Learning Stereo from Single Images

Jamie Watson[1] Oisin Mac Aodha[2] Daniyar Turmukhambetov[1]
Gabriel J. Brostow[1,3] Michael Firman[1]

[1]Niantic [2]University of Edinburgh [3]UCL

Abstract. Supervised deep networks are among the best methods for finding correspondences in stereo image pairs. Like all supervised approaches, these networks require ground truth data during training. However, collecting large quantities of accurate dense correspondence data is very challenging. We propose that it is unnecessary to have such a high reliance on ground truth depths or even corresponding stereo pairs. Inspired by recent progress in monocular depth estimation, we generate plausible disparity maps from single images. In turn, we use those flawed disparity maps in a carefully designed pipeline to generate stereo training pairs. Training in this manner makes it possible to convert any collection of single RGB images into stereo training data. This results in a significant reduction in human effort, with no need to collect real depths or to hand-design synthetic data. We can consequently train a stereo matching network from scratch on datasets like COCO, which were previously hard to exploit for stereo. Through extensive experiments we show that our approach outperforms stereo networks trained with standard synthetic datasets, when evaluated on KITTI, ETH3D, and Middlebury. Code to reproduce our results is available at https://github.com/nianticlabs/stereo-from-mono/.

Keywords: stereo matching, correspondence training data

1 Introduction

Given a pair of scanline rectified images, the goal of stereo matching is to estimate the per-pixel horizontal displacement (i.e. disparity) between the corresponding location of every pixel from the first view to the second, or vice versa. This is typically framed as a matching problem, where the current best performance is achieved by deep stereo networks e.g. [76,25,5,79]. These networks rely on expensive to obtain ground truth correspondence training data, for example captured with LiDAR scanners [20,21,54]. Synthetic data is a common alternative; given a collection of 3D scenes, large quantities of artificially rendered stereo pairs can be created with ground truth disparities [44,14,18,4]. Deployment in novel *real* scenes currently hinges on finetuning with additional correspondence data from that target domain. Assuming access to stereo pairs, a promising alternative could be to do self-supervised finetuning using image reconstruction losses and no ground truth correspondence. Unfortunately, self-supervised performance is still short of supervised methods for real domains [83,59,61].

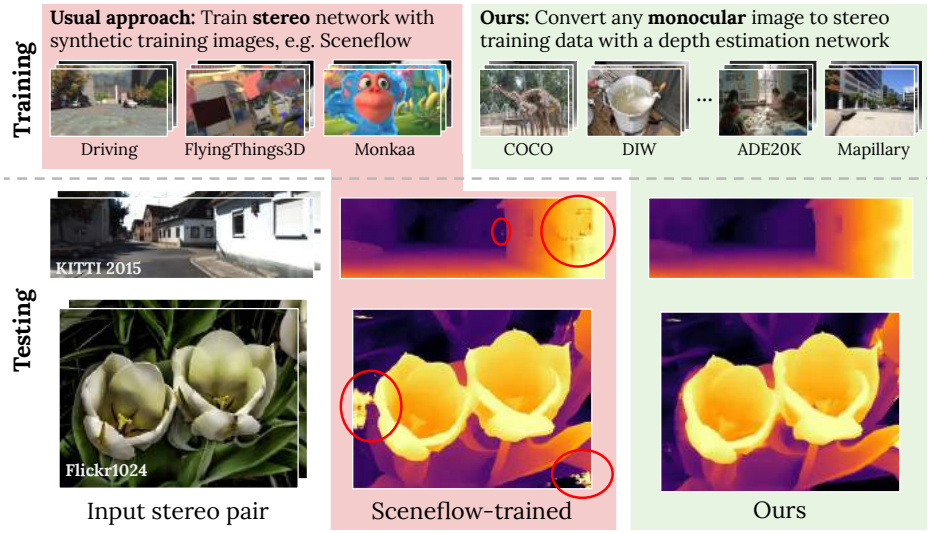


Fig. 1: Training stereo correspondence from monocular data. Deep stereo networks are typically trained on synthetic data, with per-dataset finetuning when ground truth disparity is available. We instead train on real images, which we convert to tuples of stereo training data using off-the-shelf monocular depth estimation. Our approach offers better generalization to new datasets, fewer visual artifacts, and improved scores.

We tackle the problem of generating training data for deep stereo matching networks. Recent works have shown that pretraining on large amounts of noisily labeled data improves performance on image classification [74,57,64,42]. A similar approach has not, to the best of our knowledge, been applied to training stereo matching networks. We introduce a fully automatic stereo data synthesis approach which only requires a collection of single images as input. In doing so, we open up the data available for deep stereo matching to any 2D image. Our approach uses a monocular depth estimation network to predict a disparity map for each training image. We refine this initial prediction before using it to synthesize a corresponding stereo pair from a new viewpoint, taking care to address issues associated with occlusions and collisions. Using natural images, as opposed to synthetic ones, ensures that the stereo network sees *realistic* textures and layouts, with *plausible* disparity maps during training. We show that networks trained from monocular data, or even low quality depth, provide an informative signal when generating training data. This is because the stereo network still has to learn how to perform dense matching between the real input image and our synthesized new view. In our experiments, we show that progressively better monocular depth yields gains in stereo matching accuracy. This suggests that our method will benefit 'for free' as future monocular depth networks, including self-supervised ones, improve further. We also show that increasing the amount of monocular derived training data also increases stereo performance. We present an example of a stereo network trained with our data compared to using conventional synthetic data in Fig. 1.

We make the following contributions:

1. A fully automatic pipeline for generating stereo training data from unstructured collections of single images, given a depth-from-color model.
2. A comparison to different approaches for stereo pair synthesis, showing that even simple approaches can produce useful stereo training data.
3. A detailed experimental evaluation across multiple domains with different stereo algorithms, showing that our generated training data enables better generalization to unseen domains than current synthetic datasets.

2 Related Work

Stereo Matching — Traditionally, matching between stereo image pairs was posed as an energy minimization problem consisting of separate, hand-designed, appearance matching and data smoothness terms e.g. [26,72,3]. [34,78] instead learned the appearance matching term while still retaining conventional stereo methods for the additional processing. Subsequently, fully end-to-end approaches replaced the entire conventional stereo matching pipeline [44,31]. Recent advances have focused on improving stereo network architectures and training losses. This has included innovations like spatial pyramids [5] and more sophisticated approaches for regularizing the generated cost volume [79,80,8]. In this work, we propose an approach for generating training data for deep stereo matching algorithms that is agnostic to the choice of stereo network. Our experiments across multiple different stereo networks show consistent improvements, independent of the underlying architecture used.

Just as recent works have adapted stereo models to help improve monocular depth estimation [24,41,62,71], in our work we leverage monocular models to improve stereo. This is related to concurrent work [1], where a monocular completion network is utilized to help self-supervised stereo. However, in contrast to these approaches, we do not require stereo pairs or ground truth disparity for data synthesis, but instead use monocular depth networks to generate training data for stereo networks.

Stereo Training Data — While deep stereo algorithms have proven very successful, the best performing methods are still fully supervised and thus require large quantities of stereo pairs with *ground truth* correspondence data at training time [21,54]. Acquiring this data for real world scenes typically involves specially designed depth sensing hardware, coupled with conventional intensity cameras for capturing scene appearance e.g. [20,21,54]. In addition to currently being laborious and expensive, these commonly used laser-based depth sensors are limited in the types of scenes they can operate in e.g. no direct sunlight.

Other potential sources of stereo data include static stereo cameras [73], internet videos [66], 3D movies [50], and multi-view images [38]. Crucially however, these images do not come with ground truth correspondence. Estimating disparity for these images is especially challenging due to issues such as negative disparity values in the case of 3D movies and extreme appearance changes over

time in multi-view image collections. These issues make it difficult to apply conventional stereo algorithms to process the data. It also poses a 'chicken and egg' problem, as we first need accurate stereo to extract reliable correspondences.

Synthetically rendered correspondence data is an attractive alternative as it leverages 3D graphics advances in place of specialized hardware. One of the first large-scale synthetic datasets for stereo matching consisted of simple 3D primitives floating in space [44]. Other synthetic datasets have been proposed to better match the statistics of real world domains, such as driving scenes [14,18,4] or indoor man-made environments [37,68]. With synthetic data, it is also possible to vary properties of the cameras viewing the scene (e.g. the focal length) so the trained models can be more robust to these variations at test time [80].

While dense correspondence networks can be trained from unrealistic synthetic data [13,43], the best performance is still achieved when synthetic training data is similar in appearance and depth to the target domain [80], and most networks still finetune on stereo pairs from the target domain for state-of-the-art performance e.g. [5,79,80]. This suggests that the goal of general stereo matching by training with just synthetic data has not yet been reached. It is possible to simulate stereo training data that matches the target domain, e.g. [62,14,80]. However, constructing varied scenes with realistic shape and appearance requires significant manual effort, and can be expensive to render. We perform experiments that show that our approach to generating training data generalizes better to new scenes, while requiring no manual effort to create.

Image Augmentation Based Training Data — An alternative to using computer generated synthetic data is to synthesize images by augmenting natural training images with pre-defined and known pixel-level transformations. The advantage of this approach is that it mostly retains the realistic appearance information of the input images. One simple method is to apply a random homography to an input image to create its corresponding matching pair [12]. Alternative methods include more sophisticated transformations such as pasting one image, or segmented foreground object, on top of another background image [13,43,29], random non-rigid deformations of foreground objects [30,58,35], and using supervised depth and normals to augment individual objects [2].

Augmentation-based approaches are largely heuristic and do not produce realistic correspondence data due to their inability to model plausible occlusions and depth. In contrast, we use off-the-shelf networks trained for monocular depth estimation to guide our image synthesis, so depths and occlusions are plausible. We perform experimental comparisons with existing image augmentation approaches and show superior performance with our method.

Domain Adaptation for Stereo — There is a large body of work that explores the problem of domain adaptation in stereo by addressing the domain gap between synthetic training data and real images. Existing approaches use additional supervision in the form of highly confident correspondences [59], apply iterative refinement [48], meta-learning [60], consistency checks [83,81], occlusion reasoning [36], or temporal information [61,69]. In the context of fully supervised

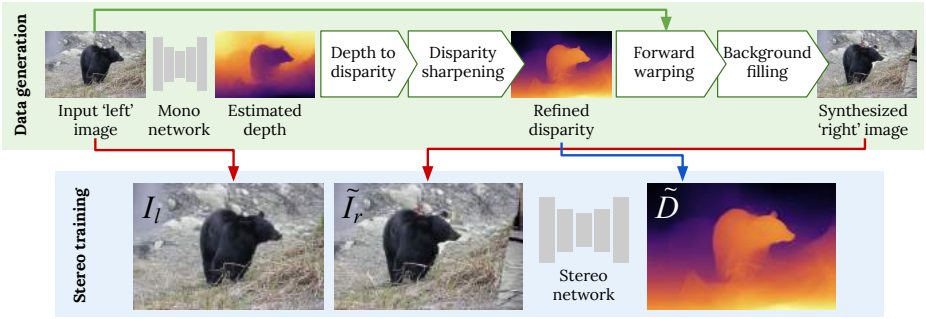


Fig. 2: Overview of our data generation approach. We use off-the-shelf monocular depth estimation to synthesize stereo training pairs from single images. We convert estimated depth to a sharpened disparity map, and then forward warp pixels from the input left image I_l to create a plausible and geometrically accurate right image $\tilde{I}_r$.

adaptation, [80] proposed a novel domain normalization layer to account for statistical differences between different data domains.

To address the domain adaptation problem we propose a method for automatically synthesizing large quantities of diverse stereo training data from single image collections. We show that our approach results in a large performance increase when testing on domains that have not been seen at all during training.

3 Method

Supervised training of deep stereo networks requires rectified pairs of color images I_l and I_r, together with a ground truth disparity map D.[1] We propose a method to create plausible tuples of training data $(I_l, \tilde{I}_r, \tilde{D})$ from collections of individual and unrelated input images. We use a pretrained monocular depth network to estimate a plausible depth map Z for an image I_l, which we then use to synthesize $\tilde{I}_r$ and $\tilde{D}$. This section describes in detail the important design decisions required at each step of this process. Our goal here is similar to works that synthesize novel views of a single image e.g. [9,85]. However, our aim is to synthesize a new view together with a geometrically correct depth map aligned with the input image, which is non-trivial to obtain from existing methods. We choose a simple and efficient algorithm for this based on pixel manipulation, which we found to be surprisingly effective.

3.1 Stereo Training Data from Monocular Depth

At test time, a monocular depth estimation network g takes as input a single color image, and for each pixel estimates the depth Z to the camera,

$$Z = g(I). \tag{1}$$

[1] For simplicity of notation we assume that disparity and depth maps are all aligned to the left input image I_l.

These monocular networks can be parameterized as deep neural networks that are trained with ground truth depth [16] or via self-supervision [19,22,84]. To train our stereo network we need to covert the estimated depth Z to a disparity map $\tilde{D}$. To achieve our aim of generalizable stereo matching, we want to simulate stereo pairs with a wide range of baselines and focal lengths. We achieve this by converting depth to disparity with $\tilde{D} = \frac{s\, Z_{\max}}{Z}$, where s is a randomly sampled (uniformly from $[d_{\min}, d_{\max}]$) scaling factor which ensures the generated disparities are in a plausible range.

Our goal is to synthesize a right image $\tilde{I}_r$ from the input 'left' image I_l and the predicted disparity $\tilde{D}$. The disparity map and the color image are both aligned to the left image I_l, so backward warping [55,19] is not appropriate. Instead, we use $\tilde{D}$ to synthesize a stereo pair $\tilde{I}_r$ via *forward* warping [55]. For each pixel i in I_l, we translate it $\tilde{D}^i$ pixels to the left, and perform interpolation of the warped pixel values to obtain $\tilde{I}_r$. Due to the nature of forward warping, $\tilde{I}_r$ will contain missing pixel artifacts due to occluded regions, and in some places collisions will occur when multiple pixels will land at the same location. Additionally, monocular networks often incorrectly predict depths around depth discontinuities, resulting in unrealistic depth maps. Handling these issues is key to synthesising realistic stereo pairs, as we demonstrate in our ablation experiments.

3.2 Handling Occlusion and Collisions

Naively synthesizing images based on the estimated disparity will result in two main sources of artifacts: occlusion holes and collisions. Occluded regions are pixels in one image in a stereo pair which are not visible in the corresponding image [67]. In the forward warping process, pixels in $\tilde{I}_r$ with no match in I_l manifest themselves as *holes* in the reconstructed image $\tilde{I}_r$. While the synthesized image gives an accurate stereo pair for I_l, the blank regions result in unnatural artifacts. We can increase the realism of the synthesized image by filling the missing regions with a texture from a randomly selected image I_b from our training set, following [15]. We perform color transfer between I_l and I_b using [51] to obtain $\hat{I}_b$. We subsequently set all missing pixels in $\tilde{I}_r$ to the values of $\hat{I}_b$ at the corresponding positions. This results in textures from natural images being copied into the missing regions of $\tilde{I}_r$. Separately, *collisions* are pixels that appear in I_l but are not in $\tilde{I}_r$, and thus multiple pixels from I_l can map to the same point in $\tilde{I}_r$. For collisions, we select the pixel from I_l which has the greater disparity value since these pixels are closer and should be visible in both images.

3.3 Depth Sharpening

As opposed to realistic sharp discontinuities, monocular depth estimation networks typically produce blurry edges at depth discontinuities [49]. When forward warping during image synthesis this leads to 'flying pixels' [52], where individual pixels are warped to empty regions between two depth surfaces. In Fig. 3 we observe this phenomenon for the monocular depth network of [50]. Not correcting this problem would result in a collection of synthesized training images where the stereo matching algorithm is expected to match pixels in I_l to individual,

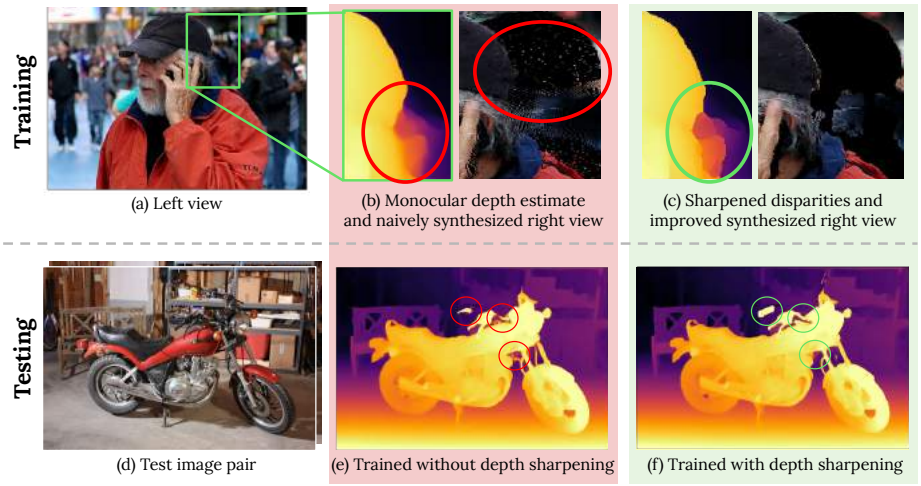


Fig. 3: Depth map sharpening. Depth maps from monocular depth networks typically have 'blurry' edges, which manifest themselves as flying pixels in $\tilde{I}_r$ (b). Our depth sharpening method reduces the effect of flying pixels (c). Models trained with depth sharpening have better scores (Table 4) and fewer visual artifacts; (e) versus (f).

isolated pixels in $\tilde{I}_r$; a highly unrealistic test-time scenario. To address flying pixels in our synthesized $\tilde{I}_r$ images we perform *depth sharpening*. We identify flying pixels as those for which the disparity map has a Sobel edge filter response of greater than 3 [56,27]. These pixels are then assigned the disparity of the nearest 'non-flying' pixel in image space. We use this corrected disparity map for both image sampling and when training our stereo networks.

3.4 Implementation Details

Training Details — Unless otherwise stated, we use the widely-used PSMNet hourglass network variant [5] for stereo matching, using the official architecture and training code. We train all models with a batch size of 2 for 250,000 steps, equivalent to 7 epochs of the Sceneflow dataset [44]. With the exception of some baselines, we do not use any synthetic data when training our models. We follow [5] in using a learning rate of 0.001 for all experiments. Unless otherwise stated, we perform monocular depth estimation using MiDaS [50], a network with a ResNeXt-101 [75] backbone, trained on a mixture of monocular and stereo datasets including 3D movies. Their training depths were generated by running a supervised optical flow network [28] on their stereo training pairs.

Our networks are set to predict a maximum disparity of 192. We set $d_{\min} = 50$ and $d_{\max} = 225$, which produces training images with a diverse range of baselines and focal lengths, including stereo pairs with disparity ranges outside the network's modeling range for further robustness to test time scenarios. During training, we take crops of size 608×320. If images are smaller than these dimensions (or significantly larger, e.g. Mapillary [45]), we isotropically resize to match

the constraining dimension. Additionally, we follow PSMNet [5] by performing ImageNet [11] normalization on all images.

Training Images — Our method allows us to train using any color images. To maximise the ability of our algorithm to learn to adapt to different test domains, we train models on a combination of varied single image datasets which we call the 'Mono for Stereo' dataset, or **MfS**. MfS comprises all the training images (without depth or semantic labels) from COCO 2017 [40], Mapillary Vistas [45], ADE20K [82], Depth in the Wild [6], and DIODE [63] (see Fig. 1 for examples). This results in 597,727 training images, an order of magnitude more than Sceneflow's 35,454 training pairs, from which we use a random subset of 500,000, unless stated otherwise.

Color Augmentation — Cameras in a stereo pair are typically nominally identical. However, each image in a pair may have slightly different white balancing, lens geometry, glare, and motion blur. To account for these differences when training stereo networks we augment $\tilde{I}_r$ with pixel-level noise with standard deviation 0.05, and we randomly adjust contrast, brightness, saturation, and hue with ranges of ± 0.2, ± 0.2, ± 0.2, and ± 0.01. Finally, with 50% probability, we add Gaussian blur to $\tilde{I}_r$ with kernel size $\sigma \sim \text{Unif}\,[0, 1]$. Interestingly, we observe that these simple augmentations also significantly improve the performance of the baseline stereo models trained on Sceneflow [44] alone.

4 Experiments

In this section we present experimental results to validate our approach to stereo data generation. Our experiments show that:

1. Our approach produces better and more general stereo matching networks compared to training on alternative sources of data, such as synthetic scenes
2. We are robust to errors in monocular depth predictions, and we perform well regardless of the source of monocular training data
3. Our design decisions are sensible, validated through ablation of our method
4. Our performance gains are agnostic to the stereo architecture used
5. As the amount of generated data increases, so does stereo performance
6. Finetuning models trained with our approach compared to synthetic data with ground truth disparity results in better performance.

4.1 Evaluation Datasets and Metrics

We evaluate multiple stereo models on a variety of datasets to demonstrate generality of our approach. Our test datasets are: (i) **KITTI 2012** and **2015** [21,20]: real-world driving scenes with LiDAR ground truth, (ii) **ETH3D** [54]: grayscale stereo pairs from indoor and outdoor environments, again with LiDAR ground truth, and (iii) **Middlebury** [53]: high-resolution stereo pairs with structured light ground truth. Middlebury and ETH3D are particularly compelling for demonstrating general-purpose stereo matching as their training data is very limited, with just 15 and 27 pairs respectively. Note that we never train on any

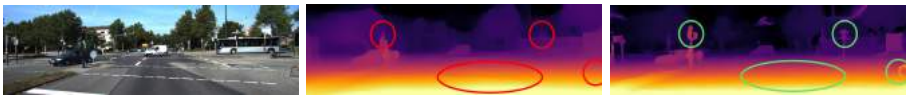
Table 1: Our approach outperforms alternative stereo training data generation methods. Each row represents a single PSMNet trained with a different data synthesis approach, without any dataset-specific finetuning. We can see that simple synthesis methods still produce valuable training data. However, our approach in the bottom row, which incorporates a monocular depth network [50], outperforms even synthetic Sceneflow data. Numbers here are directly comparable to Table 3, showing we beat the baselines no matter what depth network or training data we use.

Synthesis approach	Training data	KITTI '12		KITTI '15		Middlebury		ETH3D	
		EPE	>3px	EPE	>3px	EPE	>2px	EPE	>1px
Affine warps (e.g. [12])	MfS	3.74	14.78	2.33	14.94	21.50	61.19	1.28	24.21
Random pasted shapes (e.g. [43])	MfS	2.45	11.89	1.49	7.57	11.38	40.37	0.77	14.19
Random superpixels	MfS	1.28	6.08	1.23	5.90	12.02	32.92	2.01	13.95
Synthetic	Sceneflow [44]	1.04	5.76	1.19	5.80	9.42	34.99	1.25	13.04
SVSM selection module [41]	MfS	1.04	5.66	1.10	5.47	8.45	31.34	0.57	11.61
Ours	MfS	**0.95**	**4.43**	**1.04**	**4.75**	**6.33**	**26.42**	**0.53**	**8.17**

of these datasets, with the exception of KITTI finetuning experiments in Section 4.6. For Middlebury, we evaluate on the 15 training images provided for the official stereo benchmark. We report results on the full training set for the two view benchmark for ETH3D. For both KITTI 2012 and 2015 we evaluate on the validation set from [5], and provide the image indices in the supplementary material. For all datasets we report end-point-error (EPE) and the thresholded error rate: the % of pixels with predicted disparity more than τ pixels from the ground truth, e.g. >3px. Unless otherwise stated, we evaluate every pixel for which there is a valid ground truth disparity value (i.e. we include occluded pixels). We only report scores with the most commonly reported value of τ for each dataset. Finally, we show qualitative results on **Flickr1024** [70], a diverse dataset of stereo pairs originally captured by internet users for stereoscopic viewing.

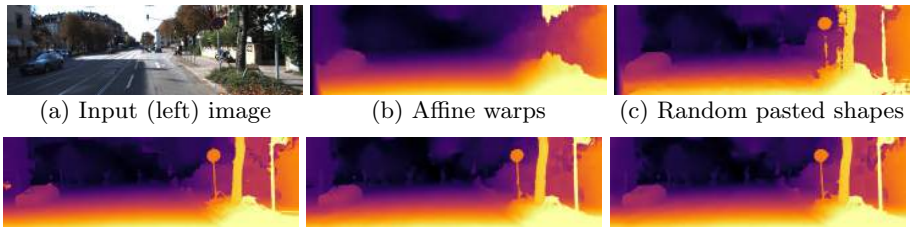
4.2 Comparison to Alternative Data Generation Methods

In our first experiment, we compare our approach to alternative baseline methods for stereo data generation. To do this, we train a PSMNet stereo network using data generated by each of the baselines and compare the network's stereo estimation performance on three held out datasets. We compare to: (i) **Affine warps**: warps of single images similar to [12], (ii) **Random pasted shapes**: our implementation of the optical flow data generation method of [43], adapted for stereo training, (iii) **Random superpixels**: where we initialise a disparity map as a plane, and assign disparity values drawn uniformly from $[d_{\min}, d_{\max}]$ to randomly selected image superpixels [17], (iv) **Synthetic**: 35K synthetic image pairs from Sceneflow [44], and (v) **SVSM selection module**: our implementation of the selection module from [41], where $\tilde{I}_r$ is the weighted sum of shifted left images, with weightings derived from the depth output of [50]. With the exception of the synthetic baseline, all results use the MfS dataset to synthesize stereo pairs. Detailed descriptions are in the supplementary material. Table 1 shows that our fully automatic approach outperforms all of these methods. Qualitative results are in Figs. 1, 6 and 5.



(a) Input (left) image (b) Mono estimation [50] (c) Our stereo estimation

Fig. 4: We can recover from errors in monocular depth estimation. Problems present in monocular depths (b), such as missing objects and uneven ground do not transfer to our eventual stereo predictions (c).



(a) Input (left) image (b) Affine warps (c) Random pasted shapes

(d) Random superpixels (e) Ours (PSMNet) (f) Ours (GANet)

Fig. 5: Comparison to baseline data generation methods. We show that simple image synthesis can be a surprisingly effective method for generating stereo training data e.g. (c) and (d). However, our approach produces sharper and cleaner disparities, whether using PSMNet (e) or the larger GANet (f).

4.3 Model Architecture Ablation

In Table 2 we use our data synthesis approach to train three different stereo matching networks. With the exception of iResNet [39], we used the same architectures and training code provided by the original authors, and trained each model for 250,000 iterations from scratch. For fair comparison, we also trained the same models without our data but instead with Sceneflow using the same color augmentations described in Section 3.4. Interestingly, we observe that this results in an improvement over the publicly released model snapshots. We see that our approach consistently outperforms Sceneflow training irrespective of the stereo network used.

Table 2: Our approach is agnostic to stereo model architecture. We compare three diverse stereo architectures, from the lightweight iResnet [39], through to the computationally expensive, but state-of-the-art, GANet [79]. Rows marked with † use the publicly released model weights. We see that our synthesized data (MfS) consistently performs better than training with the synthetic Sceneflow dataset.

Architecture	Training data	KITTI '12		KITTI '15		Middlebury		ETH3D	
		EPE	>3px	EPE	>3px	EPE	>2px	EPE	>1px
iResnet [39]	Sceneflow	1.07	6.40	**1.15**	5.98	12.13	41.65	1.57	17.90
iResnet [39]	MfS	**0.93**	**5.38**	**1.15**	**5.75**	**8.52**	**33.36**	**0.61**	**11.65**
PSMNet [5]†	Sceneflow	6.06	25.38	6.04	25.07	17.54	54.60	1.21	19.52
PSMNet [5]	Sceneflow	1.04	5.76	1.19	5.80	9.42	34.99	1.25	13.04
PSMNet [5]	MfS	**0.95**	**4.43**	**1.04**	**4.75**	**6.33**	**26.42**	**0.53**	**8.17**
GANet [79]†	Sceneflow	1.30	7.97	1.51	9.52	11.88	37.34	0.48	8.19
GANet [79]	Sceneflow	0.96	5.24	1.14	5.43	9.81	32.20	0.48	9.45
GANet [79]	MfS	**0.81**	**4.31**	**1.02**	**4.56**	**5.66**	**24.41**	**0.43**	**6.62**

Table 3: Stereo performance improves with better monocular depths. Here we train PSMNet with MfS and our data synthesis approach without any dataset-specific finetuning. Each row uses a different monocular depth network, each with a different architecture and source of supervision. As expected, better monocular depth gives improved stereo matching, but all are competitive compared to synthetic training data. Below each row we show the % difference relative to the Sceneflow baseline.

Monocular model	Monocular training data	KITTI '12		KITTI '15		Middlebury		ETH3D	
		EPE	>3px	EPE	>3px	EPE	>2px	EPE	>1px
DiW [6]	Human labelling	0.93	5.08	1.13	6.05	9.96	32.96	0.63	12.23
		-10.6	*-11.8*	*-5.0*	*4.3*	*5.7*	*-5.8*	*-49.6*	*-6.2*
Monodepth2 (M) [23]	KITTI Monocular	0.92	5.01	1.10	5.11	9.41	30.76	0.60	12.56
		-11.5	*-13.0*	*-7.6*	*-11.9*	*-0.1*	*-12.1*	*-52.0*	*-3.7*
Megadepth [38]	SfM reconstructions	**0.89**	**4.29**	1.07	4.96	7.66	28.86	0.54	10.41
		-14.4	*-25.5*	*-10.1*	*-14.5*	*-18.7*	*-17.5*	*-56.8*	*-20.2*
MiDaS [50]	3D Movies + others	0.95	4.43	**1.04**	**4.75**	**6.33**	**26.42**	**0.53**	**8.17**
		-8.7	*-23.1*	*-12.6*	*-18.1*	*-32.8*	*-24.5*	*-57.6*	*-37.3*
Sceneflow baseline		1.04	5.76	1.19	5.80	9.42	34.99	1.25	13.04

4.4 Comparing Different Monocular Depth Networks

The majority of our experiments are performed using monocular depth from MiDaS [50], which itself is trained with a variety of different datasets e.g. flow derived pseudo ground truth and structured light depth. In Table 3 we show that our approach still produces competitive stereo performance when we use alternative monocular networks that have been trained using much *weaker* supervision. This includes using monocular video only self-supervision on the KITTI dataset (Monodepth2 (M) [23]), pairwise ordinal human annotations (DiW [6]) and multi-view image collections (Megadepth [38]). It is worth noting that even though Megadepth [38] training depth is computed from unstructured image collections, without using any trained correspondence matching, using it to synthesize training data still outperforms the synthetic Sceneflow baseline. This is likely due to the fact that stereo matching is fundamentally different from monocular depth estimation, and even potentially unrealistic disparity information can help the stereo network learn how to better match pixels. While the recent MiDaS [50] performs best overall, we observe a consistent performance increase in stereo estimation with better monocular depth. This is very encouraging, as it indicates that our performance may only get better in future as monocular depth networks improve. We also show qualitatively in Fig. 4 how our stereo predictions can recover from errors made by monocular models.

4.5 Ablating Components of Our Method

In Table 4 we train stereo networks with data synthesized with and without depth sharpening and background texture filling (i.e. I_b). For the case where we do not use background texture filling we simply leave black pixels. We observe that disabling these components results in worse stereo performance compared to the full pipeline, with Fig. 3 demonstrating the qualitative impact of depth

Table 4: Depth sharpening and background filling helps. Disabling depth sharpening and background filling hurts PSMNet stereo performance when trained on MfS.

		KITTI '12		KITTI '15		Middlebury		ETH3D	
Sharpening	Background filling	EPE	>3px	EPE	>3px	EPE	>2px	EPE	>1px
✗	✓	1.02	4.75	1.06	4.76	7.57	28.16	**0.47**	8.21
✓	✗	0.96	4.81	1.10	5.27	7.11	27.22	0.68	8.70
✓	✓	**0.95**	**4.43**	**1.04**	**4.75**	**6.33**	**26.42**	0.53	**8.17**

sharpening. This highlights the importance of realistic image synthesis. A more sophisticated warping approach [46] or inpainting with a neural network [77] should increase the realism and is left as future work. We instead opt for a simple warping and background filling approach for computational efficiency, which allows us to generate new views online during training.

4.6 Adapting to the Target Domain

Our method allows us to train a generalizable stereo network which outperforms baselines. However, in cases where we have data from the target domain, e.g. in the form of single unpaired images, stereo pairs, or stereo pairs with ground truth LiDAR. We compare adaptation with each of these data sources in Table 5. First, we train on stereo pairs generated from the **KITTI left images** with our method, using the 29K KITTI images from the 33 scenes not included in the KITTI 2015 stereo training set and monocular predictions from [50]. For **KITTI self-supervised** finetuning, we follow the method proposed in [24] using the stereo image pairs from the same 33 scenes. Finally, for **KITTI LiDAR** we finetune on 160 training images and point clouds (following [5]) for 200 epochs. We give results for unoccluded pixels (*Noc*) and all pixels (*All*); see [5] for more details of the metrics. We outperform Sceneflow pretraining in all cases.

4.7 Varying the Amount of Training Data

In Table 6 we train a PSMNet stereo network by varying the amount of training images from our MfS dataset while keeping the monocular network (MiDaS [50]) fixed. We observe that as the number of training images increases so does stereo

Table 5: KITTI 2015 finetuning. All trained by us on PSMNet. Our data generation approach, when finetuned on KITTI LiDAR data (last row), results in superior performance when compared to Sceneflow pretraining and KITTI finetuning (4^{th} row). †Sceneflow models cannot be finetuned on single KITTI images without our method.

Pretraining	Finetuning method	EPE Noc	>3px Noc	EPE All	>3px All
Sceneflow	None	1.18	5.62	1.19	5.80
Sceneflow	KITTI left images	†	†	†	†
Sceneflow	KITTI self-supervised	1.03	4.90	1.05	5.07
Sceneflow	KITTI LiDAR	0.73	2.21	0.74	2.36
Ours with MfS	None	1.02	4.57	1.04	4.75
Ours with MfS	KITTI left images	1.00	4.44	1.01	4.58
Ours with MfS	KITTI self-supervised	1.00	4.57	1.01	4.71
Ours with MfS	KITTI LiDAR	0.68	2.05	0.69	2.15

Table 6: Performance improves with more synthesized images. Here we train PSMNet with varying numbers of images from our MfS dataset using our synthesis approach, without any finetuning to the test datasets. As expected, more training data improves the stereo network, even outperforming synthetic data. Note that all models were trained for 250K steps. Below each results row we show the % difference relative to the Sceneflow baseline.

Monocular model	Dataset	# of images	KITTI '12		KITTI '15		Middlebury		ETH3D	
			EPE	>3px	EPE	>3px	EPE	>2px	EPE	>1px
Megadepth [38]	MfS	35,000	0.88	4.47	1.25	5.52	9.31	28.13	0.59	12.24
			-15.4	*-22.4*	*-3.4*	*-4.8*	*-1.2*	*-19.6*	*-52.8*	*-6.1*
Megadepth [38]	MfS	500,000	0.89	**4.29**	1.07	4.96	7.66	28.86	0.54	10.41
			-14.4	*-25.5*	*-10.1*	*-14.5*	*-18.7*	*-17.5*	*-56.8*	*-20.2*
MiDaS [50]	MfS	35,000	**0.87**	4.35	1.09	5.06	6.77	28.86	0.56	8.18
			-16.3	*-24.5*	*-8.4*	*-12.8*	*-28.1*	*-17.5*	*-55.2*	*-29.6*
MiDaS [50]	MfS	500,000	0.95	4.43	**1.04**	**4.75**	**6.33**	**26.42**	**0.53**	**8.17**
			-8.7	*-23.1*	*-12.6*	*-18.1*	*-32.8*	*-24.5*	*-57.6*	*-37.3*
Sceneflow baseline		35,454	1.04	5.76	1.19	5.80	9.42	34.99	1.25	13.04

performance. Even in the smallest data regime (35K images), which is comparable in size to Sceneflow [44], we still outperform training on the synthetic data alone in all metrics. This indicates that it is not just the power of the monocular network that drives our performance, but the large and varied image set that we make available to stereo training.

5 Discussion

Many of our results used the monocular depth estimation network from [50], which was trained on a mixture of datasets including stereo pairs, which in turn had their depth estimated using an optical flow network trained on synthetic data [28]. However, experiments in Table 3 show we perform well regardless of the source of data used to train the underlying monocular estimator, with good predictions with depth estimation networks trained on SfM reconstructions [38] or monocular video sequences [23]. In fact, even the naive 'disparity estimation' methods in Table 1 generate reasonable stereo training data. Furthermore, we observe in Table 6 that as we increase the amount of training images our stereo performance improves. This indicates the power of the new additional training data that our approach makes available for stereo training.

6 Conclusion

We presented a fully automatic pipeline for synthesizing stereo training data from single monocular images. We showed that our approach is robust to the source of monocular depth and results in a significant increase in test-time generalization performance for multiple different stereo algorithms. Importantly, our approach enables the possibility of converting any monocular image into stereo training data, opening up the door to using existing large image collections to improve stereo estimation. In future we intend to explore different ways to automatically estimate the quality of our synthesized data [7] and approaches for performing

additional augmentations during training [10]. Another interesting question is if we can use our improved stereo networks to train better monocular depth networks, e.g. by providing sparse depth cues [33,71,62], which in turn could be used to further improve stereo with our method.

Acknowledgements — Large thanks to Aron Monszpart for help with baseline code, to Grace Tsai for feedback, and Sara Vicente for suggestions for baseline implementations. Also thanks to the authors who shared models and images.

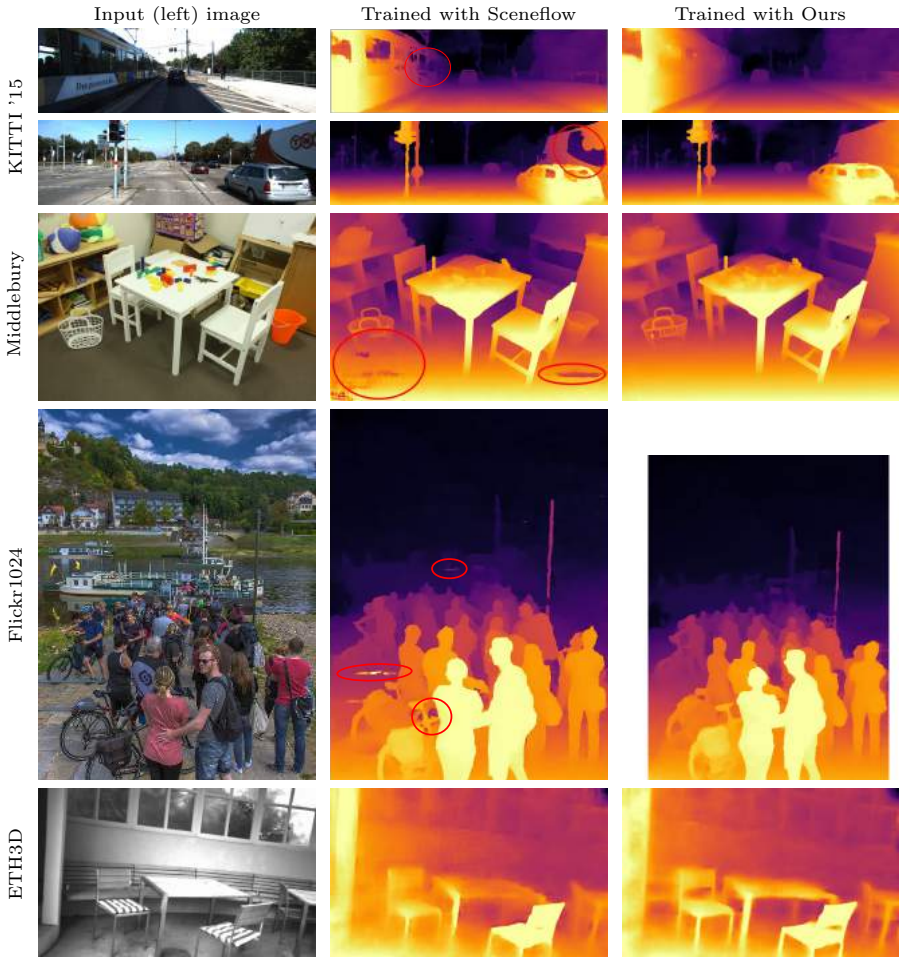


Fig. 6: Qualitative Results. Comparing our synthetic data generation approach with Sceneflow, using a PSMNet model trained on both. Our predictions benefit from semantically plausible depths at training time and have fewer artifacts.

References

1. Aleotti, F., Tosi, F., Zhang, L., Poggi, M., Mattoccia, S.: Reversing the cycle: self-supervised deep stereo. In: ECCV (2020) 3
2. Alhaija, H.A., Mustikovela, S.K., Geiger, A., Rother, C.: Geometric image synthesis. In: ACCV (2018) 4
3. Bleyer, M., Rhemann, C., Rother, C.: PatchMatch stereo - Stereo matching with slanted support windows. In: BMVC (2011) 3
4. Cabon, Y., Murray, N., Humenberger, M.: Virtual KITTI 2. arXiv:2001.10773 (2020) 1, 4
5. Chang, J.R., Chen, Y.S.: Pyramid stereo matching network. In: CVPR (2018) 1, 3, 4, 7, 8, 9, 10, 12, 27, 28
6. Chen, W., Fu, Z., Yang, D., Deng, J.: Single-image depth perception in the wild. In: NeurIPS (2016) 8, 11
7. Chen, W., Qian, S., Deng, J.: Learning single-image depth from videos using quality assessment networks. In: CVPR (2019) 13
8. Cheng, X., Wang, P., Yang, R.: Learning depth with convolutional spatial propagation network. PAMI (2019) 3
9. Choi, I., Gallo, O., Troccoli, A., Kim, M.H., Kautz, J.: Extreme view synthesis. In: ICCV (2019) 5
10. Cubuk, E.D., Zoph, B., Mane, D., Vasudevan, V., Le, Q.V.: Autoaugment: Learning augmentation strategies from data. In: CVPR (2019) 14
11. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: ImageNet: A Large-Scale Hierarchical Image Database. In: CVPR (2009) 8
12. DeTone, D., Malisiewicz, T., Rabinovich, A.: SuperPoint: Self-supervised interest point detection and description. In: CVPR Deep Learning for Visual SLAM Workshop (2018) 4, 9, 23
13. Dosovitskiy, A., Fischer, P., Ilg, E., Häusser, P., Hazırbaş, C., Golkov, V., v.d. Smagt, P., Cremers, D., Brox, T.: FlowNet: Learning optical flow with convolutional networks. In: ICCV (2015) 4
14. Dosovitskiy, A., Ros, G., Codevilla, F., Lopez, A., Koltun, V.: CARLA: An open urban driving simulator. In: CoRL (2017) 1, 4
15. Dwibedi, D., Misra, I., Hebert, M.: Cut, paste and learn: Surprisingly easy synthesis for instance detection. In: ICCV (2017) 6
16. Eigen, D., Puhrsch, C., Fergus, R.: Depth map prediction from a single image using a multi-scale deep network. In: NeurIPS (2014) 6
17. Felzenszwalb, P.F., Huttenlocher, D.P.: Efficient graph-based image segmentation. IJCV (2004) 9, 24
18. Gaidon, A., Wang, Q., Cabon, Y., Vig, E.: Virtual worlds as proxy for multi-object tracking analysis. In: CVPR (2016) 1, 4
19. Garg, R., BG, V.K., Carneiro, G., Reid, I.: Unsupervised CNN for single view depth estimation: Geometry to the rescue. In: ECCV (2016) 6
20. Geiger, A., Lenz, P., Stiller, C., Urtasun, R.: Vision meets robotics: The KITTI dataset. International Journal of Robotics Research (2013) 1, 3, 8
21. Geiger, A., Lenz, P., Urtasun, R.: Are we ready for autonomous driving? the KITTI vision benchmark suite. In: CVPR (2012) 1, 3, 8
22. Godard, C., Mac Aodha, O., Brostow, G.J.: Unsupervised monocular depth estimation with left-right consistency. In: CVPR (2017) 6
23. Godard, C., Mac Aodha, O., Firman, M., Brostow, G.J.: Digging into self-supervised monocular depth estimation. In: ICCV (2019) 11, 13, 28

24. Guo, X., Li, H., Yi, S., Ren, J., Wang, X.: Learning monocular depth by distilling cross-domain stereo networks. In: ECCV (2018) 3, 12
25. Guo, X., Yang, K., Yang, W., Wang, X., Li, H.: Group-wise correlation stereo network. In: CVPR (2019) 1
26. Hirschmuller, H.: Stereo processing by semiglobal matching and mutual information. PAMI (2007) 3
27. Hu, J., Ozay, M., Zhang, Y., Okatani, T.: Revisiting single image depth estimation: Toward higher resolution maps with accurate object boundaries. In: WACV (2019) 7
28. Ilg, E., Mayer, N., Saikia, T., Keuper, M., Dosovitskiy, A., Brox, T.: FlowNet 2.0: Evolution of optical flow estimation with deep networks. In: CVPR (2017) 7, 13
29. Janai, J., Guney, F., Ranjan, A., Black, M., Geiger, A.: Unsupervised learning of multi-frame optical flow with occlusions. In: ECCV (2018) 4
30. Kanazawa, A., Jacobs, D.W., Chandraker, M.: WarpNet: Weakly supervised matching for single-view reconstruction. In: CVPR (2016) 4
31. Kendall, A., Martirosyan, H., Dasgupta, S., Henry, P., Kennedy, R., Bachrach, A., Bry, A.: End-to-end learning of geometry and context for deep stereo regression. In: ICCV (2017) 3
32. KITTI: Kitti stereo evaluation 2015 server. http://www.cvlibs.net/datasets/kitti/eval_scene_flow.php?benchmark=stereo, accessed: 2020-03-10 20
33. Klodt, M., Vedaldi, A.: Supervising the new with the old: learning SfM from SfM. In: ECCV (2018) 14
34. Ladickỳ, L., Häne, C., Pollefeys, M.: Learning the matching function. arXiv:1502.00652 (2015) 3
35. Le, H.A., Nimbhorkar, T., Mensink, T., Baslamisli, A.S., Karaoglu, S., Gevers, T.: Unsupervised generation of optical flow datasets. arXiv:1812.01946 (2018) 4
36. Li, A., Yuan, Z.: Occlusion aware stereo matching via cooperative unsupervised learning. In: ACCV (2018) 4
37. Li, W., Saeedi, S., McCormac, J., Clark, R., Tzoumanikas, D., Ye, Q., Huang, Y., Tang, R., Leutenegger, S.: InteriorNet: Mega-scale multi-sensor photo-realistic indoor scenes dataset. In: BMVC (2018) 4
38. Li, Z., Snavely, N.: MegaDepth: Learning single-view depth prediction from internet photos. In: CVPR (2018) 3, 11, 13, 28, 31, 32
39. Liang, Z., Feng, Y., Guo, Y., Liu, H., Chen, W., Qiao, L., Zhou, L., Zhang, J.: Learning for disparity estimation through feature constancy. In: CVPR (2018) 10, 27
40. Lin, T.Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., Zitnick, C.L.: Microsoft COCO: Common objects in context. In: ECCV (2014) 8
41. Luo, Y., Ren, J., Lin, M., Pang, J., Sun, W., Li, H., Lin, L.: Single view stereo matching. In: CVPR (2018) 3, 9, 25
42. Mahajan, D., Girshick, R., Ramanathan, V., He, K., Paluri, M., Li, Y., Bharambe, A., van der Maaten, L.: Exploring the limits of weakly supervised pretraining. In: ECCV (2018) 2
43. Mayer, N., Ilg, E., Fischer, P., Hazirbas, C., Cremers, D., Dosovitskiy, A., Brox, T.: What makes good synthetic training data for learning disparity and optical flow estimation? IJCV (2018) 4, 9, 23
44. Mayer, N., Ilg, E., Hausser, P., Fischer, P., Cremers, D., Dosovitskiy, A., Brox, T.: A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation. In: CVPR (2016) 1, 3, 4, 7, 8, 9, 13, 19
45. Neuhold, G., Ollmann, T., Rota Bulò, S., Kontschieder, P.: The Mapillary Vistas Dataset for semantic understanding of street scenes. In: ICCV (2017) 7, 8

46. Niklaus, S., Liu, F.: Softmax splatting for video frame interpolation. In: CVPR (2020) 12
47. Ounsworth, M.: Algorithm to generate a random 2D polygon. https://stackoverflow.com/a/25276331/279858, accessed: 2020-03-10 24
48. Pang, J., Sun, W., Yang, C., Ren, J., Xiao, R., Zeng, J., Lin, L.: Zoom and learn: Generalizing deep stereo matching to novel domains. In: CVPR (2018) 4
49. Ramamonjisoa, M., Lepetit, V.: SharpNet: Fast and accurate recovery of occluding contours in monocular depth estimation. In: ICCV Workshops (2019) 6
50. Ranftl, R., Lasinger, K., Hafner, D., Schindler, K., Koltun, V.: Towards robust monocular depth estimation: Mixing datasets for zero-shot cross-dataset transfer. arXiv:1907.01341 (2019) 3, 6, 7, 9, 10, 11, 12, 13, 22, 25, 28, 31, 32
51. Reinhard, E., Adhikhmin, M., Gooch, B., Shirley, P.: Color transfer between images. IEEE Computer graphics and applications (2001) 6
52. Reynolds, M., Doboš, J., Peel, L., Weyrich, T., Brostow, G.J.: Capturing time-of-flight data with confidence. In: CVPR (2011) 6
53. Scharstein, D., Hirschmüller, H., Kitajima, Y., Krathwohl, G., Nešić, N., Wang, X., Westling, P.: High-resolution stereo datasets with subpixel-accurate ground truth. In: GCPR (2014) 8
54. Schops, T., Schönberger, J.L., Galliani, S., Sattler, T., Schindler, K., Pollefeys, M., Geiger, A.: A multi-view stereo benchmark with high-resolution images and multi-camera videos. In: CVPR (2017) 1, 3, 8
55. Schwarz, L.A.: Non-rigid registration using free-form deformations. Technische Universität München (2007) 6
56. Sobel, I., Feldman, G.: A 3x3 isotropic gradient operator for image processing. A talk at the Stanford Artificial Project (1968) 7
57. Sun, C., Shrivastava, A., Singh, S., Gupta, A.: Revisiting unreasonable effectiveness of data in deep learning era. In: ICCV (2017) 2
58. Thewlis, J., Bilen, H., Vedaldi, A.: Unsupervised learning of object landmarks by factorized spatial embeddings. In: ICCV (2017) 4
59. Tonioni, A., Poggi, M., Mattoccia, S., Di Stefano, L.: Unsupervised adaptation for deep stereo. In: ICCV (2017) 1, 4
60. Tonioni, A., Rahnama, O., Joy, T., di Stefano, L., Ajanthan, T., Torr, P.H.S.: Learning to adapt for stereo. In: CVPR (2019) 4
61. Tonioni, A., Tosi, F., Poggi, M., Mattoccia, S., Stefano, L.D.: Real-time self-adaptive deep stereo. In: ICCV (2019) 1, 4
62. Tosi, F., Aleotti, F., Poggi, M., Mattoccia, S.: Learning monocular depth estimation infusing traditional stereo knowledge. In: CVPR (2019) 3, 4, 14
63. Vasiljevic, I., Kolkin, N., Zhang, S., Luo, R., Wang, H., Dai, F.Z., Daniele, A.F., Mostajabi, M., Basart, S., Walter, M.R., Shakhnarovich, G.: DIODE: A Dense Indoor and Outdoor DEpth Dataset. arXiv:1908.00463 (2019) 8
64. Veit, A., Alldrin, N., Chechik, G., Krasin, I., Gupta, A., Belongie, S.: Learning from noisy large-scale datasets with minimal supervision. In: CVPR (2017) 2
65. Van der Walt, S., Schönberger, J.L., Nunez-Iglesias, J., Boulogne, F., Warner, J.D., Yager, N., Gouillart, E., Yu, T.: scikit-image: image processing in python. PeerJ (2014) 24
66. Wang, C., Lucey, S., Perazzi, F., Wang, O.: Web stereo video supervision for depth prediction from dynamic scenes. In: 3DV (2019) 3
67. Wang, J., Zickler, T.: Local detection of stereo occlusion boundaries. In: CVPR (2019) 6

68. Wang, Q., Zheng, S., Yan, Q., Deng, F., Zhao, K., Chu, X.: IRS: A large synthetic indoor robotics stereo dataset for disparity and surface normal estimation. arXiv:1912.09678 (2019) 4
69. Wang, Y., Wang, P., Yang, Z., Luo, C., Yang, Y., Xu, W.: UnOS: Unified unsupervised optical-flow and stereo-depth estimation by watching videos. In: CVPR (2019) 4
70. Wang, Y., Wang, L., Yang, J., An, W., Guo, Y.: Flickr1024: A large-scale dataset for stereo image super-resolution. In: ICCV Workshops (2019) 9, 28, 31, 32
71. Watson, J., Firman, M., Brostow, G.J., Turmukhambetov, D.: Self-supervised monocular depth hints. In: ICCV (2019) 3, 14
72. Woodford, O., Torr, P., Reid, I., Fitzgibbon, A.: Global stereo reconstruction under second-order smoothness priors. PAMI (2009) 3
73. Xian, K., Shen, C., Cao, Z., Lu, H., Xiao, Y., Li, R., Luo, Z.: Monocular relative depth perception with web stereo data supervision. In: CVPR (2018) 3
74. Xiao, T., Xia, T., Yang, Y., Huang, C., Wang, X.: Learning from massive noisy labeled data for image classification. In: CVPR (2015) 2
75. Xie, S., Girshick, R.B., Dollár, P., Tu, Z., He, K.: Aggregated residual transformations for deep neural networks. In: CVPR (2016) 7
76. Yin, Z., Darrell, T., Yu, F.: Hierarchical discrete distribution decomposition for match density estimation. In: CVPR (2019) 1
77. Yu, J., Lin, Z., Yang, J., Shen, X., Lu, X., Huang, T.S.: Free-form image inpainting with gated convolution. In: ICCV (2019) 12
78. Žbontar, J., LeCun, Y.: Stereo matching by training a convolutional neural network to compare image patches. JMLR (2016) 3
79. Zhang, F., Prisacariu, V., Yang, R., Torr, P.H.: GA-Net: Guided aggregation net for end-to-end stereo matching. In: CVPR (2019) 1, 3, 4, 10, 19, 20, 27, 28, 29, 30
80. Zhang, F., Qi, X., Yang, R., Prisacariu, V., Wah, B., Torr, P.: Domain-invariant stereo matching networks. In: ECCV (2020) 3, 4, 5
81. Zhong, Y., Dai, Y., Li, H.: Self-supervised learning for stereo matching with self-improving ability. arXiv:1709.00930 (2017) 4
82. Zhou, B., Zhao, H., Puig, X., Fidler, S., Barriuso, A., Torralba, A.: Scene parsing through ADE20K dataset. In: CVPR (2017) 8
83. Zhou, C., Zhang, H., Shen, X., Jia, J.: Unsupervised learning of stereo matching. In: ICCV (2017) 1, 4
84. Zhou, T., Brown, M., Snavely, N., Lowe, D.G.: Unsupervised learning of depth and ego-motion from video. In: CVPR (2017) 6
85. Zhou, T., Tulsiani, S., Sun, W., Malik, J., Efros, A.A.: View synthesis by appearance flow. In: ECCV (2016) 5

Supplementary Material

7 KITTI Test Server Evaluation

Figs. 7 and 8 show the complete set of available qualitative results from the held-out 2015 test set from the KITTI online server. We show the official benchmark model of (**GANet official**)[2] [79], a GANet retrained by us using only Sceneflow [44] (**Sceneflow GANet**), and GANet trained only with our MfS data (**MfS GANet (Ours)**). Note that **GANet official** uses KITTI data to finetune, as is apparent in the overly smooth predictions on transparent surfaces like car windows (see rows 6 and 7 in Fig. 7). In Fig. 8 (also rows 6 and 7), can see poor quality predictions in the right of the sky region for the finetuned model, whereas our MfS trained model produces much more sensible predictions. Our results are generally more faithful to the boundaries of the color input image and have fewer artefacts than the alternative methods despite using no LiDAR or synthetic data during training. Quantitative results are presented in Table 7.

Table 7: KITTI 2015 Benchmark. Our model performs better than the Sceneflow trained alternative. Row three gives the GANet [79] scores (after KITTI LiDAR finetuning) on the same benchmark as reported in their paper [79], and row four gives the current best GANet score on the KITTI benchmark leaderboard. Our scores are not competitive with these scores from models which have had domain-specific KITTI LiDAR finetuning, but our qualitative results are more faithful (see Figs. 7 and 8).

Model	KITTI Finetune	D1-bg Noc	D1-fg Noc	D1-all Noc	D1-bg All	D1-fg All	D1-all All
MfS GANet (Ours)		**2.96**	**15.09**	**4.97**	**3.13**	**15.57**	**5.20**
Sceneflow GANet		3.60	16.56	5.74	3.86	17.21	6.08
GANet (in paper)	✓		3.39	1.84		3.91	2.03
GANet Official	✓	1.34	3.11	1.63	1.48	3.46	1.81

8 Image Synthesis Visualisation

When converting predicted depth Z to disparity D we can vary the scaling factor s, i.e. $\tilde{D} = \frac{s\, Z_{\max}}{Z}$. In Fig. 9 we illustrate the resulting synthesized right image $\tilde{I}_r$ for different values of s.

As s becomes larger, we see that it has the effect of changing the relative camera viewpoint of the synthesized right image, e.g. we observe the front of the bus occluding the trash can on the sidewalk. Disoccluded regions are filled in with a texture from a randomly selected image.

[2] http://www.cvlibs.net/datasets/kitti/eval_scene_flow_detail.php?benchmark=stereo&result=ccb2b24d3e08ec968368f85a4eeab8b668e70b8c

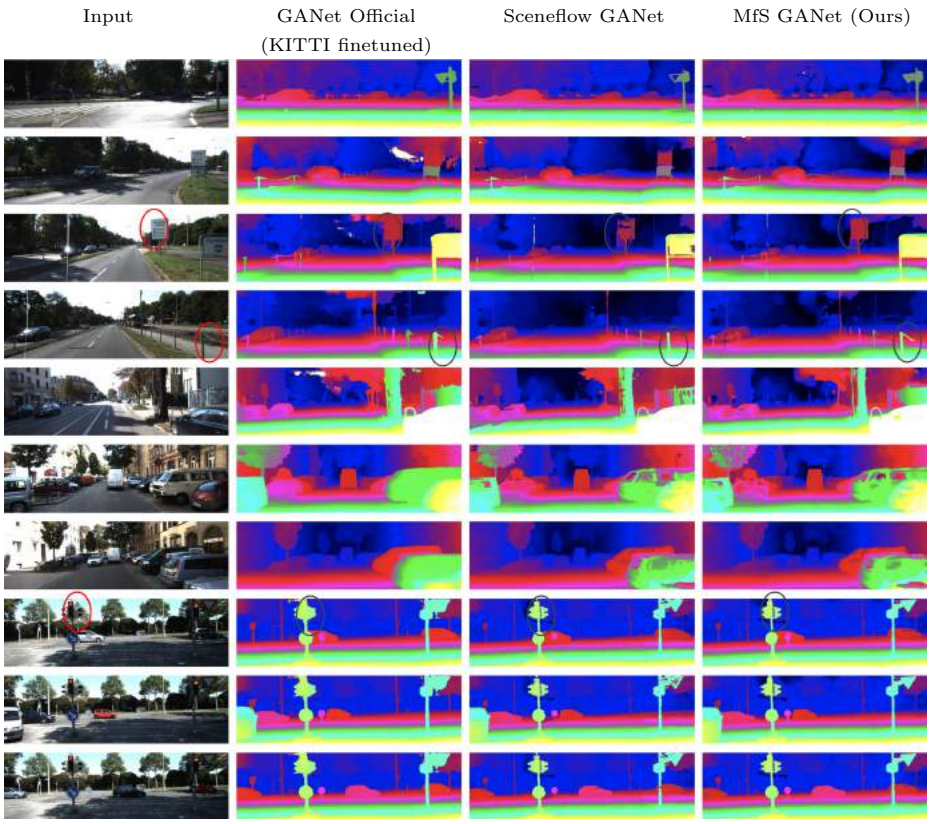


Fig. 7: KITTI 2015 benchmark qualitative comparison. On the held-out KITTI 2015 benchmark images our predictions, MfS GANet (Ours), generated without any KITTI finetuning, are qualitatively more faithful to the scene geometry than both the Sceneflow-trained model and the official KITTI-finetund GANet [79]. These visualizations are generated by the KITTI upload server. [32].

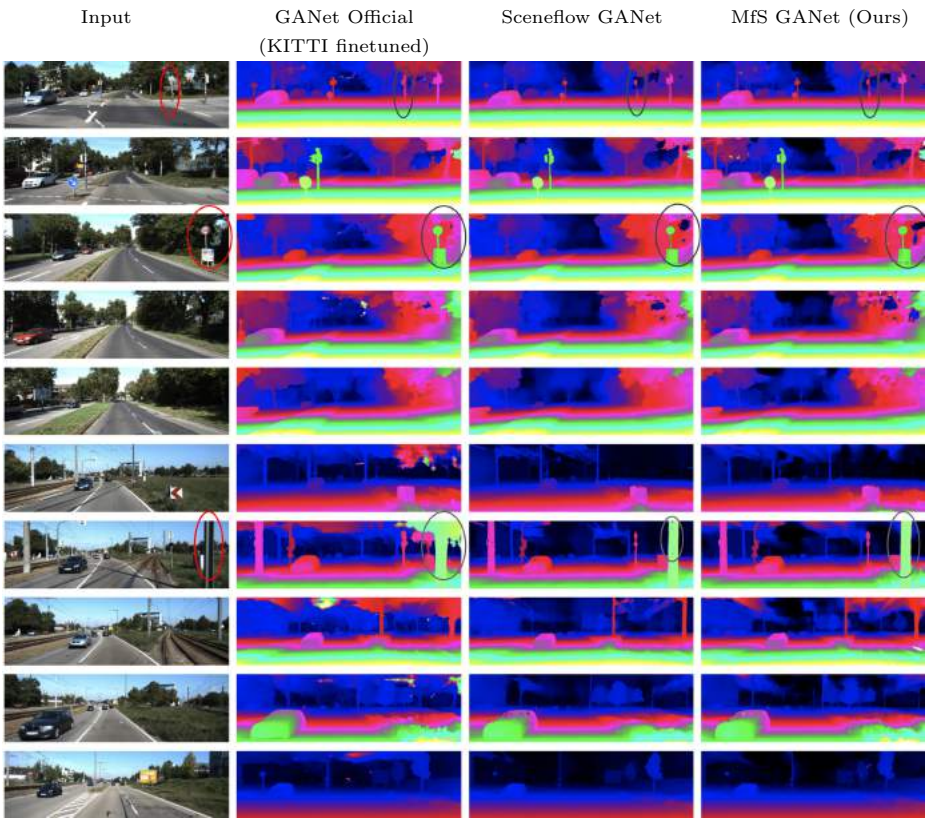
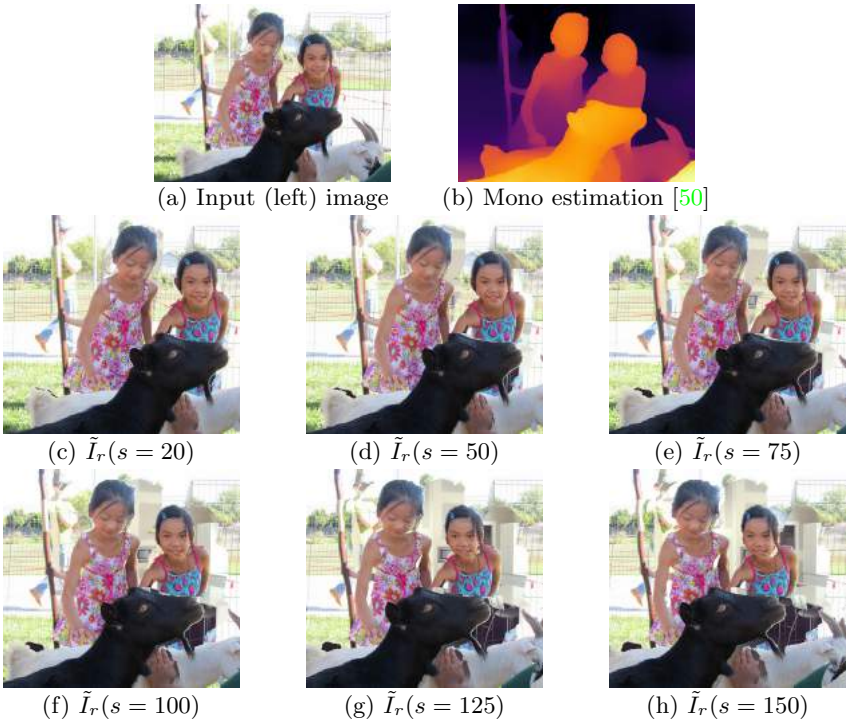


Fig. 8: KITTI 2015 benchmark qualitative comparison – continued. See Fig. 7 caption for details.



(a) Input (left) image (b) Mono estimation [50]

(c) $\tilde{I}_r(s = 20)$ (d) $\tilde{I}_r(s = 50)$ (e) $\tilde{I}_r(s = 75)$

(f) $\tilde{I}_r(s = 100)$ (g) $\tilde{I}_r(s = 125)$ (h) $\tilde{I}_r(s = 150)$

Fig. 9: Visualization of different scaling factors s**.** Here we show the effect of varying the scaling factor s (c - h) when constructing the synthesized right image $\tilde{I}_r$ from the input left image (a) and its predicted depth (b).

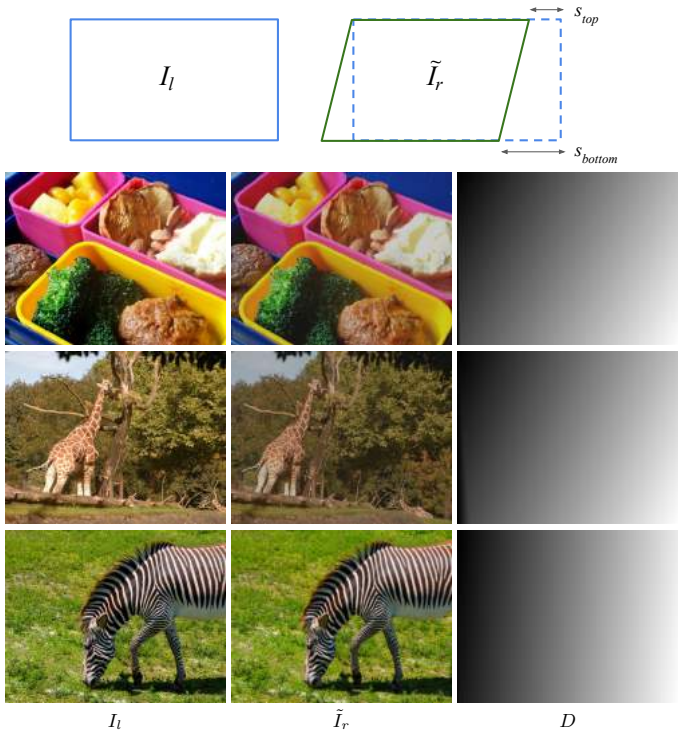


Fig. 10: Affine warp baseline. The diagram at top shows how $s_{\text{top}}, s_{\text{bottom}}$ relate to the warp applied to $\tilde{I}_r$. Below, we show examples of the resulting affine warped images.

9 Baseline Algorithm Details

Here we describe the baseline image synthesis approaches we compared to in Table 1 in the main paper. We show qualitative results for these approaches in Fig. 13 here and in Fig. 5 in the main paper.

Affine Warps — For the affine warps baseline, similar to [12], we warp each I_l with a top shift s_{top} and a bottom shift s_{bottom}, with 50% probability $s_{\text{top}} \sim \text{Unif}\,[0, d_{\max}]$ and $s_{\text{bottom}} \sim \text{Unif}\,[0, s_{\text{top}}]$; and with 50% probability $s_{\text{bottom}} \sim \text{Unif}\,[0, d_{\max}]$ and $s_{\text{top}} \sim \text{Unif}\,[0, s_{\text{bottom}}]$. Following the warping, I_l and $\tilde{I}_r$ are both cropped to avoid black borders, and the corresponding D is adjusted appropriately to account for this cropping. Details and example training images are shown in Fig. 10.

Random Pasted Shapes — For the random pasted shapes baseline we paste 'foreground' shapes onto a 'background' image in a similar manner to [43]. We start with affine warped images, but with $d_{\max} = 50$ to help to ensure that foreground 'objects' move with greater disparity than the 'background' image. We then sample num_patches $\sim$ Unif$[0, 10]$ and for each patch we load a random

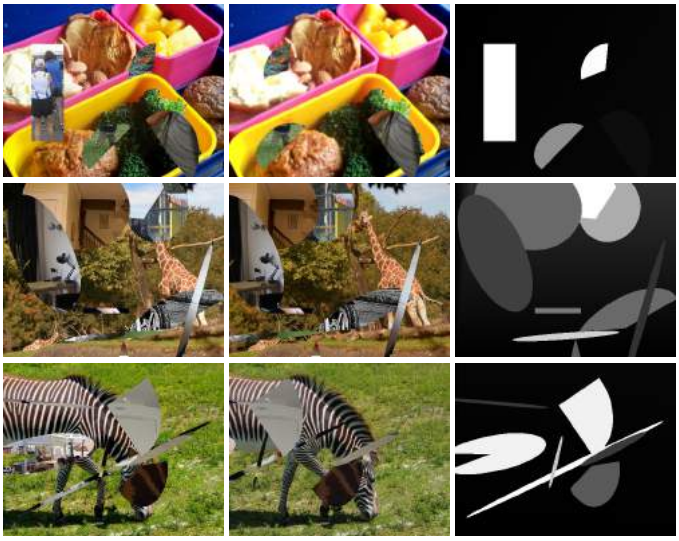


Fig. 11: Random pasted shapes. Examples images generated using this baseline.

image from the training set and mask out a region. The masked region is pasted on top of I_l, and a translated version of the masked region is placed on $\tilde{I}_r$. Masks are generated with equal probability from rectangles, partial ellipses, polygons and thin objects. With the exception of rectangles, masks are generated with OpenCV's drawing functions. Examples of random pasted shapes training tuples are shown in Fig. 11. Note that the disparity of the background image is less visible compared with the disparity maps from the Affine Warp baseline, due to the reduced $d_{\max}$ used here.

Rectangles are axis-aligned, with x-axis and y-axis bounds sampled from Unif$[0.1W, 0.9W]$ and Unif$[0.1H, 0.9H]$ respectively.

Partial ellipses have a center (x, y), where $x \sim$ Unif$[0.1W, 0.9W]$ and $y \sim$ Unif$[0.1H, 0.9H]$. With 75% probability the ellipse is a full ellipse, with start_angle $= 0$ and end_angle $= 360$. Otherwise, the ellipse is partial, with the angular bounds sampled from Unif$[0, 360]$. The rotation angle of the ellipse is sampled from Unif$[0, 360]$, and the axes sizes are sampled from Unif$[0.1W, 0.9W]$.

Polygons are generated using the approach described in [47], with a number of sides sampled from Unif$[3, 20]$, spikyness $= 0.8$, irregularity $= 0.5$, and aveRadius $\sim$ Unif$[0.01W, 0.3W]$.

Thin objects are full ellipses with with minor axis $\sim$ Unif$[0.001H, 0.025H]$ and major axis $\sim$ Unif$[0.1W, 0.5W]$.

Random Superpixels — For this baseline, we segment I_l into superpixels using [17,65], with parameters scale $\sim$ Unif$[50, 200]$, $\sigma \sim$ Unif$[0, 1]$, and

Fig. 12: Random superpixels. Synthesized stereo pairs using superpixel warping.

min_size $\sim$ Unif[75, 275]. We initialise D as a plane, where pixel i has disparity value d_i defined by $d_i = ax + by + c$ with $a \sim$ Unif[$-0.025, 0.025$], $b \sim$ Unif[$0.3, 0.4$], and $c \sim$ Unif[$15, 20$]. Superpixels s_j in I_l are chosen with probability 0.6 to act as foreground objects. We set the disparity values of these foreground superpixels as $d_j = \Sigma_{i \in j} \frac{d_i}{n} + x_j$, where $x_j \sim$ Unif[$0, 64$] and n is the number of pixels in superpixel s_j. Finally we clip the values of D to lie between 0 and $d_{\max}$. Using D, we then forward warp I_l to generate our right image $\tilde{I}_r$, handing occlusions and collisions in the way described in Section 3.2 of the main paper. Examples of this training data are shown in Fig. 12.

SVSM selection module — For this baseline, we use the selection module from [41]. Specifically, we still make use of monocular depth estimates from [50], converting the single channel monocular estimate to a one-hot encoding over possible disparities. We then forward warp the disparity and input image, subsequently taking the weighted sum. Interestingly, we did also try to train a network for image synthesis as per the paper (using KITTI), and use this to generate our synthetic right images. Whist the resulting images were reasonable, the output of the network is a distribution over disparities, and using the argmax of the distrbution as a training target for a stereo network resulted in very poor performance.

10 Evaluation Details

10.1 Evaluation Image Resolution

Due to architecture constraints, we make predictions at slightly different resolutions for each network; see Table 8 for details. Note that all predictions are resized to the native resolution of the ground truth for evaluation. When training a given architecture with different datasets we use the same resolution for all datasets for fair comparison.

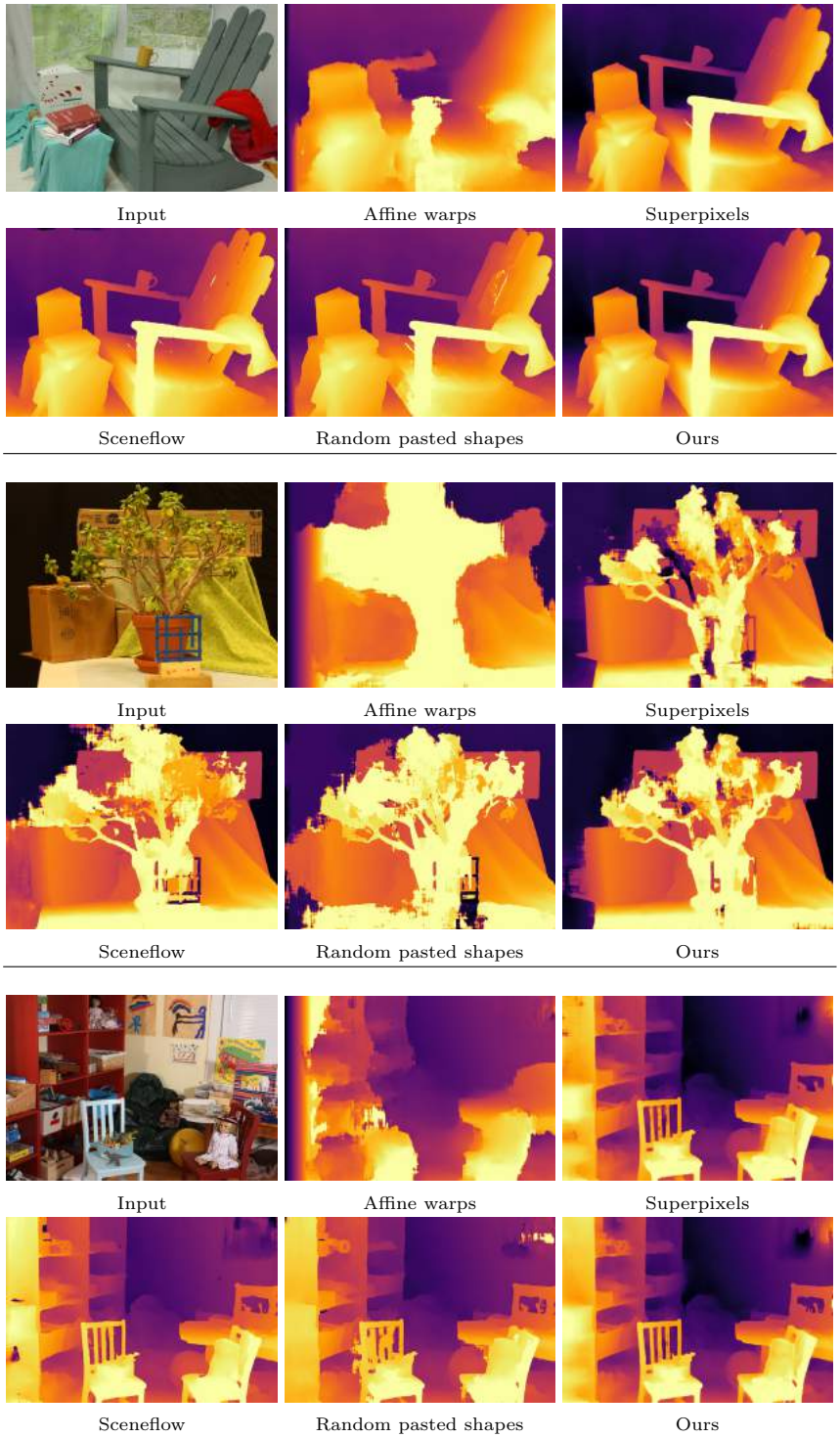


Fig. 13: Additional results comparing our method to the baseline algorithms.

Table 8: Evaluation image resolution. Here we show the prediction resolutions for each architecture. Note that all predictions are resized to the native resolution of the ground truth for evaluation.

Architecture	KITTI '12	KITTI '15	Middlebury	ETH3D
iResnet [39]	1280 × 384	1280 × 384	1280 × 768	768 × 448
PSM [5]	1280 × 384	1280 × 384	1280 × 768	768 × 448
GANet [79]	1248 × 384	1248 × 384	1248 × 768	768 × 432

10.2 KITTI Evaluation Splits

KITTI 2012 results in the paper are reported on the 40 images from the validation split of [5]. The image indices used by both us and [5] are `[3, 6, 20, 26, 38, 41, 43, 44, 49, 60, 67, 70, 81, 84, 89, 97, 109, 119, 122, 123, 129, 130, 132, 134, 141, 144, 152, 158, 165, 171, 174, 179, 184, 186]`. The KITTI 2015 indices are `[1, 3, 6, 20, 26, 35, 38, 41, 43, 44, 49, 60, 67, 70, 81, 84, 89, 97, 109, 119, 122, 123, 129, 130, 132, 134, 141, 144, 152, 158, 159, 165, 171, 174, 179, 182, 184, 186, 187, 196]`.

11 Additional KITTI Results

In Table 9 we present additional metrics for the KITTI 2012 experiments from Table 1 in the main paper where we compare different methods for generating training data. We also show additional metrics for KITTI 2015. As in the main paper, our method brings consistent improvement over the baselines.

Table 9: KITTI 2012 and 2015 Results. Additional metrics for PSMNet [5] trained with different stereo data.

Synthesis approach	Training data	EPE Noc	>3px Noc	EPE All	>3px All
KITTI 2012					
Affine warps	MfS	3.04	12.72	3.74	14.78
Random pasted shapes	MfS	1.90	9.97	2.45	11.89
Random superpixels	MfS	1.09	5.16	1.28	6.08
Synthetic	Sceneflow	0.97	5.09	1.04	5.76
Ours	MfS	**0.79**	**3.60**	**0.95**	**4.43**
KITTI 2015					
Affine warps	MfS	2.05	13.67	2.33	14.94
Random pasted shapes	MfS	1.23	6.24	1.49	7.57
Random superpixels	MfS	1.22	5.62	1.23	5.90
Synthetic	Sceneflow	1.18	5.56	1.19	5.80
Ours	MfS	**1.02**	**4.57**	**1.04**	**4.75**

12 Recovering from Monocular Depth Errors

Fig. 4 in the main paper shows that our trained stereo networks can overcome some of the errors of monocular depth estimation. In Fig. 14 here, we observe the same result across three different monocular depth networks: MiDaS [50], Megadepth [38], and Monodepth2 [23]. In each case, problems present in monocular depths, such as missing objects and uneven ground planes do not transfer to our eventual stereo predictions. We note here that although Monodepth2 [23] was trained using monocular videos from KITTI, our resulting stereo network has never seen images from this dataset.

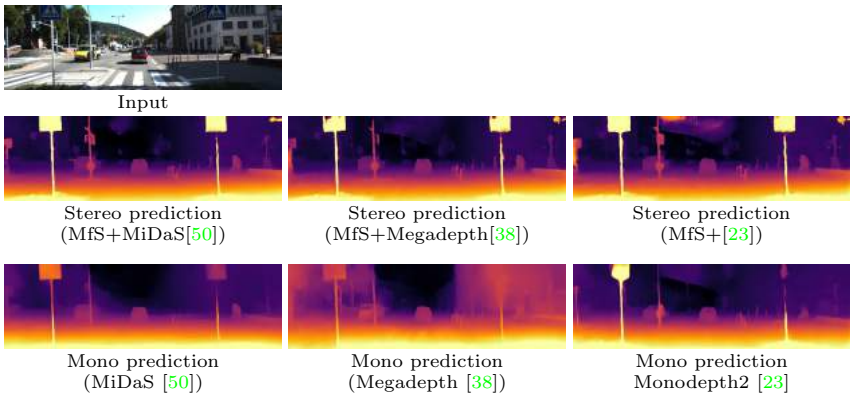


Fig. 14: We can recover from errors in monocular depth estimation.

13 Additional Qualitative Results

In Figs. 15 and 16 we present comparisons using GANet [79] to Sceneflow training versus our MfS dataset for both KITTI 2012 and KITTI 2015. In Figs. 17 and 18 we compare PSM [5] networks using Flickr1024 [70], a dataset stereo images collected online. For Flickr1024, we should results using our method using depth generated by MiDaS [50] or MegaDepth [38]. It is worth remembering that MegaDepth [38] does not use any synthetic or ground truth depth at training time but it still can be used to train a stereo model that produces high quality predictions. In all cases we see that our fully automatic data generation pipeline results in high quality stereo predictions without directly requiring any synthetic training data which is time consuming to create.

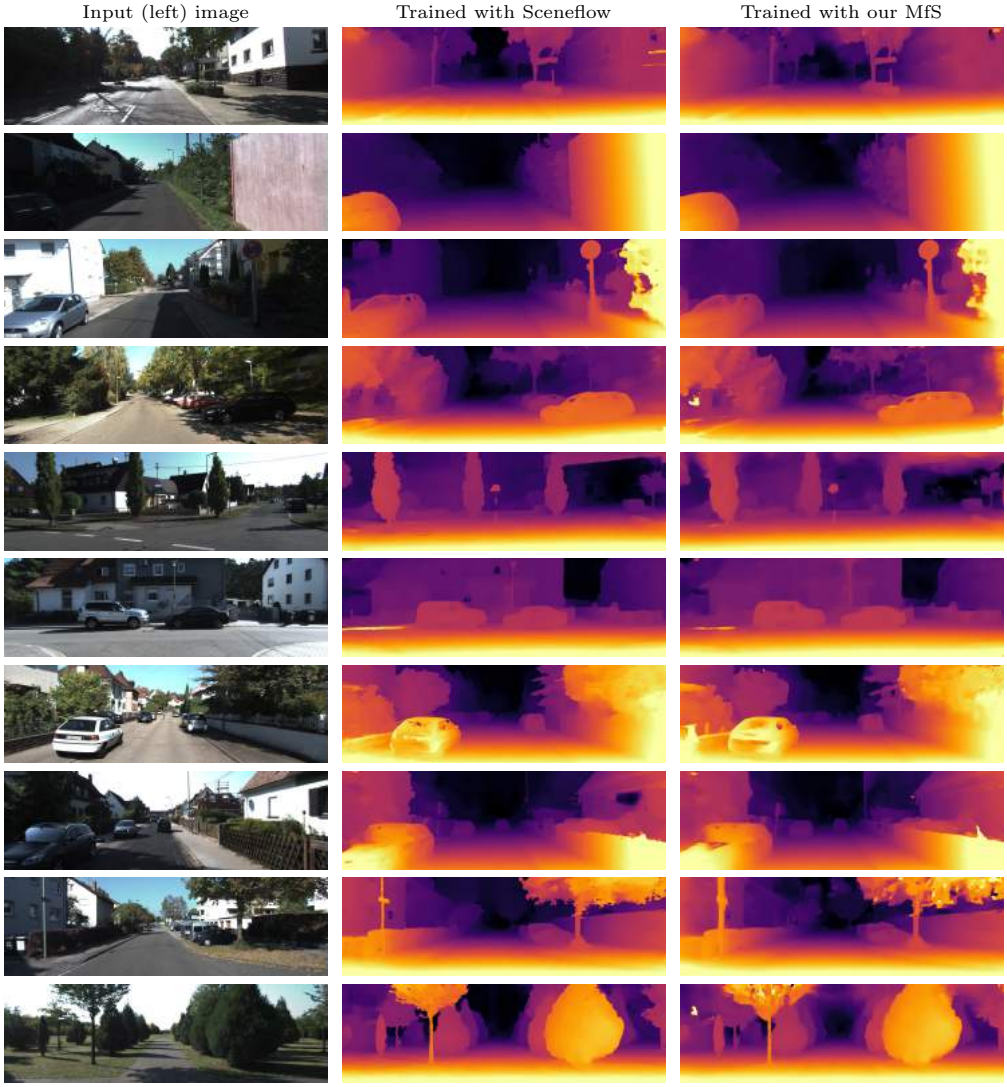


Fig. 15: Additional GANet [79] KITTI 2012 qualitative results.

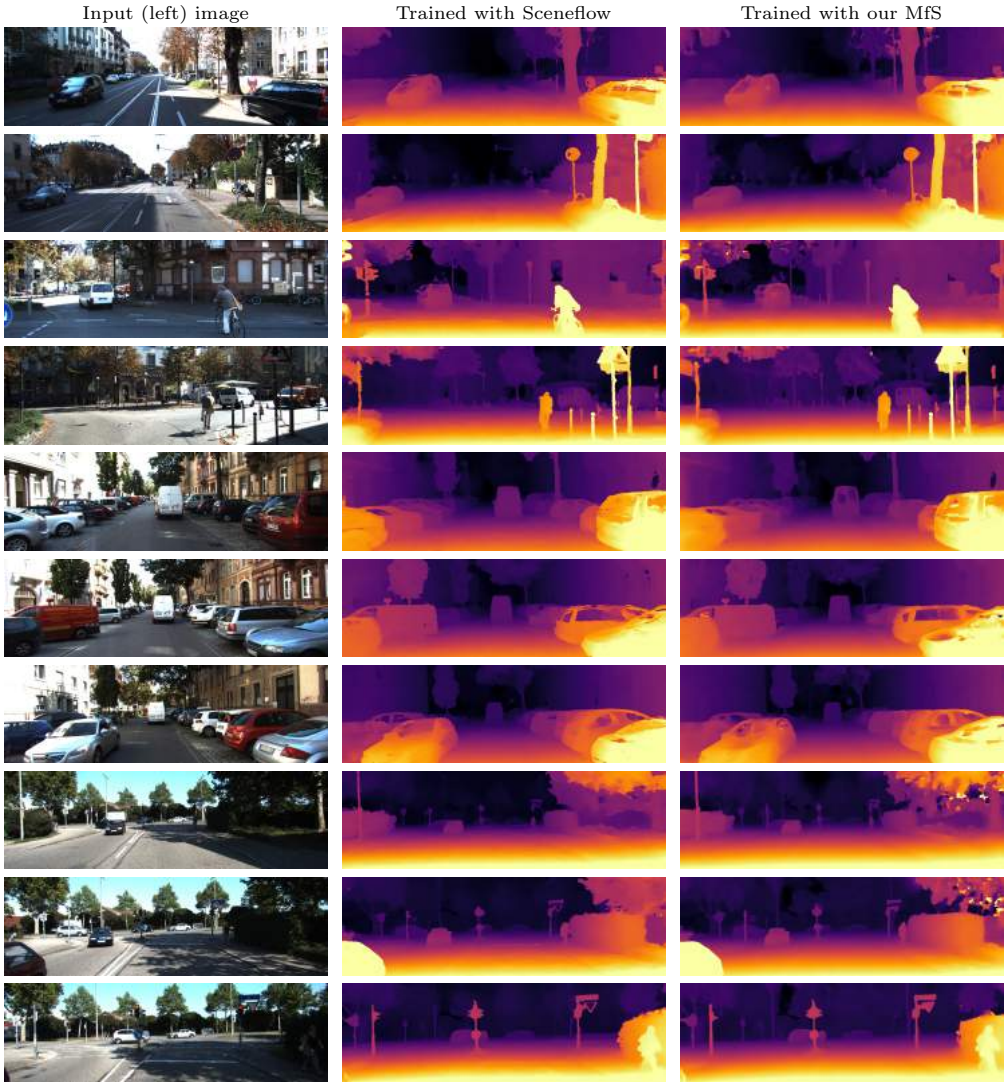


Fig. 16: Additional GANet [79] KITTI 2015 qualitative results.

Fig. 17: Additional Flickr1024 [70] qualitative results.

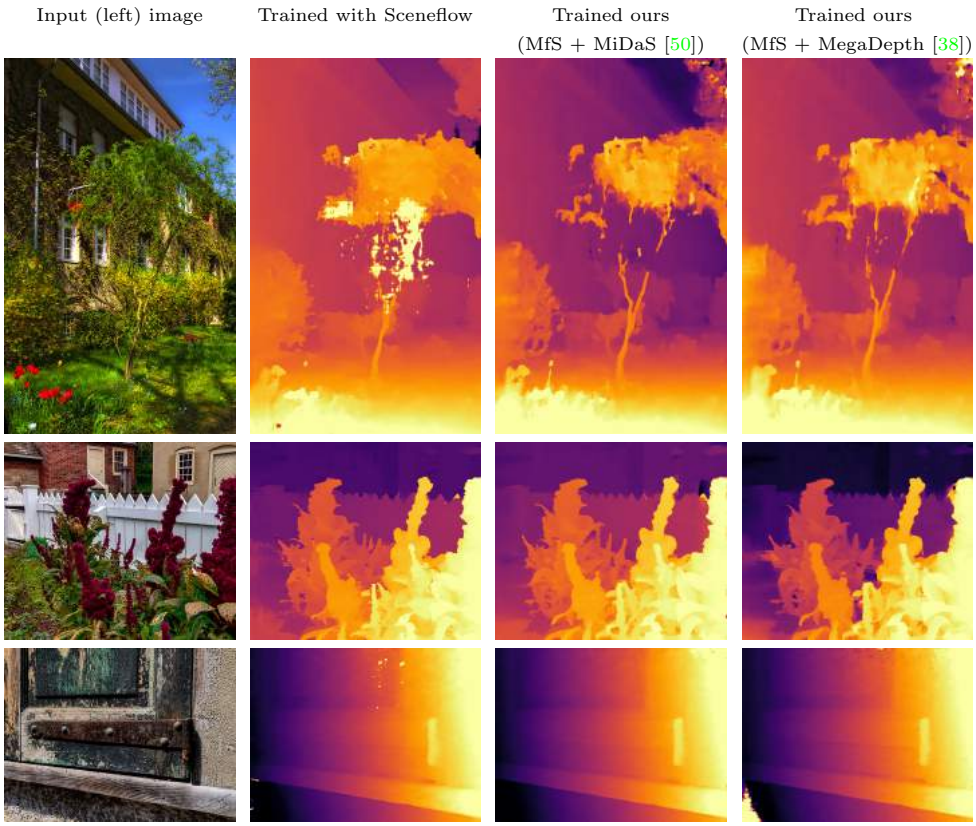


Fig. 18: Additional Flickr1024 [70] qualitative results.