Depth Completion with Uncertainty Estimation

Erik Harutyunyan Elen Vardanyan
Technical University of Munich, Department of Mathematics
erik.harutyunyan@tum.de elen.vardanyan@tum.de

Abstract

Depth completion aims to produce a dense depth map from a sparse one with guidance from the corresponding high-resolution RGB image. We propose adding uncertainty estimation to existing depth completion models to enhance their usage in high-risk industries, such as autonomous driving. We modify the decoder of the baseline model MSG-CHN [4] (Fig. 1), by adding separate branch for variance prediction and train with Laplacian NLL loss. As a result besides providing the highly-desirable uncertainty estimates, our method improves the depth prediction accuracy compared to the baseline MSG-CHN. We conduct all our experiments on the KITTI depth completion benchmark. [8].

1. Introduction

Depth estimation is a critical component for various computer vision applications. But even high-end sensors result in sparse depth maps, as most of the points become invalid pixels after the projection. To complete the missing pixels, a lot of methods were proposed in the past few years, both deep learning based and not. But some applications, such as autonomous driving and UAVs, highly depend on the reliability of the estimated depth maps. In this project we extend a state-of-the-art depth completion network with uncertainty estimates, thereby facilitating safety-critical applications that require such a notion of confidence. We investigate different approaches to learn uncertainty and their potential to improve the depth estimates at the same time.

2. Related Work

2.1. Depth Completion

Initial deep learning models for depth completion [5] employed single stream encoder-decoder architecture to process both RGB image and the sparse depth map for the dense depth map construction. More recent papers [2] mostly adopt the two-stream approach, where RGB im-

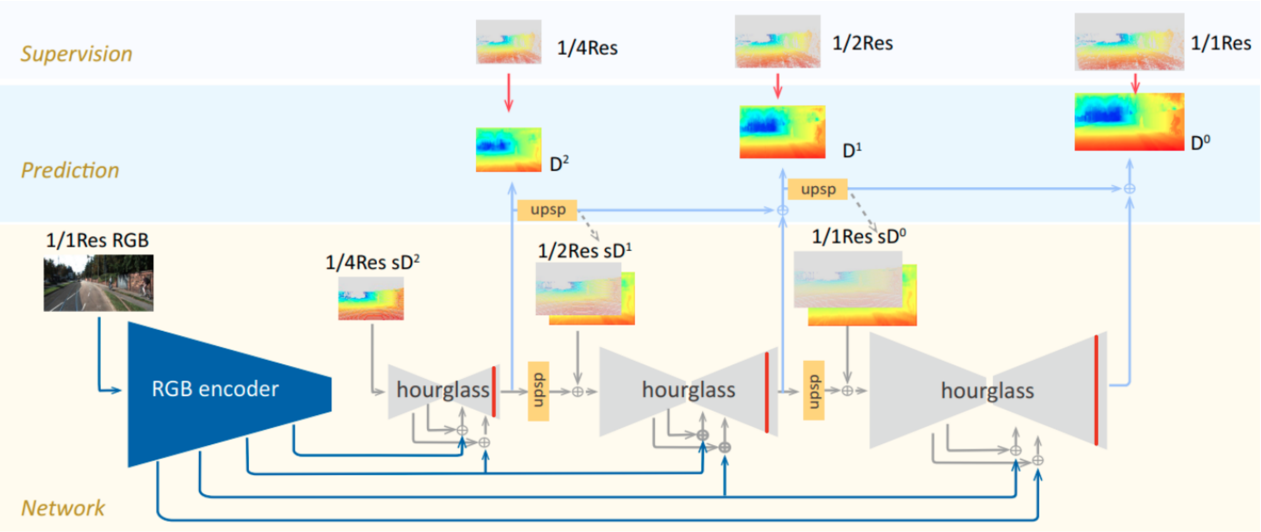
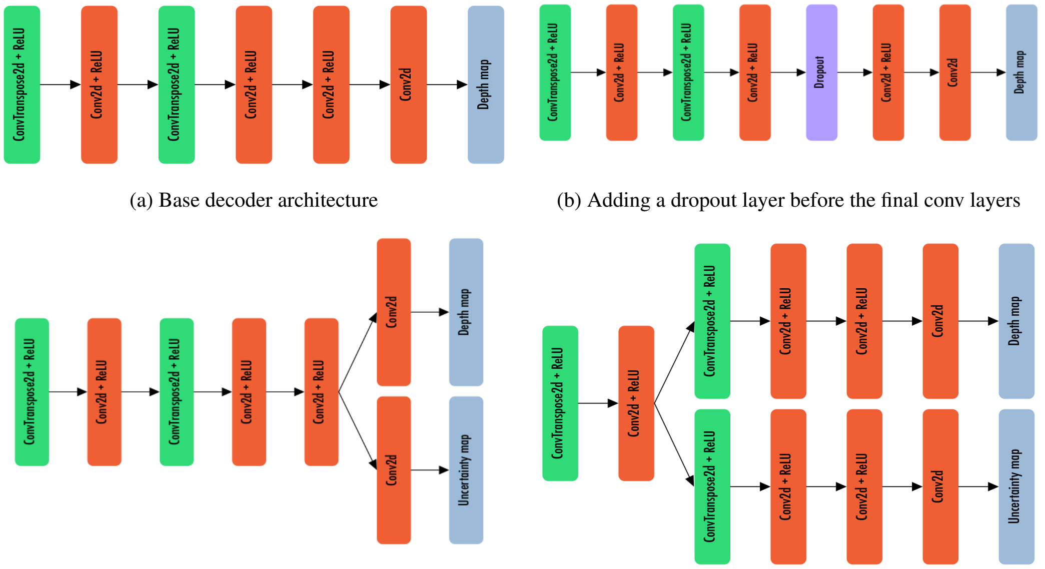


Figure 1. The architecture of the base MSG-CHN network. The decoders of the network are marked with red lines to illustrate the place where the modifications described in the paper appear.



(a) Base decoder architecture

(b) Adding a dropout layer before the final conv layers

(c) Adding separate output for std estimation

(d) Separate branch of convs and transposed convs for std estimation

Figure 2. Architecture changes for the described methods.

age and sparse depth map input are processed with separate CNNs (streams) again following the encoder-decoder paradigm. MSG-CHN [4] network that we will build upon, does not clearly fall into one of these categories, it processes the RGB image separately but does not have a full stream for that purpose – only the encoder part. While not being the top model in the current leaderboard, MSG-CHN is pretty lightweight which enables putting fast experiments on a single 11Gb GPU.

2.2. Uncertainty Estimation

One approach to pixel-wise uncertainty prediction is to extend the network output by an additional channel with exponential activation function [6]. This architecture is trained with either Gaussian or Laplacian [9] negative log-likelihood (NLL) loss function. In contrast, [1] proposes to run inference on the same input several times with active dropout layers and use the variance of the outputs as an uncertainty estimate. We will apply NLL methods to the MSG-CHN network to perform joint depth and uncertainty estimation, and utilize the MC dropout approach to estimate the depth uncertainties of the pretrained MSG-CHN network.

3. Methods

3.1. Gaussian NLL Loss

In this approach, we give the depth completion task a probabilistic formulation, which enables us to estimate per-pixel depth as well as the corresponding variance. The cornerstone assumption here is that, for an input pixel x, its corresponding depth $d_x \sim \mathcal{N}(\mu_x, \sigma_x^2)$. On a larger scale, for each input image $X \in \mathbb{R}^{H \times W \times 3}$ and sparse depth map $D \in \mathbb{R}^{H \times W}$, we need to output $\mu \in \mathbb{R}^{H \times W}$ and $\sigma^2 \in \mathbb{R}^{H \times W}$, pixel mean and pixel variance maps, respectively. This is accomplished by modifying the base MSG-CHN as described in Fig. 1, Fig. 2c and Fig. 2d. The modified network is trained with

$$\ln P(\mu|X, Y) = \sum_x \ln P(\mu_x|x, y_x) \tag{1}$$

the derivation of which is described in the Appendix A.

3.2. Laplacian NLL Loss

Similar to the Gaussian NLL approach, we can take an alternative assumption on the depths d_x of the pixels, where we again assume they are independent of each other, but come from a Laplace distribution with parameters μ_x and σ_x. While the architecture remains as in the Gaussian NLL case, the log-likelihood of individual pixel changes, that is described mathematically in the Appendix A. The total loss formula also remains the same as in Eq. (1).

3.3. Monte Carlo (MC) Dropout

It has been shown in [7] that the use of dropout in Neural Networks (NNs) can be interpreted as a Bayesian approximation of Gaussian Processes (GP). Consequently, model uncertainty can be obtained from dropout NN models.

This Gaussian explanation helps approximate the predictive distribution of the depth maps and estimate the mean and variance maps using the Monte Carlo (MC) method as described in Appendix B. In practice this is equivalent to performing T stochastic forward passes through the network and computing the mean and the variance of the outputs. We accomplish this by modifying the network as described in Fig. 1 and Fig. 2b and perform 20 forward passes through the network for each sample. We also fix the dropout probability to 0.25 for all experiments.

3.4. Evaluation Method for the Uncertainty Estimations

We evaluate uncertainty estimates by following the idea [3] that the predicted uncertainty of each pixel x, measured through standard deviation σ_x, matches the root mean squared error (RMSE) of the predicted depth. More formally,

$$\forall \sigma_x : \mathbb{E}_{x,y_x}[(\mu_x - y_x)^2 | \sigma_x^2] = \sigma_x^2. \tag{2}$$

We expect each σ_x^2 in our dataset to appear exactly once, therefore we evaluate Eq. (1) empirically using binning. Formally, let σ_t be the standard deviation of predicted output PDF p_t and assume that the examples are ordered by increasing values of σ_t. Let further N be the number of bins, that divides the number of examples, T. We divide the indices of the examples to N bins, $\{B_j\}_{j=1}^N$, such that: $B_j = \{(j-1) \cdot \frac{T}{N} + 1, ..., j \cdot \frac{T}{N}\}$.

To evaluate how calibrated the model is, we compare per bin j two quantities: the root of the mean variance

$$RMV(j) = \sqrt{\frac{1}{|B_j|} \sum_{t \in B_j} \sigma_t^2} \tag{3}$$

and the empirical root mean square error

$$RMSE(j) = \sqrt{\frac{1}{|B_j|} \sum_{t \in B_j} (y_t - \hat{y}_t)^2} \tag{4}$$

where $\hat{y}$ is the mean of the predicted PDF (p_t).

For diagnosis, we use a reliability diagram that plots the RMSE as a function of the RMV. For a calibrated model, the two should be approximately equal per bin, and hence the plot should be close to the identity function.

3.5. Uncertainty Calibration

It is natural for the trained network to have a slight increase in the MSE or MAE scores on the test set, similarly for the predicted variances it is common to have a scale mismatch on the test set. To address this issue we employ simple temperature scaling technique. For this purpose we keep a re-calibration set initially separated from the validation set. We assume that there is a constant s the same for all pixels such that $s\sigma$ results in a calibrated and hence more accurate uncertainty map. To get the constant s we solve the following equation, in case of the Gaussian NLL or MC dropout approaches

$$\frac{N_r}{2} \log s + \sum_{i=1}^{N_r} \log \sigma_i = \sum_{i=1}^{N_r} \frac{(\mu_i - y_i)^2}{2s^2\sigma_i^2} \tag{5}$$

where N_r is the number of data points in the re-calibration set. The idea is to find the constant that will balance the total uncertainty and the total MSE. The equation cannot be trivially solved analytically, so we employ the bisection method to approximate it numerically. The equation can also easily be modified for the Laplacian NLL method.

4. Dataset

We performed our experiments on the KITTI Vision Benchmark Suite. For the depth completion task, the dataset includes sparse depth maps, aligned RGB images, and semi-dense ground truth. It consists of around 86K training, 7K validation, and 1K unlabeled test data.

5. Experiments

We run all our experiments utilizing the whole training set of the KITTI dataset and validate on the selected validation set that is a 1K data points sample from the validation set. All trainings are performed on a single NVIDIA GTX 1080 GPU.

First we will discuss the MAE scores and qualitative results of the main task – the depth prediction. Then we will discuss the quality of the predicted uncertainties according to the described evaluation method.

5.1. Depth completion results

We train MSG-CHN with the modifications of architectures described in Fig. 2 with both Gaussian and Laplacian NLL losses. We will refer to the three best performing models trained with NLL loss as follows: MSG-CHN-G for the model with Fig. 2c decoder architecture trained with a Gaussian NLL loss, MSG-CHN-L for the model with Fig. 2c decoder architecture trained with a Laplacian NLL loss and MSG-CHN-L-LARGE for the model with Fig. 2d decoder architecture trained with a Laplacian NLL loss. Tab. 1 shows that changing only the loss function from the Gaussian to Laplacian drastically improves the results of the depth completion. This could be an indicator of the wrong assumption that the depths of individual pixels are coming from a Gaussian distribution. The improvement can also be visually seen in the Fig. 4. In the zoomed in section the boundaries between cars are much sharper for the MSG-CHN-L-LARGE network outputs, we clearly see that the further a car is, the lighter its color becomes. In contrast, the MSG-CHN-G outputs are blurry, not only for the pixels in the far but also for the close ones.

Initially the NLL approach of training was adopted as a separate method of uncertainty estimation. But due to the success of the Laplacian NLL, that can be seen in Tab. 1, this approach can also be considered as a better way of the network training. So the joint estimation of pixel depths and uncertainties results in better depth estimation.

MAEs of the models	
Architecture	MAE
MSG-CHN	215.14
MSG-CHN-G	240.91
MSG-CHN-L	**200.11**
MSG-CHN-L-LARGE	**194.84**
MSG-CHN-MC-3	228.03
MSG-CHN-MC-23	229.15
MSG-CHN-MC-123	230.06

Table 1. Evaluation results of the models.

5.2. Uncertainty estimation results

For the MC dropout approach we perform 3 experiments, a dropout layer is added in the decoder of a pre-trained network (Fig. 2b). The experiments are as follows: dropout added only in the third decoder (MSG-CHN-MC-3), dropout added in the last two decoders (MSG-CHN-MC-23), and dropout added in all three decoders (MSG-CHN-MC-123). In all experiments the dropout rate is set to 0.25. As you can see in Tab. 1, the more dropout layers we add, the worse MAE gets. Additionally examining the RMV vs RMSE plots we conclude that MSG-CHN-MC-3 is the best model utilizing MC dropout technique, as it has equivalent uncertainty estimation to the other 2 MC dropout networks.

We compare the estimated uncertainties of the best model trained with NLL approach, MSG-CHN-L-LARGE, with the MSG-CHN-MC-3. In Fig. 3 we visualize their uncertainty evaluations, and also how the calibration helps in case of MSG-CHN-MC-3. It is clear that the uncertainties estimated with MC dropout are more reliable as there is a clear linear correlation between the RMV and RMSE, and it becomes almost perfect identity after the calibration with temperature scaling. While in case of MSG-CHN-L-LARGE, there are some intervals of negative correlation and some of positive ones, which makes it unreliable and which cannot be corrected with the temperature scaling. We further discuss the uncertainty estimation on output visualizations in Appendix C.

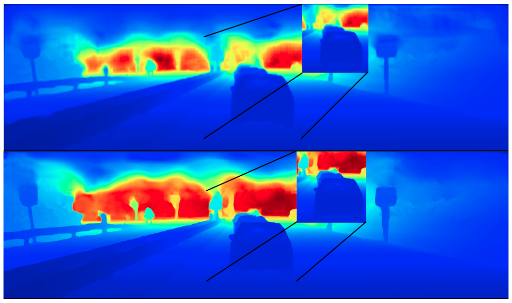
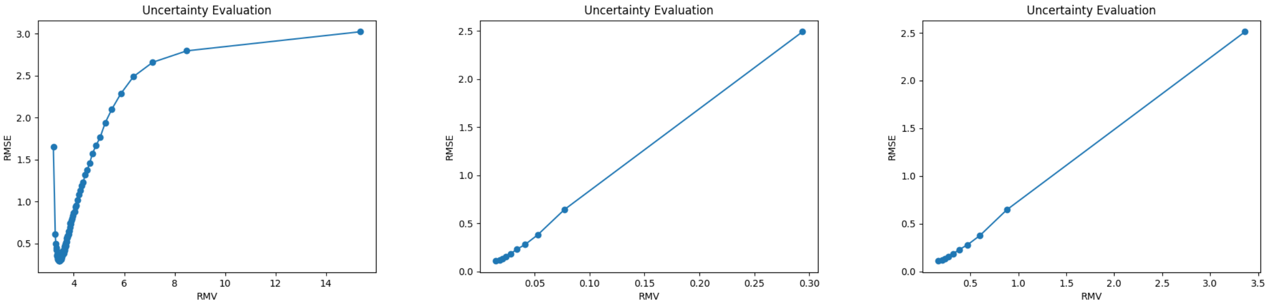


Figure 4. Above: output depth map of the network MSG-CHN-G. Below: output depth map of the network MSG-CHN-L-LARGE.

6. Conclusion

Given the experiment results, we propose the following pipeline which will both improve the depth estimation of the MSG-CHN and yield reliable uncertainty estimates: First, we train a MSG-CHN-L-LARGE model with Laplacian NLL loss. At inference time we query this model to obtain depth estimates, then we add MC dropout in the last decoder to estimate uncertainty, followed by the calibration step. This way, we decrease the MAE by 20.3 compared to the baseline MSG-CHN and on top of that obtain reliable uncertainty estimates, which are crucial for safety in autonomous driving.



(a) MSG-CHN-L-LARGE

(b) MSG-CHN-MC-3 (before calibration)

(c) MSG-CHN-MC-3 (after calibration)

Figure 3. RMV vs RMSE plots for MSG-CHN-L-LARGE and MSG-CHN-MC-3.

Appendices

A. Derivation of Gaussian and Laplacian NLLs

The Gaussian NLL loss function is derived from the initial assumption that the depth of each pixel $d_x \sim \mathcal{N}(\mu_x, \sigma_x^2)$

$$P(\mu_x|x, y_x) = \frac{1}{\sqrt{2\pi\sigma_x^2}} e^{-\frac{(\mu_x - y_x)^2}{2\sigma_x^2}} \tag{6}$$

where y_x is the observed depth of the pixel x. Hence the log-likelihood of the pixel is

$$\ln P(\mu_x|x, y_x) = -\frac{1}{2}\ln\sigma_x^2 - \frac{(\mu_x - y_x)^2}{2\sigma_x^2} + const \tag{7}$$

The simplifying assumption that the depths of pixels are independent leads to the loss function for the whole depth map as in Eq. (1).

The Laplacian NLL loss function is derived from the initial assumption that the depth of each pixel $d_x \sim Laplace(\mu_x, \sigma_x)$

$$P(\mu_x|x, y_x) = \frac{1}{2\sigma_x} e^{-\frac{|\mu_x - y_x|}{\sigma_x}} \tag{8}$$

similarly the log-likelihood becomes

$$\ln P(\mu_x|x, y_x) = -\ln\sigma_x - \frac{|\mu_x - y_x|}{\sigma_x} + const \tag{9}$$

The independence assumption again leads to the loss for the whole depth map as in Eq. (1).

B. Derivation of MC Dropout

Let $\hat{\mathbf{y}}$ be the output of a NN model with L layers, and $\mathbf{W}_i$ the NN's weight matrices of dimensions $K_i \times K_{i-1}$ for each layer $i = 1, ..., L$. We denote by $\hat{\mathbf{y}}_i$ the observed output corresponding to input $\hat{\mathbf{x}}_i$ for $1 \leq i \leq N$ data points, and the input and output sets as $\mathbf{X}$, $\mathbf{Y}$.

Given a set of random matrices $\omega = \{\mathbf{W}_i\}_{i=1}^{L}$ for a model with L layers, we have an approximate predictive distribution given by

$$q(\mathbf{y}^*|\mathbf{x}^*) = \int p(\mathbf{y}^*|\mathbf{x}^*, \omega) q(\omega) d\omega \tag{10}$$

where the variational distribution $q(\omega)$ is defined by

$$\mathbf{W}_i = \mathbf{M}_i \cdot \text{diag}([z_{i,j}]_{j=1}^{K_i}) \tag{11}$$

$$z_{i,j} \sim \text{Bernoulli}(p_i) \text{ for } i = 1, ..., L,\ j = 1, ..., K_{i-1} \tag{12}$$

given some probabilities p_i and matrices $\mathbf{M}_i$ as variational parameters. The binary variable $z_{i,j} = 0$ corresponds to unit j in layer $i - 1$ being dropped out as an input to layer i.

We perform moment-matching and estimate the first two moments of the predictive distribution empirically. More specifically, we sample T sets of vectors of realizations from the Bernoulli distribution $\{z_t^1, ..., z_t^L\}_{t=1}^{T}$ with $z_i^t = [z_{i,j}^t]_{j=1}^{K_i}$, giving $\{\mathbf{W}_1^t, ..., \mathbf{W}_L^t\}_{t=1}^{L}$. We get

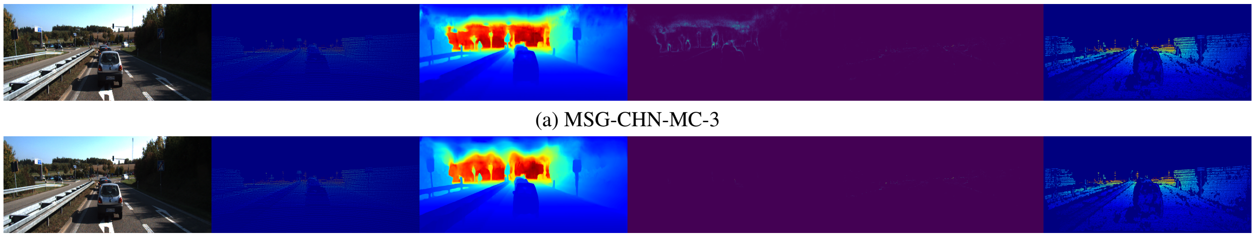
$$\mathbb{E}_{q(\mathbf{y}^*|\mathbf{x}^*)}(\mathbf{y}^*) \approx \frac{1}{T}\sum_{t=1}^{T} \hat{\mathbf{y}}^*(\mathbf{x}^*, \mathbf{W}_1^t, ..., \mathbf{W}_L^t) \tag{13}$$

and we refer to this Monte Carlo estimate as MC dropout. After estimating the second moment as well, we obtain the model's predictive variance as

$$\begin{aligned} &\text{Var}_{q(\mathbf{y}^*|\mathbf{x}^*)}(\mathbf{y}^*) \approx \tau^{-1}\mathbf{I}_D \\ &+ \frac{1}{T}\sum_{t=1}^{T} \hat{\mathbf{y}}^*(\mathbf{x}^*, \mathbf{W}_1^t, ..., \mathbf{W}_L^t)^T \hat{\mathbf{y}}^*(\mathbf{x}^*, \mathbf{W}_1^t, ..., \mathbf{W}_L^t) \\ &- \mathbb{E}_{q(\mathbf{y}^*|\mathbf{x}^*)}(\mathbf{y}^*)^T \mathbb{E}_{q(\mathbf{y}^*|\mathbf{x}^*)}(\mathbf{y}^*) \end{aligned} \tag{14}$$

where τ is a precision hyperparameter that can be recovered using a weight-decay parameter λ and length-scale parameter l as follows

$$\tau = \frac{pl^2}{2N\lambda}. \tag{15}$$



(a) MSG-CHN-MC-3

(b) MSG-CHN-L-LARGE

Figure 5. Visualizations of model outputs. From left to right are depicted: the input RGB image, the input sparse depth map, the output completed depth map, the predicted uncertainty map, the MAE error map between the predicted depth map and the semi-dense ground truth, and finally the semi-dense ground truth depth map.

C. Further visualizations and discussions

Fig. 5 contains visualizations of both uncertainty maps and the MAE error map, that can be utilized for visual evaluation of the uncertainty estimation performances. The main drawback of the error map is that it is also semi-dense which should be considered when comparing its similarity to the uncertainty map. If zoomed in, it becomes noticeable that most of the bright pixels in the error map are part of object boundaries, or objects in the far. This is also the case in the uncertainty map in Fig. 5a. There are also large bright clusters of pixels in the uncertainty map in Fig. 5a that fall in regions where the ground truth depths are missing, so we cannot conclude about their quality. In contrast the uncertainty map in Fig. 5b contains significantly less bright pixels which speaks about the overconfidence of the trained model MSG-CHN-L-LARGE. If examined meticulously one can identify several regions of high brightness which don't correspond to any bright region of the corresponding error map. Hence besides the evaluation procedure of the uncertainty estimates, we notice also visually that estimates of MSG-CHN-MC-3 are better than the ones of MSG-CHN-L-LARGE.

References

[1] Yarin Gal and Zoubin Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. *Proceedings of The 33rd International Conference on Machine Learning*, 06 2015. 2

[2] Mu Hu, Shuling Wang, Bin Li, Shiyu Ning, Li Fan, and Xiaojin Gong. Towards precise and efficient image guided depth completion. 2021. 1

[3] Dan Levi, Liran Gispan, and Ethan Fetaya Niv Giladi. Evaluating and calibrating uncertainty prediction in regression tasks. 2020. 3

[4] Ang Li, Zejian Yuan, Yonggen Ling, Wanchao Chi, Chong Zhang, et al. A multi-scale guided cascade hourglass network for depth completion. In *The IEEE Winter Conference on Applications of Computer Vision*, pages 32–40, 2020. 1, 2

[5] Fangchang Ma and Sertac Karaman. Sparse-to-dense: Depth prediction from sparse depth samples and a single image. 2018. 1

[6] D.A. Nix and A.S. Weigend. Estimating the mean and variance of the target probability distribution. In *Proceedings of 1994 IEEE International Conference on Neural Networks (ICNN'94)*, volume 1, pages 55–60 vol.1, 1994. 2

[7] Carl Edward Rasmussen and Christopher K. I. Williams. *Gaussian Processes for Machine Learning (Adaptive Computation and Machine Learning series)*. The MIT Press, 2006. 2

[8] Jonas Uhrig, Nick Schneider, Lukas Schneider, Uwe Franke, Thomas Brox, and Andreas Geiger. Sparsity invariant CNNs. In *International Conference on 3D Vision (3DV)*, 2017. 1

[9] N. Yang, L. von Stumberg, R. Wang, and D. Cremers. D3vo: Deep depth, deep pose and deep uncertainty for monocular visual odometry. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. 2