Smaller Residual Network for Single Image Depth Estimation

Andi HENDRA (a) and Yasushi KANAZAWA (b), Members

SUMMARY
We propose a new framework for estimating depth information from a single image. Our framework is relatively small and straightforward by employing a two-stage architecture: a residual network and a simple decoder network. Our residual network in this paper is a remodel of the original ResNet-50 architecture, which consists of only thirty-eight convolution layers in the residual block following by pair of two up-sampling and layers. While the simple decoder network, stack of five convolution layers, accepts the initial depth to be refined as the final output depth. During training, we monitor the loss behavior and adjust the learning rate hyperparameter in order to improve the performance. Furthermore, instead of using a single common pixel-wise loss, we also compute loss based on gradient-direction, and their structure similarity. This setting in our network can significantly reduce the number of network parameters, and simultaneously get a more accurate image depth map. The performance of our approach has been evaluated by conducting both quantitative and qualitative comparisons with several prior related methods on the publicly NYU and KITTI datasets.

key words: depth estimation, single image, deep learning, smaller model, residual network, two-stage model, multi-task loss, learning rate adjustment

1. Introduction

Depth estimation has a significant role in providing geometric information from a scene and has been used in many 3-D vision applications, to name a few, 3-D structure reconstruction [1]–[4], robotics [5], [6], and autonomous driving [7], [8]. In a frame, depth from an image can be defined as the distance of the object’s surface from the viewpoint. It can generally be obtained using highly advanced depth sensors such as stereo camera, RGB-D sensor, LiDAR, or other laser sensors. Despite using those expensive devices, computer vision researchers have been introducing a stereo image procedure to estimate depth that has reported satisfactory result. However, the popular stereo method, such as structure from motion [9], [10], structured light [11], and defocusing method [12] have limitation to the model in the case of only one view available at the test time.

To overcome this problem, various depth estimation algorithms have been introduced to handle a single view image and remain a challenging task in computer vision. Numerous efforts have been made to address these problems, whether using a conventional method or deep learning approach. Nevertheless, the famous conventional works, such as Markov Random Field (MRF) or Conditional Random Field (CRF) mainly relies on prior geometric information, which cannot be applied for general scene depth estimation. Recently, the prosperity of the depth prediction from a single image has been witnessed by utilizing deep neural networks. Employing deep neural networks to learn a relationship between image and their corresponding depth, Eigen et al. [13] work has encouraged the modern trends of single image depth estimation.

This paper proposes an effective, simple, and more accurate two-stage deep learning architecture for robust single image depth estimation encourages by Eigen et al. [13]. Differ with [13] which is using a standard CNN such as VGG or AlexNet in the first stage, our method computes feature depth from the input RGB image using a modified residual network (ResNet) based on the work in [14]. In addition to this, we employ up-sampling layer instead of fully connected layer in the last layer. Meanwhile, in the second stage, we stack 2-stride convolution layer to down-sample RGB image before concatenating it with the initial output depth.

We also utilize multi-task loss optimization to enhance the model accuracy, as outline in detail in Sect. 3.2. Furthermore, to be able to manage the speed of the neural network while the model is updating its weight, we monitor the network performance during training and adjust its learning rate parameter, as described in Sect. 3.3.

Since both models on each stage consist of a relatively shallow layer and small filter size, we can train our model with very few numbers of parameters. We confirm that the model accomplishes well to reduce training time computation while concurrently maintain to get a more accurate image depth prediction result.

The rest of this paper is organized as follows. Section 2 reviews several related works. The theory and procedure of the proposed method are described in Sect. 3. Section 4 shows the implementation of our strategy. We describe our setup for the experiment in Sect. 5. The result of our method is discussed in Sect. 6, including a comparison with the previous works. Finally, the paper is concluded in Sect. 7.

2. Related Works

In this section, we will review some of the previous related works in image depth estimation, either using deep learning
or a non-deep learning strategy. In the following, we provide brief introductions to several closely related methods.

Extracted depth information from the image has initially used a non-deep learning method relied on the stereo vision approach, such as structure from motion [9], [10], structured light [11], and defocusing method [12] which are utilizing image pairs of the same scene. On the other hand, a variety of approaches had been demonstrated the depth map prediction from a single RGB image. Following the completion of the work in [15], compose the geometric structure of the image region to reconstruct a 3-D model of outdoor images, Saxena et al. [16] had been successfully estimated depth from a single image based on learning MRF model. The achievement of their method, later extended to develop a 3-D reconstruction of scene modeling [17]. However, their approach relies on strong assumptions about scene geometry, which is work only in a particular scenario.

Recently, many researchers attempted to solve the single image depth estimation problem using Convolutional Neural Networks (CNN). Liu et al. [18] proposed image depth estimation formulating into a continuous conditional random field (CRF) approach. Their method discover the unary and pairwise potential of continuous CRF and train with a CNN network. Laina et al. [19] proposed a fully convolutional architecture to learn feature map up-sampling to produce higher resolution output dense maps. Cao et al. [20] proposed a distinctive approach to estimate depth from a single image. They utilized a fully convolutional residual networks as a classification task instead of using a standard regression procedure. Godard et al. [21] proposed unsupervised learning for a monocular image depth estimation using a deep CNN network. In their works, they generated disparity images employing a left-right consistency image reconstruction loss.

On top of those methods, the work of Eigen et al. [13], which adopted multi-scale architecture, reported remarkable results, and motivated research towards the use of the deep neural network for depth estimation. In their method, they implemented global and local information to regress dense depth maps to predict depth from an image. The initial scale plays a significant role in predicting the depth of the image using the global feature information. At the same time, the following stage is employed to improve the completion of the generated image depth map.

In this works, we propose a robust two-stage framework architecture which is faster, simpler, and achieve a more accurate result. We reconstruct residual networks (ResNet-50) following two blocks of the up-sampling layer in the first stage. While in the second stage, we stacked five instead of four convolutional layers and reduced the filter size, which is proved effectively improved the final depth prediction. Since both networks on each stage consist of a very shallow layer and small filter size, our model is inexpensive with respect to the number of network parameters and produces a more accurate output image depth prediction.

3. Proposed Method

In this section, we describe our depth estimation method in detail. We first introduce the network architecture, followed by the description of our optimization loss function. We also describe the procedure of learning rate hyperparameter adjustment utilized in our implementation. Finally, we define our augmentation procedure to transform image data to enrich our input feature to handle common overfitting problems.

3.1 Network Architecture

We formulate our model depth estimation as a two-stage deep neural network. The network architecture outline of our depth estimation model is illustrated in Fig. 1.

We employ a residual network for computing feature depth maps in the first stage. Here, we modified the original form of ResNet-50, wrapped only thirty-eight convolution layers in six residual blocks. We call this new network as remodeled ResNet-50 to avoid ambiguity with the original form. Then, we incorporate an up-sampling layer following by two blocks of convolution layer. We repeat this setting to handle the desired output map size, which has been proved effective for regression problems.

The second stage of our architecture guides the network into enhancing its output through a stack of convolution layers. We implement five instead of only four convolution layers. One of the main differences compared with [13] is the number of convolution layers before concatenating the two inputs (RGB image and initial output depth). Rather than using a stride-2 convolution layer following a max-pooling layer, we utilize stride-2 convolution in the first and second blocks to downscale the input RGB to match the initial generated depth size. Further, we reduce the kernel size from $9 \times 9$ to $7 \times 7$ to ease the process of time
Fig. 2 The detail layers of the first stage network.

Fig. 3 The detail layer of the second stage network.

computation.

Details of our proposed architecture can be seen in Fig. 2 and Fig. 3. The first part of the network is a transformed model of ResNet-50 and initialize with random weights. There are thirty-eight convolution layers employ in the residual block to encode the RGB input image into a feature vector. A max-pooling layer, together with the first convolution layer with stride-2, is utilized to downscale the input image by a factor of 4. Then we pass these output into a residual blocks and repeat the operation six times to downscale the previous output by a factor of 2. Two stack pairs of up-sampling and convolution layers are then provided before fusing the residual network’s output to the second stage network. The output size of the last residual layer will be 1/4 compared to the input RGB image.

In the second stage, we stack five blocks of convolutional layer. A stride-2 convolution is provided in the first and second block to downscale the input by a factor of 4 to match the output size from the previous stage. We use 7 × 7 filter size in the first and second block, and 5 × 5 in the remaining three blocks. The rectified linear unit (ReLU) activation function is utilized to the output of each convolution layer but in the last layer. The convolution with the filter number of one is applied in the last layer, followed by a linear activation function to refine the feature depth maps into dense depth.

3.2 Multi Loss Function

The standard loss function for depth regression problems considers the difference between the ground truth depth map \( y_t \), and the prediction of the depth regression network \( y_p \). The most common loss function used in the deep learning for regression task is mean square error or \( L_2 \) loss. These loss function measure the average of the squared difference between prediction and actual observation. However, \( L_2 \)
loss alone has insufficiency to predict data that are far away from its real values.

In our work, we implement a combination of multi-task loss functions to improve performance of the model. This idea was encouraged the practice from the work in [22] to compute loss based on pixel-wise, gradient-based, and the structured similarity. These combination losses have been proved beneficial for neural network learning optimization and demonstrated overall adequate depth performance.

However, differ from their work, instead of using mean absolute error (MAE) for computing the pixel-wise error, we utilized the Reversed Huber in Eq. (1) which is more flexible than MAE loss function that can behave either as a mean absolute error (MAE) or mean squared error (MSE) depends on the prediction error result.

\[
L_h = \begin{cases} 
|y_t - y_p|, & |y_t - y_p| \leq c, \\
\frac{1}{2c}(|y_t - y_p|^2 - c^2), & |y_t - y_p| > c,
\end{cases}
\]

where \(c = 0.2 \max(|y_t - y_p|)\).

Equation (2) presents a loss function which estimate prediction error considering its gradient direction. The gradient of ground truth depth and the depth prediction is computed along \(x\), and \(y\) direction then calculate its average.

\[
L_g = \frac{1}{N} \sum_{p} [g_x(y_t, y_p)] + [g_y(y_t, y_p)],
\]

where \(g_x\) and \(g_y\) are the gradient along \(x\) and \(y\), respectively.

The third loss function we used in this research is a structure similarity index (SSIM) loss, as presented in Eq. (3). While \(L_h\) and \(L_g\) compute error for each pixel between ground truth and generated depth, \(L_s\) measure similarity within pixels in the score range \([-1, 1]\). Eventually, the SSIM loss will compute the perceptual difference based on the visible structure of the ground truth and predicted image.

\[
L_s = 0.5 - \frac{\text{ssim}(y_t, y_p)}{2},
\]

where \(\text{ssim}(y_t, y_p) = \frac{(2\mu_y\mu_p + C_1)(2\sigma_{yp} + C_2)}{(\mu_y^2 + \mu_p^2 + C_1)(\sigma_y^2 + \sigma_p^2 + C_2)}\).

With:

\(\mu_{y_t}\) and \(\mu_{y_p}\), are the average of \(y_t\) and \(y_p\), respectively.

\(\sigma_{y_t}\) and \(\sigma_{y_p}\) are the variance of \(y_t\) and \(y_p\), respectively.

\(\sigma_{yp}\) is the covariance of \(y_t\) and \(y_p\).

We defined the loss function for training our model as the weighted sum of the three loss functions defined in Eqs. (1), (2), and (3) as:

\[
L_M = w_1L_h + w_2L_g + w_3L_s,
\]

where value of \(w_1 = w_2 = 1\) and \(w_3 = 0.1\) were arbitrarily initiate by experimentation.

3.3 Learning Rate Adjustment

One of the most challenging tasks of training deep learning neural networks comprises carefully deciding a reasonable learning rate. The learning rate is one of the critical hyper-parameters in the model, manages the speed of the neural network model to update their weights. On that account, it is necessary to provide a decent learning rate value to improve the performance while the model is training the data.

Unfortunately, the optimal learning rate cannot determine theoretically. This parameter must be observed by experimenting. Typically, a big learning rate enables the model to learn faster and converge to its local minima solution. In contrast, an extremely small learning rate may raise a lengthy training process, or the training process may get stuck. In this works, we originate our initial learning rate value to \(10^{-3}\) then reduce our learning rate depend on the loss behavior.

3.4 Overfitting and Augmentation

Overfitting is a frequent cause of the less accurate performance of general deep learning problem due to the lack of control over the learning process of the model. Training the model with more data can help the algorithm to learn more representative of input features. In recent years, the most recommended technique to enrich the characteristic of data is employing data augmentation procedure, which provides additional transformed image data.

In our training process, we practiced augmentation strategies on the fly to enrich our inputs’ feature. Particularly, with the ratio of 0.5, we randomly rearrange the channel of the input RGB images. We further applied 0.25 ratio for implementing Gaussian images to our input RGB images. Finally, affine transformation for both RGB images and depths, a horizontal flipping procedure is also performed with probability of 0.25.

4. Implementation

In this section, we describe the detailed implementation of our proposed method. First, we define the dataset used in this experiment, then explain the training process details we performed. Lastly, to confirm the achievement of our network, we represent the evaluation metric commonly used in some prior works.

4.1 Datasets

We train our model using two popular publicly depth dataset, i.e. NYU Depth v2 [23] and KITTI data [24].

The KITTI dataset contains outdoor scenes with images resolution about 376 \times 1241 captured by cameras and depth sensors in a driving car. It contains over 93K depth maps from 56 scenes with corresponding raw LiDAR scans and RGB images. We train our method on 17K images from the random scene and set depth upper bound of 80 meters. We test our model on the 697 images which are not included in the training, following the split by works of [13].
The NYU Depth v2 is an indoor dataset gathered using a Microsoft Kinect camera with resolution 640 × 480. It contains 120K raw RGB images and their corresponding depth from 464 different scenes. Only 50K images from a random scene are used for training our network. To validate the performance of our method, following the works of [13], we test on 654 from 1449 available densely labeled pairs of aligned RGB and ground-truth depth images in the maximum depth of 10 meters.

4.2 Training Details

We implement our proposed approach using the Keras framework [25] with the Tensorflow backend. Training is done on Ubuntu 16.04 with an NVIDIA GeForce GTX 1080 GPU with 8 GB memory. The network is shown in Fig. 2 and Fig. 3 has been trained using initialized random weight with approximately 12 million trainable parameters. The training is performed using 16 mini-batches and load images and their depth using an online generator for GPU memory performance.

We train our model using an adaptive moment estimation (Adam) optimizer and an initial learning rate of 10−3. During training, we monitor the model’s performance and adjust the learning rate hyperparameter depending on the loss behavior. Specifically, we degraded our learning rate by order of magnitude divided by ten on epoch 45, 60, and 75, respectively. For some other details, we trained our proposed architecture on the multi-task loss function, as described in Sect. 3.2.

4.3 Evaluation

We evaluate our model with several popular related methods to show the performance of our proposed depth estimation approach on publicly RGB-D NYU depth v2 and KITTI dataset. We adopt the following evaluation metrics, commonly used in the previous works, to evaluate our depth prediction model’s performance quantitatively. Specifically, we assess our method using metrics based on its error rate and accuracy in Eqs. (5), (6), (7), (8), (9), and (10).

- Root mean squared error (RMS):
  \[
  \sqrt{\frac{1}{N} \sum_{y_p \in \mathcal{N}} |y_p - y_t|^2}.
  \]

- Average log10 error (LOG10):
  \[
  \frac{1}{N} \sum_{y_p \in \mathcal{N}} |\log_{10}(y_p) - \log_{10}(y_t)|.
  \]

- Average relative error (REL):
  \[
  \frac{1}{N} \sum_{y_p \in \mathcal{N}} \frac{|y_p - y_t|}{y_t}.
  \]

- Root mean squared log error (RMS LOG):
  \[
  \sqrt{\frac{1}{N} \sum_{y_p \in \mathcal{N}} |\log y_p - \log y_t|^2}.
  \]

- Squared relative error (SQ REL):
  \[
  \frac{1}{N} \sum_{y_p \in \mathcal{N}} \frac{|y_p - y_t|^2}{y_t}.
  \]

- Accuracy with threshold ($P_{th}$): percentage (%) of $y_p$ to max($y_p^{\#,y_t}$) = $\delta < P_{th}$,
  \[
  P_{th} \in \{1.25, 1.25^2, 1.25^3\}.
  \]

Here, $y_p$ and $y_t$ are the predicted and ground-truth depth respectively, and $N$ is the total number of pixel.

5. Experiment

To demonstrate the performance of the proposed models for a single depth prediction, we conducted experiments on the publicly available NYU Depth v2 [23] and KITTI dataset [24] with our network using the Keras Framework [25]. We implement two different base models, remodeled ResNet-18 and remodeled ResNet-50. Our remodeled ResNet-18 consists of only fifteen convolution layers in three residual blocks. While the remodeled ResNet-50 consists of thirty-eight convolutions wrapped in six residual blocks.

5.1 Ablation Study

We perform ablation studies with different tasks considering loss function and learning rate adjustment for a detailed analysis of our framework. We train our framework with various tasks to observe the performance of the evaluation metric of interest. We implement several experimental ablation tasks, including train our model using two different loss, mean square error loss ($L_2$) and multi-tasks loss ($L_M$). To show the learning rate tune parameter’s performance, we examine the other two tasks by utilizing learning adjustment to both $L_2$ and $L_M$ losses.

We compare the performance on the NYU Depth v2 data and report the accuracy and error rate in Table 1. In these tables, $L_2^{\#}$ shows the model using $L_2$ without learning rate adjustment, $L_2^w$ shows the model using $L_2$ with learning rate adjustment, $L_M^{\#}$ shows the model using $L_M$ without learning rate adjustment, and $L_M^w$ shows the model using $L_M$ with a learning rate adjustment. Furthermore, in Fig. 4, we provide a visualization of the predicted depth to demonstrate the effectiveness of our proposed procedure.

From these results, we can see the implementation of $L_M$ achieves better results than the standard $L_2$. Likewise, utilizing learning rate adjustment can significantly enhance our proposed method’s performance in terms of accuracy and error rate. We can affirm that adopted the multi-task loss functions able to increase the accuracy as well as reduce.
Table 1 Ablation study on NYU depth V2 dataset.

| Model                  | Accuracy<1.25 | Accuracy<1.25² | Accuracy<1.25³ | Error Rate RMS | Error Rate LOG10 | Error Rate REL |
|------------------------|---------------|----------------|----------------|----------------|------------------|----------------|
| Remodeled ResNet-18+L₂ | 0.616         | 0.871          | 0.960          | 0.777          | 0.096            | 0.229          |
| Remodeled ResNet-18+L₂ ² | 0.615         | 0.875          | 0.962          | 0.770          | 0.095            | 0.221          |
| Remodeled ResNet-18+L₂ ³ | 0.629         | 0.884          | 0.966          | 0.745          | 0.092            | 0.212          |
| Remodeled ResNet-18+L₂ ⁴ | 0.642         | 0.889          | 0.969          | 0.726          | 0.090            | 0.209          |
| Remodeled ResNet-50+L₂ ⁴ | 0.784         | 0.947          | 0.984          | 0.547          | 0.065            | 0.153          |

*the higher the better.
**the lower the better.

Table 2 Accuracy comparison with previous works on NYU Depth v2.

| Method               | Accuracy<1.25 | Accuracy<1.25² | Accuracy<1.25³ |
|----------------------|---------------|----------------|----------------|
| Saxena et al. [17]   | 0.447         | 0.745          | 0.897          |
| Karsch et al. [26]   | —             | —              | —              |
| Ladicky et al. [27]  | 0.542         | 0.829          | 0.941          |
| Konrad et al. [28]   | —             | —              | —              |
| Liu et al. [29]      | —             | —              | —              |
| Eigen et al. [13]    | 0.611         | 0.887          | 0.971          |
| Wang et al. [30]     | 0.605         | 0.890          | 0.970          |
| Roy et al. [31]      | —             | —              | —              |
| Ours L²              | **0.784**     | **0.947**      | **0.984**      |

*the higher the better.

5.2 Comparison with Existing Methods

For a detailed analysis of our method, we compare with several baseline methods and report the results of accuracy and error rate in Tables 2 and 3 for NYU Depth v2, and Tables 4 and 5 for KITTI data, respectively.

NYU Depth v2. We examine our model on NYU Depth v2 with eight previous related methods in terms of accuracy as shown in Table 2. It can be stated that our strategy outperforms along with the preceding works [17], [26]–[31], and [13]. Our practice outperforms in terms of accuracy with the threshold (δ < 1.25, δ < 1.25², and δ < 1.25³), as well as the root mean square error (RMS), the average log error (LOG10), and the average relative error (REL) with significant margin as shown also in Table 3. We can affirm that our method is significantly outperforms than all the eight baseline methods on NYU Depth v2 dataset.

In addition, we compare our qualitative result with Eigen et al.’s method in Fig. 5. It can be seen that compared to their output, our method yields more visually satisfying predictions with more visible transitions, aligning to local
Table 3  Error rate comparison with previous works on NYU Depth v2.

|          | RMS  | LOG10 | REL  |
|----------|------|-------|------|
| Saxena et al. [17] | 1.214 | —     | 0.349 |
| Karsch et al. [26]  | 1.200 | 0.131 | 0.350 |
| Ladicky et al. [27] | —     | —     | —    |
| Konrad et al. [28]  | 1.300 | 0.135 | 0.368 |
| Liu et al. [29]     | 1.060 | 0.127 | 0.333 |
| Eigen et al. [13]   | 0.907 | —     | 0.215 |
| Wang et al. [30]    | 0.824 | —     | 0.220 |
| Roy et al. [31]     | 0.774 | —     | 0.187 |
| Ours $L_w^*$         | 0.547 | 0.065 | 0.153 |

*the lower the better.

Table 4  Accuracy comparison with previous works on KITTI.

|          | $\delta < 1.25$ | $\delta < 1.25^2$ | $\delta < 1.25^3$ |
|----------|-----------------|-------------------|-------------------|
| Saxena et al. [17] | 0.601           | 0.820             | 0.926             |
| Liu et al. [32]     | 0.647           | 0.882             | 0.961             |
| Eigen et al. [13]   | 0.692           | 0.899             | 0.967             |
| Godard et al. [21]  | 0.861           | 0.949             | 0.976             |
| Ours $L_w^*$         | 0.888           | 0.982             | 0.995             |

*the higher the better.

Table 5  Error rate comparison with previous works on KITTI.

|          | REL     | SQ REL   | RMS LOG  |
|----------|---------|----------|----------|
| Saxena et al. [17] | 0.280   | 3.012    | 0.361    |
| Liu et al. [32]     | 0.217   | 1.841    | 0.289    |
| Eigen et al. [13]   | 0.190   | 1.515    | 0.270    |
| Godard et al. [21]  | 0.114   | 0.898    | 0.206    |
| Ours $L_w^*$         | 0.109   | 0.651    | 0.147    |

*the lower the better.

Fig. 5 Qualitative comparison result from left to right on NYU depth v2: (a) RGB image, (b) ground truth, (c) Eigen et al. [13], (d) our proposed method. Noted that, pixel with a distance $> 10m$ are masked out.

6. Discussion

We proposed a framework that estimates depth from a single image using deep learning to employ a smaller remodel residual network in the two-stage framework design encouraged by original work of Eigen et al. [13]. However, the drawback of that approach in [13] is the fact that training neural networks with many layers such as VGG or AlexNet can cause the gradient to be too small as it is propagated backward through the network. We achieved this vanishing gradient problem using a remodel residual network as they provide residual connections straight to earlier layers in our first stage network. We also work to overcome the blurry predicted depth as resulted in [13] by utilizing two blocks of up-sampling layer with bi-linear interpolation instead of fully connected layers. Further, we found that incorporating two convolution blocks with a strides-2 before concatenating the two inputs in the second stage benefits the model to perceive more detail of the RGB input images to refine the initial depth much better than the work in [13].

Moreover, we adopted a multi-loss function (Sect. 3.2)
HENDRA and KANAZAWA: SMALLER RESIDUAL NETWORK FOR SINGLE IMAGE DEPTH ESTIMATION

Fig. 6 Qualitative comparison result with Eigen et al. and Godard et al. on KITTI data. (a) RGB image, (b) ground truth, (c) Eigen et al. [13], (d) Godard et al. [21], (e) our proposed method. Noted that, pixel with a distance > 80m are masked out.

Fig. 7 Some depth prediction output of our proposed method from 16 random images on the NYU Depth v2 dataset. The 1st and 4th rows are the input RGBs. The 2nd and 5th rows are the ground truth images. The 3rd and 6th rows are our predicted depths. Noted that, pixels with a distance > 10m are masked out.

and perform training monitoring (Sect. 3.3) for the learning rate parameter adjustment based on their loss behavior. We demonstrated that such a method setting could significantly reduce the number of network parameters while simultaneously maintaining a more accurate result. As can be seen, our model achieves state-of-the-art performance in comparison with prior related works by a significant margin on the NYU Depth v2 data and has shown very reliable achievement on the KITTI dataset.

It can be stated that our method shows adequate performance along with the previous works in the metrics of interest. The global performance of our proposed method confirmed a more accurate prediction depth. In fact, our method requires much fewer network parameters and less amount of input training data. Compare with the work in [13], our model required number of iterations about 234K vs 3.5M and with fewer input training data, 50K vs 120K for the NYU v2 depth data. While, for the KITTI data our model only need about 80K vs 3.5M iteration numbers and required 17K vs 40K samples training data.

For more analysis of our proposed method, we also demonstrated the qualitative visualization result to show that our strategy is more proficient at detecting the major depth structure of the image. In Fig. 5, we compared our predicted depth with the work of [13] on the NYU depth v2 dataset. While, the performance of our depth estimation on the KITTI data along with works of [13] and [21] is shown in Fig. 6. The result designates that our method is applicable to both indoor and outdoor data. Furthermore, Fig. 7 shows performance of our framework to generate some image depth prediction and compare it with their ground truth depth for visualization. We sample 16 images from random scenes to demonstrate the effectiveness of our method on NYU depth v2 dataset. It is demonstrated that our model is sufficient to generate much better depth visualizations, in which some results are very close to the ground-truths.
7. Conclusion and Future Works

In this paper, we proposed a distinct multi-stages architecture in the form of shallow residual network for predicting depth from a single image. Our model achieves a reliable yet better performance along with some previous related works while required fewer iteration number and input training data.

Our result accomplished a reliable performance, which is encouraging our future study that does not require a complicated model to predict a sufficient depth from a single image. Following this success, our future work to improve the performance and develop a robust and straightforward real-time single image depth estimation.

Acknowledgments

This work is supported by C-BEST program of the Japan International Cooperation Agency (JICA) and JSPS KAKENHI Grant Number JP20K06319.

References

[1] A. Saxena, S.H. Chung, and A.Y. Ng, “3-D depth reconstruction from a single still image,” Int. J. Comput. Vis., vol.76, no.1, pp.53–69, Jan. 2008.
[2] C. Frueh and A. Zakhor, “Constructing 3D city models by merging ground-based and airborne views,” Proc. IEEE on CVPR, vol.2, pp.II–562, June 2003.
[3] E. Delage, H. Lee, and A.Y. Ng, “Automatic single-image 3D reconstructions of indoor Manhattan world scenes,” Int. Symp. ISRR, vol.28, pp.305–321, Jan. 2005.
[4] E.B. Sudderth, A. Torralba, W.T. Freeman, and A.S. Willsky, “Depth from familiar objects: a hierarchical model for 3D scenes,” Proc. IEEE on CVPR, vol.2, pp.2410–2417, June 2006.
[5] E. Coupé, F. Moutarde, and S. Manitsaris, “Gesture recognition using a depth camera for human robot collaboration on assembly line,” Procedia Manuf., vol.3, pp.518–525, 2015.
[6] B. Choi, Ç. Mericli, J. Biswas, and M. Veloso, “Fast human detection for indoor mobile robots using depth images,” Proc. IEEE Int. Conf. on Robot. and Autom., pp.1108–1113, May 2013.
[7] Y. Wang, W.-L. Chao, D. Garg, B. Hartharan, M. Campbell, and K.Q. Weinberger. “Pseudo-LiDAR from visual depth estimation: bridging the gap in 3D object detection for autonomous driving,” Proc. IEEE on CVPR, June 2019.
[8] M. Mancini, G. Costante, P. Valigi, and T.A. Ciarfuglia, “Fast robust monocular depth estimation for Obstacle Detection with fully convolutional networks,” Proc. IEEE Int. Conf. Intell. Robot. Syst. (IROS), pp.4296–4303, July 2016.
[9] D. Crandall, A. Owens, N. Snively, and D. Huttenlocher, “Discrete-continuous optimization for large-scale structure from motion,” Proc. IEEE on CVPR, vol.35, pp.3001–3008, June 2011.
[10] H. Ha, S. Im, J. Park, H.-G. Ieon, and I.S. Kweon, “High-quality depth from uncalibrated small motion clip,” Proc. IEEE on CVPR, pp.5413–5421, June 2016.
[11] P. Fua and Y.G. Leclerc, “Object-centered surface reconstruction: combining multi-image stereo and shading,” Int. J. of Comput. Vis., vol.16, pp.35–56, Sept. 1995.
[12] A.N. Rajagopalan, S. Chaudhuri, and U. Mudenagudi, “Depth estimation and image restoration using defocused stereo pairs,” IEEE Trans. Pattern Anal. Mach. Intell., vol.26, no.11, pp.1521–1525, Nov. 2004.
[13] D. Eigen, C. Puhrsch, and R. Fergus, “Depth map prediction from a single image using a multi-scale deep network,” Proc. 27th Int. Conf. on Neural Inf. Process. Syst., vol.2, pp.2366–2374, Dec. 2014.
[14] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” Proc. IEEE on CVPR, pp.770–778, June 2016.
[15] D. Hoiem, A.A. Efros, and M. Hebert, “Automatic photo pop-up,” Proc. ACM SIGGRAPH, pp.577–584, Aug. 2005.
[16] A. Saxena, S.H. Chung, and A.Y. Ng, “Learning depth from single monocular images,” Proc. Int. conf. on Neural Inf. Process. Syst., vol.18, pp.1161–1168, Dec. 2005.
[17] A. Saxena, M. Sun, and A.Y. Ng, “Make3D: Learning 3D scene structure from a single still image,” IEEE Trans. Pattern Anal. Mach. Intell., vol.31, no.5, pp.824–840, May 2009.
[18] F. Liu, C. Shen, and G. Lin, “Deep convolutional neural fields for depth estimation from a single image,” Proc. IEEE on CVPR, pp.5162–5170, June 2015.
[19] I. Laina, C. Rupprecht, V. Belagiannis, F. Tombari, and N. Navab, “Deeper depth prediction with fully convolutional residual networks,” Proc. Int. Conf. on 3D Vision, pp.239–248, Oct. 2016.
[20] Y. Cao, Z. Wu, and C. Shen, “Estimating depth from monocular images as classification using deep fully convolutional residual networks,” IEEE Trans. Circuits Syst. Video Technol., vol.28, no.11, pp.3174–3182, May 2018.
[21] B.G.J.G. Clement and A.O. Mac, “Unsupervised monocular depth estimation with left-right consistency,” Proc. IEEE on CVPR, pp.6602–6611, July 2017.
[22] I. Alhashim and P. Wonka, “High quality monocular depth estimation via transfer learning,” Proc. IEEE on CVPR, pp.9799–9809, Dec. 2018.
[23] N. Silberman, D. Hoiem, P. Kohli, and R. Fergus, “Indoor segmentation and support inference from RGBD images,” Proc. Eur. Conf. on Comput. Vis. (ECCV), vol.7546, pp.746–760, Oct. 2012.
[24] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, “Vision meets robotics: The KITTI dataset,” International Journal of Robotics Research (IJRR), vol.32, no.11, pp.1231–1237, Aug. 2013.
[25] Chollet, Fran, and Others, “Keras,” https://keras.io, accessed March 30, 2020.
[26] K. Karsch, C. Liu, and S.B. Kang, “Depth transfer: depth extraction from video using non-parametric sampling,” IEEE Trans. Pattern Anal. Mach. Intell., vol.1, no.11, pp.2144–2158, Nov. 2014.
[27] L. Ladicky, J. Shi, and M. Pollefeys, “Pulling things out of perspective,” Proc. IEEE on CVPR, pp.89–96, June 2014.
[28] J. Konrad, M. Wang, and P. Ishwar, “2D-to-3D image conversion by learning depth from examples,” Proc. IEEE on CVPR, pp.16–22, June 2012.
[29] M. Liu, M. Salzmann, and X. He, “Discrete-Continuous depth estimation from a single image,” Proc. IEEE on CVPR, pp.716–723, June 2014.
[30] P. Wang, X. Shen, Z. Lin, S. Cohen, B. Price, and A. Yuille, “Towards Unified Depth and Semantic Prediction from a Single Image,” Proc. IEEE on CVPR, pp.2800–2809, June 2015.
[31] A. Roy and S. Todorovic, “Monocular Depth Estimation Using Neural Regression Forest,” Proc. IEEE on CVPR, pp.5306–5314, June 2016.
[32] F. Liu, C. Shen, G. Lin, and I. Reid, “Learning depth from single monocular images using deep convolutional neural fields,” IEEE Trans. Pattern Anal. Mach. Intell., vol.38, no.10, pp.2024–2039, 2016.
Andi Hendra received the M.S. in Information Engineering from Sepuluh Nopember Institute of Technology, Surabaya, Indonesia in 2009. He is currently a Ph.D student in computer science and engineering at Toyohashi University of Technology, Aichi, Japan. His research interests are in the area of image processing and computer vision.

Yasushi Kanazawa received his B.E. and M.S. degree in information engineering from Toyohashi University of Technology in 1985 and 1987, respectively and his Ph.D in information and computer science from Osaka University in 1997. After engaging in research and development of image processing systems at Fuji Electric Co., Tokyo, Japan, and serving as Lecturer of Information and Computer Engineering at Gunma College of Technology, Gunma, Japan, he is currently Associate Professor of computer science and engineering at Toyohashi University of Technology, Aichi, Japan. His research interests include image processing and computer vision.