CS274B Project Report

Haitham Khedr
hkhedr@uci.edu

Alex Konrad
akonrad@uci.edu

1 Overview

We implement a monocular depth estimation system using a graphical model. Patch-level texture features are computed to estimate absolute depths at particular image patches, as well as relative depth comparisons. The depths are modeled as a Gaussian Markov Random Field, with depth values considered at three different scales to simulate depth relationships between objects. A maximum likelihood estimate for the parameters is obtained by solving a linear least squares problem. Finally, prediction of depth for a test image is achieved by finding a MAP estimate for the depths in the graphical model. Because the log-likelihood is quadratic in depth values, the MAP estimate can be obtained in closed form.

2 Background

2.1 Related Work

Earlier work on monocular depth estimation from a single image is closely related to shape from shading [3], a technique in computer vision to estimate surface normals from the shading of a sampled image.

One approach to monocular depth estimation combines local (absolute) and global (relative) information from the image to predict depths. Saxena et al. [5] [4] estimate depth values for patches of an image by using absolute depth features—which incorporate local information about the depth of a patch—and relative depth features, which compare aspects of different patches of the same image to infer relative depth of patches. They train a hierarchical graphical model on both local image depth information and contextual global depth information.

Another approach is to use the semantic content of the image as input to the depth estimation. Liu et al. [1] use a two-phase approach, first performing semantic segmentation of the image, labeling each pixel as one of seven classes like `sky` or `water`. These semantic labels are then used as features for a Markov Random Field in the second phase, which incorporates geometry- and depth-related class priors to encode perceptive knowledge about these categories—e.g., that the sky is always the farthest away, or that things stand on the ground. They also condition on these labels to encode whether pixel appearance features are useful in depth estimation, e.g., whether a tree's texture is useful in depth estimation but a building's texture might not be.

As an alternative to adding other sources of information about the image, several nonparametric techniques have been developed to retrieve images with depth maps most similar to the image example for use in estimating depth. Liu et. al [2] construct a graphical model combining continuous variables reflecting image superpixel information with discrete variables containing information from similar images with known depth maps and use particle belief propagation for efficient inference.

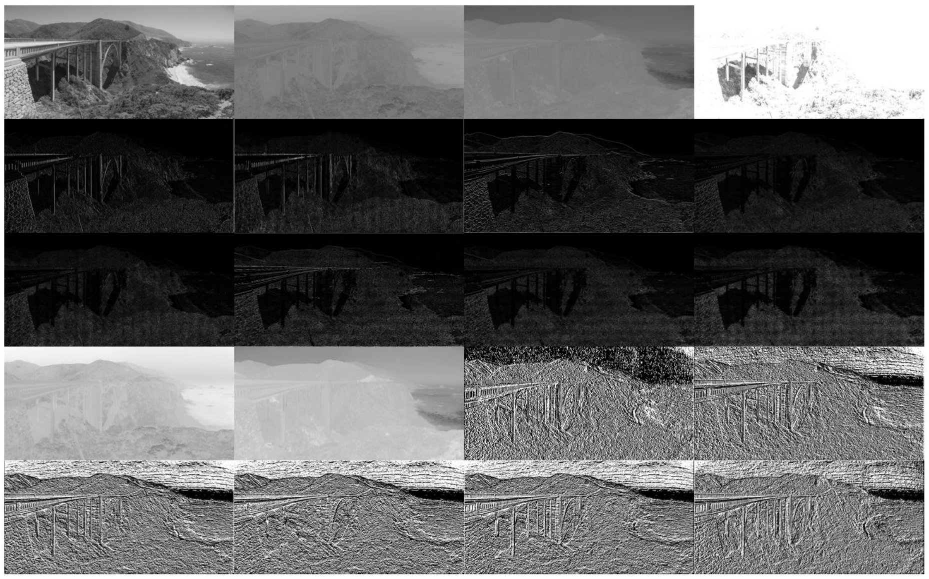

Figure 1: The intensity and two color channels of an image, followed by the 17 pixel-level features. Images three through eleven are Law's filters applied to the intensity channel, twelve and thirteen are the first Law's filter applied to each color channel, and the last are edge detection filters at different orientations.

3 Features for absolute & relative depth

3.1 Absolute Features

We begin by computing pixel-level features to represent texture gradients and color in the image. After converting the image to YCbCR colorspace, we apply 3x3 Law's masks to the intensity channel, then we apply the first Law's mask to both the Cb and Cr color channels to model haze in the image. Then we use six edge detectors rotated at 30 degree angles to model texture gradient. See 1 for an example of these filters.

Next we multiply the 17 pixel-level features element-wise with the intensity channel, taking the absolute value and sum for each of the 17 features to compute the mean absolute energy, then take the absolute value squared and sum to compute the sum squared energy of the patch. This results in 34 patch-level features.

To model contextual relationships, we append the 34 patch-level features of a given patch's 4-neighbors to the patch, creating a 170-dimensional vector. We repeat this process twice more, at scales of 3x and 9x (60x60 and 180x180 patches, respectively). The scaled features are matched back to their corresponding patch and are added to the feature vector for each patch, yielding a 510-dimensional feature vector for each patch.

Finally, the pixel features are again pooled into four columns across the image for each patch row to model the vertical connectivity of things appearing outdoors (for example, the road or a tree). These are added back to each patch feature so that each patch contains information about its vertical column. This adds another 136-dimensional vector, yielding a 646-dimensional vector for each patch.

3.2 Relative Features

For each patch, we generate relative features along with the absolute features. The relative features are computed by taking a 10-bin histogram of each of the 17 features for each scale, resulting in a 510-dimensional sparse indicator vector.

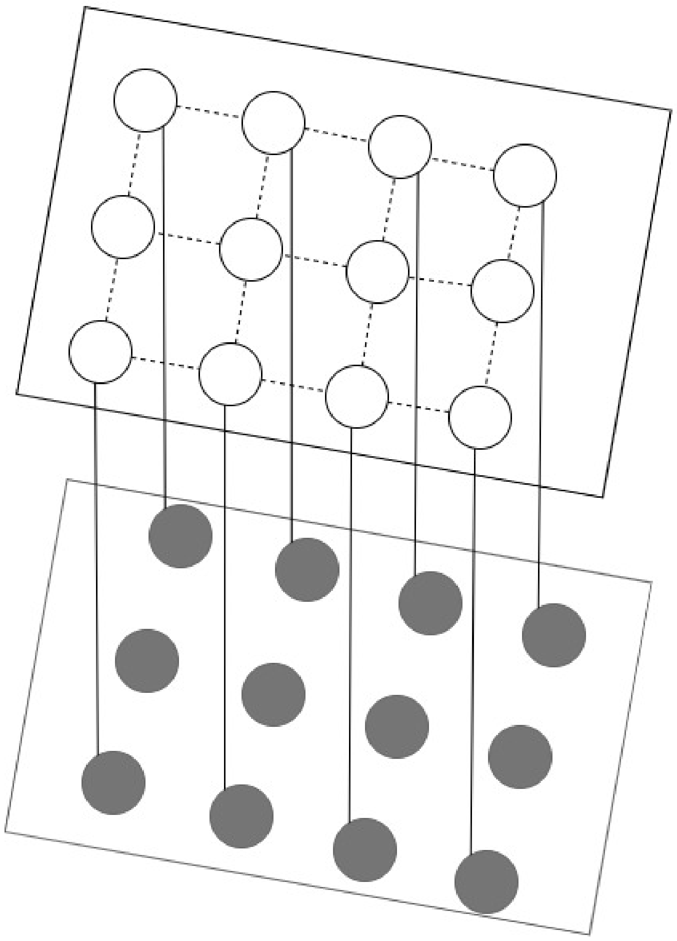

Figure 2: An MRF showing interactions between depth values and patch features as well as interactions between neighbouring patches (some connections were removed to avoid cluttering the figure). The gray nodes represent the observer patch features and the white nodes represent the latent depth values.

4 Probabilistic graphical model

Intuitively, the depth value at a certain patch of the image depends on the spatial features of this patch as well as depth values of the neighbouring patches. For example, the depth values of two patches that belong to the same rigid body should be highly correlated. We'll learn the relation between depth values of the patch and its features in a **discriminative** way. To model the interactions between depth values of neighbouring patches we use a Markov Random Field (MRF). However, one more thing that needs to be taken into account is the interaction of depth values of patches that are not immediate neighbours, that is because sometimes there is a strong interaction between patches that are not immediate neighbours. For example, patches that belong to a large object (sky, building, etc..) will have similar depths even though they might not be immediate neighbours. For this, we model interactions between depths at multiple spatial scales because some adjacent patches when viewed at smaller scales are difficult to recognize as parts of the same object. Figure 2 shows the MRF used for capturing the interactions described. Similarly, there are interactions between depth variables of adjacent patches at larger scale (not shown in figure 2).

Since we are dealing with continuous variables we choose a simple Gaussian MRF with additional hard constraints on larger scales' depth. Specifically, to capture the multi scale depth relations, we define the depth at larger scales as $d_i(s+1) = \frac{1}{5}\sum_{j \in N_s(i) \cup \{i\}} d_j(s),\ s = 1, 2, 3$, where $N_s(i)$ are the 4 neighbors of the i^{th} patch at scale s. Basically this adds a hard constraint that the depth at a larger scale is constrained to be the average of the depths at smaller scales. Instead of modeling the joint distribution of the depths at the smallest scale and the image features, we directly model the posterior distribution as follows:

$$P(d|X;\theta;\sigma) = \frac{1}{Z}\exp\left(-\sum_{i=1}^{M}\frac{(d_i(1) - x_i\theta)^2}{2\sigma_1^2} - \sum_{s=1}^{3}\sum_{i=1}^{M}\sum_{j \in N_s(i)}\frac{(d_i(s) - d_j(s))^2}{2\sigma_2^2}\right), \quad (1)$$

where Z is a the partition function, M is the total number of patches per image at the smallest scale, x_i is the computed feature vector for the i^{th} patch, and θ and σ are the model parameters.

Parameter learning. We learn the model parameters θ by maximizing the likelihood function $P(d|X;\theta;\sigma)$ of the training data. The MLE of parameters θ can be obtained by solving a linear least squares problem (no need for MCMC method). The first term in the exponent fits the dataset(i.e. depth values as a linear function of features) and the second term adds a soft constraint to enforce smoothness of depth values along patches. The terms σ_1 and σ_2 affect the importance of each term. For example, in some cases, depth cannot be reliably estimated from the image features. In this case, inference has to rely more on neighboring patches' depths. In the paper, they learned σ_1 as a linear combination of features, and σ_2 as a linear combination of relative depth features, that is if two patches' features are so distant from each other, this implies that the smoothness constraint should be less relevant. We set $\sigma_1 = 1$ and we set σ_2 to be equal the value of the relative depth feature between each pair of neighbouring patches (this captures the objective stated above).

Inference. We use MAP estimate for inferring the depth values for a new test image by maximizing (1) with respect to d. Since we are using Gaussian posteriors, that MAP estimate can be found in closed form.

5 Experiments

We tested our algorithm on Make3D Test Data set, consisting of 134 images from the Make3D dataset. Similar to the training dataset, all photos are different outdoor settings around a college campus. Figure 3 shows some examples of photos from the test set alongside ground truth depth data and the output of our model.

We observe that the model predicts with good accuracy the general structure of the depth map, but is not able to clearly distinguish objects or represent occlusions. This is likely because we do not train the variance parameters σ_1 and σ_2 for our model and instead use constants. [4] first estimate θ^{ML} holding σ_1, σ_2 fixed, then estimate σ_1, σ_2 holding θ^{ML} fixed, and so on.

The RMS error of our classifier on the test set was .213. This error rate seems accurate compared to [4], who achieve .162 on a similar model with the estimated variance terms. The best classifier they observed used a Laplacian model of the posterior distribution, which achieved .132.

6 Experience Report

6.1 Implementation

We reimplemented the Make3D system proposed by [4] in MATLAB. While the authors made their code available for reference, it was a later and more complicated version of the system described in their initial paper, which worked on superpixels and used several masks for landscape features like the ground and sky. We were hoping to use the code as a baseline to examine other approaches outlined in the previous section, but we were not able to successfully use the code for training a

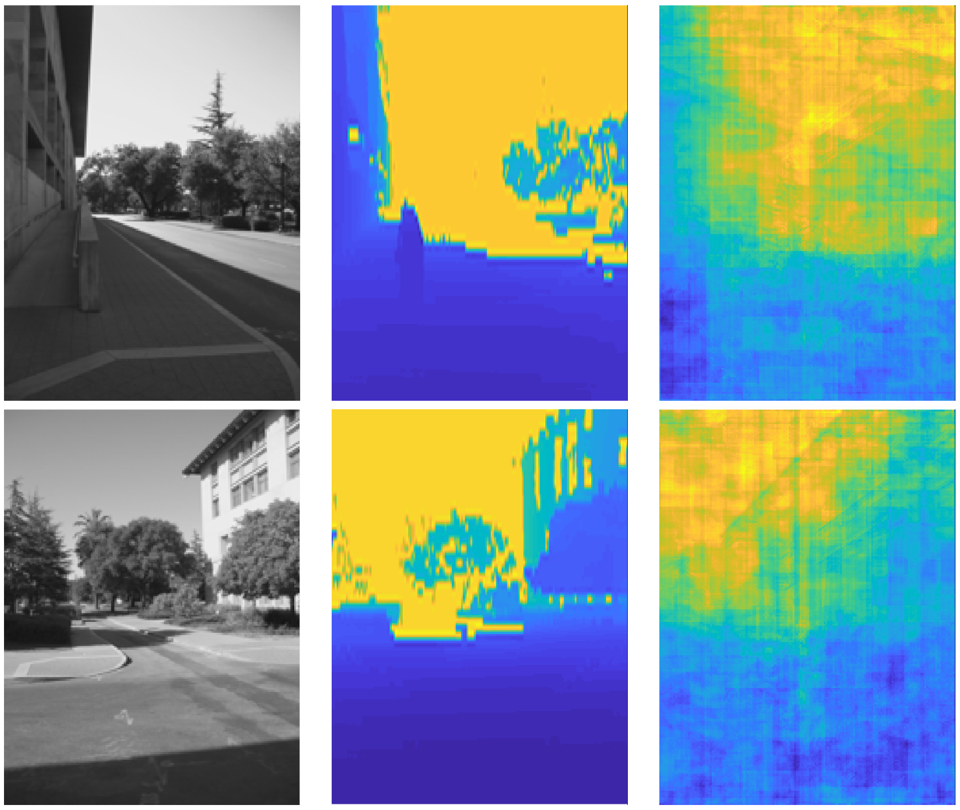

Figure 3: Shows some samples inferred from test data with (left) being the original image, (middle) is the ground truth labels and (right) is the predicted depth values. The model captures the general structure of the scene but it seems that it misses fine details (like far trees).

model because because of segmentation faults caused by MEX files. As a result most of our work involved recreating the model described in the paper in MATLAB.

We were also aiming to use the KITTI dataset to build a model specifically tuned to a road environment. Since we spent most of our time reimplementing the work of [4], we ended up running our tests with the smaller Make3D dataset provided by the authors. The Make3D dataset has 400 training images of size 1704x2202, all of which are outdoor shots of a college campus. We resized these images to 216x162 before training. We resized the corresponding depthmaps to be the same size as the image.

With the generated features, we were able to estimate parameters by simply using the MATLAB \ operator. Unfortunately the training set would be a 400x108x81x646 matrix of doubles and would require about 18GB of RAM. We were able to work around this by noticing that the paper authors estimate separate parameters for every row, so we could slice our features by row and store each row-feature on disk. This helped reduce memory usage and training time.

Our code with our final model parameters is available on GitHub.

6.2 Future Work

Our next task is to model the variance parameters for our model to improve the thresholding between depths. After that, the system could be further improved by implementing the Laplacian likelihood, which perhaps better models depths.

More generally, it would be interesting to compare this work with the other approaches outlined in the Related Work section, for example those involving semantic information or nonparametric depth retrieval. More recent approaches also make use of convolutional neural networks.

6.3 Team Member Contribution

Haitham came up with the idea for our task, and Alex did the literature review and picked the paper to base our work off of. We divided the coding work for the project for Alex to do feature generation and Haitham to do the graphical model portion. Haitham generated the sample images in Figure 3 and Alex measured the error on the test set.

References

[1] Beyang Liu, Stephen Gould, and Daphne Koller. "Single image depth estimation from predicted semantic labels". en. In: *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. San Francisco, CA, USA: IEEE, June 2010, pp. 1253–1260. ISBN: 978-1-4244-6984-0. DOI: 10.1109/CVPR.2010.5539823. URL: http://ieeexplore.ieee.org/document/5539823/ (visited on 05/14/2020).

[2] Miaomiao Liu, Mathieu Salzmann, and Xuming He. "Discrete-Continuous Depth Estimation from a Single Image". en. In: *2014 IEEE Conference on Computer Vision and Pattern Recognition*. Columbus, OH, USA: IEEE, June 2014, pp. 716–723. ISBN: 978-1-4799-5118-5. DOI: 10.1109/CVPR.2014.97. URL: http://ieeexplore.ieee.org/document/6909492/ (visited on 05/14/2020).

[3] Ruo Zhang et al. "Shape-from-shading: a survey". en. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 21.8 (Aug. 1999), pp. 690–706. ISSN: 01628828. DOI: 10.1109/34.784284. URL: http://ieeexplore.ieee.org/document/784284/ (visited on 05/14/2020).

[4] Ashutosh Saxena, Sung H. Chung, and Andrew Y. Ng. "3-D Depth Reconstruction from a Single Still Image". en. In: *International Journal of Computer Vision* 76.1 (Dec. 2007), pp. 53–69. ISSN: 0920-5691, 1573-1405. DOI: 10.1007/s11263-007-0071-y. URL: http://link.springer.com/10.1007/s11263-007-0071-y (visited on 05/14/2020).

[5] Ashutosh Saxena, Sung H Chung, and Andrew Y Ng. "Learning Depth from Single Monocular Images". en. In: (), p. 8.