Learning Fine-Scaled Depth Maps from Single RGB Images

Jun Li
http://cg.cs.uni-bonn.de/

Reinhard Klein
http://cg.cs.uni-bonn.de

Angela Yao
http://cg.cs.uni-bonn.de

Institute of Computer Science II
University of Bonn
Bonn, Germany

Abstract

Inferring the underlying depth map from a single image is an ill-posed and inherently ambiguous problem. State-of-the-art deep learning methods can now estimate accurate depth maps, but when projected into 3D, still lack local detail and are often highly distorted. We propose a multi-scale convolution neural network to learn from single RGB images fine-scaled depth maps that result in realistic 3D reconstructions. To encourage spatial coherency, we introduce spatial coordinate feature maps and a local relative depth constraint. In our network, the three scales are closely integrated with skip fusion layers, making it highly efficient to train with large-scale data. Experiments on the NYU Depth v2 dataset shows that our depth predictions are not only competitive with state-of-the-art but also leading to 3D reconstructions that are accurate and rich with detail.

1 Introduction

Estimating depth for common outdoor or indoor scenes from monocular RGB images is a challenging task in computer vision with widespread applications in scene understanding, depth-aware image editing or re-rendering, 3D modeling, robotics, etc. Given a single RGB image as input, the goal is to predict a dense depth map for each pixel or superpixel. Inferring the underlying depth is an ill-posed and inherently ambiguous problem. In particular, indoor scenes have large textural and structural variations, heavy object occlusions and rich geometric details, all of which contributes to the difficulty of accurate depth estimation.

To deal with the ambiguity of depth estimation from a single image, previous works relied on strong priors, based on either geometry or scene knowledge. Before RGB-D images were readily available, it was possible to construct simple 3D models from the image labels [5] or estimate mean depth values from image structure [17] but not predict dense depth maps. Once RGB-D data could be collected from laser or depth cameras on a large scale, it became feasible to apply learning-based approaches for dense depth estimation [6, 11, 13, 14, 21]. Nevertheless, with only hand-crafted features, the inferred depth maps can only approximate the global layout of an image and not depth at finer scales (see example in Fig. 1(b)). In recent years, deep learning methods have demonstrated their strength in a
multitude of vision tasks, including monocular depth estimation. Depth accuracy on a per-pixel basis has improved significantly with elegantly designed convolutional neural networks (CNNs) [2, 8, 10, 18]. Rather than yielding over-smoothed approximations which estimate only the coarse depth of large structures such as walls and ceilings, state-of-the-art multi-scale networks [1] can capture finer-scale items such as furniture and home accessories.

A pinnacle of success for depth estimation is the ability to generate realistic and accurate 3D scene reconstructions from the estimated depths. We observe that despite the impressive evaluation scores of recent works [1, 10], estimated depth maps suffer from artifacts at a local scale and unsatisfactory alignments between surfaces. These distortions become especially prominent when projected into 3D, for example the sloped floor in Fig. 1(c). We hypothesize that comes from a lack of explicit global spatial treatment in standard CNNs. For example, two locally similar image regions may have very different depths depending on their locations in the scene. Conversely, two visually distinct regions may have similar depths, e.g. picture frames hanging on a wall. Furthermore, despite the improvements in recovering local structure, estimated depth maps still lack detail and it is precisely this detail which makes 3D reconstructions realistic and true to life (see Fig. 1(d)).

It is with the goal of 3D reconstruction from estimated depth maps and preserving local structure and detail that we motivate our work. We want to benefit from the increased accuracy of deep learning, but with standard CNN architectures, pooling after convolution repeatedly shrinks image resolution and is contradictory to preserving detail. Other end-to-end estimation tasks such as optical flow [3] and semantic segmentation [12] face similar challenges of maintaining resolution and have introduced up-convolution strategies [12] and concatenated feature maps or predictions [3] from previous parts of the network. Inspired by these works, as well as the success of multi-scale CNNs [1, 4], we introduce a three-stage CNN that performs coarse-to-fine image understanding to predict accurate depths. We closely integrate the layers at each scale with feature map fusion, which offers significant speed benefits for network convergence in comparison to previous state-of-the-art [1]. The network architecture is shown in Fig. 2; five layer-fusions allow for feature passing between scales to accumulate image understanding from global to local.

We also make two additions to our network which make it especially suitable for preserving local structure. First, we take a two dimensional coordinate feature map as an additional network input to provide location cues. This simple tweak, inspired by [20], is trivial to implement but results in more accurate depth estimations, especially in image regions heavy...
in texture and or noise. Secondly, we introduce a relative depth constraint in the CNN loss function to encourage more coherent spatial structures. In comparison to [1] which uses only depth gradient constraints, we show that learning relative depth between neighboring pixels better captures local structures with less artifacts. The main contributions of our work can be summarized as follows:

- We explicitly integrate spatial information into our CNN network by introducing a coordinate feature map and a relative depth constraint. These additions better capture the local structural details and helps prevent artifacts in smooth planar areas.
- We propose a small batch-size training strategy, resulting in a network that is fast to train with low memory requirements.
- Our proposed method yields results that are not only competitive with state-of-the-art but also lead to realistic 3D reconstructions with fine details.

2 Related Work

Depth estimation and 3D reconstruction is a rich field of study; we focus our review only on methods using single images as input. A key strategy in early works for handling depth ambiguity given a single input image was to use strong assumptions and or prior knowledge. For example, Saxena et al. [13, 14] devised a multi-scale Markov Random Field, but assumed that all scenes are horizontally aligned with the ground plane. Hoiem et al. [5], instead of predicting depth explicitly, estimated geometric structures for major image regions and composed simple 3D models to represent the scene.

Once the capture of RGBD images became readily available, data-driven learning-based approaches gained popularity. Karsch et al. [6] proposed a non-parametric method to transfer depth from aligned exemplars and formulated depth estimation as an optimization problem with smoothness constraints. Liu et al. [11] modelled image regions as super-pixels and introduced discrete-continuous optimization for depth estimation; this was further improved by integrating mid-level region features and global scene layout [21]. Others tried to improve depth estimations by exploiting semantic labels [4, 7, 9]. All above mentioned methods use hand-crafted features and results depth estimates are still coarse and lack the finer details necessary for many applications in computer vision and graphics.

The use of deep learning has proven highly effective for depth estimation [1, 2, 8, 10, 18]. Liu et al. [10] combined CNNs and CRFs in a unified framework to jointly learn unary and pairwise potentials with CNNs. Eigen et al. [1, 2] used a multi-scale deep network to predict higher resolution depth map, which better captures structural details. Our system architecture is based on the network in [1]; however, our network has several improvements and offers faster convergence as we introduce additional layer fusions between scales. We discuss the differences in detail in Section 3.

3 Network Architecture

To learn fine-scaled depth maps from single indoor scenes, we propose the multi-scale deep network illustrated in Fig. 2. Our network model has three scales, with layers at each scale closely connected via skip layer fusion. The first scale accumulates and digests image information through a down sampling net and allows for global image understanding by using
Figure 2: Our network architecture and layer details. size is the output map resolution of each layer or layer block. #convs is the number of convolutional layers in each layer block. #chan is the channel size of output feature map. ker.sz is the filter size. l.rate is the learning rate. Specific layers with concatenated feature maps are noticed by their layer index. Reshape layer is marked with blue while prediction layers are marked with pink. The red arrows represent coordinate map concatenation at layers 2.2 and 3.2. Figure best viewed in colour.

two fully connected layers after five convolutional layer blocks. Layers in the first scale are initialized with the VGG model [16], excluding the last two fully-connected layers. Having a global view is important for accurate depth prediction. For example, Li et al. [8] used multi-scale image patch setting and found that patches of the largest scale (407 × 407 pixels) had the most contribution.

Our network is similar in structure to the state-of-the-art three-scaled network of Eigen et al. [1]. The key difference, however, is in the way we propagate information across the different scales. The network in [1] fuses outputs from the last output of previous scale. In comparison, we use five skip layer fusions across the three scales. A 5 × 5 convolution layer is added on pool3 and pool4, which are concatenated with layers in the second scale at layer 2.2 and 2.4 after upsampling. The feature vector of the last fully connected layer is reshaped to match the resolution of the second scale and fused to layer 2.6. These three links provide a hierarchical understanding of an image, which we found leads to better capturing of local structure and faster network convergence.

The second scale predicts depth at a coarse level and outputs a 53 × 70 depth map. The
Figure 3: Example of spatial consistency. The 3D point cloud is generated by re-projection from predicted output of Scale 2. (a) without coordinate map; (b) with coordinated map concatenated in network; (c) with both coordinate map and relative depth constraints.

third scale improves the prediction at a higher resolution and refines local structural details. We add additional inputs of X and Y coordinates, normalized to $[0, 1]$, for each image as a proxy for spatial information and concatenate this with the feature maps from Scale 1 and layer 2.1. Using the coordinate maps gives local filters a better sense of global spatial structure, which in turn leads to more accurate depth predictions. In addition, a $5 \times 5$ convolution layer is added to layer 2.8 and linked to Scale 3. We concatenate the upsampled feature map fused from layer 2.8 and the prediction from layer 2.9 at layer 3.2 in Scale 3.

4 Learning

4.1 Training Loss

We denote the predicted depth map as $D$ and $D^{gt}$ as ground truth depth and use depth in log-scale which we found makes training more stable. We define the loss for training as

$$E = \frac{1}{N} \sum_{i}^{N} d_{i} + \frac{1}{NK} \sum_{i}^{N} \left( \sum_{j}^{K} (d_{i} - d_{j})^{2} \right),$$

(1)

where $d = D - D^{gt}$ is an error map and $N$ is number of pixels with valid depth values. Holes with missing ground truth, typical of RGBD data collected by the Kinect, are not considering in the loss. The first term in the loss is absolute depth error; the second term is our added relative depth constraint, which enforces the difference between pixels in $D$ to be similar with the difference in $D^{gt}$. $K$ is the number of valid pixels in a $3 \times 3$ neighbourhood of pixel $i$. Preliminary experiments showed that larger neighbourhoods did not significantly improve the results, so we keep the neighbourhood fixed at $3 \times 3$ for all experiments.

4.2 Implementation details

The convolutional layers in Scale 1 are initialised with weights from the pretrained VGG network [16]. The fully connected layers in Scale 1 and other layers in Scale 2 and Scale 3 are initialized randomly. The learning rates used are shown in Fig. 2. The dropout rate of layer 1.6 is set to 0.5. All learning rates are stepped down by a factor of 10 after 0.2M gradient steps, and trained for an additional 0.2M steps. We apply two training phases; first is a joint training of Scale 1 and Scale 2 to generate coarse depths. This low-resolution map accurately captures the global structure of the scene, but lacks local detail. We fix the weights of these two scales and train Scale 3 to predict depth at higher resolutions.
4.3 Input Data

We train our network architecture using the NYU Depth v2 dataset [15] according to the standard scene split, with 335 scenes for training. Depth images and RGB images are synchronized according to time stamp; corrupt images are filtered out according to file size, which help avoid vanishing and or exploding gradients. We end up with around 280K RGB-D image pairs for training. After alignment of the depth image with its corresponding RGB image, there remains a rectangular area of 427 × 561 with valid depth values. Input RGB images are first cropped to this valid depth region and then down-sampled by half to 214 × 281. We further augment the training data on-line during training with random transforms including scaling, rotation, translation, horizontal flips, brightness and contrast. Preliminary experiments show that the horizontal flip contributes the most, likely because it generates larger variations in RGB input and depth output.

5 Experimentation

5.1 Evaluation

We evaluate our method on several common measures used in prior works [1, 2, 8, 10, 11] include mean relative error (rel): $\frac{1}{T} \sum_i^T \left| \frac{d_i^{gt} - d_i}{d_i^{gt}} \right|$, mean log10 error (log10): $\frac{1}{T} \sum_i^T |\log_{10} d_i^{gt} - \log_{10} d_i|$, root mean squared error (rms): $\sqrt{\frac{1}{T} \sum_i^T (d_i^{gt} - d_i)^2}$, and accuracy with threshold $thr$: percentage(%) of $d_i$ s.t. $\max \left( \frac{d_i^{gt}}{d_i}, \frac{d_i}{d_i^{gt}} \right) = \delta < thr$. In each of these measures, $d_i^{gt}$ are the ground-truth and $d_i$ are the estimated depths, $T$ is the total number of pixels in all evaluated images. Smaller values on rel, log10 and rms error are better, while higher
values on percentage(\%) $\delta < \text{thr}$ are better. Finally, we show qualitative evaluations by comparing the generated 3D reconstructions by projecting the estimated 2D depth maps to a corresponding 3D point cloud based on the Kinect camera projection matrix.

### 5.2 Depth Estimation Results

Table 1 shows that our prediction accuracy is competitive with state-of-the-art. Our results are marginally weaker than \cite{1} according to the common metrics. We also notice that, although the coordinate map and relative depth constraints help to preserve local details, there is no significant improvement on these metrics. However, qualitatively, one can see that our predicted depth maps are closer to ground truth and have richer details (see Fig. 4). This becomes especially evident when projected to 3D (see Fig. 6).

The work of Liu et al. \cite{10}, which used superpixels for depth estimation also preserves edges very cleanly in 2D; however when projected to 3D, it suffers from distortions and artifacts as shown in Fig. 1 and Fig. 6. The main reason is that each local (superpixel) region has the same or very similar depth even after inpainting, which leads to unsmooth surfaces and discontinuities. Comparing with Liu et al. \cite{10} and Eigen et al. \cite{1}, our 3D reconstruction is more accurate in alignment between surfaces.

Fig. 3 illustrates the benefits in spatial consistency from using coordinate inputs and the relative depth constraints. Fig. 3 (a), (b), (c) are from the predicted depth map of Scale 2. In particular, it makes the network more robust for example to the specularities and reflections which typically lead to surface misalignments. We also find that our method is able to generalize beyond the current NYU Depth v2 dataset; in Fig. 7, we show depth estimates for some image samples from internet, as well as the 3D reconstructions.

| Method            | rel | log10 | rms   | $\delta < 1.25$ | $\delta < 1.25^2$ | $\delta < 1.25^3$ |
|-------------------|-----|-------|-------|-----------------|-------------------|-------------------|
| Karsch et al. \cite{6} | 0.35 | 0.131 | 1.2   | -               | -                 | -                 |
| Liu et al. \cite{11}    | 0.335 | 0.127 | 1.06  | -               | -                 | -                 |
| Ladicky et al. \cite{7} | -   | -     | -     | 0.542           | 0.829             | 0.941             |
| Liu et al. \cite{10}    | 0.230 | 0.095 | 0.824 | 0.614           | 0.883             | 0.971             |
| Li et al. \cite{8}      | 0.232 | 0.094 | 0.821 | 0.621           | 0.886             | 0.968             |
| Eigen et al. \cite{1}   | 0.158 | -     | 0.641 | 0.769           | 0.950             | 0.988             |
| Our method             | 0.164 | 0.069 | 0.653 | 0.761           | 0.946             | 0.986             |

Table 1: Accuracy comparisons on the NYU Depth v2 dataset.
Figure 6: 3D reconstruction comparisons. In the last column, the purple point cloud is our reconstruction while dark green is the ground truth. Figure best viewed in colour.

5.3 Training Batch Size, Convergence and Timing

For common batch training, weight changes are accumulated over some \( b_s > 1 \) of instances, and then averaged for final weight updating; on-line training occurs when \( b_s = 1 \). The previous literature has conflicting claims on the impact of batch size during training. Batch training is implied to be theoretically superior to on-line training as it uses the true gradient, though others claim that on-line training learns quicker [19]. For our experiments, using a small batch size results in significantly faster convergence of the network. We speculate the following causes. First, in our training data, there is redundancy, since training frames are extracted from continuous video. If the individual filter weight changes by \( b_s \) instances in a batch are in the same or close direction, then on-line training will be \( b_s \) times as fast as batch training. Secondly, on-line learning use local gradients, which tend to be noisy and may contradict each other; however, this noise imparts an advantage in being able to easily move out of local minima and eventually converges, as long as it moves on average in the direction of the true gradient.

We use a batch size of 2 to speed up the training. It takes around 22 hours for joint training Scale 1 and Scale 2 and 6 hours for training Scale 3 on a single NVidia Titan X GPU. Each epoch cycles through the 280K training images. We find that convergence is significantly faster than previous works [1, 10] (see Fig. 5) and has lower GPU memory requirements. In particular, comparing with [10], which needed 33 hours to train their network on about 800 image samples, our network achieves the same accuracy and error by 0.5 epochs in only 3 hours. In [1], it took 2.5M gradient steps for convergence at a fine scale; in comparison, our network requires only 0.4M gradient steps with 3 epochs.

5.4 Failed Training Schemes

Training CNNs requires much patience; we report some failed training schemes in hopes that it will guide the community. First, we tried joint training of all three scales together,
but found that the network was much slower to converge and resulting depth maps were overly smooth. We hypothesize that this is due to the fact that our current loss function, despite having a relative depth term, is not aggressive enough in preserving local detail. As such, a decrease in loss does not help in improving local structure; indeed, the loss does not decrease much during the training of Scale 3. We also tried an architecture using only two scales, with Scale 2 set to a higher resolution of $107 \times 140$. This configuration did not improve depth estimates; both accuracy and details were worse and overall training converged at much higher losses. Our intuition is that the network has a hard time filling the large resolution gap between Scale 1 and Scale 2, from coarse to fine directly, instead of gradually with three scales.

6 Conclusion

We have proposed a multi-scale CNN architecture for depth estimation in which we explicitly integrate spatial information via a coordinate feature map and relative depth constraints. With a small batch-size training strategy, training our network is much faster than previous work and significantly saves GPU memory. Experiments on the NYU Depth v2 dataset demonstrate that our proposed method yields results that are not only competitive with state-of-the-art but also lead to realistic 3D reconstructions with fine details. In the current work, we have addressed only the estimation of depth from an RGB source. It is likely that by combining the task with other estimates such as surface normals as well as semantic labels, one can improve the depth estimates.

References

[1] D. Eigen and R. Fergus. Predicting depth, surface normals and semantic labels with a common multi-scale convolutional architecture. In ICCV, 2015.
[2] D. Eigen, C. Puhrsch, and R. Fergus. Depth map prediction from a single image using a multi-scale deep network. In NIPS, 2014.

[3] P. Fischer, A. Dosovitskiy, E. Ilg, P. Hausser, C. Hazirbas amd V. Golkov, P. Smagt, D. Cremers, and T. Brox. Flownet: Learning optical flow with convolutional networks. In ICCV, 2015.

[4] C. Hane, L. Ladicky, and M. Pollefeys. Direction matters: Depth estimation with a surface normal classifier. In CVPR, 2015.

[5] D. Hoiem, A. A. Efros, and M. Hebert. Automatic photo pop-up. In SIGGRAPH, 2005.

[6] K. Karsch, C. Liu, and S.B. Kang. Depth extraction from video using non-parametric sampling. In ECCV, 2012.

[7] L. Ladick, J. Shi, and M. Pollefeys. Pulling things out of perspective. In CVPR, 2014.

[8] B. Li, C. Shen, Y. Dai, A. Hengel, and M. He. Depth and surface normal estimation from monocular images using regression on deep features and hierarchical CRFs. In CVPR, 2015.

[9] B. Liu, S. Gould, and D. Koller. Single image depth estimation from predicted semantic labels. In CVPR, 2010.

[10] F. Liu, C. Shen, and I. D. Reid G. Lin. Learning depth from single monocular images using deep convolutional neural fields. In TPAMI, 2015.

[11] M. Liu, M. Salzmann, and X. He. Discrete-continuous depth estimation from a single image. In CVPR, 2014.

[12] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. In CVPR, 2015.

[13] A. Saxena, S. Chung, and A. Ng. Learning depth from single monocular images. In NIPS, 2005.

[14] A. Saxena, M. Sun, and A. Y. Ng. Make3d: Learning 3-d scene structure from a single still image. TPAMI, 2008.

[15] N. Silberman, P. Kohli, D. Hoiem, and R. Fergus. Indoor segmentation and support inference from rgbd images. In ECCV, 2012.

[16] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. In ICLR, 2015.

[17] A. Torralba and A. Oliva. Depth estimation from image structure. TPAMI, 2002.

[18] P. Wang, X. Shen, Z. Lin, S. Cohen, B. Price, and A. Yuille. Towards unified depth and semantic prediction from a single image. In CVPR, 2015.

[19] D. R. Wilson and T. R. Martinez. The general inefficiency of batch training for gradient descent learning. Neural Networks, 16:142–1451, 2003.
[20] L. Zhang, L. Lin, X. Wu, S. Ding, and L. Zhang. End-to-end photo-sketch generation via fully convolutional representation learning. In *ICMR*, 2015.

[21] W. Zhuo, M. Salzmann, X. He, and M. Liu. Indoor scene structure analysis for single image depth estimation. In *CVPR*, 2015.