CS 229 Final Project: Single Image Depth Estimation From Predicted Semantic Labels

Beyang Liu `beyangl@cs.stanford.edu`
Stephen Gould `sgould@stanford.edu`
Prof. Daphne Koller `koller@cs.stanford.edu`

December 10, 2009

1 Introduction

Recovering the 3D structure of a scene from a single image is a fundamental problem in computer vision that has applications in robotics, surveillance, and general scene understanding. However, estimating structure from raw image features is notoriously difficult since local appearance is insufficient to resolve depth ambiguities (e.g., sky and water regions in an image can have similar appearance but dramatically different geometric placement within a scene). Intuitively, semantic understanding of a scene plays an important role in our own perception of scale and 3D structure.

Our goal is to estimate the depth of each pixel in an image. We employ a two phase approach: In the first phase, we use a learned multi-class image labeling MRF to estimate the semantic class for each pixel in the image. We currently label pixels as one of: *sky*, *tree*, *road*, *grass*, *water*, *building*, *mountain*, and *foreground object*.

In the second phase, we use the predicted semantic class labels to inform our depth reconstruction model. Here, we first learn a separate depth estimator for each semantic class. We incorporate these predictions in a Markov random field (MRF) that includes semantic-aware reconstruction priors such as smoothness and orientation. Motivated by the work of Saxena et. al., [6], we explore both pixel-based and superpixel-based variants of our model.

2 Depth Estimation Model

As mentioned above, our algorithm works in two phases. Out of concern for length, we forgo a detailed discussion of the semantic labeling phase here. Briefly, our model can use any multi-class image labeling method that produces pixel-level semantic annotations [7, 3, 2]. The particular algorithm we employ defines an MRF over the semantic class labels (*sky*, *road*, *water*, *grass*, *tree*, *building*, *mountain* and *foreground object*) of each pixel in the image. The MRF includes singleton potential terms, which are simply the confidence of a multi-class boosted decision tree classifier in the predicted semantic label of a pixel. This classifier is trained on a standard set of 17 filter response features [7] computed in a small neighborhood around each pixel. The MRF also includes pairwise potential terms, which define a contrast-dependent smoothing prior between adjacent pixels, encouraging them to take the same label. The weighting between the singleton terms and the pairwise terms is determined via cross-validation on the training set. Given the semantic labeling of the image, we also predict the location of the horizon. We now begin our discussion of the second phase of our algorithm with an overview of the geometry of image formation.

2.1 Image Formation and Scene Geometry

Consider an ideal camera model (i.e., with no lens distortion). Then, a pixel p with coordinates (u_p, v_p) (in the camera plane) is the image of a point in 3D space that lies on the ray extending from the camera origin through (u_p, v_p) in the camera plane. The ray r_p in the world coordinate system is given by

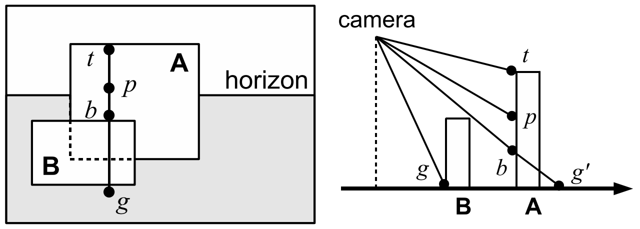
$$r_p \propto \mathbf{R}^{-1}\mathbf{K}^{-1}\begin{bmatrix} u_p \\ v_p \\ 1 \end{bmatrix} \tag{1}$$

$$\mathbf{K} = \begin{bmatrix} f_u & 0 & u_0 \\ 0 & f_v & v_0 \\ 0 & 0 & 1 \end{bmatrix} \tag{2}$$

$$\mathbf{R} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\theta & \sin\theta \\ 0 & -\sin\theta & cos\theta \end{bmatrix} \tag{3}$$

Here, $\mathbf{R}$ defines the rotation matrix from camera coordinate system to world coordinate system and $\mathbf{K}$ is the camera matrix [5]. [1] f_u and f_v are the (u- and v-scaled) focal

[1] In our model, we assume that there is no translation between the world coordinate system and the camera coordinate system, i.e., that the images were taken from approximately the same height above the ground.



(a) Image view (b) Side view

Figure 1: Illustration of semantically derived geometric constraints. See text for details.

lengths of the camera, and the principal point (u_0, v_0) is the center pixel in the image. As in Saxena et. al., [6] we assume a reasonable value for the focal length (in our experiments we set $f_u = f_v = 348$ for a 240×320 image). The form of our rotation matrix assumes the camera's horizontal x axis is parallel to the ground and we estimate the yz-rotation of the camera plane from the predicted location of the horizon (assumed to be at depth ∞): $\theta = \tan^{-1}(\frac{1}{f_v}(v^{\text{hz}} - v_0))$. In the sequel, we will assume that r_p has been normalized (i.e., $\|r_p\|_2 = 1$).

We now describe constraints about the geometry of a scene. Consider, for example, the simple scene in Figure 1, and assume that we would like to estimate the depth of some pixel p on a vertical object **A** attached to the ground. We define three key points that are strong indicators of the depth of p. First, let g be the topmost ground pixel in the same column below p. The depth of g is a lower bound on the depth of p. Second, let b be the bottommost visible pixel b on the object **A**. By extending the camera ray through b to the ground, we can calculate an upper bound on the depth of p. Third, the topmost point t on the object may also be useful since a non-sky pixel high in the image (e.g., an overhanging tree) tends to be close to the camera.

Simple geometric reasoning allows us to encode the first two constraints as

$$d_g \left(\frac{r_g^T e_3}{r_p^T e_3} \right) \leq d_p \leq d_g \left(\frac{r_g^T e_2}{r_b^T e_2} \right) \left(\frac{r_b^T e_3}{r_p^T e_3} \right) \quad (4)$$

where d_p and d_g are the distances to the points p and g, respectively, and e_i is the i-th standard basis vector. The third constraint can similarly be encoded as $d_t r_t^T e_3 \approx d_p r_p^T e_3$. We incorporate these constraints both implicitly as features and explicitly in our pixel MRF model.

2.2 Features and Pointwise Depth Estimation

The basis of our model is linear regression toward the log-depth of each pixel in the image. The output from the linear regression becomes the singleton terms of our MRFs.

Our feature vector includes the same 17 raw pixel filter features from the semantic labeler phase and also the log of these features. These describe the local appearance of the pixel. We also include the (u, v) coordinates of the pixel and an a priori estimated log-depth determined by pixel coordinates (u, v) and semantic label L_p. The prior log-depth is learned, for each semantic class, by averaging the log-depth at a particular (u, v)-pixel location over the set of training images (see Figure 2). Since not all semantic class labels appear in all pixel locations, we smooth the priors with a global log-depth prior (the average of the log-depths over all the classes at the particular location). We encode additional geometric constraints as features by examining the three key pixels discussed in Section 2.1. For each of these pixels (bottommost and topmost pixel with class L_p and topmost ground pixel), we use the pixel's prior log-depth to calculate a depth estimate for p (assuming that most objects are roughly vertical) and include this estimate as a feature. We also include the (horizon-adjusted) vertical coordinate of these pixels as features. Note that our verticality assumption is not a hard constraint, but rather a soft one that can be overridden by the strength of other features. By including these as features, we allow our model to learn the strength of these constraints. Finally, we add the square of each feature, allowing us to learn quadratic depth correlations. We note this set of features is significantly simpler than those used in previous works such as Saxena et. al., [6]. For numerical stability, we also normalize each feature to zero mean and unit variance.

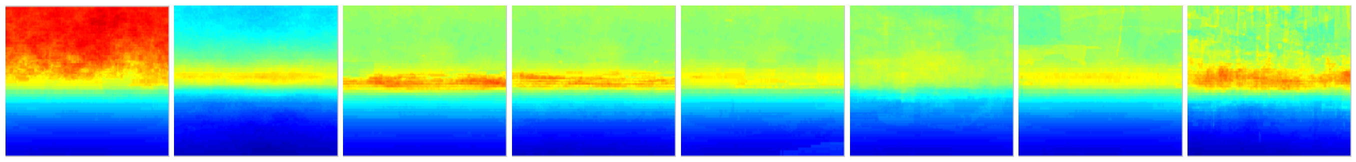
We learn a different local depth predictor (i.e., a different linear regression) for each semantic class. Motivated by the desire to more accurately model the depth of nearby objects and the fact that relative depth is more appropriate for scene understanding, we learn a model to predict log-depth rather than depth itself. We thus estimate pointwise log-depth as a linear function of the pixel features (given the pixel's semantic class),

$$\log \hat{d}_p = \theta_{L_p}^T f_p \quad (5)$$

where $\hat{d}_p$ is the pointwise estimated depth for pixel p, L_p is its predicted semantic class label, $f_p \in \mathbb{R}^n$ is the pixel feature vector, and $\{\theta_l\}_{l \in \mathcal{L}}$ are the learned parameters of the model.

2.3 MRF Models for Depth Reconstruction

The pointwise depth estimation provided by Eq. (5) is somewhat noisy and can be improved by including priors that constrain the structure of the scene. We develop two different MRF models—one pixel-based and one superpixel-based—for the inclusion of such priors.



(a) sky (b) tree (c) road (d) grass (e) water (f) building (g) mountain (h) fg. obj.

Figure 2: Smoothed per-pixel log-depth prior for each semantic class with horizon rotated to center of image. Colors indicate distance (red is further away and blue is closer). The classes "water" and "mountain" had very few samples and so are close to the global log-depth prior (not shown). See text for details.

2.3.1 Pixel-based Markov Random Field

Our pixel-based MRF includes a prior for smoothness. Here, we add a potential over three consecutive pixels (in the same row or column) that prefers co-linearity. We also encode semantically-derived depth constraints which penalize vertical surfaces from deviating from geometrically plausible depths (as described in Section 2.1). Formally, we define the energy function over pixel depths $\mathbf{D}$ as

$$\mathbf{E}(\mathbf{D} \mid \mathcal{I}, \mathbf{L}) = \underbrace{\sum_p \psi_p(d_p)}_{\text{data term}} + \underbrace{\sum_{pqr} \psi_{pqr}(d_p, d_q, d_r)}_{\text{smoothness}} + \underbrace{\sum_p \psi_{pg}(d_p, d_g) + \sum_p \psi_{pb}(d_p, d_b) + \sum_p \psi_{pt}(d_p, d_t)}_{\text{geometry (see §2.1)}} \quad (6)$$

where the data term, ψ_p, attempts to match the depth for each pixel d_p to the pointwise estimate $\hat{d}_p$, and ψ_{pqr} represents the co-linearity prior. The terms ψ_{pg}, ψ_{pb} and ψ_{pt} represent the geometry constraints described in Section 2.1 above. Recall that the pixel indices g, b and t are determined from p and the semantic labels.

The data term in our model is given by

$$\psi_p(d_p) = h(d_p - \hat{d}_p; \beta) \quad (7)$$

where $h(x; \beta)$ is the Huber penalty, which takes the value x^2 for $-\beta \leq x \leq \beta$ and $\beta(2|x| - \beta)$ otherwise. We choose the Huber penalty because it is more robust to outliers than the more commonly used ℓ_2-penalty and, unlike the robust ℓ_1-penalty, is continuously differentiable (which simplifies inference). In our model, we set $\beta = 10^{-3}$.

Our smoothness prior encodes a preference for co-linearity of adjacent pixels within uniform regions. Assuming pixels p, q, and r are three consecutive pixels (in any row or column), we have

$$\psi_{pqr}(d_p, d_q, d_r) = \lambda^{\text{smooth}} \cdot \sqrt{\gamma_{pq}\gamma_{qr}} \cdot h(2d_q - d_p - d_r; \beta) \quad (8)$$

where the smoothness penalty is weighted by a contrast-dependent term and the prior strength λ^{smooth}. Here, $\gamma_{pq} = \exp\left(-c^{-1}\|x_p - x_q\|^2\right)$ measures the contrast between two adjacent pixels, where x_p and x_q are the CIELab color vectors for pixels p and q, respectively, and c is the mean square-difference over all adjacent pixels in the image. (Incidentally, this is the same contrast term used by the MRF in the semantic labeling phase.) We choose the prior strength by cross-validation on a set of training images.

The soft geometry constraints ψ_{pg}, ψ_{pt} and ψ_{pb} model our prior belief that certain semantic classes are vertically oriented (e.g., buildings, trees and foreground objects). Here, we impose the soft constraint that a pixel within such a region should be the same depth as other pixels in the region (i.e., via the constraint on the topmost and bottommost pixels in the region), and be between the nearest and farthest ground plane points g and g' defined in Section 2.1. The constraints are encoded using the Huber penalty, (e.g., $h(d_p - d_g; \beta)$ for the nearest ground pixel constraint). Each term is weighted by a semantic-specific prior strength $\{\lambda_l^{\text{g}}, \lambda_l^{\text{t}}, \lambda_l^{\text{b}}\}_{l \in \mathcal{L}}$.

2.3.2 Superpixel-based Markov Random Field

In the superpixel model, we segment the image into a set of non-overlapping regions (or superpixels) using a bottom-up over-segmentation algorithm. In our experiments we used mean-shift [1], but could equally have used a graph-based approach or normalized cuts. Each superpixel S_i is assumed to be planar, a constraint that we strictly enforce. Instead of defining an MRF over pixel depths, we define an MRF over superpixel plane parameters, $\{\alpha_i\}$, where any point $x \in \mathbb{R}^3$ on the plane satisfies $\alpha_i^T x = 1$. The depth of pixel p corresponds to the intersection of the ray r_p and the plane, and is given by $(\alpha_i^T r_p)^{-1}$.

We define an energy function that includes terms that penalize the distance between the superpixel planes and the pointwise depth estimates $\hat{d}_p$ (Eq. (5)) and terms that enforce soft connectivity, co-planarity, and orientation constraints over the planes. All of these are conditioned on the semantic class of the superpixel (which we define as the majority semantic class over the superpixel's constituent pixels). Formally, we have

$$\mathbf{E}(\boldsymbol{\alpha} \mid \mathcal{I}, \mathbf{L}, \mathbf{S}) = \underbrace{\sum_p \psi_p(\alpha_{i\sim p})}_{\text{data term}} + \underbrace{\sum_i \psi_i(\alpha_i)}_{\text{orientation prior}} + \underbrace{\sum_{ij} \psi_{ij}(\alpha_i, \alpha_j)}_{\substack{\text{connectivity and}\\ \text{co-planarity prior}}} \quad (9)$$

Here $\alpha_{i\sim p}$ indicates the α_i associated with the superpixel containing pixel p, i.e., $\alpha_i : p \in S_i$.

Region Data Term. The data term penalizes the plane parameters from deviating away from the pointwise depth estimates. It takes the form

$$\psi_p(\alpha_i) = \frac{1}{\hat{d}_p} h\left(\hat{d}_p \cdot \alpha_i^T r_p - 1; \beta\right) \quad (10)$$

where $h(x;\beta)$ is the Huber penalty as defined in Section 2.3.1 above. We weight each pixel term by the inverse pointwise depth estimate to give higher preference to nearby regions.

Orientation Prior. The orientation prior enables us to encode a preference for orientation of different semantic surfaces, e.g., ground plane surfaces ("road", "grass", etc.,) should be horizontal while buildings should be vertical. We encode this preference as

$$\psi_i(\alpha_i) = N_i \cdot \lambda_l \cdot \|P_l(\alpha_i - \bar{\alpha}_l)\|^2 \quad (11)$$

where P_l projects onto the planar directions that we would like to constrain and $\bar{\alpha}_l$ is the prior estimate for the orientation of a surface with semantic class label $L_i = l$. We weight this term by the number of pixels (N_i) in the superpixel and a semantic-class-specific prior strength (λ_l). The latter captures our confidence in a semantic class's orientation prior (for example, we are very confident that ground is horizontal, but we are less certain a priori about the orientation of tree regions).

Connectivity and Co-planarity Prior. The connectivity and co-planarity term captures the relationship between two adjacent superpixels. For example, we would not expect adjacent "sky" and "building" superpixels to be connected, whereas we would expect "road" and "building" to be connected. Defining B_{ij} to be the set of pixels along the boundary between superpixels i and j, we have

$$\psi_{ij}(\alpha_i, \alpha_j) = \frac{N_i + N_j}{2|B_{ij}|} \lambda_{lk}^{\text{conn}} \cdot \sum_{p \in B_{ij}} \|\alpha_i^T r_p - \alpha_j^T r_p\|^2 + \frac{N_i + N_j}{2} \lambda_{lk}^{\text{co-plnr}} \cdot \|\alpha_i - \alpha_j\|^2 \quad (12)$$

where we weight each term by the average number of pixels in the associated superpixels and a semantic-class-specific prior strength.

2.4 Inference and Learning

Both of our MRF formulations (Eq. (6) and Eq. (9)) define convex objectives which we solve using the L-BFGS algorithm [4] to obtain a depth prediction for every pixel in the image—for the superpixel-based model we compute pixel depths as $d_p = \frac{1}{\alpha_i^T r_p}$ where α_i are the inferred plane parameters for the superpixel containing pixel p. In our experiments, inference takes about 2 minutes per image for the pixel-based MRF and under 30 seconds for the superpixel-based model (on a 240×320 image).

The various prior strengths (λ^{smooth}, etc.,) are learned by cross-validation on the training data set. To make this process computationally tractable, we add terms in an incremental fashion, freezing each weight before adding the next term. This coordinate-wise optimization seemed to yield good parameters.

3 Experimental Results and Discussion

We run experiments on the publicly available dataset from Saxena et. al., [6]. The dataset consists of 534 images with corresponding depth maps and is divided into 400 training and 134 testing images. We hand-annotated the 400 training images with semantic class labels. The 400 training images were then used for learning the parameters of the semantic and depth models. All images were resized to 240×320 before running our algorithm.[2]

We report results on the 134 test images. Since the maximum range of the sensor used to collect ground truth measurements was 81m, we truncate our predictions to the range $[0, 81]$. Table 3 shows our results compared against previous published results. We compare both the average log-error and average relative error, defined as $|\log_{10} g_p - \log_{10} d_p|$ and $\frac{|g_p - d_p|}{g_p}$, respectively, where g_p is the ground truth depth for pixel p. We also compare our results to our own baseline implementation which does not use any semantic information.

Both our pixel-based and superpixel-based models achieve state-of-the-art performance for the $\log_{10}$ metric and comparable performance to state-of-the-art for the relative error metric. Importantly, they achieve good results on both metrics unlike the previous results which perform well at either one or the other. This can be clearly seen in Figure 4 where we have plotted the performance metrics on the same graph.

Having semantic labels allows us to break down our results by class. Our best performing results are the ground plane classes (especially road), which are easily identified

[2]Note that, although the horizon in the dataset tends to be fixed at the center of the image, we still adjust our camera rays to the predicted horizon location.

METHOD	$\log_{10}$	REL.
SCN †	0.198	0.530
HEH †	0.320	1.423
Pointwise MRF †	0.149	0.458
PP-MRF †	0.187	**0.370**
Pixel MRF Baseline	0.206	0.464
Pixel MRF Model (§2.3.1)	0.149	0.375
Superpixel MRF Baseline	0.209	0.471
Superpixel MRF Model (§2.3.2)	**0.148**	0.379

† Results reported in Saxena et. al., [6].

Figure 3: Quantitative results comparing variants of our "semantic-aware" approach with strong baselines and other state-of-the-art methods. Baseline models do not use semantic class information.

by our semantic model and tightly constrained geometrically. We achieve poor performance on the foreground class which we attribute to the lack of foreground objects in the training set (less than 1% of the pixels).

Unexpectedly, we also perform poorly on sky pixels which are easy to predict and should always be positioned at the maximum depth. This is due, in part, to errors in the groundtruth measurements (caused by sensor misalignment) and the occasional misclassification of the reflective surfaces of buildings as sky by our semantic model. Note that the nature of the relative error metric is to magnify these mistakes since the ground truth measurement in these cases is always closer than the maximum depth.

Finally, we show some qualitative results in Figure 5. The results show that we correctly model co-planarity of the ground plane and building surfaces. Notice our accurate prediction of the sky (which is sometimes penalized by misalignment in the groundtruth, e.g., third example). Our algorithm also makes mistakes, such as positioning the building too close in the second example and missing the ledge in the foreground (a mistake that many humans would also make).

Overall, our model attains state-of-the-art results, though it utilizes relatively simple image features, because it incorporates semantic reasoning about the scene.

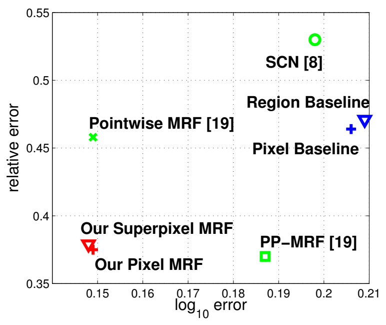


Figure 4: Plot of log_{10} error metric versus relative error metric comparing algorithms from Table 3 Bottom-left indicates better performance.

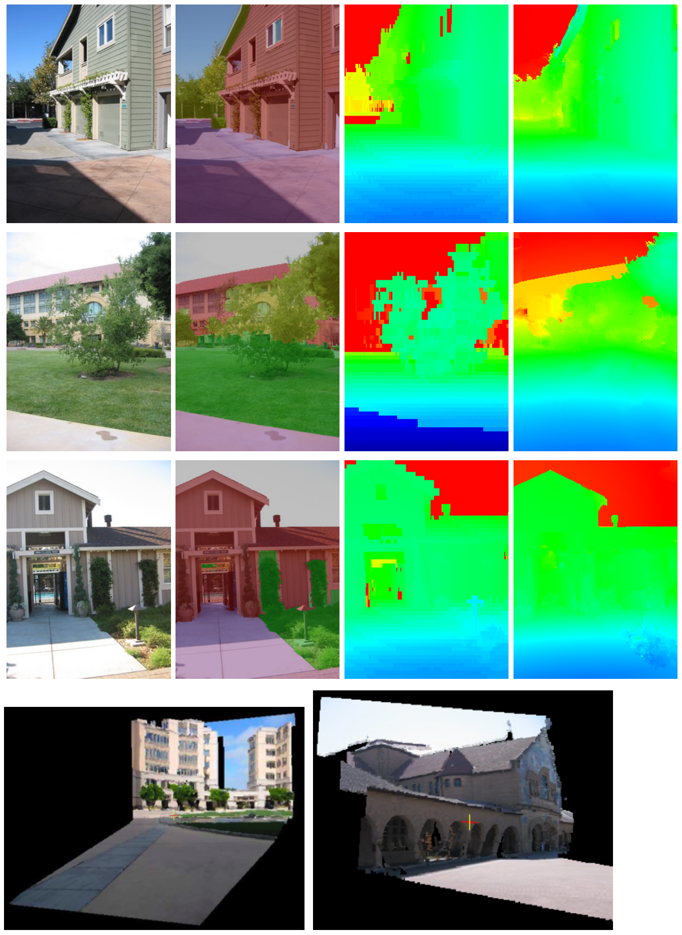


Figure 5: (Above) Some qualitative depth reconstructions from our model showing (from left to right) the image, semantic overlay, ground truth depth measurements, and our predicted depths. Red signifies a distance of 80m, black 0m. (Below) Example 3D reconstructions.

References

[1] D. Comaniciu and P. Meer. Mean shift: A robust approach toward feature space analysis. *PAMI*, 2002.

[2] S. Gould, R. Fulton, and D. Koller. Decompsing a scene into geometric and semantically consistent regions. In *ICCV*, 2009.

[3] X. He, R. Zemel, and M. Carreira-Perpinan. Multiscale CRFs for image labeling. In *CVPR*, 2004.

[4] D. Liu and J. Nocedal. On the limited memory method for large scale optimization. In *Mathematical Programming B*, 1989.

[5] Y. Ma, S. Soatto, J. Kosecka, and S. S. Sastry. *An Invitation to 3-D Vision*. Springer, 2005.

[6] A. Saxena, M. Sun, and A. Y. Ng. Make3D: Learning 3-D scene structure from a single still image. In *PAMI*, 2008.

[7] J. Shotton, J. Winn, C. Rother, and A. Criminisi. TextonBoost: Joint appearance, shape and context modeling for multi-class object recognition and segmentation. In *ECCV*, 2006.