Coarse-to-fine Depth Estimation from a Single Image via Coupled Regression and Dictionary Learning

Mohammad Haris Baig  
Dartmouth College  
Hanover, New Hampshire. USA  
haris@cs.dartmouth.edu

Lorenzo Torresani  
Dartmouth College  
Hanover, New Hampshire. USA  
lt@dartmouth.edu

Abstract

In this work we present a novel coarse-to-fine approach to depth prediction from single images. The coarse estimation part of the framework predicts a low-resolution depth map by means of a holistic method that considers the whole image at once in order to exploit global information and context in the scene. This initial estimate is then used as a prior to guide subsequent local refinements at higher resolutions that add missing details and improve the overall accuracy of the system. Both the coarse estimation as well as the successive refinements formulate the task as a regression problem from the image domain to the depth space. However, rather than directly regressing on depth, we propose to learn a compact depth dictionary and a mapping from image features to reconstructive depth weights. The dictionary and the mapping are coupled and jointly optimized by our learning scheme. We demonstrate that this results in a significant improvement in accuracy compared to direct depth regression or approaches using depth dictionaries learned disjointly from the mapping. Experiments on the NYUv2 [17] and KITTI Depth [7] datasets show that our method outperforms the existing state-of-the-art by a large margin.

1. Introduction

Over the last few years depth estimation has been the subject of active research by the machine learning and computer vision community. This can be primarily attributed to the introduction of inexpensive but high-quality depth sensors, such as the Microsoft Kinect which have enabled the acquisition of large collections of RGBD images, i.e., photos encoding the depth of the scene in addition to the traditional color information. Most of these sensors were introduced to assist machine vision algorithms by providing a more robust cue i.e. depth. Algorithms operating on RGBD data have shown to yield great improvements and advancements in challenging computer vision tasks, including semantic segmentation [14], and pose estimation [21] over their RGB counterparts. However most of our current inputs are still RGB images and therefore algorithms operating on these are unable to benefit from the tremendous boosts in performance enjoyed by RGBD algorithms. This has spurred the creation of methods that can leverage these RGBD datasets as training data in order to learn predictive models of depth from RGB channels, which may then be used to improve performance of machine vision algorithms on challenging problems. This paper belongs to this area and focuses on the specific problem of depth estimation from single images. While inferring depth from a single view is ill-posed in general (an infinite number of 3D geometric interpretations can fit perfectly well any given
photo), physical constraints and statistical regularities can be exploited to learn to predict depth from an input photo with good overall accuracy.

We argue that the primary challenge in estimating depth from a single view lies in effectively integrating global and local cues. Global cues extracted by analyzing the image as a whole should inform the method about the spatial context and structure of the scene, such as the presence and location of wall corners and ceiling edges. Conversely, local patches in the image reveal fine depth gradients and 3D geometric details created by the presence of small objects or secondary scene components (e.g., a coffee table or a cabinet) not accounted by the global structure estimated at the coarse level. While a few methods have attempted to combine global and local cues for depth prediction, the goal of producing an integrated robust framework has remained by and large elusive. In Figure 1 we show the coarse and refined outputs produced from our system.

In the pioneering method of Saxena et al. [19, 20], the depth map of a scene was estimated by spatially regularizing the disjoint depth predictions at individual patches of the images with an MRF. However, the MRF is simply used to perform a global smoothing of the noisy and brittle local predictions. More recent approaches [12, 11] have leveraged the massive scale of modern RGBD repositories by turning depth estimation into the task of finding the nearest neighbors of the query image in the labeled database. If the dataset is sufficiently large and/or it exhibits small variations in depths, then it will be possible to find with high probability pictures that match both the global structure as well as the local details of the query and thus to transfer the depth maps of the neighbors to the unlabeled input photo. However, the simplifying assumptions made by such non-parametric models are typically violated in practice. Furthermore, these methods are expensive to run as they require a massive search for neighbors in the large collection.

These earlier approaches advocated the use of a strategy for inferring local depth and following it by an optimization that adds context. In our approach we propose the exact opposite philosophy by first estimating a coarse global estimate and then doing local refinements that are guided by the Global refinement. We propose to naturally combine global and local visual cues by means of a coarse-to-fine approach to depth estimation. Initially, a holistic representation of the image (encoded by convolutional feature maps of a deep network) is used to perform prediction of the depth map at a coarse resolution. Note that this prediction model is trained to estimate depth using the entire spatial extent of the image, i.e., by considering the entire picture at once.

We utilize the convolutional feature maps of the deep neural network trained by [28] (using the Caffe software described in [4]), which has been trained on 2.5 million images from the PLACES dataset [28]. In section 4.3 we show a sensitivity study of design choices which elaborates on the choice of the PLACES features. This initial coarse depth estimate is then used as a prior to perform local refinements at higher resolutions. In the refinement stage we effectively shrink the spatial extent considered by each model (to a part of the image instead of the whole image) but we predict at a higher resolution thus effectively forcing the mapping to fill-in the missing details of the initial coarse estimate.

Both the coarse estimation and the refinement procedure are formulated as regression functions and trained to predict reconstruction weights on a compact dictionary of the depth space. Crucially, we learn the depth dictionary and the regression model jointly by optimizing a single learning objective. Our approach makes parsimonious use of the available training data via a mechanism that shrinks the spatial extent of the region when resolution is increased. Thanks to the fact that 1) the coarse estimator is learned at a very small resolution and 2) the refinement models are trained over smaller spatial extents, our approach can yield state of the art results even with small training sets. For example, on the NYUv2 benchmark, our approach outperforms recent deep network approaches [6], which use a training set that is 150 times bigger than ours.

2. Related Work

Initial approaches to the problem of depth prediction were designed to exploit specific cues like shading [27] and geometry [9] for estimating shape based on our understanding of the image formation process. Their use of specific cues placed restrictions on the applicability of these approaches in real-world scenarios where assumptions made regarding the input may not hold.

Recently the focus has shifted toward employing machine learning methods for estimating absolute depth in addition to shape. The first milestone work in this regard was the work in [19]. This approach exploited texture cues for estimating depth for outdoor scenes and was followed up by [20] which was a refinement of the original work to allow predictions to be made for image regions by assuming each superpixel to be a plane. This reduced the amount of information to be estimated and produced significant accuracy improvements. While this was a very promising direction, these methods were developed in the day and age of limited training data and unable to effectively leverage larger training sets.

Recently, with the advent of much larger RGB datasets like NYUV1 [22], NYUV2 [17], SUN3D [25] there has been an increased interest in non-parametric models that can make effective use of large datasets. Using their own dataset the authors of [11] were able to develop a non-parametric approach to depth estimation that generalizes well to indoor scenes, where methods like [20] did not produce good results since indoor scenes are composed of large
texture-less regions. This was followed by the work of Konrad et al. [12], who also adopted a non-parametric approach that uses kNN to find highly similar images and then applies a bilateral filter to create the final result. Approaches such as these rely on finding near-duplicate matches in the training set and so do not generalize well unless the test set is collected from the same environment as the training set. Another problem with non-parametric approaches is that they require all of the data to be stored on disk at inference time in order to perform the database search, which can be time-consuming.

In [11] the authors use exemplars as basis to come up with a compact representation of depth maps and then use global image features to estimate the weights on their exemplar basis for a query image. Our approach differs from this prior work in three key ways. First, we propose to estimate coarse depth as opposed to full resolution depth maps. This reduces the number of unknowns to estimate. We argue that a full resolution depth map cannot be reliably estimated from a holistic view of the image, which capture the global structure but not fine details. Second, we propose an optimization framework that couples the depth dictionary and the prediction model. Finally, unlike [11] and [6] both of which perform refinement at the same scale as their coarse estimate using superpixels and features respectively, our work performs upsampling and refinement simultaneously, by looking at a complementary source of information (raw image patches) and using reduced spatial extent but a higher resolution to introduce details on top of the coarse estimate.

In [14] perspective effects in the image are exploited as a cue for extracting depth and semantic labels jointly. Another line of recent investigation is represented by [10] which is a bottom up approach estimating continuous valued depth for super-pixels and then discretizing the relations between them to solve for the full image depth.

The approach that is most closely related to ours is the method of Eigen et al. [6]. It shares similarities with our work in the use of deep network features and in the application of a refinement strategy to the initial coarse estimate. However, our approach critically differs from this prior work in three fundamental aspects. First, it regresses on a small set of depth reconstruction weights rather than the full depth maps. Our design choice makes the regression problem easier as the target dimensionality is much lower and the output space is more structured (we predict sparse reconstruction weights). Second, we use the initial coarse depth map as an explicit prior for refinement, while [6] appends the initial estimate to the image fed back to the refinement model, thus putting the crux of discovering the relationship between the coarse and the high resolution map on the shoulders of the training procedure. Third, our refinement models are trained to predict local patches rather than the whole depth map. These three contributions reduce the number of parameters to estimate and enable effective learning even with scarce training data.

Our joint optimization of depth dictionary and regression is inspired by the semi-coupled dictionary learning scheme described in [24], where two pairs of dictionaries and a mapping between them were simultaneously learned. This was applied to the problems of image super-resolution and photo-sketch synthesis. Vondrick et al. [23] used this framework to visualize object detection features. In our work we borrow this optimization scheme to perform joint learning of a depth dictionary and a regressor from the image space to the dictionary weights.

3. Technical Approach

3.1. Overview

In this section we provide an overview of our technical approach. Let \( D = \{ (X_1, D_1), \ldots, (X_N, D_N) \} \) be the training set used to learn our model, where \( X_i \in \mathbb{R}^{R\times C \times 3} \) represents the \( i \)-th image (consisting of \( R \) rows, \( C \) columns and 3 color channels) and \( D_i \in \mathbb{R}^{R\times C} \) is its associated ground-truth depthmap.

We first preprocess the data by subtracting from each depth map the mean depth map computed from the entire training set. This effectively shifts the depth map values at each individual pixel to have zero mean. With a slight abuse of notation from now on we will indicate with \( D_i \) the mean-subtracted version of the depth map. We train our model on this zero-mean depth data. Thus, we optimize the model to predict the deviations from the mean depth map. We store the mean depth map and at test time we simply add it to the prediction of our model in order to generate the final output.

3.2. Coarse Depth Estimation

3.2.1 Learning the Coarse Depth Model

To learn the coarse depth model, we resize the training depth maps to size 32 × 43 for NYUv2 and 32 × 156 for KITTI (in order to maintain aspect ratio of ground truth), via bilinear interpolation. This has the effect of removing fine detail features (object boundaries, fine gradients denoting local shape etc.). We denote with \( d_i \in \mathbb{R}^{LR} \) the vector obtained by vectorizing the resized depth map \( D_i \), where \( LR \) represents the low resolution depth map dimensionality, 1,376 for NYUv2 and 4,992 for KITTI. Similarly, we indicate with \( x_i \in \mathbb{R}^{RC \times 3} \) the vector obtained by stacking the pixel values of the image one on top of the other. Our objective is to train a model that given an input image \( x \) (at full resolution) predicts its coarse depth map \( d \).

The first assumption we make is that the coarse depth map \( d \) can be expressed as a linear combination of basis
vectors from a depth dictionary \( B = [b_1, \ldots, b_m] \):

\[
d = Bw
\]  

(1)

where the \( b_k \in \mathbb{R}^{LR} \) are the dictionary atoms and \( w = [w_1, \ldots, w_m]^T \) is the vector containing the image-specific mixing coefficients (or weights). We propose to learn a linear combination of bases in terms of feature descriptors \( f(x) \), \( f(c) \) extracted from the images \( x, c \). We use as image representation \( f(x) \) the features computed by layer “pool5” of the deep network of the PLACES model [28]. This is the max-pooled output of the networks fifth and final convolutional layer. The feature map has dimensionality \( 6 \times 6 \times 256 = 9216 \). While prior work [8] [26] [10] has shown that the subsequent (fully connected) layers of the Krizhevsky [13] network (same architecture, different dataset) produce higher level representations that yield improved recognition accuracy, pool5 is the most appropriate feature map to use in our setting since it is the last layer preserving explicit location information before the “spatial scrambling” of the fully connection layers (note that a spatially variant representation is crucially necessary to predict the depth at each pixel). We validated experimentally this intuition and observed that using the feature maps from the fully-connected layers produced consistently poorer depth prediction accuracy. Using this representation for feature vector \( f(x) \), we then compute \( \phi_j(x) = \exp(-||f(x) - f(c_j)||^2/2\sigma^2) \).

Given this model, a naive approach to training our depth estimator is to learn disjointly the depth dictionary and the regression mapping. This would involve first learning the dictionary \( B \) and the weights \( w \) of Eq. [1] (e.g., by minimizing the reconstruction error on training depths \( d_1, \ldots, d_N \)) and then regressing on these learned weights to estimate the transformation \( T \) of Eq. [2]. While straightforward, in our experiments we demonstrate that this two-step process yields much inferior results compared to a joint optimization over \( B, w, T \) using a single learning objective that couples all of parameters together. We refer to this learning objective as \( J(B, w, T) \) and define it as follows:

\[
J(B, w, T) = \sum_{i=1}^{N} ||d_i - Bw_i||_2 + \lambda_w \sum_{i=1}^{N} ||w_i||_1 + \lambda_r \sum_{i=1}^{N} ||w_i - T\phi(x_i)||_2 + \lambda_T ||T||_F.
\]

The first two terms of \( J \) encourage reconstruction of the depth maps using sparse weights and are equivalent to those in the traditional sparse coding objective [15]. The third term imposes the requirement that the depth weights be “predictable” from the image input. The final term is a regularizer over the transformation \( T \). Thus, joint optimization of \( J \) over all parameters will yield a depth dictionary \( B \), depth weights \( w \), and transformation \( T \) that simultaneously minimize 1) sparse reconstruction of depths maps and 2) regression error from the image domain to the depth space, subject to appropriate regularizations. In practice we minimize \( J(B, w, T) \) subject to the constraints \( ||b_j||_2 \leq 1 \) for \( j = 1, \ldots, n \) in order to avoid scale degeneracies on \( B \). Furthermore we enforce positivity constraints on the sparse weights \( w_{ij} \) in order to define a purely additive model of depth estimation. We have found experimentally that yields better results than leaving the weights unconstrained. Also, we have tried replacing the L1 sparsity over the weights \( w_i \) with an L2 regularization term but found that this yielded consistently slightly worse results (see Table 5, as also reported in prior articles [18].

While our learning objective is not jointly convex over \( w, B, T \), it is convex for each of these individual parameters when we keep the other two fixed. Based on this, we optimize our learning objective via block-coordinate descent by minimizing in turn with respect to 1) the dictionary, 2) the depth weights and 3) the transformation. These three alternating steps are considered in detail below:

1. **Estimate weights** \( w \) **given parameters** \( B, T \). It is easy to verify that minimizing \( J \) with respect to \( w \) while keeping \( B, T \) fixed (at the current estimate) reduces to a problem of the form:

\[
\text{arg min}_w \sum_{i=1}^{N} || \alpha_i - Cw_i ||_2 + \lambda_w \sum_{i=1}^{N} ||w_i||_1
\]

(4)

where \( \alpha_i, C \) are constants expressed in terms of \( B \) and \( T \). We optimize this problem via least angle regression (LARS) [5].

2. **Learning the depth dictionary** \( B \) **given** \( w, T \). This amounts to a L2-constrained least-squares problem, which we solve using the Lagrange dual formulation, as described in [15].
3. Learning the transformation $T$ given $w$, $B$. This reduced to a L2-regularized least-squares problem, which can be solved in closed-form as shown in [24].

We initialize this optimization by setting $B$ and $w$ to the solution computed via sparse coding [15], thus neglecting the terms in $J$ depending on transformation $T$. We then compute $T$ by solving step 3 above.

3.2.2 Coarse Depth Map Inference

At inference time, given a new input image $x$, we compute its coarse depth map $d$ by finding the sparse depth weights $w$ that best fit the image-based prediction, i.e., by solving the optimization problem

$$\arg\min_w \|w - T\phi(x)\|_2 + \lambda_w\|w\|_1.$$ (5)

The coarse depth map is then generated as $d = BW$. We then add to this estimate the mean depth computed from the training set to produce the final output.

3.3. Depth Refinement

The objective of the depth refinement is to take the low-resolution depth map $d$ produced using our coarse depth model (described in the previous section) and generate a higher resolution depth map $d^\dagger$ containing finer details.

While $d \in \mathbb{R}^{LR}$ is the vectorized version of our coarse depth map at resolution $32 \times 43$ for NYUv2 and $32 \times 156$ for KITTI, the vector $d^\dagger$ encodes the depth map at resolution $128 \times 128$ for NYUv2 and $64 \times 311$ for KITTI, i.e., $d^\dagger \in \mathbb{R}^{HR}$, where $HR$ represents dimensionality of the high resolution vectorized depth map which is 16,384 for NYUv2 and 19,904 for KITTI. We want to stress that $d^\dagger$ is not simply an upsampled version of $d$, but instead it is directly predicted from the image $x$. In the prediction of $d^\dagger$ the coarse estimate $d$ is utilized as a prior to ensure that the higher resolution depth map will not differ too much from the coarse low-resolution estimate and is able to incorporate the depth relations discovered in the coarse estimation step in the refinement step.

We denote with $x^\dagger$ the original image $x$ resized to size $HR$, i.e., the same resolution as the depth map $d^\dagger$ that we want to generate. We then extract from $x$ all patches of size $p_x \times p_y$, where $(p_x, p_y)$ is (16, 16) for NYUv2 and (16, 77) for KITTI. We represent with $PR$ the dimensionality of the vectorized version of the patches which is 256 for NYUv2 and 1,232 for KITTI. We denote with $x^{\dagger(j)}$ the $j$-th patch and with $P$ the total number of patches. For each $x^{\dagger(j)}$, with $j = 1, \ldots, P$, we want to predict its associated depth map patch $d^{\dagger(j)}$. For this purpose we train a patch-based depth predictor $h^\dagger$ such that $h^\dagger(x^{\dagger(j)}) = d^{\dagger(j)}$. The advantage of working with patches (rather than the whole image $x^\dagger$) is that it reduces considerably the number of output variables that our refinement model needs to estimate, only $PR$ instead of $HR$, a much smaller number. Furthermore, it allows us to obtain a much larger training set: hundreds of examples (patches) per training image! The next subsection discusses our algorithm for learning $h^\dagger$.

The final high-resolution depth map $d^\dagger$ is generated by piecing together the individual patch depth predictions $d^{\dagger(j)}$ for $j = 1, \ldots, P$. Specifically for each pixel $p$, we select all patches that include that pixel and average their depth predictions at pixel $p$.

3.3.1 Learning the Depth Refinement Model

The patch-based refinement model $h^\dagger(x^{\dagger(j)})$ is trained by optimizing a learning objective analogous to that described in [3.2.1] for the coarse depth estimation. Specifically, we minimize the following objective function:

$$J^\dagger(B^\dagger, w^\dagger, T^\dagger) = \sum_{i=1}^{N} \sum_{j=1}^{P} \|d^{\dagger(j)}_i - B^\dagger w^{\dagger(j)}_i\|_2$$

$$+ \lambda_w \sum_{i=1}^{N} \sum_{j=1}^{P} \|w^{\dagger(j)}_i\|_1$$

$$+ \lambda_f \sum_{i=1}^{N} \sum_{j=1}^{P} \|w^{\dagger(j)}_i - T^\dagger \phi(x^{\dagger(j)}_i)\|_2$$

$$+ \lambda_T \|T^\dagger\|_F$$

where $d^{\dagger(j)}_i \in \mathbb{R}^{PR}$ is the vectorized ground-truth depth patch $j$ obtained from the training depth map $D_i$. The vector $\phi(x^{\dagger(j)}) = [\phi_1(x^{\dagger(j)}), \ldots, \phi_{n^\dagger}(x^{\dagger(j)})]^T$ contains radial basis functions computed with respect to $n^\dagger$ centers. Note that while the radial basis functions for the coarse estimate were defined as $\phi_k(x) = \exp(-||f(x) - f(x_k)||^2/2\sigma^2)$ in terms of deep features $f(x)$ computed from the whole image, deep global features are clearly not appropriate to represent individual patches. Instead, we obtained good results by representing the patches via sparse coding and using the sparse coefficients in the calculation of the radial basis function, i.e., $\phi_k(x^{\dagger(j)}) = \exp(-||\alpha(x^{\dagger(j)}) - \alpha(c^\dagger_k)||^2/2\nu^2)$ where $\alpha()$ denotes a function that extracts the sparse code coefficients from an input patch and $c^\dagger_k$ is the $k$-th center, itself a patch. In section 3.4, we discuss the details of how we choose the centers $c^\dagger_k$ for $k = 1, \ldots, n^\dagger$.

Joint minimization of $J^\dagger$ with respect to $B^\dagger$, $w^\dagger$, $T^\dagger$ will yield a dictionary optimized for depth reconstruction and such that the reconstruction weights can be reliably estimated from the patch image features $\phi(x^{\dagger(j)})$ via transformation $T^\dagger$. As before, we optimize this objective via block-coordinate descent where at each step we globally optimize $J^\dagger$ with respect to one of the three sets of parameters, while keeping the other two fixed.
3.3.2 Inferring the fine depth map

At test time, given the input image \( x \) and its coarse depth map \( d \) estimated using the method described in Section 3.2.1, we need to generate the fine depth maps \( d^{(j)} \) for all patches \( x^{(j)} \). For ease of explanation, we consider the case of refinement in the NYUv2. In order to obtain the fine depth map we solve for each patch \( j \) the optimization

\[
\arg \min_{\mathbf{w}^{(j)}} \quad ||d^{(j)} - s(B^\top \mathbf{w}^{(j)})||_2^2 \tag{7}
+ \lambda_v||\mathbf{w}^{(j)} - T^\top \phi(x^{(j)})||_2^2
+ \lambda_r||\mathbf{w}^{(j)}||_1
\]

where \( d^{(j)} \) is the vectorized \( 4 \times 4 \) patch \( j \) taken from the coarse estimate \( d \) produced by our model and \( s() \) is a function that down-samples the high-resolution patch \( B^\top \mathbf{w}^{(j)} \) to the coarse resolution. The first term in the minimization above is essentially a prior constraining the predicted high-resolution \( 16 \times 16 \) patch to be similar to \( d^{(j)} \) when subsampled down to resolution \( 4 \times 4 \). This optimization can be solved efficiently via the LARS algorithm [5].

The depth patch is then generated as \( d^{(j)} = B^\top \mathbf{w}^{(j)} \). As already discussed, from all these patch predictions \( d^{(j)} \) we generate the global high-resolution depth map \( d \) by averaging the individual depth patch predictions at each pixel.

3.4. Implementation Details

In this section we provide implementation level details for our algorithm that are common to all datasets, in future sections we talk about specific design choices that were made with regard to each dataset. In [13] the deep network was applied to multiple crops of the image and the predictions on the individual crops were then averaged. Inspired by this approach, we defined five distinct image crops (Center(C), Upper Left(UL), Upper Right(UR), Down Left(DL), Down Right(DR)) of size \( 227 \times 227 \) and learned a distinct coarse model for each of the crops. However, note that all 5 models are trained to predict the complete depth depth map at a coarse resolution (thereby estimating also depth at pixels not in the crop). At inference, we generate the final depth at each pixel as a weighted average of the predictions from the 5 crops.

We use spatially-varying weighting function of the 5 estimates at the coarse size. The weight of crop \( i \) at pixel location \( p \) is computed as

\[
\beta_i(p) = \exp\left(-||p - p_i||/\gamma^2\right)/\sum_{j=1}^5 \exp\left(-||p - p_j||/\gamma^2\right)
\]

where \( p_i \) is the center pixel of crop \( i \). Thus, at each pixel we give more importance to the predictions of crops that are closer to the pixel. For each crop, we form the vector centers \( c_j \) used in the radial basis functions \( \phi_j(x) \) by taking image examples from the two nearest crops.

We use (UL,UR) as centers for C, (C,UR) as centers for UL, (C,UL) as centers for UR, (C,DR) as centers for DL and (C,DL) as centers for DR. We augment the set of centers with the mirrored version of each example. The \( \sigma \) in the kernel is set to be half of the maximum pairwise distance between centers.

For refinement, instead of learning a single shared model for all patches of the image, we trained a separate patch model for each block of rows of the image. In total, we learn a model for each block of 20 rows of the image. This is motivated by the observation that patches within a row (or in neighboring rows) of the image tend to have similar depth statistics but patches coming from distant rows often exhibit large depth variations, as already noted in [19]. This is the mere consequence of ceilings being typically at the top of the image, walls in the middle and floors at the bottom of the picture. For each model we use a depth-patch dictionary \( B^\top \) with 24 atoms for NYUv2 and 48 atoms for KITTI. Figure 3 shows the learned dictionary of depth patches. As features to represent the image patches \( \phi(x^{(j)}) \) we use weights of sparse codes learned for a dictionary of size 384. We then create 2,048 clusters of the image patches using the weights of the image patches on the dictionary. The closest exemplar to the centroid of each cluster is used as the center for creating the radial basis functions. This is done to account for the rich diversity in image patches.

In order to encourage sparsity in weights and at the same time be able to learn how to reconstruct the depth well, we scale the depth maps by dividing by 10 each value in the depth map for NYUv2 and dividing by 50 for each value in KITTI. This allows us to learn better weights. At inference time we estimate the weights for these scaled depth maps and multiply the estimated depth map by the appropriate scaling factor before adding the mean depth to produce the final output.

4. Experiments

In the paper we apply our proposed approach to the NYUv2 [17] and KITTI [7] datasets and show that it produces state of the art results on both. While NYUv2 is a widely used benchmark dataset for depth estimation methods, the KITTI dataset [7] serves as an outdoor scene dataset that illustrates the generalization capability of our approach to different kinds of depths, since KITTI has very different characteristics from the NYUv2.

We begin by presenting in Figure 1 an illustrative example of depth reconstruction for a sample image of this dataset. Notice how the output of refinement is not merely an upsampling of the coarse estimate but it contains added details.

The quality of such depth estimates can be assessed according to multiple criterion. Whereas some metrics inform us of the ability of a method to predict accurate absolute depth, other metrics measure the ability to recover relative scene shape. In this work, we report results on multiple metrics that are widely used: RMSE [6], L1 error [1], Absolute Relative error [6], Scale Invariant error [6], Threshold.
Table 1. Visualization of depth estimates from NYUv2. Details, last column is the pixel-wise difference between the coarse estimate computed by CCDLR and the refinement produced by FCDLR, showing the details introduced by the refinement. It can be noticed that this enables the introduction of additional details.

Table 2. Visualization of depth estimates from KITTI. Details, last column is the pixel-wise difference between the coarse estimate computed by CCDLR and the refinement produced by FCDLR, showing the details introduced by the refinement. It can be noticed that this enables the introduction of additional details.

Table 3. Quantitative Evaluation on KITTI [7] Dataset. Second Row in the table indicates the number of training examples that were used for each method. The 19,852 examples used for training were the ones used by [6].

This allows to compare the coarse and the fine estimates on the same ground.

First, we will describe the settings that were used from the KITTI dataset and then look at the performance of our method and state-of-the-art methods on this dataset. Then we will move on to consider the setup that was used for
evaluation of the NYUv2 dataset, followed by a comparison of the performance of many methods on the NYUv2. Our discussion on experimental setup and the results is based on the design choices that yielded the best results. In section 4.3, we revisit a sensitivity study conducted to analyze these design choices.

4.1. KITTI

The KITTI dataset is an outdoor scene dataset consisting of videos along with depth measurements provided by a LiDaR sensor corresponding to certain frames of the videos. For evaluating the ability of our method to estimate depths on the KITTI dataset we used the train/test split proposed by Eigen et al. [6] consisting of 19852 training examples and 697 test examples. The training and test set comprises of examples from the "city"; "residential" and "road" sequences. Some of the qualitative results from our evaluation can be seen in Table 3.

For this dataset, the input images have size 375 × 1242 and the depths captured by the LiDaR sensor are sparsely sampled in the lower part of the image. We consider the lower crop of the image of size 256 × 1242, which contains all of the sparsely sampled depth measurements and use the colorization method of Levin et.al [2] to construct complete depth maps. Then we resize the depth maps to size 32 × 156 and the input images to size 256 × 256 with bilinear interpolation for learning by our coarse framework as described in section 3.2. For the coarse framework, we use a depth dictionary $B$ consisting of $m = 96$ atoms. Figure 2 shows the learned basis for the center crop. For KITTI we make use of the 654 examples used by [6] to train Make3D as centers from the different crops as described in section 3.4 and use all the other examples in the optimization procedure. The use of few examples as clusters is done to prevent memorization and encourage the optimization framework to produce the improvement desired.

For refinement we resize the original depth maps and the cropped images of size 256 × 1242 to size 64 × 311 for learning. For our final evaluation of both coarse and refinement frameworks we upsample with bilinear interpolation to 256 × 1242 and compare against the depth map we constructed by colorization. Performance for our coarse approach can be seen in Tables 3.

These results highlight that given the same amount of training data, our method significantly outperforms the deep network approach described in [6] as can be seen by our results on the KITTI dataset.

4.2. NYUv2

The NYUv2 dataset [17] consists of examples from 27 different indoor scene categories taken from a total of 464 different scenes. The original images and depth maps are captured at 480 × 640, however after the stereo alignment of the two modalities a portion of the image and depth map contains no data. After removing this portion by using a mask provided by the authors of NYUv2 we get an image and depth map of size 427 × 561. The input $x$ to our coarse estimation is formed by resizing the image to size 256 × 256 using bilinear interpolation. As for the depth map $d$ used by our coarse training procedure, it is obtained by resizing down the depth map with bilinear interpolation to resolution 32 × 43 so that learning can take place with limited data. We evaluated our methods for coarse estimate and refinement of depth maps using the standard train/test split provided by the authors of NYUv2 (795 training examples, 654 test examples) [17]. Some of the qualitative results from our evaluation can be seen in Table 4.

For the coarse framework, we use a depth dictionary $B$ consisting of $m = 48$ atoms for NYUv2. Figure 2 shows the learned basis. For NYUv2 as we have very few examples, 795, to train our framework with we use all of these examples as centers for constructing rbf kernel features. For training the refinement model we use images and depth maps resized to size 128 × 128.

Next we present a comprehensive empirical comparison of our coarse-to-fine approach to the existing state-of-the-art. We have attempted to include in this comparison all prior methods that have been evaluated on the standard training/test splits of the NYUv2 benchmark. The results are summarized in Table 4. We denote our coarse estimation method as CCDLR (Coarse Coupled Dictionary Learning and Regression) and our refinement procedure as FCDLR (Fine Coupled Dictionary Learning and Regression). The last column reports the performance obtained by simply predicting the constant average depth map (computed from the training set) for any input, as this is an interesting base-
We include in the separate Table 5 a side-by-side comparison of our coarse-to-fine methods with the approach of Eigen et al. [6], both evaluated on the test set of NYUv2. Note that we did not include this technique in Table 4 because it does not use the standard training set of the NYUv2 split. Instead, it trains a deep network by making use of 120K frames taken from videos of the NYUv2 dataset. Thus, it uses a training set that is 150 times larger than the one we used in this work (only 795 images). Despite this, it can be seen from Table 5 that our coarse depth predictor produces much better results compared to those obtained with their massively trained deep model.

### 4.3. Revisiting Model Choices

Now, we study the impact of the various design choices made to validate the framework. In Table 6, we conduct a comparative study of different variants of our model in order to validate our design and model choices. For all these choices, the final output is produced via merging the results of depth estimation from multiple crops as described in section 3.4.

We chose the convolutional features from the “pool5” layer produced by the PLACES model to construct our rbf kernel features. In Table 6, we show the impact of using features from the “pool5” layer but trained for different objectives. We compare the model of [13] trained on the imagenet dataset with the objective of object recognition with the model of [28] trained on the PLACES dataset with the objective of scene recognition.

The method CCDLR-L2 (second column) is identical to our model CCDLR, except that we replaced the L1 penalty over weights \( w \) with an L2 regularization. We see that the sparsity enforced by the L1 penalty is advantageous over the L2 penalty. We believe that this happens as the L1 term reduces the number of non-zero weights and thus simplifies the task of predicting the weights from the image space.

In this work we proposed that in order to capture the structure in the output space (spatial smoothness, rejection of unlikely depth maps), one must make use of a depth dictionary that is learned from the data rather than directly regressing on depth directly. The third column of Table 6 (Direct Regr) shows the performance obtained by learning a mapping that uses our image features \( \phi(x) \) to directly regress on the depth \( d \). This leads to a significant drop in accuracy according to all the metrics considered here.

Finally, we suggested that optimizing the dictionary and the transformation jointly should be beneficial over a two-step process. Our hypothesis is confirmed by the results shown in the last column of Table 6 (Disjoint DL&R), which reports the performance obtained by learning a dictionary via sparse coding and then regressing on the weights of the dictionary. Although the degradation in performance with respect to CCDLR is relatively small, we found this difference to be consistently present in all of our experiments, i.e., coupling the dictionary learning and the regression yields always better results.

### 4.4. Analysis of Computational Cost

We also compare the computational cost of training our procedure to the method proposed in [6], which produces the best published results on the NYUv2 dataset by using 136,847 training examples (150 times more examples than our approach) and the best published results on KITTI [7] by using 19,852 examples. The method in [6] requires training a Deep Convolutional Neural Network. Although the time to train this model was not reported in this prior paper, similar architectures with comparable training set sizes have been learned [13] for other tasks and required five to six days even when using GPU accelerations. In contrast, our coarse framework (CCDLR) requires approximately \( \frac{1}{2} \) hour for feature extraction on 1449 images for training and testing of NYUv2 and approximately 5 hours for feature extraction of approximately 20,500 images in the training and testing set of KITTI. Training for NYUv2 which has 795 training examples takes only 5 minutes whereas even in the case of KITTI which has 19,852 training examples, our coarse estimation model is trained in under 20 minutes.
Table 5. Side-by-side comparison between the coarse-to-fine approach of Eigen et al. [6] and our approach on the test set of NYUv2. Note that the model of Eigen et al. was trained using a training set that is 150 times bigger than the one used by our models (CCDLR and FCDLR). Despite this our models achieve nearly comparable results.

This further goes to show how our algorithm scales gracefully to increasing dataset sizes.

For refinement, the method proposed in [6] learns 3 convolutional layers as opposed to an entire network and therefore their training times should be less than that of their coarse framework. Our fine estimate method (FCDLR) learns 24 models corresponding to different groups of rows in the image. Each model is learned from approximately 80,000 patches and requires 2 hours to train. We train these independent models in parallel using a cluster, which makes the total training still 2 hours. Inference using CCDLR takes 1 second per image and FCDLR refinement takes 3 second per image on top of that.

5. Conclusion

We presented a novel approach to depth prediction from single image that naturally integrates global and local information. Global cues in the form of deep convolutional features are used to predict the coarse map. In a subsequent stage the estimated coarse depth map is used as a prior to constrain local refinement at a higher resolution. Both coarse and fine estimation are formulated as joint learning problem over a depth dictionary and a regression mapping from the image space to the dictionary weights. Our approach yields a vast improvement over the state-of-the-art on the standard train/test split of the NYUv2. While our approach learns the depth representation jointly with regression, it currently relies on a fixed set of image features (the deep convolutional features for the coarse estimation and the sparse image codes for patches). In future work we intend to investigate the possibility of using the full framework of semi-coupled dictionary learning to obtain image codes in addition to the depth dictionary and regression within the same training optimization.

6. Acknowledgements

We thank Dr. Loris Bazzani for helpful discussions. This research was funded in part by NSF award CNS-1205521 and a Neukom CompX Faculty Grant. We gratefully acknowledge NVIDIA for the donation of GPUs used for portions of this work.

References

[1] M. H. Baig, V. Jagadeesh, P. Piramuthu, A. Bhardwaj, W. Di, and N. Sundaresan. Im2depth: Scalable exemplar based depth transfer. In IEEE Winter Conference on Applications of Computer Vision, Steamboat Springs, CO, USA, March 24-26, 2014, pages 145–152. IEEE, 2014.
[2] A. L. Dani, D. Lischinski, and Y. Weiss. Colorization using optimization. ACM Transactions on Graphics, 23:689–694, 2004.
[3] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. ImageNet: A Large-Scale Hierarchical Image Database. In CVPR09, 2009.
[4] J. Donahue, Y. Jia, O. Vinyals, J. Hoffman, N. Zhang, E. Tzeng, and T. Darrell. Decaf: A deep convolutional activation feature for generic visual recognition. arXiv preprint arXiv:1310.1531, 2013.
[5] B. Efron, T. Hastie, I. Johnstone, and R. Tibshirani. Least angle regression. Annals of Statistics, 32:407–499, 2004.
[6] D. Eigen, C. Puhrsch, and R. Fergus. Depth map prediction from a single image using a multi-scale deep network. CoRR, abs/1406.2283, 2014.
[7] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun. Vision meets robotics: The kitti dataset. International Journal of Robotics Research (IJRR), 2013.
[8] R. B. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In 2014 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2014, Columbus, OH, USA, June 23-28, 2014, pages 580–587, 2014.
[9] V. Hedau, D. Hoiem, and D. Forsyth. Recovering the spatial layout of cluttered rooms. In Proceedings of the IEEE International Conference on Computer Vision (ICCV ’09), 2009.
[10] S. Karayev, A. Hertzmann, H. Winnemoeller, A. Agarwala, and T. Darrell. Recognizing image style. CoRR, abs/1311.3715, 2013.
[11] K. Karsch, C. Liu, and S. B. Kang. Depth extraction from video using non-parametric sampling. In ECCV, 2012.
Table 6. Testing different variant of our coarse estimation framework on NYUv2 and KITTI. The table shows the variation in RMSE based on different design choices. CCDLR is our framework from Section 3.2. CCDLR-L2 is the same model except that it uses an L2 regularizer (instead of the L1 penalty) on the weights \( w \). “Direct Regr” uses the image features to directly regress on depth (no dictionary learning). “Disjoint DL&R” learns the dictionary via sparse coding and then trains a regression model on the learned weights using our features \( \phi(x) \). CCDLR-Imagenet corresponds to the use of imagenet features obtained from the [13] model.

| Metrics | CCDLR | CCDLR-L2 | Direct Regr | Disjoint DL&R | CCDLR-Imagenet |
|---------|------|---------|-------------|---------------|----------------|
| NYUv2   | 0.8390 | 0.8429  | 0.9746      | 0.8445        | 0.8908         |
| KITTI   | 6.6078 | 6.6495  | 7.5039      | 6.7138        | 6.9923         |

[12] J. Konrad, M. Wang, P. Ishwar, C. Wu, and D. Mukherjee. Learning-based, automatic 2d-to-3d image and video conversion. *Image Processing, IEEE Transactions on*, 22(9):3485–3496, 2013.

[13] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In F. Pereira, C. Burges, L. Bottou, and K. Weinberger, editors, *Advances in Neural Information Processing Systems 25*, pages 1097–1105. Curran Associates, Inc., 2012.

[14] L. Ladicky, J. Shi, and M. Pollefeys. Pulling things out of perspective. June 2014.

[15] H. Lee, A. Battle, R. Raina, and A. Y. Ng. Efficient sparse coding algorithms. In *Advances in Neural Information Processing Systems 19*, pages 801–808. 2007.

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

[17] P. K. Nathan Silberman, Derek Hoiem and R. Fergus. Indoor segmentation and support inference from rgbd images. In *ECCV*, 2012.

[18] A. Ng. Feature selection, l1 vs. l2 regularization, and rotational invariance. In *Proceedings of the twenty-first international conference on Machine learning*, page 78. ACM, 2004.

[19] A. Saxena, S. Chung, and A. Ng. Learning depth from single monocular images. In *Neural Information Processing Systems 18*, 2005.

[20] A. Saxena, M. Sun, and A. Ng. Make3d: Learning 3d scene structure from a single still image. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 31(5):824–840, 2009.

[21] J. Shotton, R. Girshick, A. Fitzgibbon, T. Sharp, M. Cook, M. Finocchio, R. Moore, P. Kohli, A. Criminisi, A. Kipman, and A. Blake. Efficient human pose estimation from single depth images. *Pattern Analysis and Machine Intelligence*, 35(12), 2013.

[22] N. Silberman and R. Fergus. Indoor scene segmentation using a structured light sensor. In *Proceedings of the International Conference on Computer Vision - Workshop on 3D Representation and Recognition*, 2011.

[23] C. Vondrick, A. Khosla, T. Malisiewicz, and A. Torralba. Hoggles: Visualizing object detection features. In *IEEE International Conference on Computer Vision, ICCV 2013, Sydney, Australia, December 1-8, 2013*, pages 1–8, 2013.

[24] S. Wang, L. Zhang, L. Y., and Q. Pan. Semi-coupled dictionary learning with applications in image super-resolution and photo-sketch synthesis. In *International Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2012.

[25] J. Xiao, A. Owens, and A. Torralba. SUN3D: A database of big spaces reconstructed using SIM and object labels. 2013.

[26] N. Zhang, M. Paluri, M. Ranzato, T. Darrell, and L. Bourdev. Panda: Pose aligned networks for deep attribute modeling. *CoRR*, abs/1311.5591, 2013.

[27] R. Zhang, P.-S. Tsai, J. E. Cryer, and M. Shah. Shape from shading: A survey. *IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE*, 21(8):706–717, 1999.

[28] B. Zhou, A. Lapedriza, J. Xiao, A. Torralba, and A. Oliva. Learning deep features for scene recognition using places database. *NIPS*, 2014.