3x3 window over the two images, with a stride and zero-padding of 1.

$$\text{SSIM}(x, y) = \frac{(2\mu_x\mu_y + c_1)(2\sigma_{xy} + c_2)}{(\mu_x^2 + \mu_y^2 + c_1)(\sigma_x^2 + \sigma_y^2 + c2)} \quad (2)$$

Reconstruction of an image from its stereo pair and the corresponding disparity map is prone to an artifact called "texture-copy" by Godard et al.[36]. Examples of such artifacts are shown in Figure 4. In reconstruction, texture-copy occurs at depth discontinuities, such as the edge of a nearby object against distant backgrounds. In such cases, regions of the background are visible for one of the cameras but are occluded for the other. So, the nearby object will be mapped once to its correct location (high disparity area), and once more to the occluded area (low disparity area). If two pixels on the synthetic left-view image originate from the same location on the right-view image, the one with the largest disparity is invalid [37]. An incorrectly mapped pixel can be identified by the algorithm below.

```
minShift ← 4
maxShift ← 192
invalid ← False
for n ← minShift, maxShift do
    shifted ← shift(disparity, n)
    locDif ← (shifted − disparity − n)
    if absoluteValue(locDif) < 0.5 then
        invalid ← True
    end if
end for
```

In this algorithm, the shift function shifts the given array along the horizontal axis by a number of pixels. The minimum disparity taken into account is 4 to prevent unnecessary noise filtering in the input disparity map. The maximum number of pixels for shifting is set to 192, which is the maximum disparity that can be predicted by the chosen SDE and MDE models. The confidence map values for the pixels identified as invalid are set to zero. Note that the presence of the for loop means that this algorithm is not parallelized, significantly impacting the runtime performance of the fusion, further discussed in subsection IV-C.

B. Fusion Network Architecture

The proposed algorithm falls into the category of early fusion (according to the definition of Jaritz et al. [6]), where the multi-modal inputs are concatenated along the channel dimension, reminiscent of the depth completion network of Ma et al. [27]. Our fusion algorithm employs a deep convolutional neural net, which follows the U-net-style architecture, inspired by Garg et al. [10] and Laina et al. [20]. as shown in Figure 3. Based on this shallow base architecture, a number of alternate, deeper networks were also implemented and tested. These

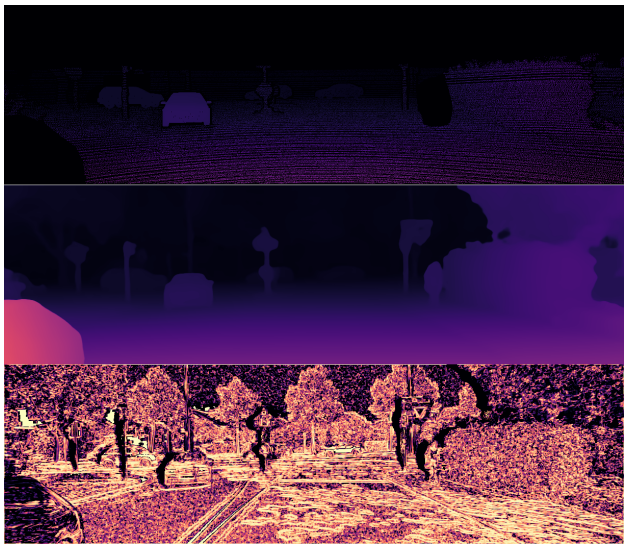


Fig. 5. Example for input depth prediction and the corresponding confidence maps. From top to bottom: label, stereo estimate, and stereo estimate confidence map. Brighter pixels on the confidence map represent higher confidence values. Note the black patches around some objects, the result of reconstructed pixel invalidation.

follow the same design, only with increased size, making them considerably deeper. The network is designed to be trained in a supervised manner, with smooth L1 loss [38].

The role of the encoder is to serve as a feature extractor. By incrementally downsizing the feature maps through increased stride, the receptive field of the convolution operator is effectively increased. This allows the network to aggregate to more global (i.e., long spatial range) relationships. Similarly to the depth completion network of Ma et al. [27], the encoder is based on ResNet18 [39], with the final fully connected layers removed. In alternate ResNet networks (e.g. ResNet34), where the encoder size is increased, the number of residual blocks is simply doubled or tripled, but the distribution of layers are kept the same. Thus, while the doubled encoder has the same number of residual blocks as ResNet34, they are not completely equivalent.

In order to obtain the desired full scale depth or disparity map, the encoder's deep feature space is passed through a series of upconvolutional layers, the decoder. The upscaling follows the same increments as the downscaling in the encoder, allowing for skip connections between the two. The finer details of the encoder input, while still present in the early layers, are then lost during repeated downsampling operations. This leads to lower-quality predictions. This loss of spatial resolution can be mitigated through the use of skip connections, fusing the encoder layers with the corresponding decoder layers [22]. In the proposed fusion network, the skip connections are concatenated with the intermediate outputs of the decoder, just before the upsampling operation. The concatenated feature space is passed through a size 1 convolution, halving the number of channels, in order to maintain the appropriate feature dimensions. The use of transposed convolution results in what are known as checkerboard artifacts in the output [40]. Therefore, instead of using transposed convolution, deconvolution

Layer	Input	1x en - 1x de	3x en - 3x de
en0	MDE, SDE, add. in.	conv(k=7, s=2, p=3) MaxPool2d(k=3, s=2, p=1) BatchNorm2D() ReLU()	
en1	en0	2x ResBlock(ds=False)	6x ResBlock(ds=False)
en2	en1	ResBlock(ds=True) ResBlock(ds=False)	ResBlock(ds=True) 5x ResBlock(ds=False)
en3	en2	ResBlock(ds=True) ResBlock(ds=False)	ResBlock(ds=True) 5x ResBlock(ds=False)
en4	en3	ResBlock(ds=True) ResBlock(ds=False)	ResBlock(ds=True) 5x ResBlock(ds=False)
de1	en4	interp(upscale=2) conv(k=3, s=2, p=1) ELU() BatchNorm2D()	interp(upscale=2) conv(k=3, s=2, p=1) ELU() BatchNorm2D() conv(k=3, s=1, p=1) 2x ELU() BatchNorm2D()
skip1	de1, en3	concat(de1, en3) conv(k=3, s=1, p=1) ELU()	
de2	skip1	interp(upscale=2) conv(k=3, s=2, p=1) ELU() BatchNorm2D()	interp(upscale=2) conv(k=3, s=2, p=1) ELU() BatchNorm2D() conv(k=3, s=1, p=1) 2x ELU() BatchNorm2D()
skip2	de2, en2	concat(de2, en2) conv(k=3, s=1, p=1) ELU()	
de3	skip2	interp(upscale=2) conv(k=3, s=2, p=1) ELU() BatchNorm2D()	interp(upscale=2) conv(k=3, s=2, p=1) ELU() BatchNorm2D() conv(k=3, s=1, p=1) 2x ELU() BatchNorm2D()
skip3	de3, en1	concat(de3, en1) conv(k=3, s=1, p=1) ELU()	
de4	skip3	interp(upscale=2) conv(k=3, s=2, p=1) ELU() conv(k=3, s=1, p=1) Sigmoid()	interp(upscale=2) conv(k=3, s=2, p=1) ELU() conv(k=3, s=1, p=1) 2x ELU() BatchNorm2D() conv(k=3, s=1, p=1) Sigmoid()

Fig. 6. The basic (1x en - 1x de) and best performing (3x en - 3x de) architectures

is performed through separate upsampling (nearest-neighbor interpolation) followed by a size 3 convolution. Decoder layers making use of transposed convolution will be referred to as deconvolution, while the ones with upscaling followed by separate convolution will be referred to as upconvolution. Instead of ReLU, the decoder makes use of the exponential linear units (ELU) [41] as activation functions, which was found to be performing better in the decoder of a pixel-to-pixel predicting encoder-decoder network [42]. The only exception is the final layer, where a sigmoid function is used, in order to obtain strictly positive depth or disparity predictions. After scaling, these correspond to the minimum and maximum estimates. While the encoder's convolutions incorporate biases as per the ResNet architecture, the decoder omits them, as the presence of batch normalization layers means that biases would be ignored anyways [43]. The doubled and tripled versions of the decoder include one or two extra convolutional layers after each upconvolutions, correspondingly. The simple and the triple-depth architectures are shown in Figure 6, and all considered architectures are presented in section A in detail.

IV. Experimental Results

The fusion algorithm described above was implemented in the PyTorch machine learning framework, trained, and evaluated on the KITTI2015 stereo disparity estimation dataset [30]. KITTI2015 contains 200 color image pairs with corresponding sparse ground truth disparity (captured with and converted from LIDAR sensors) for the learning process. The ground truth disparity maps are sparse and only include labels for the bottom two-thirds of the scene. The dataset is split into a training batch of 180 images and an evaluation batch of 20 images. The original data set includes sequential images; these are, however, not included in either the training or the evaluation set. Although there are 200 additional test images, these are used for benchmarking on the KITTI leaderboard, and their labels are not publicly accessible. The images have a resolution of 1242x375, which is then cropped down to 1216x352, according to the KITTI benchmark specifications. The ground truth disparity maps do not have valid labels in the top one-third band of each scene. When this upper area is included in the training images, the resulting model learns to produce completely invalid estimates for that section, as can be seen in Figure 7. Although this artifact does not influence the metrics during evaluation, as the predictions are only evaluated where valid labels are available, it is still considered undesirable. In order to deal with it, the training images are cropped to 1216x224 pixel resolution, removing the top one-third of the scene. These are further cropped to a resolution of 704x224, where the center of the cropping is chosen randomly for each case. This serves both as data augmentation and reduces the computing load in training. The setup of the stereo camera rig used for these images is well documented, allowing the conversion between disparity and depth. The results presented here are from the evaluation batch, after training the fusion network on the training batch for 100 epochs (note that the input networks are pre-trained, and their parameters are not modified any further during the training process of the fusion network). The hardware used for training and evaluation is a NVIDIA T4 and 12.7GB RAM via the cloud computing service of Google Colab.

A. Quantitative analysis

The primary results are listed in Table I. With the ground truth and prediction for pixel i being y_i and $\hat{y_i}$, and n number of pixels in a scene with valid label, the metrics shown on that table are defined as follows [19]:

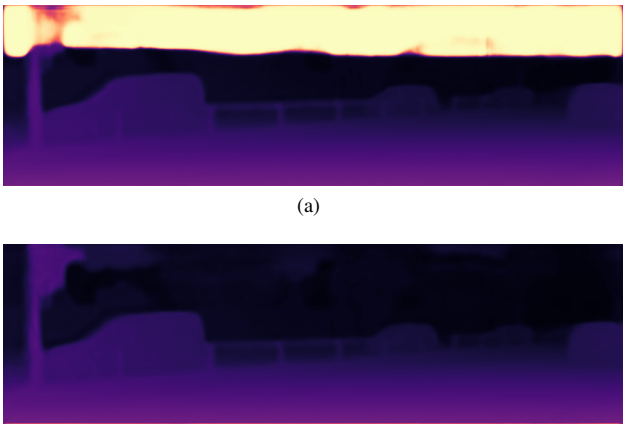
- absolute relative error (abs. rel.): $\frac{\sum_i^n (|y_i - \hat{y}_i|/y_i)}{n}$
- square relative error (sq. rel.): $\frac{\sum_i^n (y_i - \hat{y}_i)^2/y_i)}{n}$
- root mean square error (RMSE): $\sqrt{\frac{\sum_i^n (y_i - \hat{y}_i)^2}{n}}$
- end point error (EPE): $\frac{\sum_i^n y_i - \hat{y}_i}{n}$

Network	Fusion depth	Additional data	Lower is better					Higher is better		
			Abs. Rel.	Sq. Rel.	RMSE	EPE	3px err.	δ_1	δ_2	δ_3
Fusion	1x en-decoder	-	0.044	0.313	2.083	1.094	0.067	0.959	0.983	0.992
	1x en-decoder	LV	0.043	0.198	1.975	1.014	0.062	0.968	**0.994**	**0.998**
	1x en-decoder	SSIM	0.044	0.303	2.041	1.067	0.064	0.963	0.988	0.996
	1x en-decoder	SSIM+LV	0.044	0.233	2.083	1.097	0.067	0.961	0.992	**0.998**
Fusion	2x en-decoder	SSIM+LV	0.040	0.199	**1.904**	0.931	0.056	0.971	**0.994**	**0.998**
Fusion	3x en-decoder	SSIM+LV	0.040	**0.194**	1.917	**0.928**	0.055	0.974	0.993	**0.998**
MDE (BTS)	N/A	N/A	0.235	9.929	7.891	5.812	0.362	0.846	0.926	0.950
SDE (LEAStereo)	N/A	N/A	**0.038**	0.226	2.399	0.948	**0.036**	**0.976**	0.988	0.992
Martins et al. [2018]*	N/A	N/A	0.20	3.00	5.47	-	-	0.85	0.96	0.98

TABLE I

EXPERIMENTAL RESULTS OF SOME SELECT ARCHITECTURES, AVERAGED OVER THE 20 SCENE EVALUATION SET FROM KITTI2015. THE BEST RESULTS ARE BOLDFACED, AND THE SECOND-BEST ONES ARE IN RED.

* AS REPORTED



(a)

(b)

Fig. 7. Fused disparity estimate before ((a)) and after ((b)) removing the top band with no valid labels from the training data. Although fixing the non-sensical top band has no impact on the performance metrics, as these regions are not evaluated due to the lack of ground truth, it does improve the visual appeal of the fused disparity maps.

- three pixel error (3px err.): $1 - \frac{\text{count}(|y_i - \hat{y}_i| < 3 \ \vee \ |y_i - \hat{y}_i| < 0.05 * y_i)}{n}$
- threshold $\delta < 1.25$ (δ_1): $\frac{\text{count}\left(\left(\max(\frac{y_i}{\hat{y}_i}, \frac{\hat{y}_i}{y_i})\right) < 1.25\right)}{n}$
- threshold $\delta < 1.25^2$ (δ_2): $\frac{\text{count}\left(\left(\max(\frac{y_i}{\hat{y}_i}, \frac{\hat{y}_i}{y_i})\right) < 1.25^2\right)}{n}$
- threshold $\delta < 1.25^3$ (δ_3): $\frac{\text{count}\left(\left(\max(\frac{y_i}{\hat{y}_i}, \frac{\hat{y}_i}{y_i})\right) < 1.25^3\right)}{n}$

The MDE and SDE in Table I are pre-trained models, which were evaluated the same way as the fusion. However, the metrics for Martins et al. [7] are taken directly from their published article. Their monocular estimator was trained through self-supervised learning and, for the stereo estimator, a traditional stereo matching algorithm was chosen.

Between the SDE and MDE input networks, the former is clearly better. This is not surprising, as stereo matching is a considerably easier task than monocular estimation, due to the mathematically constrained nature of the problem. Note that in contrast to LEAStereo SDE, the BTS MDE was trained on the KITTI depth prediction/completion dataset [31], a much more extensive array of depth-annotated images. As this and KITTI2015 share the same environment and scenery, it was assumed that the model can generalize between the two dataset without issues. However, both LEAStereo and BTS have not performed as well compared to their reported metrics.

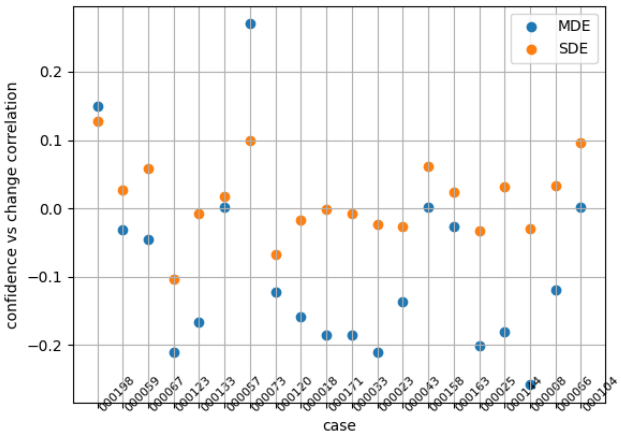
Examining the metrics in Table I, it is evident that the fusion improved the stereo estimates according to several metrics. Considering that fusion exhibits better performance in outlier ratios of a wider threshold (δ_2, δ_3) and in RMSE (which is more sensitive to outliers due to the square term), the fusion seems to reduce the number of predicted pixels that are far off from their ground truth values. However, the fusion lost some accuracy when it came to pixels that SDE predicted with high precision. This can be deduced from the reduced performance in the outlier measures with tighter thresholds (three-pixel-error and δ_1). The assumption that this particular method of fusion leads to fewer outliers but also fewer high-accuracy pixels is further reinforced by the fact that it achieves a better score in square relative error but worse score in absolute relative error. Like with RMSE, the former is more sensitive to outliers than the latter. Between the absolute relative error and the end-point error, the former is less penalizing for the same disparity error with a corresponding high disparity ground truth (i.e. nearby objects). As the fusion network presents a lower EPE but higher absolute relative error compared to the input SDE, the fused disparity maps are more accurate at sections close but slightly worse at larger distances.

Note that due to the small size of the evaluation set, the above-mentioned results are highly skewed by a single data point, the predicted depth map corresponding to the dark tunnel scene (shown in the last row of Figure 10). One would expect that any vision-based depth estimation would perform poorly in such low-visibility conditions. It is nearly impossible to see and distinguish objects that are used to provide a frame of reference, thus it is not surprising that the MDE performs so poorly. The SDE on the other hand, while greatly suffering from a lack of discernible pixels to match, is able to produce a fairly accurate depth map. An examination of the data revealed that such dark scenes are not at all represented in the training set, and thus the fusion was unable to learn to properly deal with similar situations. Due to the low number of images in the validation set, even a single outlier can substantially skew the results. In fact, when the fusion network is evaluated on a modified validation set, where the dark tunnel is not included, its detrimental effect on the averaged metrics becomes clear. As can be seen in Table II, both the MDE and the Fusion networks present increased accuracy, with the latter improving the input SDE with respect to all metrics considered, except for the three-pixel-error. Thus, while the SDE maintains a slight

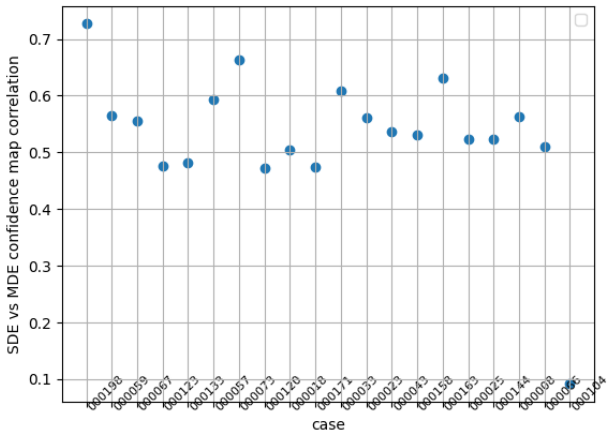
Network	Fusion depth	Additional data	Lower is better					Higher is better		
			Abs. Rel.	Sq. Rel.	RMSE	EPE	3px err.	δ_1	δ_2	δ_3
Fusion	3x en-decoder	SSIM+LV	**0.036**	**0.165**	**1.772**	**0.822**	0.038	**0.978**	**0.995**	**0.998**
MDE (BTS)	N/A	N/A	0.132	1.252	4.684	2.966	0.328	0.885	0.964	0.984
SDE (LEAStereo)	N/A	N/A	0.038	0.231	2.416	0.920	**0.031**	0.975	0.987	0.992

TABLE II
AVERAGED RESULTS ON THE VALIDATION SET WITH THE DARK TUNNEL SCENE REMOVED.

edge in predicting high accuracy pixels, the fusion network is competitive in that regard as well, as long as it is used in a well-lit environment. Note that while the SDE and the fusion networks were trained on the same training-evaluation split, the MDE was trained on the much more extensive KITTI monocular estimation dataset.



(a)



(b)

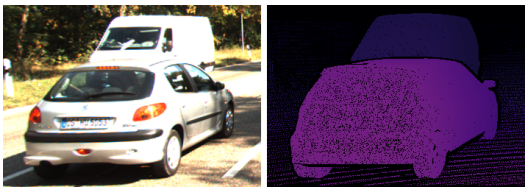
Fig. 8. Confidence map related correlation coefficients with the horizontal depicting a series of KITTI images, used as the evaluation set: (a) shows the correlation between the confidence maps and the absolute difference between the corresponding MDE and SDE input, and fused disparity maps; while (b) shows the correlation between the confidence maps corresponding to the stereo and monocular methods.

The extent to which the fusion exploits the available confidence maps in order to improve on the initial estimate is measured through the correlation coefficient between the said confidence map and the corresponding change map (i.e. the absolute disparity difference between the initial estimate and the fused one). This coefficient is expected to be negative in optimal cases (anti-correlation), as areas of high confidence should not be changed much by the fusion. A positive correlation coefficient may indicate that the fusion network made significant changes to estimates that were found to have a high level of certainty by the reconstruction error-based confidence mechanism. From 8a, it can be seen that for most evaluation cases, the fusion network could take only a limited advantage of the confidence maps, as most of the data points hover in the low negative range. The fact that the above correlations are significantly lower in magnitude for the SDE than for the MDE indicates that the fusion relies on the former more than on the latter. This is to be expected, as the chosen stereo method outperforms the monocular one (see Table I). Note that there are a few outliers (most importantly, cases 000073 and 000198), with considerably high, positive coefficients. It was found that at certain scenes, the confident predictions of SDE largely overlap with the ones of the MDE, and vice versa. This overlap was measured through the correlation coefficient between the two confidence maps, which is shown in 8b. Lower correlation between the confidence maps indicates a better utilization of this additional information by the fusion network, as it suggests that the two input networks perform better in different sections of the scene, which can then be harnessed in the fusion. However, a high correlation coefficient may be a sign that the two input methods produce similarly uncertain estimates corresponding to the same areas of the input image.

B. *Qualitative analysis*

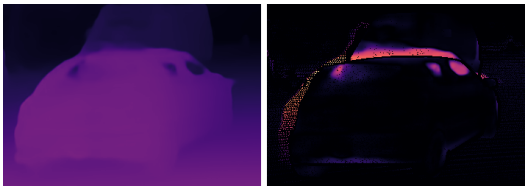
When taking a close look at the fused estimates (e.g. in Figure 9), the synergistic combination of MDE and SDE becomes clear: on the one hand, the errors due to occlusion from the stereo estimates are greatly reduced; while on the other, artifacts from the ambiguity of monocular cues are disregarded in favor of the more reliable stereo matching.

As discussed in subsection II-A, stereo matching in regions that are not visible for both cameras is limited at a fundamental level. Thus, it is expected that the SDE that makes predictions for the left-view image will have large errors at the left edges of objects, as these pixels are not present in the right-view image. SDE disparity maps are expected to accumulate larger errors than usual in scenes where objects are in the foreground, close to the camera. The effect of occlusion is more prevalent in such cases, because the disparities are larger and the occluded areas take up a larger portion of the image. This can be clearly seen in 9b, where the occluded sections of the vehicle contour show up brightly on the error map, indicating large discrepancies with respect to the label. However, this is significantly (although not completely) reduced on the fused disparity map, 9d. In low texture areas, the stereo matching algorithm was expected to produce lower quality predictions

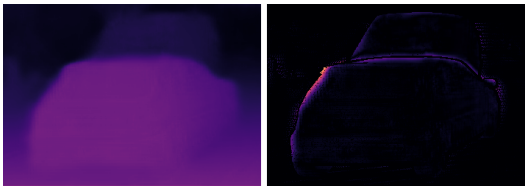


(a) Input and ground truth

(b) SDE



(c) MDE



(d) Fusion

Fig. 9. Row (a) shows a section of an input scene and the corresponding ground truth. Rows (b), (c) and (d) show the resulting stereo, monocular, and fused disparity and error maps. Note, how fusion greatly improves the errors corresponding to occlusion in SDE and reflection in MDE. The disparity and the error maps use two different color scales, in order to make the erronous regions more apparent.

than the MDE, the opposite was found to be the case. SDE disparities in smooth surfaces, such as asphalt roads and car bodywork, had lower error and higher values on the confidence map than the ones from MDE. From the above, it is concluded that the primary limitation of current state-of-the-art SDE is occlusion, which can be addressed by the fusion network.

Some undesired artifacts of monocular methods are corrected as well. in 9c, one can see that the MDE makes the wrong predictions at the car windows. Due to the reflecting or see-through nature of glass windows, the corresponding monocular cues are utterly misleading, resulting in much larger depth estimates than the true distance. Seemingly, such image sections do not pose a problem for LEAStereo (see 9b). This difference between the two methods was successfully captured in the fusion algorithm model, as the disparities at the car windows in 9d are correctly predicted. Further examples of the same artifact can be seen in the 3rd and 4th rows of Figure 10

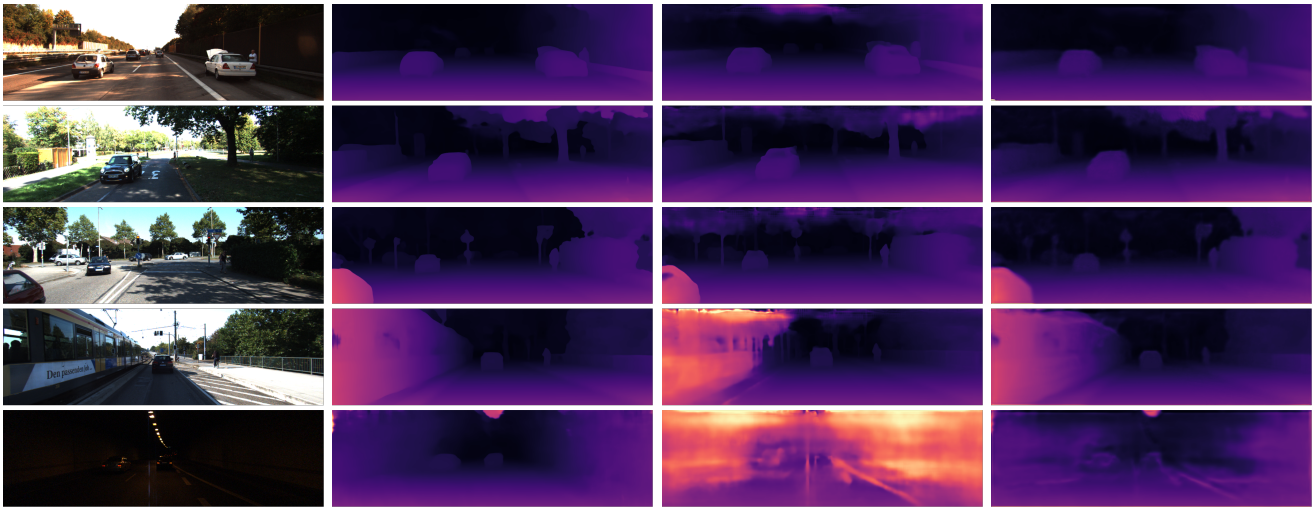
However, the fused disparity maps depict a glaring shortcoming: the edges of scene objects are markedly more blurred in comparison to the input disparity maps, particularly those derived from the stereo solution. This has an obvious negative impact on both accuracy and visual "looks" of the disparity maps. The source of this problem is two fold: the loss of local structure (small-scale details, such as object boundaries) as the feature map is gradually downscaled through the encoder section; [44], [45] and the insensitivity regular depth losses (such as L1) to prediction shifts in the spatial directions. The former can be mitigated by augmenting the standard loss function with a gradient-based term, which is more penalizing for errors around edges [44]. The latter, which is a fundamental weakness of CNNs, was expected behavior, the mitigation of which was attempted via skip connection, as explained in subsection III-B. Evidently, skip connections at their current placement were not enough. Note from Figure 3, that the first skip connection is already preceded by the initial downsampling layer two ResBlocks of the encoder. Thus, it is possible that the skip connection with the highest spatial resolution is already too deep and lacking the local structure for a sharp disparity map. The alternative solution found by Hu et al.[44] and Chen et al. [45] is the use of multi-scale schemes, as opposed to the more traditional encoder-decoder architecture.

C. Ablation study

For the ablation study, all combinations of several network depths and additional information were considered. The network depth is varied by combining an encoder and a decoder of certain depths. The base network, as presented in subsection III-B, is a combination of the smallest encoder and decoder. Larger networks are made up of a combination of single-, double-, or triple-sized encoders and decoders. The detailed design of each of these is given in section A. These result in nine different encoder-decoder combinations, with an increasing number of learning parameters. Furthermore, two different priors, the grayscale left-view image and the confidence maps corresponding to the SDE and MDE maps, may be used as additional channels in the input. These led to a total of 36 unique architectures, which were tested, examining their performance in inference time and prediction accuracy. The detailed numerical results of these experiments are presented in section C.

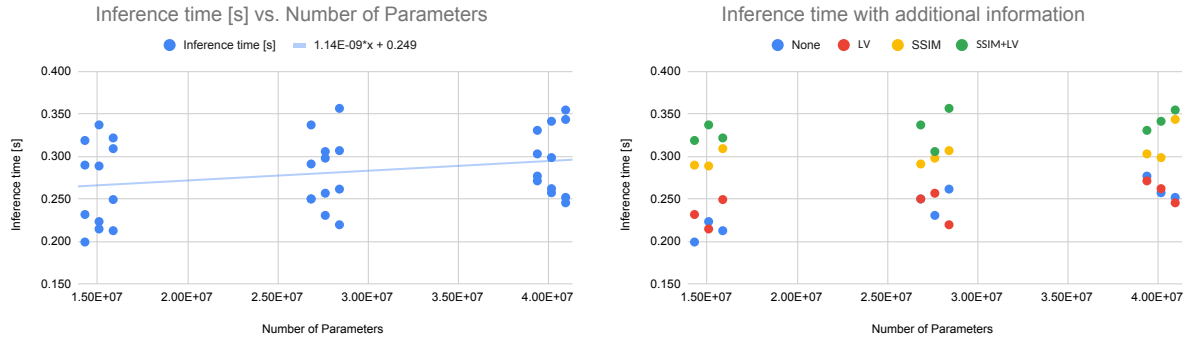
Regarding the inference time, the network depth (i.e. the number of parameters), and the number and type of additional information all have an impact on the performance. Each additional 10^7 parameter was found to increase the inference time by around 1.4 milliseconds, as can be seen in 11a. Considering that the number of parameters of the tested networks ranges between $1.43 * 10^7$ and $4.10 * 10^7$, this is relatively insignificant. However, the inclusion of SSIM-based confidence maps has a substantial impact, as it increases the inference time by more than 50 milliseconds. The extra cost comes from the computational complexity of the windowed SSIM operation (see Equation 2), and the reconstructed pixel validity checking algorithm (see subsection III-A).

When it comes to network depth, bigger is not always better. As shown in Table III, smaller networks frequently surpass larger ones in performance. However, the largest network, comprising a 3x encoder and a 3x decoder, is arguably the



(a) Input (b) SDE (c) MDE (d) Fusion

Fig. 10. Some additional results: Column (a) shows the input scene, and columns (b), (c) and (d) show the SDE, MDE and fused disparity maps, respectively. For each row, the color scale of the predictions are the same.



(a) (b)

Fig. 11. Inference time for each case considered in the ablation study: 9 different network depths and 4 different combinations of additional data.The general trend with respect to the network depth and the additional information are shown on (a) and (b), respectively.

best, as indicated in Table VI. Interestingly, the combinations with balanced encoders and decoders (i.e. 1en - 1de, 2en - 2de, 3en - 3de) tend to outperform the rest. Note that the number of parameters between the encoder and decoder is widely different, with the encoder increasing in size by $125 * 10^5$ between the single and double, and the double and triple variants, while the decoder only increases by $7.84 * 10^5$. Thus, it is assumed that an even distribution in parameters between the encoder and decoder is not necessary. On the contrary, a ratio of about $16 : 1$ in favor of the encoder may be closer to optimal. The ratio in the number of convolutional layers between the encoder and the decoder is $17 : 9$ and $49 : 17$ in the 1en - 1de and the 3en - 3de architectures. The large disparity in parameters is due to the fact that ResBlocks in the encoder include weights and biases, while the upconvolutional layers in the decoder only include weights.

Finally, it can be determined from Table V, that the right information as additional channels has a positive impact on the accuracy of the disparity in general. Including the left-view image in the input gives a significant boost to the accuracy of the estimation. It is assumed that it provides contextual information to the fusion network, in order to better recognize areas in the scene that may be problematic for either the SDE or the MDE. Surprisingly, the confidence maps not only do not improve accuracy on their own, but even degrade the quality of the fused disparity map. The fused estimate, however, is further improved by supplying the confidence maps in addition to the left-view image, leading to the top performing architecture, 3en - 3de with SSIM+LV.

V. Conclusion

In this paper, an encoder-decoder style CNN was presented for the purposes of improving dense monocular and stereo depth estimates through the fusion of the two. When tested on the KITTI2015 stereo benchmark, it was found that such an algorithm can fulfill this task, when sufficient contextual information is provided alongside the stereo and monocular estimates. The network was tested with the gray-scale image

Additional data	Lower is better					Higher is better			Lower is better
	Abs. Rel.	Sq. Rel.	RMSE	EPE	3px err.	δ_1	δ_2	δ_3	Inference time [s]
1en - 1de	0.046	**0.302**	2.081	1.081	0.065	0.963	**0.990**	**0.997**	**0.260**
2en - 1de	0.047	0.319	2.137	1.138	0.067	0.963	0.987	0.996	0.282
1en - 2de	0.052	0.474	2.382	1.294	0.068	0.959	0.984	0.994	0.268
2en - 2de	**0.044**	0.308	**2.032**	**1.041**	**0.061**	**0.966**	0.988	0.994	0.270
3en - 1de	0.061	1.809	2.689	1.458	0.074	0.963	0.986	0.993	0.294
1en - 3de	0.058	0.807	2.489	1.371	0.068	0.954	0.980	0.991	0.272
2en - 3de	0.051	0.587	2.330	1.233	0.064	0.962	0.984	0.993	0.287
3en - 2de	0.057	0.624	2.403	1.331	0.071	0.960	0.985	0.993	0.292
3en - 3de	0.048	0.477	2.254	1.166	0.063	0.964	0.988	0.995	0.288

TABLE III

METRICS AVERAGED ACROSS ALL COMBINATIONS OF ADDITIONAL DATA, WITH THE DIFFERENT ARCHITECTURE SIZES USED IN THE EXPERIMENTS. THE BEST RESULTS ARE BOLDFACED, AND THE SECOND-BEST ONES ARE IN RED.

Additional data	Lower is better					Higher is better			Lower is better
	Abs. Rel.	Sq. Rel.	RMSE	EPE	3px err.	δ_1	δ_2	δ_3	Inference time [s]
1en - 1de	0.044	0.209	2.015	1.048	0.063	0.966	0.993	**0.998**	**0.199**
2en - 1de	0.044	0.245	1.981	1.020	0.063	0.968	0.990	0.997	0.250
1en - 2de	0.045	0.302	2.093	1.064	0.059	0.965	0.988	0.996	0.215
2en - 2de	**0.040**	0.199	**1.904**	0.931	0.056	0.971	**0.994**	**0.998**	0.231
3en - 1de	0.044	0.233	1.966	1.053	0.063	0.967	0.991	0.997	0.271
1en - 3de	0.050	0.377	2.218	1.170	0.060	0.959	0.985	0.996	0.213
2en - 3de	0.043	0.205	1.953	1.010	0.058	0.972	0.993	**0.998**	0.220
3en - 2de	0.043	0.219	2.004	1.036	0.063	0.968	0.993	**0.998**	0.257
3en - 3de	**0.040**	**0.194**	1.917	**0.928**	**0.055**	**0.974**	0.993	**0.998**	0.245

TABLE IV

BEST METRICS AMONG ALL COMBINATIONS OF ADDITIONAL DATA, WITH THE DIFFERENT ARCHITECTURE SIZES USED IN THE EXPERIMENTS. THE BEST RESULTS ARE BOLDFACED, AND THE SECOND-BEST ONES ARE IN RED.

Additional data	Lower is better					Higher is better			Lower is better
	Abs. Rel.	Sq. Rel.	RMSE	EPE	3px err.	δ_1	δ_2	δ_3	Inference time [s]
N/A	0.053	0.544	2.323	1.264	0.068	0.959	0.983	0.993	**0.240**
LV	0.048	0.406	2.208	1.158	0.065	**0.965**	**0.989**	0.996	0.249
SSIM	0.062	1.418	2.722	1.495	0.071	0.957	0.980	0.989	0.303
SSIM+LV	**0.046**	**0.275**	**2.069**	**1.074**	**0.063**	**0.965**	**0.989**	**0.997**	0.334

TABLE V

METRICS AVERAGED ACROSS ALL ARCHITECTURE SIZES, WITH THE DIFFERENT ADDITIONAL DATA USED IN THE EXPERIMENTS. THE BEST RESULTS ARE BOLDFACED, AND THE SECOND-BEST ONES ARE IN RED.

Additional data	Lower is better					Higher is better			Lower is better
	Abs. Rel.	Sq. Rel.	RMSE	EPE	3px err.	δ_1	δ_2	δ_3	Inference time [s]
N/A	0.043	0.346	2.006	1.003	0.059	0.955	0.979	0.990	**0.199**
LV	0.043	0.205	1.953	1.010	0.058	0.957	0.984	0.991	0.220
SSIM	0.044	0.266	2.035	1.063	0.064	0.968	0.990	0.996	0.289
SSIM+LV	**0.040**	**0.194**	**1.904**	**0.928**	**0.055**	**0.974**	**0.994**	**0.998**	0.306

TABLE VI

BEST METRICS AMONG ALL ARCHITECTURE SIZES, WITH THE DIFFERENT ADDITIONAL DATA USED IN THE EXPERIMENTS. THE BEST RESULTS ARE BOLDFACED, AND THE SECOND-BEST ONES ARE IN RED.

and a pair of reconstruction error based confidence maps corresponding to the input disparity maps, and was found that it performs the best with the combination of both. The fusion algorithm represents an improvement on the input (reference) algorithms in almost all metrics. Compared to the input stereo disparity maps, while the fusion outputs feature fewer extreme outliers, accuracy of some very precise prediction from the stereo input is washed out. The network managed to successfully address fundamental limitations of both stereo matching and single-view methods by relying on the other one at the problematic areas: it was proven to mitigate the errors resulting from occlusion and reflecting or see-through glass windows. These improvements, however, come at a cost. The inference time, especially considering that the fusion network (0.355 second per scene) relies on the outputs of two other deep neural nets (0.212 and 0.073 seconds per scene for the MDE and SDE, respectively), limits its application in robotics or in tasks where real-time prediction is required. Additionally, in the dark tunnel scene, while the SDE managed to produce a reasonably good disparity map in a scene not represented in the training set, the fused disparity was considerably less accurate. This suggests a deteriorated generalization capability of the fused network compared to the input stereo matching network.

During experiments, a number of points of improvement were identified regarding the fusion network. The SDE, MDE and fusion inference times are measured as separate entities, i.e. each of them having their own separate overhead (such as dataloaders, and pre- and post-processing algorithms). As they perform similar tasks with similar input data, the combined inference time can probably be significantly reduced via more thorough integration. Furthermore, the confidence

map generation may be completely parallelized, as it currently includes a for-loop in the reconstructed pixel validity check, creating a significant bottleneck. The currently blurry fused depth map output may be refined by including a gradient-based term in the loss function, or by adopting a multi-scale refinement architecture for fusion, such as the feature pyramid of Chen et al.[45]. Finally, a more sophisticated architecture, such as the two-branch backbone of PENet[28], may be used to improve the fusion performance when the left-view image or confidence maps are included as additional data, as the simple concatenation is less efficient in multi-modal fusion. This, however, will increase the computational complexity, and thus, the inference time.

With the above being stated, learning-based fusion is clearly beneficial for accurate depth or disparity measurements, but future work in either or both the hardware and the network itself is necessary for wide-spread deployment in robotics, and autonomous vehicles in particular.

REFERENCES

[1] J. N. Yasin, S. A. Mohamed, M.-H. Haghbayan, J. Heikkonen, H. Tenhunen, and J. Plosila, "Unmanned aerial vehicles (uavs): Collision avoidance systems and approaches," *IEEE access*, vol. 8, pp. 105 139–105 155, 2020.

[2] P. Baraldi, E. De Micheli, and S. Uras, "Motion and depth from optical flow.," in *Alvey Vision Conference*, 1989, pp. 1–4.

[3] J. L. Barron, A. D. Jepson, and J. K. Tsotsos, "The feasibility of motion and structure from noisy time-varying image velocity information," *International Journal of Computer Vision*, vol. 5, no. 3, pp. 239–269, 1990.

[4] P. Batista, C. Silvestre, P. Oliveira, and B. Cardeira, "Accelerometer calibration and dynamic bias and gravity estimation: Analysis, design, and experimental evaluation," *IEEE transactions on control systems technology*, vol. 19, no. 5, pp. 1128–1137, 2010.

[5] K. Souhila and A. Karim, "Optical flow based robot obstacle avoidance," *International Journal of Advanced Robotic Systems*, vol. 4, no. 1, p. 2, 2007.

[6] M. Jaritz, R. De Charette, E. Wirbel, X. Perrotton, and F. Nashashibi, "Sparse and dense data with cnns: Depth completion and semantic segmentation," in *2018 International Conference on 3D Vision (3DV)*, IEEE, 2018, pp. 52–60.

[7] D. Martins, K. van Hecke, and G. de Croon, "Fusion of stereo and still monocular depth estimates in a self-supervised learning context," *2018 IEEE International Conference on Robotics and Automation (ICRA)*, 2018. DOI: 10.1109/icra.2018.8461116.

[8] J. M. Facil, A. Concha, L. Montesano, and J. Civera, "Deep single and direct multi-view depth fusion," *CoRR, abs*, vol. 1611, p. 30, 2016.

[9] J. Li, P. Wang, P. Xiong, *et al.*, "Practical stereo matching via cascaded recurrent network with adaptive correlation," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2022, pp. 16 263–16 272.

[10] R. Garg, V. K. B.G., G. Carneiro, and I. Reid, "Unsupervised cnn for single view depth estimation: Geometry to the rescue," in *Computer Vision – ECCV 2016*, B. Leibe, J. Matas, N. Sebe, and M. Welling, Eds., Cham: Springer International Publishing, 2016, pp. 740–756, ISBN: 978-3-319-46484-8.

[11] H. Hirschmuller, "Stereo processing by semiglobal matching and mutual information," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 30, no. 2, pp. 328–341, 2007.

[12] N. Mayer, E. Ilg, P. Hausser, *et al.*, "A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 4040–4048.

[13] A. Kendall, H. Martirosyan, S. Dasgupta, *et al.*, "End-to-end learning of geometry and context for deep stereo regression," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 66–75.

[14] X. Cheng, Y. Zhong, M. Harandi, *et al.*, "Hierarchical neural architecture search for deep stereo matching," *Advances in Neural Information Processing Systems*, vol. 33, 2020.

[15] X. Fan, S. Jeon, and B. Fidan, "Occlusion-aware self-supervised stereo matching with confidence guided raw disparity fusion," in *Conference on Robots and Vision*, 2022.

[16] J. Li, P. Wang, P. Xiong, *et al.*, "Practical stereo matching via cascaded recurrent network with adaptive correlation," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2022, pp. 16 263–16 272.

[17] A. Saxena, S. Chung, and A. Ng, "Learning depth from single monocular images," *Advances in neural information processing systems*, vol. 18, 2005.

[18] G. De Croon, C. De Wagter, B. Remes, and R. Ruijsink, "Sky segmentation approach to obstacle avoidance," in *2011 Aerospace Conference*, IEEE, 2011, pp. 1–16.

[19] D. Eigen, C. Puhrsch, and R. Fergus, "Depth map prediction from a single image using a multi-scale deep network," in *Advances in Neural Information Processing Systems*, Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K. Weinberger, Eds., vol. 27, Curran Associates, Inc., 2014. [Online]. Available: https : / / proceedings . neurips . cc / paper / 2014 / file / 7bccfde7714a1ebadf06c5f4cea752c1-Paper.pdf.

[20] I. Laina, C. Rupprecht, V. Belagiannis, F. Tombari, and N. Navab, "Deeper depth prediction with fully convolutional residual networks," in *2016 Fourth international conference on 3D vision (3DV)*, IEEE, 2016, pp. 239–248.

[21] M. Mancini, G. Costante, P. Valigi, T. A. Ciarfuglia, J. Delmerico, and D. Scaramuzza, "Toward domain independence for learning-based monocular depth estimation," *IEEE Robotics and Automation Letters*, vol. 2, no. 3, pp. 1778–1785, 2017.

[22] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 3431–3440.

[23] J. Xie, R. Girshick, and A. Farhadi, "Deep3d: Fully automatic 2d-to-3d video conversion with deep convolutional neural networks," in *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part IV 14*, Springer, 2016, pp. 842–857.

[24] C. Peng, X. Zhang, G. Yu, G. Luo, and J. Sun, "Large kernel matters–improve semantic segmentation by global convolutional network," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 4353–4361.

[25] H. Fu, M. Gong, C. Wang, K. Batmanghelich, and D. Tao, "Deep ordinal regression network for monocular depth estimation," in *Proceedings of the IEEE confer-*

ence on computer vision and pattern recognition, 2018, pp. 2002–2011.

[26] R. Ranftl, A. Bochkovskiy, and V. Koltun, "Vision transformers for dense prediction," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2021, pp. 12 179–12 188.

[27] F. Ma and S. Karaman, "Sparse-to-dense: Depth prediction from sparse depth samples and a single image," in *2018 IEEE international conference on robotics and automation (ICRA)*, IEEE, 2018, pp. 4796–4803.

[28] M. Hu, S. Wang, B. Li, S. Ning, L. Fan, and X. Gong, "Penet: Towards precise and efficient image guided depth completion," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, 2021, pp. 13 656–13 662.

[29] J. H. Lee, M.-K. Han, D. W. Ko, and I. H. Suh, "From big to small: Multi-scale local planar guidance for monocular depth estimation," *arXiv preprint arXiv:1907.10326*, 2019.

[30] M. Menze and A. Geiger, "Object scene flow for autonomous vehicles," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 3061–3070.

[31] J. Uhrig, N. Schneider, L. Schneider, U. Franke, T. Brox, and A. Geiger, "Sparsity invariant cnns," in *International Conference on 3D Vision (3DV)*, 2017.

[32] T. v. Dijk and G. d. Croon, "How do neural networks see depth in single images?" In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 2183–2191.

[33] F. Ma, G. V. Cavalheiro, and S. Karaman, "Self-supervised sparse-to-dense: Self-supervised depth completion from lidar and monocular camera," in *2019 International Conference on Robotics and Automation (ICRA)*, IEEE, 2019, pp. 3288–3295.

[34] J. Qiu, Z. Cui, Y. Zhang, *et al.*, "Deeplidar: Deep surface normal guided depth prediction for outdoor scene from sparse lidar data and single color image," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 3313–3322.

[35] Z. Wang, E. P. Simoncelli, and A. C. Bovik, "Multiscale structural similarity for image quality assessment," in *The Thrity-Seventh Asilomar Conference on Signals, Systems & Computers, 2003*, Ieee, vol. 2, 2003, pp. 1398–1402.

[36] C. Godard, O. Mac Aodha, and G. J. Brostow, "Unsupervised monocular depth estimation with left-right consistency," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 270–279.

[37] R. Stikker, "Self-supervised finetuning of stereo matching algorithms," M.S. thesis, Faculty of Aerospace Engineering, Delft University of Technology, 2022.

[38] R. Girshick, *Fast r-cnn*, 2015. arXiv: 1504.08083 [cs.CV].

[39] K. He, X. Zhang, S. Ren, and J. Sun, *Deep residual learning for image recognition*, 2015. arXiv: 1512.03385 [cs.CV].

[40] A. Odena, V. Dumoulin, and C. Olah, "Deconvolution and checkerboard artifacts," *Distill*, 2016. DOI: 10.23915/distill.00003. [Online]. Available: http://distill.pub/2016/deconv-checkerboard.

[41] D.-A. Clevert, T. Unterthiner, and S. Hochreiter, *Fast and accurate deep network learning by exponential linear units (elus)*, 2016. arXiv: 1511.07289 [cs.LG].

[42] R. Yasrab, N. Gu, and X. Zhang, "An encoder-decoder based convolution neural network (cnn) for future advanced driver assistance system (adas)," *Applied Sciences*, vol. 7, no. 4, p. 312, 2017.

[43] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *International conference on machine learning*, pmlr, 2015, pp. 448–456.

[44] J. Hu, M. Ozay, Y. Zhang, and T. Okatani, "Revisiting single image depth estimation: Toward higher resolution maps with accurate object boundaries," in *2019 IEEE winter conference on applications of computer vision (WACV)*, IEEE, 2019, pp. 1043–1051.

[45] X. Chen, X. Chen, and Z.-J. Zha, "Structure-aware residual pyramid network for monocular depth estimation," *arXiv preprint arXiv:1907.06023*, 2019.

APPENDIX A
NETWORK ARCHITECTURE DETAILS

Layer	Input	Input Resolution	1x encoder
en0	MDE+SDE	[2, 352, 1216]	Conv2d(kernel=7, stride=2, padding=3) MaxPool2d(kernel=3, stride=2, padding=1) BatchNorm2D() ReLU()
	MDE+SDE+LV	[3, 352, 1216]	
	MDE+SDE+SSIM	[4, 352, 1216]	
	MDE+SDE+SSIM+LV	[5, 352, 1216]	
en1	en0	[64, 88, 304]	ResBlock(downsample=False) ResBlock(downsample=False)
en2	en1	[64, 88, 304]	ResBlock(downsample=True) ResBlock(downsample=False)
en3	en2	[128, 44, 152]	ResBlock(downsample=True) ResBlock(downsample=False)
en4	en3	[256, 22, 76]	ResBlock(downsample=True) ResBlock(downsample=False)

TABLE VII
DETAILED OVERVIEW OF THE SIMPLE (1X ENCODER) ENCODER ARCHITECTURE USED IN THE EXPERIMENTS

Layer	Input	Input Resolution	2x encoder
en0	MDE+SDE	[2, 352, 1216]	Conv2d(kernel=7, stride=2, padding=3) MaxPool2d(kernel=3, stride=2, padding=1) BatchNorm2D() ReLU()
	MDE+SDE+LV	[3, 352, 1216]	
	MDE+SDE+SSIM	[4, 352, 1216]	
	MDE+SDE+SSIM+LV	[5, 352, 1216]	
en1	en0	[64, 88, 304]	ResBlock(downsample=False) ResBlock(downsample=False) ResBlock(downsample=False) ResBlock(downsample=False)
en2	en1	[64, 88, 304]	ResBlock(downsample=True) ResBlock(downsample=False) ResBlock(downsample=False) ResBlock(downsample=False)
en3	en2	[128, 44, 152]	ResBlock(downsample=True) ResBlock(downsample=False) ResBlock(downsample=False) ResBlock(downsample=False)
en4	en3	[256, 22, 76]	ResBlock(downsample=True) ResBlock(downsample=False) ResBlock(downsample=False) ResBlock(downsample=False)

TABLE VIII
DETAILED OVERVIEW OF THE DOUBLE (2X ENCODER) ENCODER ARCHITECTURE USED IN THE EXPERIMENTS

Layer	Input	Input Resolution	3x encoder
en0	MDE+SDE	[2, 352, 1216]	Conv2d(kernel=7, stride=2, padding=3) MaxPool2d(kernel=3, stride=2, padding=1) BatchNorm2D() ReLU()
	MDE+SDE+LV	[3, 352, 1216]	
	MDE+SDE+SSIM	[4, 352, 1216]	
	MDE+SDE+SSIM+LV	[5, 352, 1216]	
en1	en0	[64, 88, 304]	ResBlock(downsample=False) ResBlock(downsample=False) ResBlock(downsample=False) ResBlock(downsample=False) ResBlock(downsample=False) ResBlock(downsample=False)
en2	en1	[64, 88, 304]	ResBlock(downsample=True) ResBlock(downsample=False) ResBlock(downsample=False) ResBlock(downsample=False) ResBlock(downsample=False) ResBlock(downsample=False)
en3	en2	[128, 44, 152]	ResBlock(downsample=True) ResBlock(downsample=False) ResBlock(downsample=False) ResBlock(downsample=False) ResBlock(downsample=False) ResBlock(downsample=False)
en4	en3	[256, 22, 76]	ResBlock(downsample=True) ResBlock(downsample=False) ResBlock(downsample=False) ResBlock(downsample=False) ResBlock(downsample=False) ResBlock(downsample=False)

TABLE IX

DETAILED OVERVIEW OF THE TRIPLE (3X ENCODER) ENCODER ARCHITECTURE USED IN THE EXPERIMENTS

Layer	Input	Input Resolution	1 x decoder	2 x decoder	3 x decoder
de1	en4	[512, 11, 38]	interpolate(scale=2, mode=nearest) Conv2d(kernel=3, stride=2, padding=1) ELU() BatchNorm2D()	interpolate(scale=2, mode=nearest) Conv2d(kernel=3, stride=2, padding=1) ELU() BatchNorm2D() Conv2d(kernel=3, stride=1, padding=1) ELU() BatchNorm2D()	interpolate(scale=2, mode=nearest) Conv2d(kernel=3, stride=2, padding=1) ELU() BatchNorm2D() Conv2d(kernel=3, stride=1, padding=1) ELU() BatchNorm2D() Conv2d(kernel=3, stride=1, padding=1) ELU() BatchNorm2D()
skip1	de1, en3	256, 22, 76], [256, 22, 76]	concat(de1, en3) Conv2d(kernel=3, stride=1, padding=1) ELU()	concat(de1, en3) Conv2d(kernel=3, stride=1, padding=1) ELU()	concat(de1, en3) Conv2d(kernel=3, stride=1, padding=1) ELU()
de2	skip1	[256, 22, 76]	interpolate(scale=2, mode=nearest) Conv2d(kernel=3, stride=2, padding=1) ELU() BatchNorm2D()	interpolate(scale=2, mode=nearest) Conv2d(kernel=3, stride=2, padding=1) ELU() BatchNorm2D() Conv2d(kernel=3, stride=1, padding=1) ELU() BatchNorm2D()	interpolate(scale=2, mode=nearest) Conv2d(kernel=3, stride=2, padding=1) ELU() BatchNorm2D() Conv2d(kernel=3, stride=1, padding=1) ELU() BatchNorm2D() Conv2d(kernel=3, stride=1, padding=1) ELU() BatchNorm2D()
skip2	de2, en2	128, 44, 152], [128, 44, 152]	concat(de2, en2) Conv2d(kernel=3, stride=1, padding=1) ELU()	concat(de2, en2) Conv2d(kernel=3, stride=1, padding=1) ELU()	concat(de2, en2) Conv2d(kernel=3, stride=1, padding=1) ELU()
de3	skip2	[128, 44, 152]	interpolate(scale=2, mode=nearest) Conv2d(kernel=3, stride=2, padding=1) ELU() BatchNorm2D()	interpolate(scale=2, mode=nearest) Conv2d(kernel=3, stride=2, padding=1) ELU() BatchNorm2D() Conv2d(kernel=3, stride=1, padding=1) ELU() BatchNorm2D()	interpolate(scale=2, mode=nearest) Conv2d(kernel=3, stride=2, padding=1) ELU() BatchNorm2D() Conv2d(kernel=3, stride=1, padding=1) ELU() BatchNorm2D() Conv2d(kernel=3, stride=1, padding=1) ELU() BatchNorm2D()
skip3	de3, en1	[128, 88, 304]	concat(de3, en1) Conv2d(kernel=3, stride=1, padding=1) ELU()	concat(de3, en1) Conv2d(kernel=3, stride=1, padding=1) ELU()	concat(de3, en1) Conv2d(kernel=3, stride=1, padding=1) ELU()
de4	skip3	[64, 88, 304]	interpolate(scale=2, mode=nearest) Conv2d(kernel=3, stride=2, padding=1) ELU() Conv2d(kernel=3, stride=1, padding=1) ELU() Conv2d(kernel=3, stride=1, padding=1) Sigmoid()	interpolate(scale=2, mode=nearest) Conv2d(kernel=3, stride=2, padding=1) ELU() Conv2d(kernel=3, stride=1, padding=1) ELU() BatchNorm2D() Conv2d(kernel=3, stride=1, padding=1) ELU() Conv2d(kernel=3, stride=1, padding=1) Sigmoid()	interpolate(scale=2, mode=nearest) Conv2d(kernel=3, stride=2, padding=1) ELU() Conv2d(kernel=3, stride=1, padding=1) ELU() BatchNorm2D() Conv2d(kernel=3, stride=1, padding=1) ELU() BatchNorm2D() Conv2d(kernel=3, stride=1, padding=1) ELU() Conv2d(kernel=3, stride=1, padding=1) Sigmoid()
Output		[1, 352, 1216]			

Fig. 12. Detailed overview of all decoder architectures used in the experiments

APPENDIX B
HYPERPARAMETERS AND TRAINING DETAILS

Name	Value
batch_size	4
num_epochs	100
init_learning_rate	1e-4
end_learning_rate	-1
weight_decay	1e-2
adam_eps	1e-3
min_disp	1e-3
max_disp	192

APPENDIX C
ABLATION STUDY DETAILS

UNet depth	Parameters	Additional data	SILog ↓	abs rel ↓	sq rel ↓	rmse ↓	EPE ↓	3px acc ↓	d1 ↑	d2 ↑	d3 ↑	Inference time [s] ↓
1x encoder - 1x decoder	1.43E+07	**N/A**	8.561	0.050	0.462	2.186	1.110	0.064	0.963	0.985	0.994	0.199
2x encoder - 1x decoder	2.68E+07	**N/A**	8.933	0.054	0.486	2.421	1.374	0.076	0.956	0.981	0.993	0.250
1x encoder - 2x decoder	1.51E+07	**N/A**	9.170	0.062	0.680	2.604	1.544	0.079	0.955	0.979	0.991	0.223
2x encoder - 2x decoder	2.76E+07	**N/A**	7.939	0.043	0.346	2.006	1.003	0.059	0.967	0.988	0.995	0.231
3x encoder - 1x decoder	3.94E+07	**N/A**	8.601	0.044	0.349	2.089	1.053	0.064	0.964	0.987	0.994	0.277
1x encoder - 3x decoder	1.59E+07	**N/A**	8.318	0.051	0.560	2.342	1.243	0.060	0.959	0.983	0.993	0.213
2x encoder - 3x decoder	2.84E+07	**N/A**	8.873	0.055	0.694	2.437	1.318	0.066	0.960	0.981	0.990	0.261
3x encoder - 2x decoder	4.02E+07	**N/A**	9.447	0.063	0.815	2.552	1.472	0.073	0.955	0.979	0.990	0.257
3x encoder - 3x decoder	4.10E+07	**N/A**	8.400	0.054	0.506	2.266	1.259	0.067	0.956	0.982	0.993	0.252
1x encoder - 1x decoder	1.43E+07	**LW**	8.324	0.044	0.209	2.015	1.048	0.063	0.966	0.993	0.998	0.232
2x encoder - 1x decoder	2.68E+07	**LW**	8.822	0.046	0.280	2.112	1.095	0.066	0.961	0.988	0.996	0.250
1x encoder - 2x decoder	1.51E+07	**LW**	9.416	0.052	0.429	2.487	1.309	0.069	0.957	0.986	0.996	0.215
2x encoder - 2x decoder	2.76E+07	**LW**	8.373	0.043	0.208	1.923	0.968	0.059	0.971	0.992	0.997	0.257
3x encoder - 1x decoder	3.94E+07	**LW**	8.602	0.049	0.254	2.115	1.194	0.071	0.963	0.991	0.997	0.271
1x encoder - 3x decoder	1.59E+07	**LW**	8.391	0.050	0.377	2.218	1.170	0.070	0.957	0.985	0.996	0.249
2x encoder - 3x decoder	2.84E+07	**LW**	8.085	0.043	0.205	1.953	1.010	0.058	0.972	0.993	0.998	0.220
3x encoder - 2x decoder	4.02E+07	**LW**	8.386	0.043	0.219	2.004	1.036	0.063	0.968	0.993	0.998	0.262
3x encoder - 3x decoder	4.10E+07	**LW**	8.972	0.055	0.975	2.750	1.380	0.064	0.964	0.984	0.991	0.245
1x encoder - 1x decoder	1.43E+07	**SSIM**	8.166	0.044	0.303	2.041	1.067	0.064	0.963	0.988	0.996	0.290
2x encoder - 1x decoder	2.68E+07	**SSIM**	8.173	0.044	0.266	2.035	1.063	0.064	0.968	0.990	0.996	0.291
1x encoder - 2x decoder	1.51E+07	**SSIM**	8.560	0.055	0.694	2.497	1.349	0.069	0.954	0.980	0.991	0.289
2x encoder - 2x decoder	2.76E+07	**SSIM**	8.423	0.050	0.484	2.342	1.260	0.065	0.959	0.984	0.993	0.298
3x encoder - 1x decoder	3.94E+07	**SSIM**	9.589	0.051	0.479	2.293	1.261	0.069	0.956	0.978	0.987	0.303
1x encoder - 3x decoder	1.59E+07	**SSIM**	10.557	0.103	6.399	4.586	2.524	0.097	0.956	0.976	0.983	0.309
2x encoder - 3x decoder	2.84E+07	**SSIM**	9.292	0.073	1.732	3.047	1.742	0.067	0.949	0.971	0.982	0.307
3x encoder - 2x decoder	4.02E+07	**SSIM**	8.985	0.061	1.165	2.727	1.459	0.068	0.956	0.979	0.989	0.299
3x encoder - 3x decoder	4.10E+07	**SSIM**	9.801	0.074	1.240	2.930	1.728	0.080	0.949	0.975	0.985	0.344
1x encoder - 1x decoder	1.43E+07	**SSIM+LW**	8.418	0.044	0.233	2.083	1.097	0.067	0.961	0.992	0.998	0.319
2x encoder - 1x decoder	2.68E+07	**SSIM+LW**	8.368	0.045	0.245	1.981	1.020	0.063	0.967	0.990	0.997	0.337
1x encoder - 2x decoder	1.51E+07	**SSIM+LW**	8.296	0.045	0.302	2.093	1.064	0.059	0.965	0.988	0.996	0.337
2x encoder - 2x decoder	2.76E+07	**SSIM+LW**	8.102	0.040	0.199	1.904	0.931	0.056	0.971	0.994	0.998	0.306
3x encoder - 1x decoder	3.94E+07	**SSIM+LW**	8.295	0.047	0.233	1.966	1.062	0.063	0.967	0.991	0.997	0.331
1x encoder - 3x decoder	1.59E+07	**SSIM+LW**	8.769	0.057	0.560	2.350	1.330	0.075	0.951	0.980	0.992	0.322
2x encoder - 3x decoder	2.84E+07	**SSIM+LW**	8.711	0.046	0.284	2.201	1.143	0.065	0.959	0.984	0.995	0.357
3x encoder - 2x decoder	4.02E+07	**SSIM+LW**	8.647	0.046	0.222	2.124	1.089	0.066	0.967	0.993	0.998	0.341
3x encoder - 3x decoder	4.10E+07	**SSIM+LW**	7.994	0.040	0.194	1.917	0.928	0.055	0.974	0.993	0.998	0.355

Fig. 13. Experiment details of the ablation study

Part II

Literature Study

1

Micro Aerial Vehicles around us

The mobility provided by aviation has been exploited by humanity for over a century. Not having to accommodate crew allows drones to be smaller, and thus more agile in their employment. Once the hardware on-board required for remote control or autonomous operation is smaller and lighter than the human crew it is replacing, the deployment of such machines becomes potentially more economical. This is especially true for flying vehicles, where an increase in local mass always leads to a 3-6 times greater increase in the total mass of the aircraft [9], a phenomenon known as the snowball effect. By eliminating the need for a crew, these vehicles can safely go to places that would otherwise be dangerous or even inaccessible to humans.

The fact that in the last 7 years, the global UAV market increased 11 times the size it was in 2015[1] indicates that drones have proven their usefulness for many purposes. Four major areas of application can be identified: hobby, industrial and commercial, public services, and military. Civilian use for hobby purposes includes aerial photography and racing[2][3], among others. Many public services make use of the mobility and birds-eye view provided by rotary wing aircraft (helicopters). UAVs may take over some of these applications where the large size and the human crew are not necessary, being a cheaper alternative to manned crafts. Such applications include wildfire detection in support of fire departments [4]. Search & rescue party may make use of UAVs to cover a large area[5], which would require an expensive fleet of manned aircraft otherwise. According to Silvagni et al. [52], in addition to locating missing persons through visible and thermal cameras, UAVs can fill other domain and task-specific niches, such as initiating controlled avalanches or delivering emergency survival kits. One of the most well-known commercial applications of drones is aerial delivery, as companies such as Amazon have shown serious interest in the technology[6]. There is much more potential and already established applications for commercial drones, especially in the inspection and maintenance of infrastructure. As proposed by Quater et al. [48], UAVs equipped with thermal sensors can be used to detect several failure modes of photovoltaic plants. Small UAVs (such as Micro Aerial Vehicles) are also used for inspection of indoor ducts, interiors of pressure vessels, and other hard-to-access areas [7]. The concept of wind turbine blade inspection via UAVs has been tested and validated by Car et al. [7], expanding the potential application of drones even further.

The small mass and dimensions of micro aerial vehicles (or MAVs, small lightweight drones, sometimes as small as the 3 gram DelFly [14]), offer these aircraft a high level of transportability and the ability to deploy them on short notice, with very little specialized equipment and infrastructure [19]. Additionally, due to their light weight (when coupled with slow speed), MAVs are unlikely to cause serious

[1] www.statista.com/statistics/913668/projected-global-uav-market-size/ [Retrieved on 16/11/2022]
[2] www.thedroneracingleague.com/ [Retrieved on 16/11/2022]
[3] www.multigp.com/ [Retrieved on 16/11/2022]
[4] www.dronewatch.eu/chc-and-vrnhn-drones-wildfire-monitoring/ [Retrieved on 16/11/2022]
[5] www.visionaerial.com/drones-for-search-and-rescue/ [Retrieved on 16/11/2022]
[6] www.aboutamazon.com/news/transportation/how-amazon-is-building-its-drone-delivery-system [Retrieved 16/11/2022]
[7] www.ostservices.com/uav-inspection [Retrieved 16/11/2022]

injury or damage in collision ([26]). The inherently safer design means a simpler authorization process for operation according to EASA regulations [2], as the risks are considerably lower to begin with. In case of a remotely piloted MAV, the vehicle would fall into the *open* category due to its small size and weight. In the case of autonomous operation, the MAV would fall into the *specific* category, requiring official authorization after a meticulous risk assessment process. As the energy and dimensions of the MAVs are smaller than those of other larger UAVs, the intrinsic ground risk of the UAS will be lower; thus, fewer and simpler risk mitigation measures may be required to obtain authorization for the same operation.

2

Collision avoidance in the air

While MAVs can be designed to be inherently safe, they also are expected to operate at very low altitude (due to their limited range and endurance), where they are likely to encounter a dynamic and obstacle-ridden environment. EASA regulations on UAV operations require the use of detect and avoid systems for tactical air risk mitigation measure, when the UAV is operating beyond visual line-of-sight, in an airspace where the chance to encounter a manned aircraft is any higher than extremely low [2] . Safe operation must be pursued even beyond the level set out by regulation, in order to further improve the feasibility of autonomous vehicles, and their perception by the public. Thus, drones should preferably be equipped with some kind of collision avoidance system, when applicable. This subsection provides a quick overview of the primary methods used for sensing obstacles.

Trajectory conflict resolution in traditional aircraft is done in a cooperative manner. In order to detect a conflict, aircraft continuously share information about their state (location, altitude, heading, speed, etc.), while also receiving similar information from nearby aircraft. The evasion maneuvers are then agreed upon and executed by the aircraft involved. This procedure is often facilitated by the Air Traffic Services or on-board systems, such as TCAS [1]. Cooperative methods build on the assumption that objects involved in a trajectory conflict broadcast state information and will collaborate in the evasion. As these are often not given at low-altitude operation, small drones must be prepared for non-cooperative collision avoidance: the aircraft must be able to independently sense its environment, recognize potential obstacles, project the future state of obstacles, identify trajectory conflicts, and plan and execute an appropriate evasion maneuver, when necessary [4].

Yasin et al. [62] divide non-cooperative sensing methods into two categories, based on whether it makes use of active or passive sensors. The former emits some radiation which reflects off of objects and is then picked up by a receiver. Several properties of the object can be conferred from knowledge of the emitted and received signals. Most importantly, the bearing is found from the receiver orientation when the reflected signal is detected, and the range is calculated from the time that passed between the transmission and reception of the signal [12]. More advanced systems may be able to find additional information, such as relative speed, calculated from the Doppler shift between the emitted and received signal [3]. Laser-based LIDAR, radio wave-based RADAR, and sound-based SONAR are a few commonly used methods of this category. Active sensors tend to be accurate and require little processing power. They have the additional advantage of not having to rely on external sources for illumination.

Passive sensors rely on signals emitted by the scenery, which are then reflected by objects, or signals emitted by the objects themselves. The signals are captured and processed by the aircraft in order to create a picture of its environment. These sensors are cameras that are most commonly operating in the visible, infrared, and ultraviolet ranges. Passive methods are not exclusive to the electromagnetic domain; Mizumachi et al. [44] for example experimented with the detection of car traffic with acoustic sensors. Traditional cameras, which are a very common consumer product, are well developed and inexpensive. In addition, they are small, lightweight and can capture high-resolution images in three

channels. However, they are heavily dependent on environmental conditions, such as illumination and a clear atmosphere. Infrared sensors are not limited to daylight, but are most often more blurry and of lower resolution than traditional cameras. Both types of cameras capture a very large amount of information, the processing and interpretation of which is a challenging problem. This research attempts to address this challenge.

The performance of a sensor must be measured against the associated size, weight, required power and cost (SWaP-C)[1]. It is useful to describe these properties with a single metric, as often one can be traded for the other three. SWaP-C is of special concern for drones, as additional size and weight leads to the snowball effect similar to other aircraft, power is often limited on UAVs, and the use of expensive components degrades one of the primary selling points of drones, the low cost. Active sensors have a prohibitively high SWaP-C for deployment in small UAVs [62]. Thus, most research, including this one, investigates detection with passive sensors, such as cameras.

[1] `www.trentonsystems.com/blog/what-is-swap-c` [Retrieved 22/11/2022]

3

Vision-based depth perception

In this chapter, the theoretical background and the evolution of learning-based depth estimation are discussed icomplementins of monocular and stereo vision. Only solutions with available publications detailing the design, implementation and testing of the program are considered. As primary performance indicators, absolute relative error, root mean square error, and run time are used, as reported by the authors on the KITTI2015 or NYU-DepthV2 dataset for monocular networks and KITTI stereo for stereo networks. In section 3.1 and in section 3.3, the evolution and the state-of-the-art in deep learning-based monocular and stereo depth estimation are presented, respectively. In section 3.5, a few more algorithms are introduced, which were assumed to perform better with the limited compute available on-board of MAVs. Section 3.6 discusses the few architectures in which the fusion of multiple different branches has been utilized in depth estimation. Finally, section 3.7 discusses the relevance of self-supervision and presents a couple of studies that have achieved notable results with it.

Depth perception is the ability of the observer to judge the distance of an object[1]. In computer vision, the task of dense depth estimation is to find the pixel-wise depth map associated with a projection of the scene. There are three subcategories of this task, the depth estimation from a single image, images taken at the same time with multiple cameras, and consecutively taken images with a single camera as it traverses in space. This subsection introduces the underlying principles, advantages, and disadvantages of each of these.

3.1. Stereo depth estimation

Figure 3.1: The principles of disparity in stereo vision has been harnessed for depth estimation for much longer than the emergence of computer vision. ”Target practice. Range finder at work.” by Harris & Ewing, 1913[2]

[1] `https://www.aao.org/eye-health/anatomy/depth-perception` [Retrieved 22/11/2022]

Stereo depth estimation (SDE) works with multiple cameras with a known and fixed focal length and offset. Due to this offset between the cameras, when a pair of pictures is taken simultaneously, the position of objects on the produced image pairs is shifted. The shift in corresponding pixels between the images is the disparity. After obtaining the disparity, the depth of the corresponding pixel can be computed through triangulation as shown in figure 3.2. This leads to the expression $\hat{z} = B\ f\ d^{-1}$, where $\hat{z}$ is the estimated depth. The baseline B and focal length f are both known. The choice of these properties will affect the performance of the setup. For example, a wider baseline generally results in more accurate long-distance measurements but also requires a larger structure to mount the cameras on. Given the above expression, the challenge of SDE is to obtain an accurate and reliable disparity for each pixel that makes up the disparity map. The depth map is limited to the field-of-view (FoV) overlap of the two cameras. Consequently, traditional stereo matching is not able to resolve points that are occluded to one of the two cameras. The reliability of pixel matching is greatly reduced in areas of repetitive, fin, or no texture, since the algorithm is likely to find several similarly good matches. When relying only on discrete disparities, the disparity error will have a theoretical lower limit of 0.5 pixel ([46]). Disparity errors have a larger impact on the depth estimate for distant pixels (as the disparity decreases inversely with increasing range), and so the maximum range of practical stereo matching is limited by the camera resolution. This can be mitigated both on the hardware and software level. The former means higher resolution cameras, cameras of larger focal length, or longer baseline. On the software level, instead of discrete disparities, calculating continuous disparities (for example, via *soft argmin*[32]) improves the sub-pixel level accuracy. Finally, the addition of a second camera and the corresponding structure for spacing them result in an increase in weight compared to monocular methods.

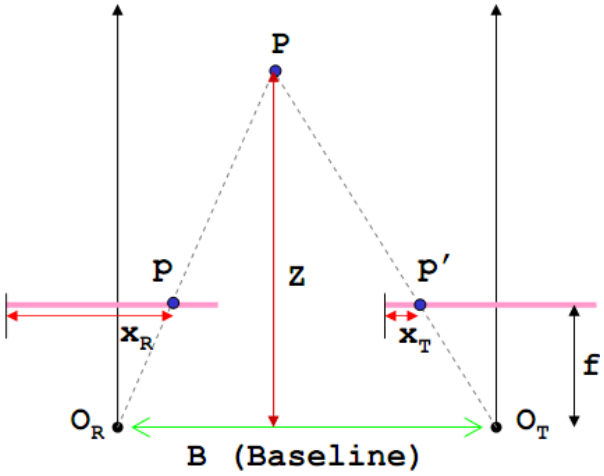


Figure 3.2: Relation between depth z, disparity $d = x_R - x_T$, distance between cameras B, and focal length f[3]

3.2. Model-based stereo matching

Finding the disparity at distinct points, such as edges and corners is quite straight forward, and non-learning-based computational methods were developed before the popularization of deep learning techniques. Local methods calculate the matching cost (often based on the color and intensity of the pixel) for each pixel between the two images. Patches centered on the pixel in question instead of individual pixels may be used for matching in order to obtain a more robust cost. Global methods find the minimum of a single cost function defined by the disparities corresponding to every pixel. This is often referred to as the energy function. Hirschmuller[28] proposes an in-between solution, semi-global matching (SGM). The cost for a given pixel is defined as the sum of the pixel matching cost and a smoothness constraint, penalizing disparity changes of the neighboring pixels. The aggregated cost for a pixel is the sum of the costs to reach that pixel from 8 directions. The disparity of that pixel is for which this aggregated cost function is minimum. The disparity is found at the minima of the aggregated cost function.

[2] `https://www.loc.gov/pictures/item/2016864982/` [Retrieved 09/06/2024]
[3] `http://ashutoshdtu.blogspot.com/2012/10/stereo-vision-introduction.html` [Retrieved 07/11/2022]

3.2.1. Convolutional Networks in stereo matching

While due to their ease of application, semi-global matching algorithms stay relevant today, they produce noisy results, littered with outliers. Model-based matching algorithms such as SGM stay relevant even today, due to their ease of application, no required training, and consequently their ability to generalize to all domains. Substantial improvement in terms of accuracy was achieved with the proliferation of deep convolutional networks. One of the first solutions for disparity map estimation from stereo image pairs via CNNs was DispNetCorr (a version of their base architecture DispNet, with modified for the task of disparity estimation), a demonstrator network by Mayer et al.[42] for their stereo scene flow training dataset. An image pair is first passed through a feature extractor involving a single convolutional layer. The two produced feature spaces are then fused via 1D correlation along the epipolar line. The rest of their architecture follows the traditional encoder-decoder style fully convolutional network with skip connections. It is worth noting that the decoder is made of alternating convolutional and upconvolutional layers, as opposed to pure upconvolutions. In training, their dataset was augmented via spatial and chromatic transformations, such as cropping, and color, contrast and brightness filters. GCNet by Kendall et al.[32] manage to achieve better accuracy through more elaborate architecture design, based on engineering insight into the geometry of disparity calculation. Consequently, the GCNet pipeline is reminiscent of that of traditional stereo matching algorithms. Once again, instead of using raw image pixels, the stereo pair is fed into a feature extractor. For each image, a 4D cost volume is constructed by concatenating each extracted feature with their corresponding feature from the other image. Kendall et al.[32] find that the use of such cost volumes leads to better performance than the direct distance or correlation computation between the features (as it is done in DispNetCorr). The cost volume includes the matching cost for each pixel for each possible disparity. The regularization, aimed at refining the disparity estimates by learning the context, is performed via a 3D encoder-decoder network. The use of 3D convolutions greatly increases the computation cost compared to 2D convolutions. At the same time, however, subsampling also decreases the feature space much faster, as it is done along all three dimensions. Such encoder-decoder architecture comes with the same drawback as the CNNs used for MDE: the fine details are washed out at the deeper layers. This was addressed in GCNet via the use of skip connections between the encoder and decoder layers with outputs of the same scale. Finally, in order to obtain the disparity map from the final feature space, the differentiable operator *soft argmin* is defined as $\sum_{d=0}^{d=D_{max}} d \times \sigma(-c_d)$, where D_{max} is the maximum disparity, and c_d is the predicted cost for disparity d. It has the advantages over the traditional *argmin* operator that it is differentiable and continuous, allowing back-propagation training and sub-pixel level disparity estimation.

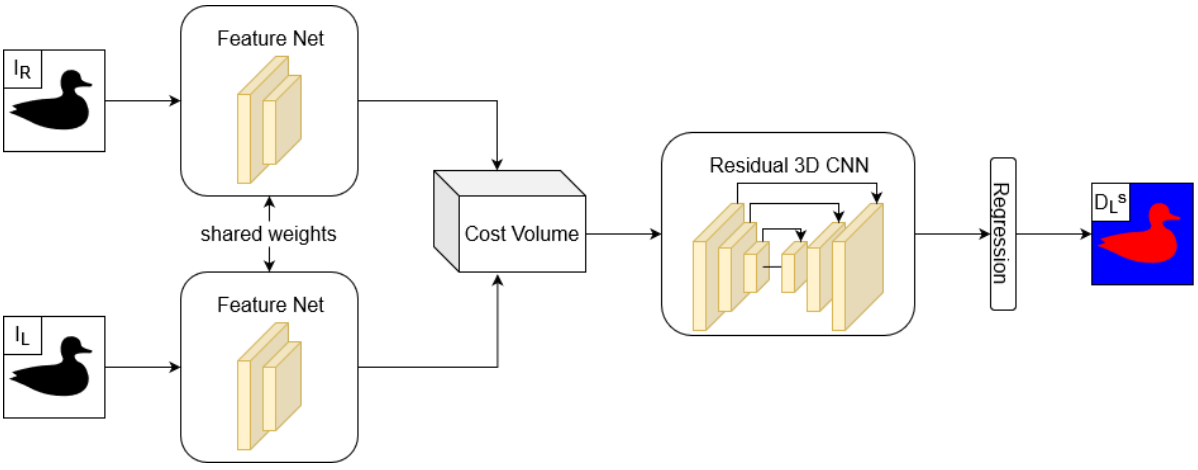


Figure 3.3: The general procedure that is commonly utilized by cost volume-based stereo matching algorithms: extraction of features, construction of a cost volume, residual encoder-decoder of 3D convolutions, and the solution to the cost minimization problem.

The methodology and pipeline of GCNet is commonly used by other stereo matching algorithms, making use of the 3D convolutional regularization module of a cost volume. A more general architecture following the same concept is shown in figure 3.3.

Li et al.[34] identify the three primary challenges of SDE as (1) dealing with imperfect rectification, (2) accurately estimating the disparity in areas with repetitive or no texture, and (3) finding an accurate disparity in fine details such as thin nets. In their proposed architecture, CREStereo, Li et al.[34] address each of these difficulties. (1) In cases where some distortion remains in the image pair even after rectification, the corresponding points will often not be on the same search line. To address this vulnerability of SDE, a windowed local search is applied along the epipolar line, with alternating linear and square search windows. Due to the local correlation computations, the generated 3D cost volume is much smaller than the 4D cost volume of GCNet in the previous paragraph. (2) In an attempt to reduce the disparity error in repetitive or textureless areas, the sample points in the search window are offset by a certain amount. The offset parameters are learned during the training phase. (3) Finally, the matching of fine and high detail points needs to happen at high resolution and high level features, as such details would wash out at the lower scales of deep levels. Thus, a cascaded recurrent network is adopted, which upsamples the previous, lower scale predicted disparity map and uses it as the initialization of the disparity map, predicted from the larger-scale input image. The very first cascade is initialized with a zero space at the 1/16th scale of the original input. As Li et al.[34] conclude, such a network is highly dependent on the careful design of the training data, while also suffering from efficiency issues, as it is not able to run in real time on small devices. Still, this algorithm runs over twice as fast as GCNet.

Cheng et al.[11] build on the idea of constructing a large cost volume and processing it via 3D convolutions, similar to GCNet and the mock architecture in figure 3.3. The four major steps of the pipeline are the extraction of image features with a 2D convolutional feature net, the construction of the cost volume from the left and right feature maps, the refinement of this volume via a 3D convolutional feature net, and the disparity prediction (note that, as opposed to Kendall et al.[32], Cheng et al.[11] refer to the volume constructed from the left and right features as ”feature volume”, and the output of the 3D convolutional net as ”cost volume”). Instead of carefully designing the feature extraction network and the matching network, they apply neural architecture search (NAS) to automatically design them. The search operates on two hierarchical levels. First, on the cell level, it defines what operation shall be performed in a cell. A cell may have two inputs, three operations, and one output. The possible cell operations are skip connection and 2D or 3D convolution for the feature net and the matching net, respectively. Second, on the network level, specifying the spatial resolution of each layer. The search space for the network level search is constructed by defining the lowest spatial resolution and the number of desired layers. By constraining the search space that the NAS has to parse through (limiting the possible operations in the cells, and pre-defining the smallest scale and the depth of the networks), the GPU memory requirement is kept at a manageable level. The architecture search revealed a number of lessons that can be learned for future stereo matching networks. The feature extraction network does not need to be very deep (as LEAStereo already achieves good results with a depth of six layers). However, keeping the resolution high during the feature extraction and thus constructing a larger volume from the extracted features was found to lead to better accuracy. And finally, the encoder-decoder structure, which is made from a series of convolution operations, followed by deconvolution operations, was found to be not necessary for either the feature extraction or the matching net, as the architectures found by the NAS include both at a seemingly random order. LEAStereo[11] by Cheng et al. represents the state of the art on the KITTI stereo benchmark.

3.3. Monocular depth estimation

Monocular depth estimation (MDE) is the task of dense depth estimation from a single view image. While humans innately perform well in this task (relying on monocular visual cues such as occlusion, perspective, relative size of familiar objects, etc.), it has been a challenging problem in robotics: since infinitely many different 3D scenes can produce a given 2D projection, the reconstruction of depth is a mathematically ill-posed problem. Using task-specific insight and prior knowledge of the environment, some depth information can be extracted from simple machine learning approaches, such as sky segmentation. First proposed by McGee et al. in 2005,[43] a sky segmentation algorithm classifies each pixel of the input image as either sky or no-sky, where the everything falling into the latter category is considered to be obstacles. The classification is based on hand-crafted weights calculated from the

[4] https://en.wikipedia.org/wiki/File:Madurodam_21.jpg [Retrieved 25/11/2022]

Figure 3.4: The miniature aircraft appear to be much bigger and thus further away than they really are. Madurodam in 2007, by Malis - Ondřej Málek[4]

color of the pixels. Such algorithms are limited in scope and provide only crude estimates. For example, the sky segmentation approach is only viable with a visible sky (thus only operable outdoors), and the depth resolution is limited to two extremes: very large (sky) and relatively small (no-sky)[13].

When looking at approaches estimating the true depth, one of the primary challenges of MDE is the fact that most monocular visual cues, such as interposition (or occultation), perspective, or relative size, only describe the relative depth between the objects. The global scale is captured through familiar size, relating objects of known size to all others in the scene. For this task, the MDE algorithm must also consider contextual information and long (spatial) range dependencies in order to produce an accurate depth map. For example, semantic information may be utilized to improve the depth estimate[51, 20].

Saxena et al.[50] realized the importance of contextual information, which they attempted to capture by extracting features from an image (via hand-crafted convolutional filters) and relating them to each other via the Markov Random Field method. Although still relying on task-specific prior knowledge (shown in the design of the convolutional filters), the Make3D of Saxena et al.[50] was intended to be general purpose, not limited to a single type of environment. With the rise of popularity of deep neural networks, MDE has drawn a lot of attention as it offers more alternatives to extract features and capture contextual information in an image. MDE only requires a single camera, promising a very low weight solution for range estimation (see **??**), which is always of special interest in aviation. In addition, it is a standalone system that does not require any knowledge of the state of the vehicle it is deployed on. In theory, MDE does not have a limited depth range, and can predict the distance to both very far and very close points. However, it struggles to predict the global scale of the scene. MDEs use the scale of objects with known size as they appear in the image to determine the scale of the scene[40]. However, the familiar object used as reference may not have the same size as inferred from prior knowledge or training data. For example, the presence of people in the background and the miniature aircraft in the foreground of figure 3.4 create an uncertainty of global scale. This significantly increases the absolute error of the estimated depth map. Finally, reliance on known objects limits the generalization capability of learning-based MDEs, making them reliable only in domains that are well represented in the training data.

Some of the first and arguably the most influential pioneers of tackling MDE via deep learning algorithms were Eigen et al.[18]. In their paper, they propose a convolutional neural network (CNN) with two branches: a coarse and a fine network. The coarse-scale convolutional network increases the receptive field of the convolution operator via a series of pooling operations, in order to extract global information. The last layer in this branch is a fully connected layer, which rescales the 4096 channel deep feature space into a single channel depth map. The fine-scale network does not downscale anymore after an initial convolution with pooling, which reduces the resolution of the feature space to match the one of the coarse output. The course depth map is concatenated with the fine feature

space, which is then passed through two more layers of convolutions. As mentioned before, these do not involve pooling, thus the resolution is retained. Note that the final depth map is still considerably smaller in resolution than the input image, e.g. an input of 576x172 pixels from the KITTI benchmark only leads to a 142x27 depth map. The other major contribution of this paper is the scale-invariant error, equation (3.1) where y_i and y_i^* are the predicted and the ground truth depths, respectively, at pixel i. This measure represents the depth error of the pixels relative to each other. This means that the prediction of the global scale, which accounts for about one-fifth of the total error, has less of an impact on the scale-invariant error. This measure, or some versions of it, still remains a popular option both in benchmarking and as a training loss function. Incorporating the scale-invariant error in the training loss has the beneficial effect of forcing the system to focus on more accurate relative depth prediction, which MDEs are generally good at.

$$D(y, y^*) = \frac{1}{n}\sum_i d_i^2 - \frac{1}{n^2}\left(\sum_i d_i\right)^2, \quad \text{where } d_i = \log y_i - \log y_i^* \tag{3.1}$$

Fully convolutional networks in monocular depth estimation

The major breakthrough in the evolution of MDE was the adoption of fully convolutional networks with architecture resembling the encoder-decoder scheme of autoencoders. Such architectures, also known as U-nets, were pioneered by Garg et al.[24] and Laina et al.[33] Early U-nets often made use of already existing deep networks, originally designed for other tasks such as semantic segmentation or image classification, as the encoder (also referred to as backbone). For example, Laina et al.[33] build on ResNet-50[27] as the backbone. Encoders incrementally increase the receptive field of the convolutional operator by reducing the spatial resolution of the feature space via down-sampling (spatial pooling) at each layer. In the meantime, the number of channels is increased in order to retain more information. A typical encoder-decoder architecture is shown in figure 3.5. A large receptive field is essential for dense pixel-wise prediction tasks, since global information can only be evaluated when the receptive field is sufficiently large. MDE infers the global scale by comparing the relative size of known objects (from training) to the scene[40]. The spatial resolution of such networks at the deepest layers is too small to be used in accurate, pixel-wise prediction, thus an additional step is required: the decoder. Using a fully connected layer for transforming the deep feature map into a high spatial resolution depth map would lead to an unreasonably large number of parameters. Instead, Laina et al.[33] replace the fully connected layers of ResNet with a series of deconvolutional layers (convolution with up-sampling), making up the decoder section of the architecture. The deep feature space is passed through a series of deconvolutional layers (convolution with up-sampling) in order to obtain a spatial resolution matching the input. When properly trained, the result is a dense depth map with considerably higher resolution than before.

Two commonly used techniques for lessening the loss of spatial resolution in the encoder are the use of skip connections between the encoder and the decoder, and the use of a multi-scale network. The architecture proposed by Long et al.[38] for semantic segmentation is an encoder-like CNN with a single up-convolution layer to obtain a pixel-wise prediction. They hypothesized that finer details of the feature space, while still present in the upper layers, are lost at the deeper layers. This was proven correct, as they managed to obtain more detailed semantic maps by fusing the output of an earlier layer with the final prediction layer through a skip connection. It was also found that the earlier layer output (i.e. feature map with higher spatial resolution) they fuse, the finer the final semantic map will be, further proving their hypothesis. Garg et al.[24] adopted this idea and applied it to their encoder-decoder with AlexNet backbone. As can be seen in figure 3.5, skip connections are used to take the output of the second and fifth layers of the encoder and merge them into the second and third layers of the decoder. Following the coarse-to-fine approach, the decoder feature maps are upsampled by a factor of 4 (twice both width and height), while the encoder feature space is collapsed into a single channel feature map via a 1-by-1 convolution. The two are then added together element-wise to reach the refined feature map. Similarly, Xie et al.[59] sample their deep convolutional network at five different depths, before each pooling operation. These are brought to the same scale through deconvolution and merged via element-wise addition. The resulting feature map is then used to synthesize a right-view image from a

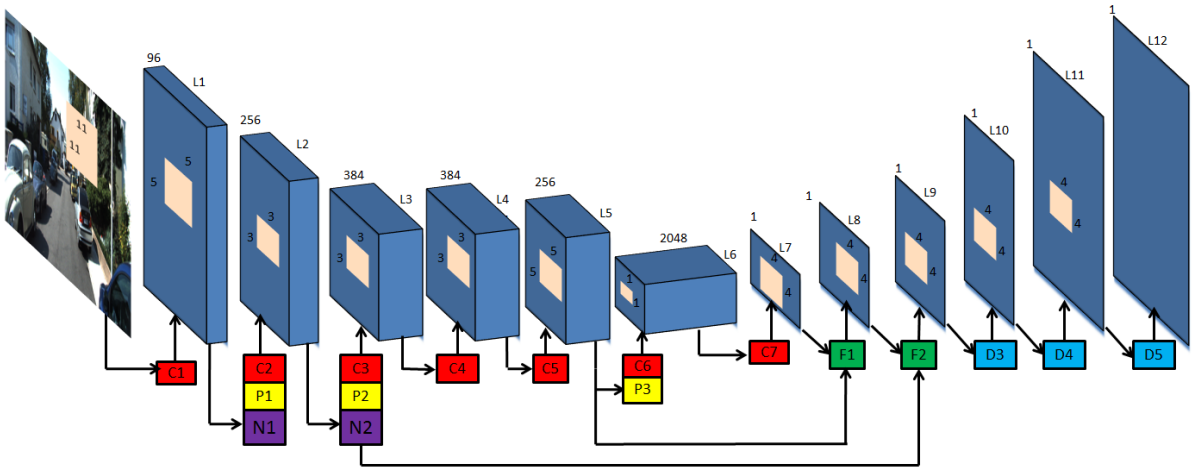


Figure 3.5: The encoder-decoder architecture of Garg et al.[24]. The colored blocks represent layer operations, specifically red being convolution, yellow being pooling, purple being normalization, green being fully connected layers, and blue being upsampling.

left-view image (more about image warping and left-right consistency in section 3.5.2).

Multi-scale networks have already been touched upon, as the work of Eigen et al.[18] makes use of a two-scale architecture. This was then taken one step further by Eigen and Fergus[17] via the addition of a third scale, and was shown to produce good results with fine details. Many modern multi-scale MDE CNNs are based on pyramid pooling (for example DORN[23], or the architectures by Song et al.[53] and Chen et al.[10]) This method was first adopted to dense pixel-wise prediction in PSPNet by Zhao et al.[64], in order to improve the processing of global information. As noted by Chen et al.[10], the challenges in semantic segmentation are often very similar to those of MDE. They build a feature pyramid by scaling the input and then extracting features via a deep convolutional encoder. The smallest feature space is decoded with a residual convolutional network in order to obtain an initial coarse depth map estimate. This is then progressively up-scaled and refined by concatenating subsequent levels of the feature pyramid and passing it through a residual convolutional network.

Ranftl et al.[49] state that the inherent problem with the deep CNN encoder is the loss of feature resolution and granularity at the deepest layers. As opposed to classification tasks (which most commonly used backbone architectures were originally designed for), for dense pixel-wise prediction a large spatial resolution (to preserve detail and nuance in the resulting depth map) and a large receptive field (to account for long-range relationships in the image and/or feature map) are both essential for accurate predictions. The latter may be addressed by increasing the kernel size[45]. Still, a large and dense kernel quickly increases the number of parameters to an unreasonable level. This can be somewhat mitigated via dilated convolution, as proposed by Yu and Koltun[63]. The dilated convolution increases the size of the kernel, and thus the receptive field of the convolution, while keeping the number of parameters the same. This is achieved by skipping every l-th pixel, according to $(F *_l k)(p) = \sum s + l \cdot t = pF(s)k(t)$, where l is the dilation factor. Fu et al.[23] have successfully applied this concept to MDE in their DORN network.

Transformers in monocular depth estimation

However, the accuracy of fully convolutional MDE networks was limited by drawbacks in the backbone. On one hand, deep architectures were necessary in order to increase the receptive field of the convolution to identify long-range relationships. On the other hand, the repeated downsampling needed to achieve this leads to the loss of detail and granularity that the decoder struggles to retrieve[49, 61]. This granularity is essential for accurate pixel-to-pixel mapping of the input image to the output depth map[18]. The solution came with the invention of the Transformer module. Transformers were originally intended for the field of natural language processing; the concept was adopted for image processing, in order to harness its outstanding ability to capture long-range relationships. The visual Transformer

(ViT) was first introduced by Dosovitskiy et al.[16] for the task of image classification. ViT applies the concept of “bag-of-words” from natural language processing by splitting the image into non-overlapping square regions. These patches are flattened into vectors of the same size via a trainable linear projection. The vectors are appended with a positional embedding. Ranftl et al.[49] was among the first to successfully apply Visual Transformers to the task of MDE. They found that better performance can be achieved when a convolutional feature extractor is applied to the image instead of directly embedding the image patches. In this case, the desired vectors are created by applying the trainable embedding function to the resulting feature space. Every “word” is related to all others through the multi-headed self-attention mechanism of the Transformers, giving ViT a global receptive field. Since the “words” perfectly correspond to their image region of origin, ViT outputs can simply be reconstructed by ordering the “words” according to the original spatial location. Multiple layers of Transformers are used in order to create a multi-scale feature pyramid. The “words” are then reassembled into an image-like feature map at multiple scales of the original input image. The multi-scaled feature maps are fused into each other via a series of convolutions and upsampling operations. Ranftl et al.[49] outperformed the state-of-the-art solutions of the time, with only a minor increase in computational cost.

When it comes to accuracy in learning-based MDE, the current state of the art is dominated by solutions incorporating both convolutional layers and Transformers. The validity of the combination of these two methods is shown by Yang et al.[60], referring to ”*lack of spatial inductive bias*” of pure Transformer architectures, a weakness that was noted by Dosovitskiy et al.[16] previously. In such a combination network, the convolutional operators are commonly used for feature extraction and the fusion and upsampling of multi-scaled feature maps after the Transformer-based architecture. Researchers have, however, found a variety of ways to apply Transformers in MDE architectures. In the pilot study of Li et al.[36], they observed that architectures using ViT in their encoders perform better on distant objects than fully convolutional ones. Interestingly enough, this is switched when estimating the depth on near objects. In order to capture the best of both methods, they make use of two separate encoder branches, one based on Transformers (proceeding patch embedding, just as described in the previous paragraph) aimed at modeling long-range spatial correlations, and a CNN encoder aimed at extracting local information. As the CNN branch is only supposed to collect local information in this architecture, it is kept very shallow, only incorporating the first block of ResNet. This improves the runtime, and avoids the loss of detail. The fusion of the outputs of these two independent branches is not an easy task, as they operate on different ”languages”, so one must ”interpret” between the two. The hierarchical aggregation and heterogeneous interaction (HAHI) module designed for the DepthFormer[36] fulfills this purpose: HAHI learns to model the affinity between the features of the two branches via cross-attention. Finally, the decoder is a regular up-convolutional network. One drawback of this architecture is the liberal use of Transformers, leading to elevated memory requirements. The authors mitigated this issue via the use of Swin Transformers[37] replacing regular ViT. To further reduce the memory cost of such large-scale attention, a deformable attention operation is proposed, which only attends only to key sampling vectors, the selection of which is done in a learnable manner. A contribution of DepthFormer from Li et al.[36] which is of special interest to the topic of fusion of stereo and monocular depth cues, is the fact that the HAHI module, being input-agnostic, may be used to enhance a monocular estimator with the features of a stereo one.

So far, most discussed solutions have framed MDE as a regression problem, i.e. the architecture is trained to predict the exact depth value corresponding to a pixel. An alternative way to frame the problem is classification, where the architecture is trained to select one from a predefined set of categories (classes) which it predicts is the most likely to fit the input data. In terms of depth prediction as a pixel-wise classification problem, the set of categories is often defined as discrete intervals of depth[23]. These intervals will be referred to as bins. Such classifiers, while managed to achieve better overall accuracy than previous regression models, are, however, limited in depth resolution due to pre-defined discrete bins. Bhat et al.[6] propose a classification-regression architecture named AdaBins, which sets the intervals through predicting the depth distribution of a given input. Instead of making use of predefined bins, it uses this predicted distribution to define the intervals, making the bins adaptive. Bhat et al.[6] note the susceptibility of this classification strategy to depth artifacts and propose to calculate the final depth for each pixel as a linear combination of bin centers and the probability corresponding to those bins (Softmax from the classifier). BinsFormer by Li et al.[35] improves on AdaBins by achieving

a better global understanding of the scene for bin generation, through the integration of the Transformer module with the convolutional encoder-decoder. Instead of placing the CNN and the transformer module one after the other, the transformer inputs are taken from the convolutional decoder layers. In this way, the bins are generated by attending to the multi-scale decoder feature pyramid. Additionally, there is a major difference in the way AdaBins and BinsFromer use the convolutional features. The CNN module of the former produces a deep feature representation with multiple channels only to be used for bin generation. The latter, however, finds a similarity map between output of the convolutional branch and the bin embeddings (similarity is calculated via the dot product), from which the probability distributions are calculated (via softmax, just like in AdaBins). Thus, a higher level of integration between the CNN and the transformer branches is achieved both for the bin generation and the final depth prediction. At the writing of this summary, BinsFormer achieves the lowest SIerror and lowest RMSE on the KITTI and the NYU-Depth v2, respectively.

3.4. Optical flow

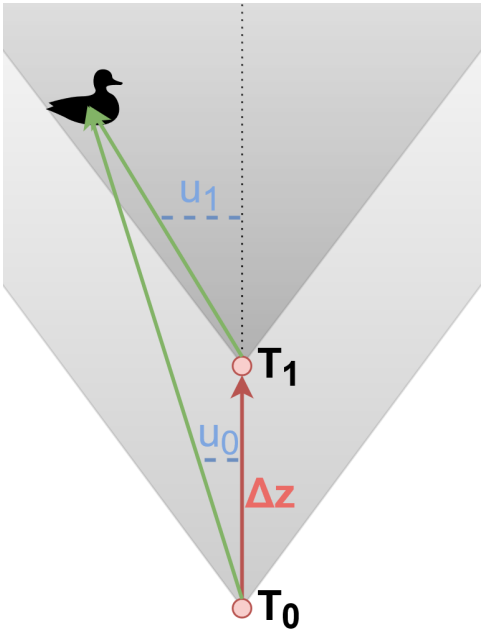


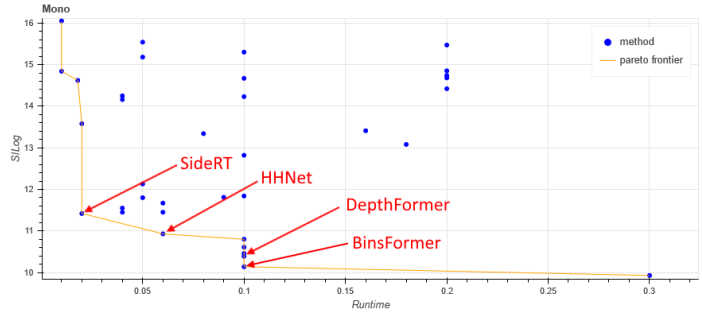
Figure 3.6: Shift in feature location between images with temporal offset and one-dimensional ego-motion

The third option is the use of optical flow. Similarly to stereo vision, it calculates the depth from the shift in feature positions between an image pair. The image pairs are produced by a single camera traversing through space, taking pictures, one after the other. Thus, the shift in image features between images is the result of a temporal offset in the camera trajectory. In such a case, instead of disparity, the shift is called optical flow. With accurate knowledge of the camera trajectory, the scene depth can be calculated via triangulation, similar to stereo matching[47]. A simplified example can be seen in figure 3.6, where the observer is limited to forward motion only. u denotes the location where the feature appears on the image. The depth estimate $\hat{z}$ can be found as $\hat{z} = u_0 \frac{\Delta z}{\Delta u}$, where $\Delta u = u_1 - u_0$. There are three very important limitations to the application of this method. First, the disparity-to-depth operation relies on accurate knowledge of the motion of the camera. Drones commonly rely on fused measurements from inertial sensors and satellite-based positioning systems for navigation. The position estimate error of such a set-up can be measured in the order of $10^{-1}[m]$ or higher[8, 58]. Second, the optical flow is approaching zero near the direction of the velocity vector (Focus of Expansion). When the optical flow is small, even small errors in flow estimation can lead to large errors in depth. This is very troubling, as obstacles in that exact area are the most likely to make contact with the vehicle. Finally, depth estimation from optical flow is only possible when all objects in the scene are stationary or if their position is known at all times. Otherwise, the extra flow resulting from the unknown motion of the object will lead to erroneous depth estimates. Based on these limitations,

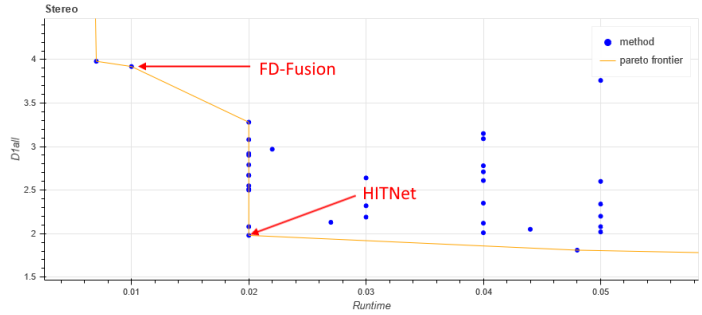
optical flow-based methods will not be considered for the proposed solution of this research project.

3.5. Accuracy-runtime Pareto optimal depth estimation

Until this point, section 3.5.1 and section 3.5.2 have been exploring the evolution and the currently best performing solutions in depth estimation with respect to precision. While the state-of-the-art may produce amazingly detailed depth maps, they are often designed with little or no regard to the hardware limitations constraining their real-world application. When making use of depth prediction on board of MAVs, the processing of images must be done in real time, with computers and batteries light enough so that the drone can maintain its mobility. Thus, a trade-off must be made between accuracy and runtime when choosing from possible depth estimation solutions. The Pareto frontier is constructed for all mono and stereo depth estimators (see figure 3.7). Both plots have been cropped for better readability. A selection of corner cases are to be concisely introduced in this subsection. A trend can be found, which generally applies to both MDE and SDE solutions: high accuracy architectures tend to be designed for CPUs, while architectures with fast runtime tend to utilize GPUs. Additionally, SDE solutions appear to be almost an order of magnitude faster than MDE solutions. Note, however, that direct comparison is difficult, as MDE and SDE use different metrics to evaluate accuracy.



(a) Monocular



(b) Stereo

Figure 3.7: Accuracy - runtime plots, with Pareto frontier and some select corner cases

3.5.1. Monocular Depth Estimation

Both the DepthFormer and the BinsFormer have been introduced in section 3.5.1. SideRT[51] is based on the creation of a feature pyramid via a set of Swin Transformers, which are pre-trained on ImageNet by Deng et al.[15] The linearly embedded non-overlapping image patches are direct inputs to the transformers, as opposed to applying a convolutional feature extractor first. Swin Transformers are used as more efficient (in terms of memory) alternatives to ViTs. The feature pyramid of 4 levels corresponds to decreasing scales and increasing channel dimensions. The lowest scale feature map is then progressively fused with the higher scale feature maps in order to refine the prediction. This fusion is performed in two stages. Cross attention between the two adjacent feature maps is used for the first fusion, which was shown to be able to capture dependencies of long spatial range, such as semantic information. The second stage of the fusion is designed to focus on local details. After upsampling the lower scale feature map to match its neighbor, the two are summed up element-wise. The resulting map is further refined by passing it through a relatively shallow network of perceptrons. This is intended to correct for

aliasing that may have occurred in upsampling. This second step of upsampling and anti-aliasing is repeated until the desired output resolution is reached. SideRT achieves a good compromise between accuracy and speed: when making use of Swin-T (the smallest of Swin Transformers), it runs 2.5 to 5 times faster than solutions of similar accuracy (DPT[49] and LapDepth[54], respectively), reaching over 83 images processed per second, on an NVIDIA GeForce RTX 2080 Ti GPU, i.e. a runtime of 0.012 seconds.

3.5.2. Stereo Depth Estimation

With HITNet, Tankovich et al.[55] developed an SDE architecture that is able to generate accurate predictions at a fast pace, allowing for real-time application. Both images of the pair are passed through a relatively shallow encoder-decoder style CNN for feature extraction. The encoder layers of each scale are connected to the corresponding decoder layers with skip connections. The feature maps of different scales from the decoder layers form a multi-scale feature pyramid. The disparity between the feature map pairs (from the left and right images) at each scale is calculated in a windowed manner, producing a set of local disparity maps. While the matching cost is computed for all disparities, only the location of the best match is kept, eliminating the need to store and process a large 3D cost volume. These crude disparity estimates will be used as initializations for each scale for further refinement. The disparity maps are refined from small to large. Using a CNN module, the confidence in the prediction of the disparities is calculated for each tile, and the full map is upscaled to match the resolution of the next disparity map. For each location, between the tiles of the upscaled prediction and the initial prediction, the one with the highest confidence is selected. The final refinement step is performed two more times in order to reach the input resolution. In the ablation study (in the supplementary material[5] of Tankovich et al.[55]), it was found that a larger network, i.e. a deeper feature extractor with more scales and thus more refinement steps, does not lead to better results on the KITTI dataset due to overfitting. The runtime of HITNet for a stereo pair from KITTI was found to be 0.019 seconds when tested on the Titan V GPU.

Ferrera et al.[21] propose a fairly shallow (and thus light weight) CNN named Dil-Net. It generates a disparity map from the stereo image pair, reinforced with preliminary disparity predictions from a model-based branch (such as SGM, introduced in section 3.5.2) and a data-based branch (a CNN, pre-trained on the benchmark dataset). The full architecture, including the fusion module and the 2 branches, is named FD-Fusion. The three sources of data (the image pair and the two preliminary disparity maps from the two branches) are concatenated and are passed through Dil-Net. Although the spatial resolution is reduced by a factor of 4 at first, instead of further gradual downsampling, it achieves a sufficiently large receptive field through the use of dilated convolution (introduced in section 3.5.1). The extra information obtained from the model-based branch enables the employed neural networks to remain relatively shallow. The use of dilated convolution further limits the number of parameters, which is essential for fast runtime.

3.6. Fusion

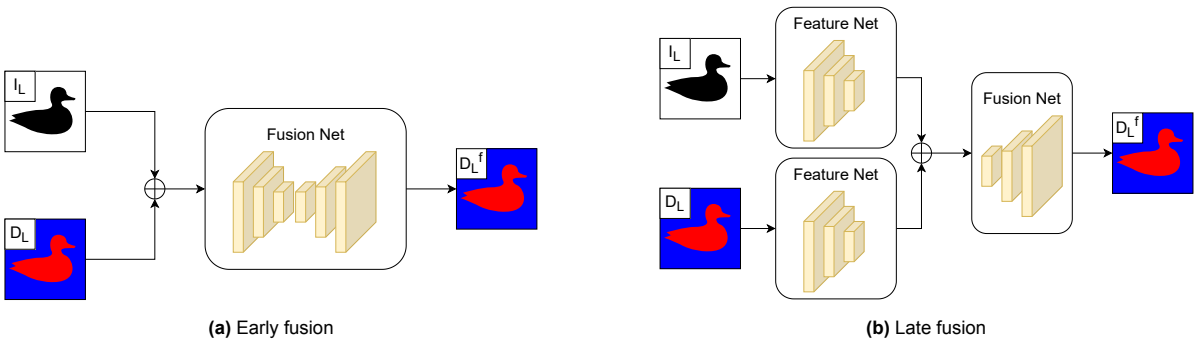
As mentioned in chapter 3, the strengths and weaknesses of the stereo and monocular methods are such that the two could potentially complement each other very well when fused properly. Very little research has been done on the explicit fusion of stereo and monocular methods in deep learning-based depth estimation. One of the first to consider the topic was Facil et al.[20]. Their solution creates two separate depth estimates, one via a convolutional MDE, and one with a traditional model-based stereo matching algorithm. The two are then fused based on a hand-crafted rule, a weighted interpolation of depths for pixels calculated by their traditional stereo matcher. The weighing model is intended to capture semantic information, in order to assign similar depths to neighboring pixels, when those are likely to belong to the *"same local structure"*. The calculation of the probability of pixels belonging to the same structure is based on their relative proximity and depth gradients. Martins et al.[41] proposed a network fusing two dense depth maps, one from a monocular CNN and one from model-based stereo matching algorithm. Just like before, the fusion follows a set of hand-crafted rules: the stereo measurements

[5] `www.openaccess.thecvf.com/content/CVPR2021/supplemental/Tankovich_HITNet_Hierarchical_Iterative_CVPR_2021_supplemental.pdf` [Retrieved on 16/11/2022]

are used where its confidence is high or where the measured depth is similar to the corresponding MDE estimate; and the MDE estimate is used at occluded areas, and where the confidence of the corresponding stereo measurement is low. Unlike the interpolated fusion of Facil et al.[20], the fusion of Martins et al.[41] results in a sort of patchwork of stereo measurements and monocular estimates.

The general fusion of feature maps is a much more studied topic. One of the simplest ways to merge two feature maps, as demonstrated by Ferrera et al.[21], is to just concatenate them, and have a deep convolutional network to merge them into an improved feature map. The convolutional SDE network of Zhong et al.[65] concatenates the feature map extracted from one image with the feature map of the other, but shifted by a disparity in a given range. This is repeated until the full disparity range is exhausted, resulting in a 4D feature volume, which is then processed with a 3D CNN. Finally, DepthFormer[36] makes use of fusion to combine the benefits of fully convolutional and transformer-based networks for MDE. They accomplish this fusion of two separate branches at the deep feature level, i.e. at the bottleneck of encoder-decoder structure. As the two branches learn very different feature representations, a cross-attention based hierarchical aggregation and heterogeneous interaction (HAHI) module is proposed to resolve the difference.

Depth completion is the task of creating a dense depth map from a single still image and a corresponding sparse but reliable input depth map (such as a LIDAR point cloud). Solutions for this task most often involve some sort of fusion between an image, and the corresponding sparse depth map. One of the simplest approaches, termed *"early fusion"* by Jaritz et al.[31] is the method of concatenating the sparse map and the RGB image before feeding it into a learning network. Mat et al.[39] did just that, with the deep network following an encoder-decoder architecture. *"Late fusion"* is the process of applying feature extractors with independent weights to each of the two inputs in order to translate them into a joint representation and upscaling the fused feature map into the desired dense depth map, as was done by Jaritz et al.[31]. As the extracted feature spaces share modalities, they are simply summed up element vise before being fed in the decoder. A visual representation of the general early and late fusion networks is shown in figure 3.8. Similar multi-branch architectures are preferred in order to deal with the differing modalities between the inputs. The branches were expanded to two complete encoder-decoder branches in PENet[30], a "color-dominant" and a "depth-dominant" one, each generating a depth map estimate. The former takes the concatenated RGB image and sparse depth map as inputs in order to generate one depth estimate. This is then concatenated with the sparse depth map in order to generate another depth map estimate through the "depth-dominant" branch. The two depth map estimates are summed element-wise, resulting in the fused depth map. Since the "color-dominant" branch takes multi-modal inputs (concatenated RGB image and sparse depth map), this part of the architecture falls in the early-fusion category. Hu et al. found that early fusion (i.e. their color-dominant branch) is sensitive to change in color or texture of the input image, lowering the confidence of the depth estimate at certain areas of the scene.



(a) Early fusion **(b)** Late fusion

Figure 3.8: Early and late fusion of multi-modal channels, as defined by Jaritz et al.[31]

3.7. Self-supervised learning for depth estimation

Data-based depth estimators achieving the best results are generally trained in a supervised manner. This requires labeled data, i.e. a reference for the algorithm to learn what the desired output is for each

input. Due to the large number of parameters in deep neural networks, proper training requires a large amount of training data. The labeling of this data is expensive and time-consuming, making sufficiently large training datasets for depth estimation scarce, and the domains represented by them limited. For example, the KITTI dataset was recorded with equipment mounted on a car, resulting in images primarily showcasing roads, traffic, and built-in areas. Such a domain is not fully representative of the environments in which MAVs are expected to operate. As was shown by Hochreiter and Schmidhuber[29], supervised learning-based convolutional algorithms suffer from generalization: Compared to their benchmark performance, their accuracy is greatly degraded when tested on images from different environments. Synthetic dataset, such as MidAir by Fonder and Droogenbroeck[22], can provide diverse and highly realistic computer-generated images while also being much cheaper than traditionally labeled ones. However, even the most diverse training set can only prepare the algorithm for anticipated domains, and thus supervised depth estimators will never be truly domain-independent. The solution is to have an algorithm generate its own labels for a given input. This process is based on additional information (e.g. stereo image pair instead of single view image) and task-specific knowledge. This is known as self-supervised or unsupervised learning. Online learning, i.e., the act of adapting to new domains during operation instead of solely relying on the parameters learned prior to deployment, is an additional challenge that will be beyond the scope of this thesis project[57]. In this subsection, three self-supervision strategies will be presented, used for the training of both SDE and MDE networks. The main challenge of self-supervised depth estimation is the generation of reliable labels. The architecture of the deep network making use of the labels is of secondary importance. Thus, the rest of the subsection will focus on the way the labels are generated but will ignore the estimation networks.

Garg et al.[24] was one of the pioneering solutions for self-supervised depth estimation. It makes use of a stereo image pair, one of which, namely the left image, is put in an encoder-decoder convolutional MDE. The produced depth map (or disparity map as there is an inverse correlation between the two) is then used to warp the right image into a synthetization of the left image. This warping $I'_l(x) = I_r(x + Bfd'^{-1}(x))$ (where $I_r(x)$ is a pixel of the right image at location x along the scan line, I'_l is the synthesized left image, $d'(x)$ is the predicted depth at location x, and B and f are the baseline and the focal length of the stereo setup respectively), is based on the inverse of the operation that is used to generate depth maps from disparity maps, as explained in chapter 3. The synthesized left image is then compared to the original, input left image in order to calculate the reconstruction error. This, along with a regularization term to encourage smoothness, is then used to train CNN.[25] found that the above method often produces artifacts at depth discontinuities, at the boundaries of objects. They took the reconstruction method a step further in order to address this issue: the encoder-decoder CNN was designed to predict both the left-to-right and the right-to-left disparity maps with only a single image as input. These disparity maps were then used to synthesize both the right and the left images. The reconstruction errors of both are then included in the training loss.

Aleotti et al.[5] propose a nested self-supervised solution: the final depth map is produced by a stereo matching network, supervised by a monocular completion network, which in turn is supervised by a model-based matching algorithm (namely semi-global matching) . An initial depth map is calculated via traditional matching algorithms. The left-right consistency check is used to filter out the unreliable points of this initial prediction, resulting in a semi-dense but highly reliable depth map. A randomly selected subset of these points is then fed into a monocular completion network, the training of which is supervised via the reliable semi-dense depth map. As the completed depth prediction is still likely to contain artifacts and inconsistencies, the above procedure is applied to a set of augmented image pairs, generated through randomly exposing the original stereo pair to color augmentation and horizontal flipping, resulting in an ensemble of depth estimates. As the augmentation should not affect the depth in theory, it is assumed that any inconsistency in the ensemble is the result of artifacts, and thus can safely be ignored. Thus, a combined semi-dense depth map can be created: a given pixel of the combined map takes the value of the corresponding ensemble pixel average only if those pixel values have a variance less than a certain cut-off value. The pixel is left blank otherwise. This combined semi-dense depth map is then used to supervise the training of an SDE network.

Finally, the Pyramid Voting Module designed by Wang et al.[56] is used to generate labels in their self-supervised stereo depth estimation network, PVStereo. Once again, the goal of the label generator

is to produce a semi-dense disparity map, devoid of unreliable points and artifacts. The mechanism behind the Pyramid Voting Module is based on the assumption that when the disparity is calculated at multiple scales, if their values are similar and their matching costs are consistent at every scale, the calculated disparity will be accurate and reliable. So, disparity maps of a given multi-scale image pair are calculated via some model-based stereo matching algorithm. Then a point on the resulting disparity map is accepted when the root mean square difference both in value and in normalized inverse matching cost is under a certain threshold. If a pixel does not meet either of the two criteria, it is left blank, leading to a semi-dense disparity map of confident values. The above method is applied to generate both a left-to-right and a right-to-left semi-dense disparity map, which are checked against each other via left-right disparity consistency check for further vetting. The final semi-dense disparity map is used to supervise the deep convolutional stereo matching network of PVStereo.

4

Research Questions and Objectives

This chapter specifies the goal of this research. The research question and the sub-questions following from it are listed in section 4.1. In order to find the answer to these questions, a number of research objectives need to be met, which are described in section 4.2.

4.1. Research Question

The primary research question is stated in bold, with the additional sub-question broken down beneath.

1. **Can the fusion of monocular and stereo depth cues through a learning-based approach enhance the performance of its input monocular and stereo networks?**

   (a) How does the proposed depth estimator (simply referred to as the estimator) perform on the KITTI Stereo 2015 benchmark?

      i. What kind of accuracy can be achieved with the estimator?
      ii. How does the accuracy of the fused monocular and stereo estimates compare to the monocular and stereo input estimates?
      iii. How does the addition of the fusion network affect the inference time?

4.2. Research Objective

The objective of this research is to explore if the supervised learning based fusion of monocular and stereo methods of disparity estimates results in improved performance. The proposed architecture is expected perform better than the baseline monocular and stereo estimators. The inference time impact of the addition of the proposed fusion network shall be investigated, as that may pose a bottleneck in its application in small robotic and autonomous vehicles.

The above primary objective can be broken down into smaller milestones or sub-objectives. The first objective is to gain insight into the strengths and weaknesses of the currently existing monocular and stereo depth and disparity estimation solutions. This will guide the design decisions of the fusion network. The architecture design of a set of proposed fusion networks must be completed after the evaluation of prior research on fusion in dense estimation networks in the field of computer vision. Finally, the affect of the fusion is to be analyzed by comparing test results of the proposed architectures with comparable, but purely monocular and purely stereo solutions. In this way, the impact of the fusion module on the architecture performance can be analyzed by directly comparing it to the performance of the base networks.

Note that the goal is to produce a proof-of-concept fusion network, and the research is considered successful, as long as the proposed network offers improvements to the input networks. Thus, design choices that favor simplicity and ease of implementation or testing may be preferred over alternatives.

References

[1] Federal Aviation Administration. *Aeronautical Information Manual*. 4-4 ATC Clearances and Aircraft Separation. Apr. 2023.

[2] European Union Aviation Safety Agency. *Easy Access Rules for Unmanned Aircraft Systems*. https://www.easa.europa.eu/en/downloads/110913/en. 2022.

[3] Clive Alabaster. *Pulse Doppler radar: Principles, technology, applications*. pages 43–53. Edison, NJ: SciTech publ., 2012. ISBN: 9781613531518, 1613531516.

[4] BM Albaker and NA Rahim. "A survey of collision avoidance approaches for unmanned aerial vehicles". In: *2009 international conference for technical postgraduates (TECHPOS)*. IEEE. 2009, pp. 1–7.

[5] Filippo Aleotti et al. "Reversing the cycle: self-supervised deep stereo through enhanced monocular distillation". In: *European Conference on Computer Vision*. Springer. 2020, pp. 614–632.

[6] Shariq Farooq Bhat, Ibraheem Alhashim, and Peter Wonka. "Adabins: Depth estimation using adaptive bins". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2021, pp. 4009–4018.

[7] Marko Car et al. "Autonomous wind-turbine blade inspection using LiDAR-equipped unmanned aerial vehicle". In: *IEEE Access* 8 (2020), pp. 131380–131387.

[8] Francois Caron et al. "GPS/IMU data fusion using multisensor Kalman filtering: introduction of contextual aspects". In: *Information fusion* 7.2 (2006), pp. 221–230.

[9] John Singh Cheema. *The Mass Growth Factor–Snowball Effects in Aircraft Design*. 2020.

[10] Xiaotian Chen, Xuejin Chen, and Zheng-Jun Zha. "Structure-aware residual pyramid network for monocular depth estimation". In: *arXiv preprint arXiv:1907.06023* (2019).

[11] Xuelian Cheng et al. "Hierarchical neural architecture search for deep stereo matching". In: *Advances in Neural Information Processing Systems* 33 (2020), pp. 22158–22169.

[12] dr. Clark Borst. *AE4302 Avionics and Operations: Communication, Navigation & Surveillance*. 2020.

[13] GCHE De Croon et al. "Sky segmentation approach to obstacle avoidance". In: *2011 Aerospace Conference*. IEEE. 2011, pp. 1–16.

[14] GCHE De Croon et al. "The delfly". In: *Dordrecht: Springer Netherlands. doi* 10 (2016), pp. 978–94.

[15] Jia Deng et al. "Imagenet: A large-scale hierarchical image database". In: *2009 IEEE conference on computer vision and pattern recognition*. Ieee. 2009, pp. 248–255.

[16] Alexey Dosovitskiy et al. "An image is worth 16x16 words: Transformers for image recognition at scale". In: *arXiv preprint arXiv:2010.11929* (2020).

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

[18] David Eigen, Christian Puhrsch, and Rob Fergus. "Depth Map Prediction from a Single Image using a Multi-Scale Deep Network". In: *Advances in Neural Information Processing Systems*. Ed. by Z. Ghahramani et al. Vol. 27. Curran Associates, Inc., 2014. URL: https://proceedings.neurips.cc/paper/2014/file/7bccfde7714a1ebadf06c5f4cea752c1-Paper.pdf.

[19] Mohamed Elbanhawi et al. "Enabling technologies for autonomous MAV operations". In: *Progress in Aerospace Sciences* 91 (2017), pp. 27–52.

[20] Jose M Facil et al. "Deep single and direct multi-view depth fusion". In: *CoRR, abs* 1611 (2016), p. 30.

[21] Maxime Ferrera, Alexandre Boulch, and Julien Moras. “Fast stereo disparity maps refinement by fusion of data-based and model-based estimations”. In: *2019 International Conference on 3D Vision (3DV)*. IEEE. 2019, pp. 9–17.

[22] Michael Fonder and Marc Van Droogenbroeck. “Mid-Air: A multi-modal dataset for extremely low altitude drone flights”. In: *Conference on Computer Vision and Pattern Recognition Workshop (CVPRW)*. June 2019.

[23] Huan Fu et al. “Deep ordinal regression network for monocular depth estimation”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2018, pp. 2002–2011.

[24] Ravi Garg et al. “Unsupervised CNN for Single View Depth Estimation: Geometry to the Rescue”. In: *Computer Vision – ECCV 2016*. Ed. by Bastian Leibe et al. Cham: Springer International Publishing, 2016, pp. 740–756. ISBN: 978-3-319-46484-8.

[25] Clément Godard, Oisin Mac Aodha, and Gabriel J Brostow. “Unsupervised monocular depth estimation with left-right consistency”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017, pp. 270–279.

[26] Prof. dr. Guido de Croon. *AE4317 Autonomous Flight of Micro Air Vehicles: Introduction*. 2020.

[27] Kaiming He et al. “Deep residual learning for image recognition”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 770–778.

[28] Heiko Hirschmuller. “Stereo processing by semiglobal matching and mutual information”. In: *IEEE Transactions on pattern analysis and machine intelligence* 30.2 (2007), pp. 328–341.

[29] Sepp Hochreiter and Jürgen Schmidhuber. “Long short-term memory”. In: *Neural computation* 9.8 (1997), pp. 1735–1780.

[30] Mu Hu et al. “Penet: Towards precise and efficient image guided depth completion”. In: *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2021, pp. 13656–13662.

[31] Maximilian Jaritz et al. “Sparse and dense data with cnns: Depth completion and semantic segmentation”. In: *2018 International Conference on 3D Vision (3DV)*. IEEE. 2018, pp. 52–60.

[32] Alex Kendall et al. “End-to-end learning of geometry and context for deep stereo regression”. In: *Proceedings of the IEEE international conference on computer vision*. 2017, pp. 66–75.

[33] Iro Laina et al. “Deeper Depth Prediction with Fully Convolutional Residual Networks”. In: *2016 Fourth International Conference on 3D Vision (3DV)*. 2016, pp. 239–248. DOI: 10.1109/3DV.2016.32.

[34] Jiankun Li et al. “Practical stereo matching via cascaded recurrent network with adaptive correlation”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2022, pp. 16263–16272.

[35] Zhenyu Li et al. “BinsFormer: Revisiting Adaptive Bins for Monocular Depth Estimation”. In: *arXiv preprint arXiv:2204.00987* (2022).

[36] Zhenyu Li et al. “DepthFormer: Exploiting Long-Range Correlation and Local Information for Accurate Monocular Depth Estimation”. In: *arXiv preprint arXiv:2203.14211* (2022).

[37] Ze Liu et al. “Swin transformer: Hierarchical vision transformer using shifted windows”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2021, pp. 10012–10022.

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

[39] 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)*. IEEE. 2018, pp. 4796–4803.

[40] Michele Mancini et al. “Toward domain independence for learning-based monocular depth estimation”. In: *IEEE Robotics and Automation Letters* 2.3 (2017), pp. 1778–1785.

[41] Diogo Martins, Kevin Van Hecke, and Guido De Croon. “Fusion of stereo and still monocular depth estimates in a self-supervised learning context”. In: *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2018, pp. 849–856.

[42] Nikolaus Mayer et al. “A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 4040–4048.

[43] Tim G McGee, Raja Sengupta, and Karl Hedrick. “Obstacle detection for small autonomous aircraft using sky segmentation”. In: *Proceedings of the 2005 IEEE International Conference on Robotics and Automation*. IEEE. 2005, pp. 4679–4684.

[44] Mitsunori Mizumachi et al. “Robust sensing of approaching vehicles relying on acoustic cues”. In: *Sensors* 14.6 (2014), pp. 9546–9561.

[45] Chao Peng et al. “Large kernel matters–improve semantic segmentation by global convolutional network”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017, pp. 4353–4361.

[46] Peter Pinggera et al. “Know your limits: Accuracy of long range stereoscopic object measurements in practice”. In: *European conference on computer vision*. Springer. 2014, pp. 96–111.

[47] K Prazdny. “Egomotion and relative depth map from optical flow”. In: *Biological cybernetics* 36.2 (1980), pp. 87–102.

[48] Paolo Bellezza Quater et al. “Light Unmanned Aerial Vehicles (UAVs) for cooperative inspection of PV plants”. In: *IEEE Journal of Photovoltaics* 4.4 (2014), pp. 1107–1113.

[49] René Ranftl, Alexey Bochkovskiy, and Vladlen Koltun. “Vision transformers for dense prediction”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2021, pp. 12179–12188.

[50] Ashutosh Saxena, Sung Chung, and Andrew Ng. “Learning depth from single monocular images”. In: *Advances in neural information processing systems* 18 (2005).

[51] Chang Shu et al. “SideRT: A Real-time Pure Transformer Architecture for Single Image Depth Estimation”. In: *arXiv preprint arXiv:2204.13892* (2022).

[52] Mario Silvagni et al. “Multipurpose UAV for search and rescue operations in mountain avalanche events”. In: *Geomatics, Natural Hazards and Risk* 8.1 (2017), pp. 18–33.

[53] Minsoo Song, Seokjae Lim, and Wonjun Kim. “Monocular Depth Estimation Using Laplacian Pyramid-Based Depth Residuals”. In: *IEEE Transactions on Circuits and Systems for Video Technology* 31.11 (2021), pp. 4381–4393. DOI: 10.1109/TCSVT.2021.3049869.

[54] Minsoo Song, Seokjae Lim, and Wonjun Kim. “Monocular Depth Estimation Using Laplacian Pyramid-Based Depth Residuals”. In: *IEEE Transactions on Circuits and Systems for Video Technology* 31.11 (2021), pp. 4381–4393. DOI: 10.1109/TCSVT.2021.3049869.

[55] Vladimir Tankovich et al. “Hitnet: Hierarchical iterative tile refinement network for real-time stereo matching”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2021, pp. 14362–14372.

[56] Hengli Wang et al. “PVStereo: Pyramid voting module for end-to-end self-supervised stereo matching”. In: *IEEE Robotics and Automation Letters* 6.3 (2021), pp. 4353–4360.

[57] Qin Wang et al. “Domain adaptive semantic segmentation with self-supervised depth estimation”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2021, pp. 8515–8525.

[58] Shiyao Wang, Zhidong Deng, and Gang Yin. “An accurate GPS-IMU/DR data fusion method for driverless car based on a set of predictive models and grid constraints”. In: *Sensors* 16.3 (2016), p. 280.

[59] Junyuan Xie, Ross Girshick, and Ali Farhadi. “Deep3d: Fully automatic 2d-to-3d video conversion with deep convolutional neural networks”. In: *European conference on computer vision*. Springer. 2016, pp. 842–857.

[60] Guanglei Yang et al. *Transformer-Based Attention Networks for Continuous Pixel-Wise Prediction*. 2021. DOI: 10.48550/ARXIV.2103.12091. URL: https://arxiv.org/abs/2103.12091.

[61] Guanglei Yang et al. “Transformer-based attention networks for continuous pixel-wise prediction”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2021, pp. 16269–16279.

[62] Jawad N Yasin et al. “Unmanned aerial vehicles (uavs): Collision avoidance systems and approaches”. In: *IEEE access* 8 (2020), pp. 105139–105155.

[63] Fisher Yu and Vladlen Koltun. “Multi-scale context aggregation by dilated convolutions”. In: *arXiv preprint arXiv:1511.07122* (2015).

[64] Hengshuang Zhao et al. “Pyramid scene parsing network”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017, pp. 2881–2890.

[65] Yiran Zhong, Yuchao Dai, and Hongdong Li. “Self-supervised learning for stereo matching with self-improving ability”. In: *arXiv preprint arXiv:1709.00930* (2017).