Sparse-to-Dense: Depth Prediction from Sparse Depth Samples and a Single Image

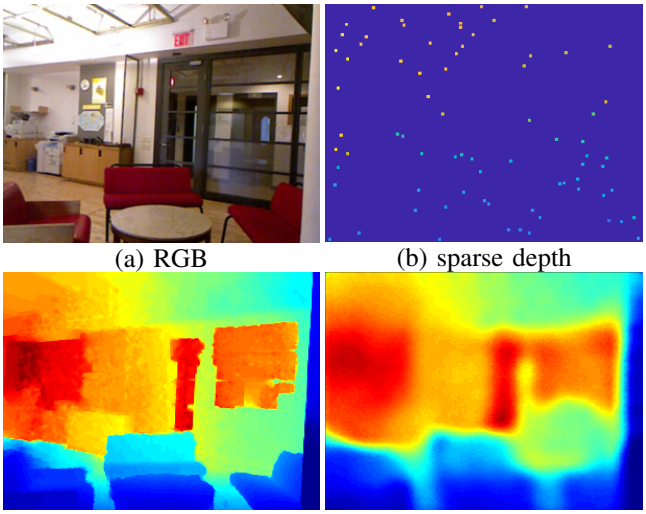
Fangchang Ma[1] and Sertac Karaman[1]

***Abstract*— We consider the problem of dense depth prediction from a sparse set of depth measurements and a single RGB image. Since depth estimation from monocular images alone is inherently ambiguous and unreliable, to attain a higher level of robustness and accuracy, we introduce additional sparse depth samples, which are either acquired with a low-resolution depth sensor or computed via visual Simultaneous Localization and Mapping (SLAM) algorithms. We propose the use of a single deep regression network to learn directly from the RGB-D raw data, and explore the impact of number of depth samples on prediction accuracy. Our experiments show that, compared to using only RGB images, the addition of 100 spatially random depth samples reduces the prediction root-mean-square error by 50% on the NYU-Depth-v2 indoor dataset. It also boosts the percentage of reliable prediction from 59% to 92% on the KITTI dataset. We demonstrate two applications of the proposed algorithm: a plug-in module in SLAM to convert sparse maps to dense maps, and super-resolution for LiDARs. Software[2] and video demonstration[3] are publicly available.**

I. Introduction

Depth sensing and estimation is of vital importance in a wide range of engineering applications, such as robotics, autonomous driving, augmented reality (AR) and 3D mapping. However, existing depth sensors, including LiDARs, structured-light-based depth sensors, and stereo cameras, all have their own limitations. For instance, the top-of-the-range 3D LiDARs are cost-prohibitive (with up to $75,000 cost per unit), and yet provide only sparse measurements for distant objects. Structured-light-based depth sensors (*e.g.* Kinect) are sunlight-sensitive and power-consuming, with a short ranging distance. Finally, stereo cameras require a large baseline and careful calibration for accurate triangulation, which demands large amount of computation and usually fails at featureless regions. Because of these limitations, there has always been a strong interest in depth estimation using a single camera, which is small, low-cost, energy-efficient, and ubiquitous in consumer electronic products.

However, the accuracy and reliability of such methods is still far from being practical, despite over a decade of research effort devoted to RGB-based depth prediction including the recent improvements with deep learning approaches. For instance, the state-of-the-art RGB-based depth prediction methods [1–3] produce an average error (measured by the root mean squared error) of over 50cm in indoor scenarios (*e.g.*, on the NYU-Depth-v2 dataset [4]). Such methods perform even worse outdoors, with at least 4 meters of average error on Make3D and KITTI datasets [5, 6].



(a) RGB (b) sparse depth (c) ground truth (d) prediction

Fig. 1: We develop a deep regression model to predict dense depth image from a single RGB image and a set of sparse depth samples. Our method significantly outperforms RGB-based and other fusion-based algorithms.

To address the potential fundamental limitations of RGB-based depth estimation, we consider the utilization of sparse depth measurements, along with RGB data, to reconstruct depth in full resolution. Sparse depth measurements are readily available in many applications. For instance, low-resolution depth sensors (*e.g.*, a low-cost LiDARs) provide such measurements. Sparse depth measurements can also be computed from the output of SLAM[4] and visual-inertial odometry algorithms. In this work, we demonstrate the effectiveness of using sparse depth measurements, in addition to the RGB images, as part of the input to the system. We use a single convolutional neural network to learn a deep regression model for depth image prediction. Our experimental results show that the addition of as few as 100 depth samples reduces the root mean squared error by over 50% on the NYU-Depth-v2 dataset, and boosts the percentage of reliable prediction from 59% to 92% on the more challenging KITTI outdoor dataset. In general, our results show that the addition of a few sparse depth samples drastically improves depth reconstruction performance. Our quantitative results may help inform the development of

[1]F. Ma and S. Karaman are with the Laboratory for Information & Decision Systems, Massachusetts Institute of Technology, Cambridge, MA, USA. {fcma, sertac}@mit.edu

[2]https://github.com/fangchangma/sparse-to-dense

[3]https://www.youtube.com/watch?v=vNIIT_M7x7Y

[4]A typical feature-based SLAM algorithm, such as ORB-SLAM [7], keeps track of hundreds of 3D landmarks in each frame.

sensors for future robotic vehicles and consumer devices.

The main contribution of this paper is a deep regression model that takes both a sparse set of depth samples and RGB images as input and predicts a full-resolution depth image. The prediction accuracy of our method significantly outperforms state-of-the-art methods, including both RGB-based and fusion-based techniques. Furthermore, we demonstrate in experiments that our method can be used as a plug-in module to sparse visual odometry / SLAM algorithms to create an accurate, dense point cloud. In addition, we show that our method can also be used in 3D LiDARs to create much denser measurements.

II. Related Work

RGB-based depth prediction Early works on depth estimation using RGB images usually relied on hand-crafted features and probabilistic graphical models. For instance, Saxena et al. [8] estimated the absolute scales of different image patches and inferred the depth image using a Markov Random Field model. Non-parametric approaches [9–12] were also exploited to estimate the depth of a query image by combining the depths of images with similar photometric content retrieved from a database.

Recently, deep learning has been successfully applied to the depth estimation problem. Eigen et al. [13] suggest a two-stack convolutional neural network (CNN), with one predicting the global coarse scale and the other refining local details. Eigen and Fergus [2] further incorporate other auxiliary prediction tasks into the same architecture. Liu et al. [1] combined a deep CNN and a continuous conditional random field, and attained visually sharper transitions and local details. Laina et al. [3] developed a deep residual network based on the ResNet [14] and achieved higher accuracy than [1, 2]. Semi-supervised [15] and unsupervised learning [16–18] setups have also been explored for disparity image prediction. For instance, Godard et al. [18] formulated disparity estimation as an image reconstruction problem, where neural networks were trained to warp left images to match the right.

Depth reconstruction from sparse samples Another line of related work is depth reconstruction from sparse samples. A common ground of many approaches in this area is the use of sparse representations for depth signals. For instance, Hawe et al. [19] assumed that disparity maps were sparse on the Wavelet basis and reconstructed a dense disparity image with a conjugate sub-gradient method. Liu et al. [20] combined wavelet and contourlet dictionaries for more accurate reconstruction. Our previous work on sparse depth sensing [21, 22] exploited the sparsity underlying the second-order derivatives of depth images, and outperformed both [1, 19] in reconstruction accuracy and speed.

Sensor fusion A wide range of techniques attempted to improve depth prediction by fusing additional information from different sensor modalities. For instance, Mancini et al. [23] proposed a CNN that took both RGB images and optical flow images as input to predict distance. Liao et al. [24] studied the use of a 2D laser scanner mounted on a mobile ground robot to provide an additional reference depth signal as input and obtained higher accuracy than using RGB images alone. Compared to the approach by Liao et al. [24], this work makes no assumption regarding the orientation or position of sensors, nor the spatial distribution of input depth samples in the pixel space. Cadena et al. [25] developed a multi-modal auto-encoder to learn from three input modalities, including RGB, depth, and semantic labels. In their experiments, Cadena et al. [25] used sparse depth on extracted FAST corner features as part of the input to the system to produce a low-resolution depth prediction. The accuracy was comparable to using RGB alone. In comparison, our method predicts a full-resolution depth image, learns a better cross-modality representation for RGB and sparse depth, and attains a significantly higher accuracy.

III. Methodology

In this section, we describe the architecture of the convolutional neural network. We also discuss the depth sampling strategy, the data augmentation techniques, and the loss functions used for training.

A. CNN Architecture

We found in our experiments that many bottleneck architectures (with an encoder and a decoder) could result in good performance. We chose the final structure based on [3] for the sake of benchmarking, because it achieved state-of-the-art accuracy in RGB-based depth prediction. The network is tailed to our problem with input data of different modalities, sizes and dimensions. We use two different networks for KITTI and NYU-Depth-v2. This is because the KITTI image is triple the size of NYU-Depth-v2 and consequently the same architecture would require 3 times of GPU memory, exceeding the current hardware capacity. The final structure is illustrated in Figure 2.

The feature extraction (encoding) layers of the network, highlighted in blue, consist of a ResNet [14] followed by a convolution layer. More specifically, the ResNet-18 is used for KITTI, and ResNet-50 is used for NYU-Depth-v2. The last average pooling layer and linear transformation layer of the original ResNet have been removed. The second component of the encoding structure, the convolution layer, has a kernel size of 3-by-3.

The decoding layers, highlighted in yellow, are composed of 4 upsampling layers followed by a bilinear upsampling layer. We use the UpProj module proposed by Laina et al. [3] as our upsampling layer, but a deconvolution with larger kernel size can also achieve the same level of accuracy. An empirical comparison of different upsampling layers is shown in Section V-A.

B. Depth Sampling

In this section, we introduce the sampling strategy for creating the input sparse depth image from the ground truth.

During training, the input sparse depth D is sampled randomly from the ground truth depth image D^* on the fly. In particular, for any targeted number of depth samples m

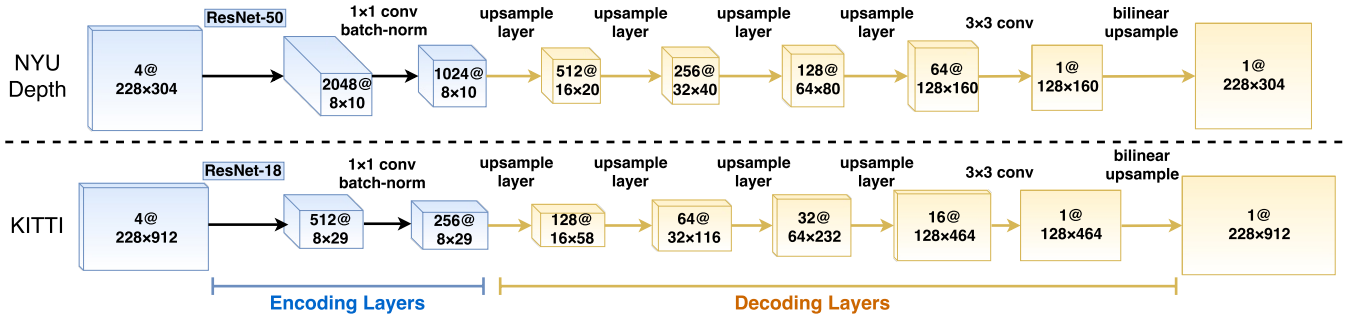


Fig. 2: CNN architecture for NYU-Depth-v2 and KITTI datasets, respectively. Cubes are feature maps, with dimensions represented as #features@height×width. The encoding layers in blue consist of a ResNet [14] and a 3×3 convolution. The decoding layers in yellow are composed of 4 upsampling layers (UpProj) followed by a bilinear upsampling.

(fixed during training), we compute a Bernoulli probability $p = \frac{m}{n}$, where n is the total number of valid depth pixels in D^*. Then, for any pixel (i, j),

$$D(i,j) = \begin{cases} D^*(i,j), & \text{with probability } p \\ 0, & \text{otherwise} \end{cases} \quad (1)$$

With this sampling strategy, the actual number of non-zero depth pixels varies for each training sample around the expectation m. Note that this sampling strategy is different from dropout [26], which scales up the output by $1/p$ during training to compensate for deactivated neurons. The purpose of our sampling strategy is to increase robustness of the network against different number of inputs and to create more training data (*i.e.*, a data augmentation technique). It is worth exploring how injection of random noise and a different sampling strategy (*e.g.*, feature points) would affect the performance of the network.

C. Data Augmentation

We augment the training data in an online manner with random transformations, including

- *Scale*: color images are scaled by a random number $s \in [1, 1.5]$, and depths are divided by s.
- *Rotation*: color and depths are both rotated with a random degree $r \in [-5, 5]$.
- *Color Jitter*: the brightness, contrast, and saturation of color images are each scaled by $k_i \in [0.6, 1.4]$.
- *Color Normalization*: RGB is normalized through mean subtraction and division by standard deviation.
- *Flips*: color and depths are both horizontally flipped with a 50% chance.

Nearest neighbor interpolation, rather than the more common bi-linear or bi-cubic interpolation, is used in both scaling and rotation to avoid creating spurious sparse depth points. We take the center crop from the augmented image so that the input size to the network is consistent.

D. Loss Function

One common and default choice of loss function for regression problems is the mean squared error ($\mathcal{L}_2$). $\mathcal{L}_2$ is sensitive to outliers in the training data since it penalizes more heavily on larger errors. During our experiments we found that the $\mathcal{L}_2$ loss function also yields visually undesirable, over-smooth boundaries instead of sharp transitions.

Another common choice is the Reversed Huber (denoted as berHu) loss function [27], defined as

$$\mathcal{B}(e) = \begin{cases} |e|, & \text{if } |e| \le c \\ \frac{e^2+c^2}{2c}, & \text{otherwise} \end{cases} \quad (2)$$

[3] uses a batch-dependent parameter c, computed as 20% of the maximum absolute error over all pixels in a batch. Intuitively, berHu acts as the mean absolute error ($\mathcal{L}_1$) when the element-wise error falls below c, and behaves approximately as $\mathcal{L}_2$ when the error exceeds c.

In our experiments, besides the aforementioned two loss functions, we also tested $\mathcal{L}_1$ and found that it produced slightly better results on the RGB-based depth prediction problem. The empirical comparison is shown in Section V-A. As a result, we use $\mathcal{L}_1$ as our default choice throughout the paper for its simplicity and performance.

IV. Experiments

We implement the network using Torch [28]. Our models are trained on the NYU-Depth-v2 and KITTI odometry datasets using a NVIDIA Tesla P100 GPU with 16GB memory. The weights of the ResNet in the encoding layers (except for the first layer which has different number of input channels) are initialized with models pretrained on the ImageNet dataset [29]. We use a small batch size of 16 and train for 20 epochs. The learning rate starts at 0.01, and is reduced to 20% every 5 epochs. A small weight decay of 10^{-4} is applied for regularization.

A. The NYU-Depth-v2 Dataset

The NYU-Depth-v2 dataset [4] consists of RGB and depth images collected from 464 different indoor scenes with a Microsoft Kinect. We use the official split of data, where 249 scenes are used for training and the remaining 215 for testing. In particular, for the sake of benchmarking, the small labeled test dataset with 654 images is used for evaluating the final performance, as seen in previous work [3, 13].

For training, we sample spatially evenly from each raw video sequence from the training dataset, generating roughly

48k synchronized depth-RGB image pairs. The depth values are projected onto the RGB image and in-painted with a cross-bilateral filter using the official toolbox. Following [3, 13], the original frames of size 640×480 are first down-sampled to half and then center-cropped, producing a final size of 304×228.

B. The KITTI Odometry Dataset

In this work we use the *odometry* dataset, which includes both camera and LiDAR measurements . The *odometry* dataset consists of 22 sequences. Among them, one half is used for training while the other half is for evaluation. We use all 46k images from the training sequences for training the neural network, and a random subset of 3200 images from the test sequences for the final evaluation.

We use both left and right RGB cameras as unassociated shots. The Velodyne LiDAR measurements are projected onto the RGB images. Only the bottom crop (912×228) is used, since the LiDAR returns no measurement to the upper part of the images. Compared with NYU-Depth-v2, even the ground truth is sparse for KITTI, typically with only 18k projected measurements out of the 208k image pixels.

C. Error Metrics

We evaluate each method using the following metrics:

- RMSE: root mean squared error
- REL: mean absolute relative error
- δ_i: percentage of predicted pixels where the relative error is within a threshold. Specifically,

$$\delta_i = \frac{\mathbf{card}\left(\left\{\hat{y}_i : \max\left\{\frac{\hat{y}_i}{y_i}, \frac{y_i}{\hat{y}_i}\right\} < 1.25^i\right\}\right)}{\mathbf{card}\left(\{y_i\}\right)},$$

where y_i and $\hat{y}_i$ are respectively the ground truth and the prediction, and $\mathbf{card}$ is the cardinality of a set. A higher δ_i indicates better prediction.

V. Results

In this section we present all experimental results. First, we evaluate the performance of our proposed method with different loss functions and network components on the prediction accuracy in Section V-A. Second, we compare the proposed method with state-of-the-art methods on both the NYU-Depth-v2 and the KITTI datasets in Section V-B. Third, In Section V-C, we explore the impact of number of sparse depth samples on the performance. Finally, in Section V-D and Section V-E, we demonstrate two use cases of our proposed algorithm in creating dense maps and LiDAR super-resolution.

A. Architecture Evaluation

In this section we present an empirical study on the impact of different loss functions and network components on the depth prediction accuracy. The results are listed in Table I.

Problem	Loss	Encoder	Decoder	RMSE	REL	δ_1	δ_2	δ_3
RGB	$\mathcal{L}_2$	Conv	DeConv_2	0.610	0.185	71.8	93.4	98.3
	berHu	Conv	DeConv_2	0.554	0.163	**77.5**	94.8	**98.7**
	$\mathcal{L}_1$	Conv	DeConv_2	**0.552**	**0.159**	**77.5**	**95.0**	**98.7**
		Conv	DeConv_3	0.533	0.151	79.0	95.4	98.8
		Conv	UpConv	0.529	0.149	79.4	**95.5**	**98.9**
		Conv	UpProj	**0.528**	**0.144**	**80.3**	95.2	98.7
RGBd	$\mathcal{L}_1$	ChanDrop	UpProj	0.361	0.105	90.8	98.4	99.6
		DepthWise	UpProj	**0.261**	0.054	**96.2**	**99.2**	99.7
		Conv	UpProj	0.264	**0.053**	96.1	**99.2**	**99.8**

TABLE I: Evaluation of loss functions, upsampling layers and the first convolution layer. RGBd has an average sparse depth input of 100 samples. (a) comparison of loss functions is listed in Row 1 - 3; (b) comparison of upsampling layers is in Row 2 - 4; (c) comparison of the first convolution layers is in the 3 bottom rows.

1) Loss Functions: To compare the loss functions we use the same network architecture, where the upsampling layers are simple deconvolution with a 2×2 kernel (denoted as DeConv_2). $\mathcal{L}_2$, berHu and $\mathcal{L}_1$ loss functions are listed in the first three rows in Table I for comparison. As shown in the table, both berHu and $\mathcal{L}_1$ significantly outperform $\mathcal{L}_2$. In addition, $\mathcal{L}_1$ produces slightly better results than berHu. Therefore, we use $\mathcal{L}_1$ as our default choice of loss function.

2) Upsampling Layers: We perform an empirical evaluation of different upsampling layers, including deconvolution with kernels of different sizes (DeConv_2 and DeConv_3), as well as the UpConv and UpProj modules proposed by Laina et al. [3]. The results are listed from row 3 to 6 in Table I.

We make several observations. Firstly, deconvolution with a 3×3 kernel (*i.e.*, DeConv_3) outperforms the same component with only a 2×2 kernel (*i.e.*, DeConv_2) in every single metric. Secondly, since both DeConv_3 and UpConv have a receptive field of 3×3 (meaning each output neuron is computed from a neighborhood of 9 input neurons), they have comparable performance. Thirdly, with an even larger receptive field of 4×4, the UpProj module outperforms the others. We choose to use UpProj as a default choice.

3) First Convolution Layer: Since our RGBd input data comes from different sensing modalities, its 4 input channels (R, G, B, and depth) have vastly different distributions and support. We perform a simple analysis on the first convolution layer and explore three different options.

The first option is the regular spatial convolution (Conv). The second option is depthwise separable convolution (denoted as DepthWise), which consists of a spatial convolution performed independently on each input channel, followed by a pointwise convolution across different channels with a window size of 1. The third choice is channel dropout (denoted as ChanDrop), through which each input channel is preserved as is with some probability p, and zeroed out with probability $1 - p$.

The bottom 3 rows compare the results from the 3 options. The networks are trained using RGBd input with an average of 100 sparse input samples. DepthWise and Conv yield very similar results, and both significantly outperform the ChanDrop layer. Since the difference is small, for the sake

of comparison consistency, we will use the convolution layer for all experiments.

B. Comparison with the State-of-the-Art

In this section, we compare with existing methods.

1) NYU-Depth-v2 Dataset: We compare with RGB-based approaches [3, 13, 30], as well as the fusion approach [24] that utilizes an additional 2D laser scanner mounted on a ground robot. The quantitative results are listed in Table II.

Problem	#Samples	Method	RMSE	REL	δ_1	δ_2	δ_3
RGB	0	Roy *et al.* [30]	0.744	0.187	-	-	-
	0	Eigen *et al.* [2]	0.641	0.158	76.9	95.0	98.8
	0	Laina *et al.* [3]	0.573	**0.127**	**81.1**	95.3	98.8
	0	Ours-RGB	**0.514**	0.143	81.0	**95.9**	**98.9**
sd	20	Ours-sd	0.461	0.110	87.2	96.1	98.8
	50	Ours-sd	0.347	0.076	92.8	98.2	99.5
	200	Ours-sd	**0.259**	**0.054**	**96.3**	**99.2**	**99.8**
RGBd	225	Liao *et al.* [24]	0.442	0.104	87.8	96.4	98.9
	20	Ours-RGBd	0.351	0.078	92.8	98.4	99.6
	50	Ours-RGBd	0.281	0.059	95.5	99.0	99.7
	200	Ours-RGBd	**0.230**	**0.044**	**97.1**	**99.4**	**99.8**

TABLE II: Comparison with state-of-the-art on the NYU-Depth-v2 dataset. The values are those originally reported by the authors in their respective paper

Our first observation from Row 2 and Row 3 is that, with the same network architecture, we can achieve a slightly better result (albeit higher REL) by replacing the berHu loss function proposed in [3] with a simple $\mathcal{L}_1$. Secondly, by comparing problem group RGB (Row 3) and problem group sd (*e.g.*, Row 4), we draw the conclusion that an extremely small set of 20 sparse depth samples (without color information) already produces significantly better predictions than using RGB. Thirdly, by comparing problem group sd and proble group RGBd row by row with the same number of samples, it is clear that the color information does help improve the prediction accuracy. In other words, our proposed method is able to learn a suitable representation from both the RGB images and the sparse depth images. Finally, we compare against [24] (bottom row). Our proposed method, even using only 100 samples, outperforms [24] with 225 laser measurements. This is because our samples are spatially uniform, and thus provides more information than a line measurement. A few examples of our predictions with different inputs are displayed in Figure 3.

2) KITTI Dataset: The KITTI dataset is more challenging for depth prediction, since the maximum distance is 100 meters as opposed to only 10 meters in the NYU-Depth-v2 dataset. A greater performance boost can be obtained from using our approach. Although the training and test data are not the same across different methods, the scenes are similar in the sense that they all come from the same sensor setup on a car and the data were collected during driving. We report the values from each work in Table III.

The results in the first RGB group demonstrate that RGB-based depth prediction methods fail in outdoor scenarios, with a pixel-wise RMSE of close to 7 meters. Note that we

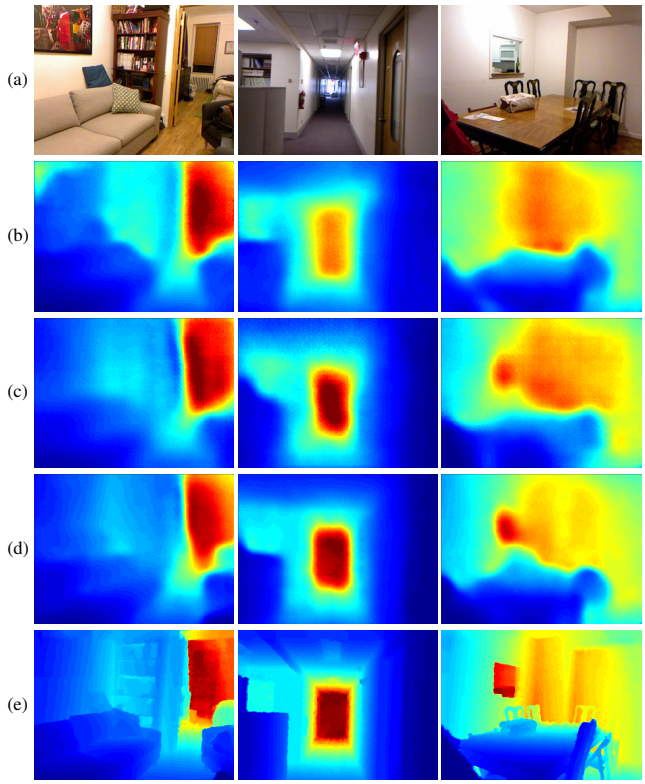


Fig. 3: Predictions on NYU-Depth-v2. From top to bottom: (a) rgb images; (b) RGB-based prediction; (c) sd prediction with 200 and no rgb; (d) RGBd prediction with 200 sparse depth and rgb; (e) ground truth depth.

Problem	#Samples	Method	RMSE	REL	δ_1	δ_2	δ_3
RGB	0	Make3D [3]	8.734	0.280	60.1	82.0	92.6
	0	Mancini [23]	7.508	-	31.8	61.7	81.3
	0	Eigen et al. [13]	7.156	**0.190**	**69.2**	89.9	**96.7**
	0	Ours-RGB	**6.266**	0.208	59.1	**90.0**	96.2
RGBd	∼650	full-MAE [25]	7.14	0.179	70.9	88.8	95.6
	50	Ours-RGBd	4.884	0.109	87.1	95.2	97.9
	225	Liao *et al.* [24]	4.50	0.113	87.4	96.0	98.4
	100	Ours-RGBd	4.303	0.095	90.0	96.3	98.3
	200	Ours-RGBd	3.851	0.083	91.9	97.0	98.6
	500	Ours-RGBd	**3.378**	**0.073**	**93.5**	**97.6**	**98.9**

TABLE III: Comparison with state-of-the-art on the KITTI dataset. The Make3D values are those reported in [13]

use sparsely labeled depth image projected from LiDAR, instead of dense disparity maps computed from stereo cameras as in [13]. In other words, we have a much smaller training dataset compared with [13, 23].

An additional 500 depth samples bring the RMSE to 3.3 meters, a half of the RGB approach, and boosts δ_1 from only 59.1% to 93.5%. Our performance also compares favorably to other fusion techniques including [24, 25], and at the same time demands fewer samples.

C. On Number of Depth Samples

In this section, we explore the relation between the prediction accuracy and the number of available depth samples. We train a network for each different input size for optimal

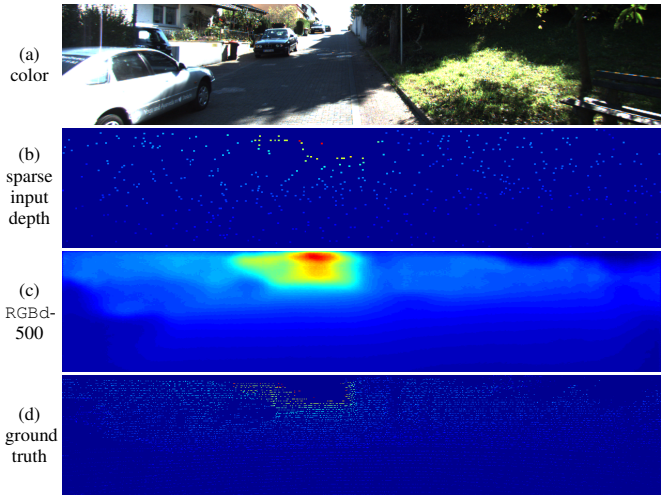


Fig. 4: Example of prediction on KITTI. From top to bottom: (a) RGB; (b) sparse depth; (c) RGBd dense prediction; (d) ground truth depth projected from LiDAR.

performance. We compare the performance for all three kinds of input data, including RGB, sd, and RGBd. The performance of RGB-based depth prediction is independent of input sample size and is thus plotted as a horizontal line for benchmarking.

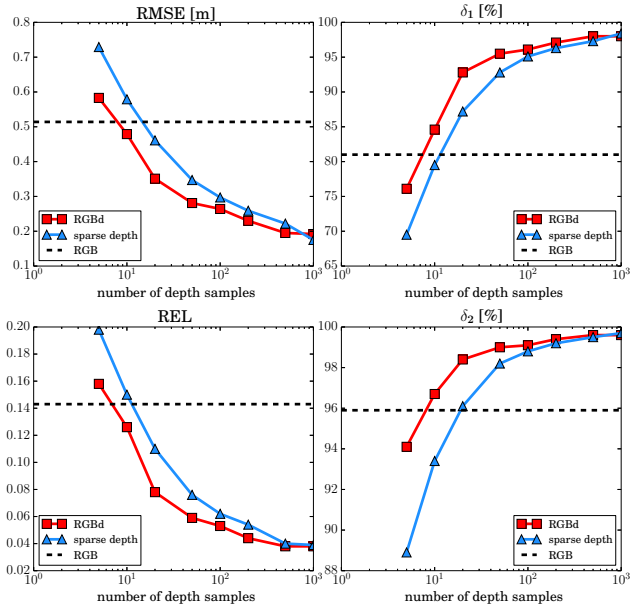


Fig. 5: Impact of number of depth sample on the prediction accuracy on the NYU-Depth-v2 dataset. Left column: lower is better; right column: higher is better.

On the NYU-Depth-v2 dataset in Figure 5, the RGBd outperforms RGB with over 10 depth samples and the performance gap quickly increases with the number of samples. With a set of 100 samples, the RMSE of RGBd decreases to around 25cm, half of RGB (51cm). The REL sees a larger

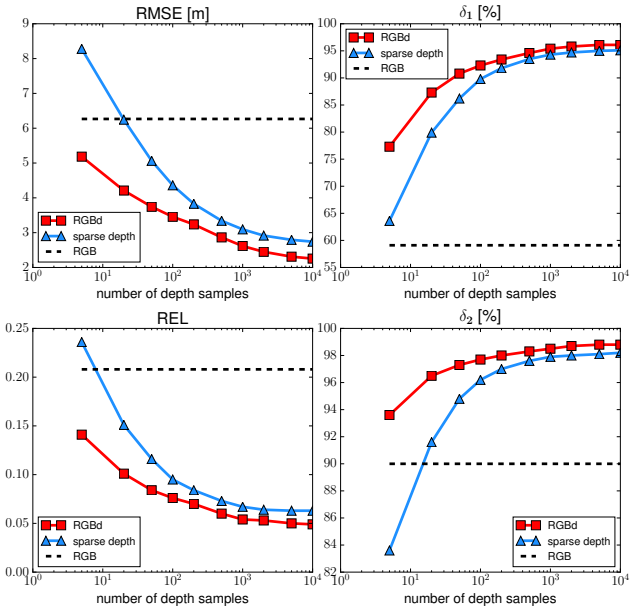


Fig. 6: Impact of number of depth sample on the prediction accuracy on the KITTI dataset. Left column: lower is better; right column: higher is better.

improvement (from 0.15 to 0.05, reduced by two thirds). On one hand, the RGBd approach consistently outperforms sd, which indicates that the learned model is indeed able to extract information not only from the sparse samples alone, but also from the colors. On the other hand, the performance gap between RGBd and sd shrinks as the sample size increases. Both approaches perform equally well when sample size goes up to 1000, which accounts for less than 1.5% of the image pixels and is still a small number compared with the image size. This observation indicates that the information extracted from the sparse sample set dominates the prediction when the sample size is sufficiently large, and in this case the color cue becomes almost irrelevant.

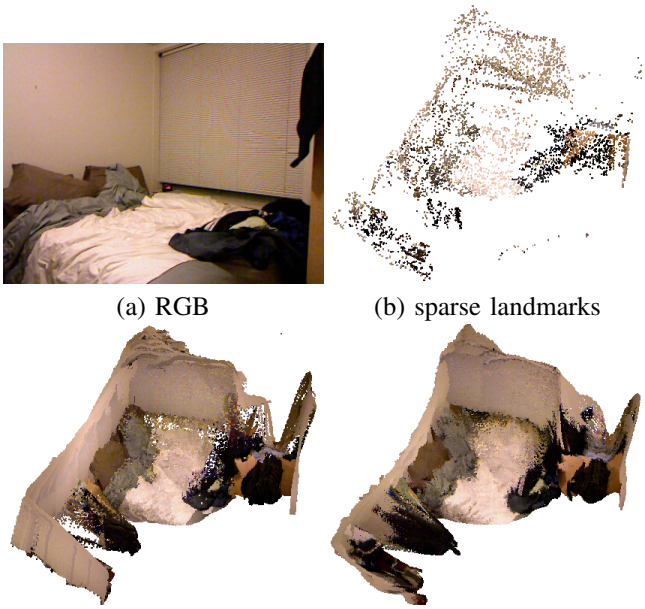
The performance gain on the KITTI dataset is almost identical to NYU-Depth-v2, as shown in Figure 6. With 100 samples the RMSE of RGBd decreases from 7 meters to a half, 3.5 meters. This is the same percentage of improvement as on the NYU-Depth-v2 dataset. Similarly, the REL is reduced from 0.21 to 0.07, again the same percentage of improvement as the NYU-Depth-v2.

On both datasets, the accuracy saturates as the number of depth samples increases. Additionally, the prediction has blurry boundaries even with many depth samples (see Figure 8). We believe both phenomena can be attributed to the fact that fine details are lost in bottleneck network architectures. It remains further study if additional skip connections from encoders to decoders help improve performance.

D. Application: Dense Map from Visual Odometry Features

In this section, we demonstrate a use case of our proposed method in sparse visual SLAM and visual inertial odometry (VIO). The best-performing algorithms for SLAM and VIO

are usually sparse methods, which represent the environment with sparse 3D landmarks. Although sparse SLAM/VIO algorithms are robust and efficient, the output map is in the form of sparse point clouds and is not useful for other applications (*e.g.* motion planning).



(a) RGB (b) sparse landmarks

(c) ground truth map (d) prediction

Fig. 7: Application in sparse SLAM and visual inertial odometry (VIO) to create dense point clouds from sparse landmarks. (a) RGB (b) sparse landmarks (c) ground truth point cloud (d) prediction point cloud, created by stitching `RGBd` predictions from each frame.

To demonstrate the effectiveness of our proposed methods, we implement a simple visual odometry (VO) algorithm with data from one of the test scenes in the NYU-Depth-v2 dataset. For simplicity, the absolute scale is derived from ground truth depth image of the first frame. The 3D landmarks produced by VO are back-projected onto the RGB image space to create a sparse depth image. We use both RGB and sparse depth images as input for prediction. Only pixels within a trusted region, which we define as the convex hull on the pixel space formed by the input sparse depth samples, are preserved since they are well constrained and thus more reliable. Dense point clouds are then reconstructed from these reliable predictions, and are stitched together using the trajectory estimation from VIO.

The results are displayed in Figure 7. The prediction map resembles closely to the ground truth map, and is much denser than the sparse point cloud from VO. The major difference between our prediction and the ground truth is that the prediction map has few points on the white wall, where no feature is extracted or tracked by the VO. As a result, pixels corresponding to the white walls fall outside the trusted region and are thus removed.

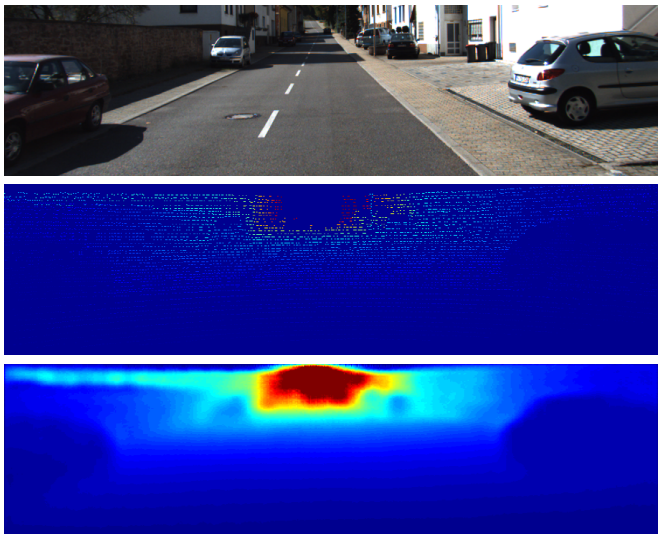


Fig. 8: Application to LiDAR super-resolution, creating denser point cloud than the raw measurements. From top to bottom: RGB, raw depth, and predicted depth. Distant cars are almost invisible in the raw depth, but are easily recognizable in the predicted depth.

E. Application: LiDAR Super-Resolution

We present another demonstration of our method in super-resolution of LiDAR measurements. 3D LiDARs have a low vertical angular resolution and thus generate a vertically sparse point cloud. We use all measurements in the sparse depth image and RGB images as input to our network. The average REL is 4.9%, as compared to 20.8% when using only RGB. An example is shown in Figure 8. Cars are much more recognizable in the prediction than in the raw scans.

VI. CONCLUSION

We introduced a new depth prediction method for predicting dense depth images from both RGB images and sparse depth images, which is well suited for sensor fusion and sparse SLAM. We demonstrated that this method significantly outperforms depth prediction using only RGB images, and other existing RGB-D fusion techniques. This method can be used as a plug-in module in sparse SLAM and visual inertial odometry algorithms, as well as in super-resolution of LiDAR measurements. We believe that this new method opens up an important avenue for research into RGB-D learning and the more general 3D perception problems, which might benefit substantially from sparse depth samples.

ACKNOWLEDGMENT

This work was supported in part by the Office of Naval Research (ONR) through the ONR YIP program. We also gratefully acknowledge the support of NVIDIA Corporation with the donation of the DGX-1 used for this research.

REFERENCES

[1] F. Liu, C. Shen, and G. Lin, "Deep convolutional neural fields for depth estimation from a single image,"

in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 5162–5170.

[2] D. Eigen and R. 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.

[3] I. Laina, C. Rupprecht *et al.*, "Deeper depth prediction with fully convolutional residual networks," in *3D Vision (3DV), 2016 Fourth International Conference on*. IEEE, 2016, pp. 239–248.

[4] N. Silberman, D. Hoiem *et al.*, "Indoor segmentation and support inference from rgbd images," *Computer Vision–ECCV 2012*, pp. 746–760, 2012.

[5] A. Saxena, M. Sun, and A. Y. Ng, "Make3d: Learning 3d scene structure from a single still image," *IEEE transactions on pattern analysis and machine intelligence*, vol. 31, no. 5, pp. 824–840, 2009.

[6] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? the kitti vision benchmark suite," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012.

[7] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardos, "Orb-slam: a versatile and accurate monocular slam system," *IEEE Transactions on Robotics*, vol. 31, no. 5, pp. 1147–1163, 2015.

[8] A. Saxena, S. H. Chung, and A. Y. Ng, "Learning depth from single monocular images," in *Advances in neural information processing systems*, 2006, pp. 1161–1168.

[9] K. Karsch, C. Liu, and S. B. Kang, "Depth extraction from video using non-parametric sampling," in *European Conference on Computer Vision*. Springer, 2012, pp. 775–788.

[10] J. Konrad, M. Wang, and P. Ishwar, "2d-to-3d image conversion by learning depth from examples," in *Computer Vision and Pattern Recognition Workshops (CVPRW), 2012 IEEE Computer Society Conference on*. IEEE, 2012, pp. 16–22.

[11] K. Karsch, C. Liu, and S. Kang, "Depthtransfer: Depth extraction from video using non-parametric sampling," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, no. 99, pp. 1–1, 2014.

[12] M. Liu, M. Salzmann, and X. He, "Discrete-continuous depth estimation from a single image," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 716–723.

[13] 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*, 2014, pp. 2366–2374.

[14] K. He, X. Zhang *et al.*, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[15] Y. Kuznietsov, J. Stückler, and B. Leibe, "Semi-supervised deep learning for monocular depth map prediction," *arXiv preprint arXiv:1702.02706*, 2017.

[16] T. Zhou, M. Brown *et al.*, "Unsupervised learning of depth and ego-motion from video," *arXiv preprint arXiv:1704.07813*, 2017.

[17] R. Garg, G. Carneiro, and I. Reid, "Unsupervised cnn for single view depth estimation: Geometry to the rescue," in *European Conference on Computer Vision*. Springer, 2016, pp. 740–756.

[18] C. Godard, O. Mac Aodha, and G. J. Brostow, "Unsupervised monocular depth estimation with left-right consistency," *arXiv preprint arXiv:1609.03677*, 2016.

[19] S. Hawe, M. Kleinsteuber, and K. Diepold, "Dense disparity maps from sparse disparity measurements," in *Computer Vision (ICCV), 2011 IEEE International Conference on*. IEEE, 2011, pp. 2126–2133.

[20] L.-K. Liu, S. H. Chan, and T. Q. Nguyen, "Depth reconstruction from sparse samples: Representation, algorithm, and sampling," *IEEE Transactions on Image Processing*, vol. 24, no. 6, pp. 1983–1996, 2015.

[21] F. Ma, L. Carlone *et al.*, "Sparse sensing for resource-constrained depth reconstruction," in *Intelligent Robots and Systems (IROS), 2016 IEEE/RSJ International Conference on*. IEEE, 2016, pp. 96–103.

[22] ——, "Sparse depth sensing for resource-constrained robots," *arXiv preprint arXiv:1703.01398*, 2017.

[23] M. Mancini, G. Costante *et al.*, "Fast robust monocular depth estimation for obstacle detection with fully convolutional networks," in *Intelligent Robots and Systems (IROS), 2016 IEEE/RSJ International Conference on*. IEEE, 2016, pp. 4296–4303.

[24] Y. Liao, L. Huang *et al.*, "Parse geometry from a line: Monocular depth estimation with partial laser observation," in *Robotics and Automation (ICRA), 2017 IEEE International Conference on*. IEEE, 2017, pp. 5059–5066.

[25] C. Cadena, A. R. Dick, and I. D. Reid, "Multi-modal auto-encoders as joint estimators for robotics scene understanding." in *Robotics: Science and Systems*, 2016.

[26] N. Srivastava, G. E. Hinton *et al.*, "Dropout: a simple way to prevent neural networks from overfitting." *Journal of machine learning research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[27] A. B. Owen, "A robust hybrid of lasso and ridge regression," *Contemporary Mathematics*, vol. 443, pp. 59–72, 2007.

[28] R. Collobert, K. Kavukcuoglu, and C. Farabet, "Torch7: A matlab-like environment for machine learning," in *BigLearn, NIPS Workshop*, 2011.

[29] O. Russakovsky, J. Deng *et al.*, "Imagenet large scale visual recognition challenge," *International Journal of Computer Vision*, vol. 115, no. 3, pp. 211–252, 2015.

[30] A. Roy and S. Todorovic, "Monocular depth estimation using neural regression forest," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 5506–5514.