Unsupervised Learning of Stereo Vision with Monocular Cues

Hoang Trinh
http://ttic.uchcago.edu/~trinh
David McAllester
http://ttic.uchicago.edu/~dmcallester

The Toyota Technological Institute at Chicago
6045 Kenwood Ave
Chicago IL 60637

Abstract

We demonstrate unsupervised learning of a 62 parameter slanted plane stereo vision model involving shape from texture cues. Our approach to unsupervised learning is based on maximizing conditional likelihood. The shift from joint likelihood to conditional likelihood in unsupervised learning is analogous to the shift from Markov random fields (MRFs) to conditional random fields (CRFs). The performance achieved with unsupervised learning is close to that achieved with supervised learning for this model.

1 Introduction

We demonstrate unsupervised learning of a 62 parameter stereo vision model involving shape from texture cues. Texture is one example of a monocular depth cue — evidence for depth based on a single image. Our work can therefore be interpreted as training monocular depth cues from unlabeled stereo pair training data. However, the stereo pair data is not viewed as a simple surrogate for depth information — the stereo algorithm itself is viewed as the object being trained and the training of monocular depth cues happens as a byproduct of stereo training.

Our training method is a form of unsupervised learning. In unsupervised learning one usually formulates a parameterized probability model and seeks parameter values maximizing the likelihood of the unlabeled training data. For stereo vision it seems appropriate to formulate a conditional probability model rather than a joint model. In particular, the model should define, say, the probability of the right image given the left. This conditional model need not model any probability distribution over images — it only models the conditional distribution of the right image given the left.

The move from maximizing the (joint) likelihood of the given data to maximizing a conditional likelihood is related to the move from Markov random fields (MRFs) to conditional random fields (CRFs). MRFs model joint probabilities while CRFs model conditional probabilities. MRFs have been widely used for decades as statistical models in a variety of application areas [11, 15, 19]. An MRF defines a probability distribution on the joint assignments to (configurations of) a set of random variables. Conditional random fields (CRFs) [18] are similar to MRFs except that in a CRF the variables are divided into two groups — exogenous and dependent. A CRF defines a conditional probability of the dependent variables given the exogenous variables and (importantly) does not model the distribution of the

exogenous variables. Although the difference between an MRF and a CRF is mathematically simple, the shift from joint modeling to conditional modeling has significant consequences which has lead to a rapid replacement of MRFs by CRFs in practice. Perhaps most significantly, since a conditional model does not attempt to model the distribution of the exogenous variables, there is no danger of corrupting the model by modeling the exogenous variables poorly. In the case of stereo vision one might expect that it is easier to model the probability distribution of the right image given the left image than to model a probability distribution over images.

The most closely related earlier work seems to be that of Zhang and Seitz [24]. They give a method for adapting five parameters of a stereo vision model including the weights for the match and smoothness energies as well as robustness parameters. The five parameters are tuned to each individual input stereo pair, although the method could be used to tune a single parameter setting over a corpus of stereo pairs. The main difference between their work and ours is that we train highly parameterized monocular depth cues. Another difference is that we formulate a general CRF-like model for unsupervised learning based on maximizing conditional likelihood and avoid the need for the independence assumptions used by Zhang and Seitz by using contrastive divergence — a general method for optimizing loopy CRFs [7, 12].

There is also related work by Saxena et al. on learning highly parameterized monocular depth cues [3, 4]. The main difference between this work and ours is that we use unsupervised learning while they use laser range finder data to train their system. One might argue that stereo pairs constitute supervised training of monocular depth cues. A standard stereo depth algorithm could be used to infer a depth map for each pair which could then be used in a supervised learning mode to train monocular depth cues. However, we demonstrate that training monocular depth cues from stereo pair data improves *stereo* depth estimation. Hence the method can be legitimately viewed as unsupervised learning of a stereo depth. Also the general formulation of unsupervised learning by maximizing conditional likelihood, like the shift from MRFs to CRFs, may have significance beyond computer vision.

Other related work includes that of Scharstein and Pal [22] and Kong and Tao [17]. In these cases somewhat more highly parameterized stereo models are trained using methods developed for general CRFs. However, the training uses ground truth depth data rather than unlabeled stereo pairs.

Our stereo vision model is a slanted plane model involving shape from texture cues. The slanted plane model is similar to that described in [5] but where we use a fixed oversegmentation for the left image as in [16]. The stereo algorithm infers a slanted plane for each segment. This is done by minimizing an energy functional with 62 parameters - 10 correspondence parameters, 2 smoothness parameters, and 50 texture parameters. We learn MRF parameters using contrastive divergence [7, 12], a general MRF learning algorithm capable of training large models. Our stereo model involves three terms — a correspondence energy measuring the degree to which the left and right images agree under the induced disparity map, a smoothness energy measuring the smoothness of the induced depth map, and an texture energy measuring the degree to which the surface orientation at each point agrees with a certain (monocular) texture based surface orientation cue. For surface orientation cue we use histograms of oriented gradients (HOG) [8]. We derive a formal relationship between a variant of HOG features and surface orientation. Although our observation that there should be a statistical relationship between HOG features and surface orientation is a simple result in the area of shape from texture [1, 6, 20, 21, 23], HOG features have only recently gained popularity and to our knowledge the possibility of using HOG as a surface orientation cue

has not been previously noted.

2 The Slanted Plane Stereo Model

We now take x to be a segmented left image, y to be a right image, and take z to be an assignment of a disparity plane to each segment of x. More specifically, for each segment i of x we have that z specifies three plane parameters A_i B_i, and C_i. Given an assignment z of plane parameters to segments we define the disparity $d(p)$ for any pixel p as follows where $i(p)$ is the segment containing p and x_p and y_p are the image coordinates of p.

$$d(p) = A_{i(p)}x_p + B_{i(p)}y_p + C_{i(p)} \tag{1}$$

So by equation (1) we have that z assigns a disparity to each pixel.

The model is defined by three energies: a smoothness energy, a match energy, and a texture energy. The energy $Z_z(x,z,\beta_z)$, which determines $P(z|x,\beta_z)$, consists of the smoothness energy and the texture energy. The energy $E_y(x,y,z,\beta_y)$, which determines $P(y|x,z,\beta_y)$, consists solely of the match energy. To define the smoothness energy we write $(p,q) \in B_{i,j}$ if p is a pixel in segment i, q is a pixel in segment j, and p and q are adjacent pixels (p is directly above, below, left or right of q). The smoothness energy is defined as follows where τ_S and λ_S are parameters of the energy.

$$E_S = \sum_{i,j} \min\left(\tau_S, \sum_{(p,q)\in B_{i,j}} \lambda_S |d(p) - d(q)|\right) \tag{2}$$

Intuitively the minimization with τ_S corresponds to interpreting the entire boundary between i and j as either an occlusion boundary or as a joining of two planes on the same object.

Next we consider the match energy. We write $p + d(p)$ for the pixel in y that corresponds to the pixel p in image x under the disparity $d(p)$. For color images we construct a nine dimensional feature vector $\Phi^x(p)$ and $\Phi^y(p)$ for the pixel p in the images x and y respectively. The vector $\Phi^x(p)$ consists of three (bias gain corrected) color values plus a six dimensional color gradient vector and similarly for Φ^y_p. We write $\Phi^x_k(p)$ for the kth component of the vector $\Phi^x(p)$. The match energy is defined as follows where λ_k are nine scalar parameters of the match energy.

$$E_M = \sum_p \sum_k \lambda_k (\Phi^x_k(p) - \Phi^y_k(p + d(p)))^2 \tag{3}$$

Finally we consider the texture energy. At each pixel p we also compute a HOG vector $H(p)$ which is a 24 dimensional feature vector consisting of three 8 dimensional normalized edge orientation histograms — an 8 dimensional orientation histogram is computed at three different scales. The texture energy is defined as follows where $i(p)$ is the segment containing pixel p and where the scalars τ_T, λ_A, λ_B, and the vectors β_A and β_B are parameters of the energy. The form of this energy is justified in section 2.1.

$$E_T = \sum_p \min\left(\tau_T, \begin{array}{rl} & \lambda_A \left(d(p)(\beta_A \cdot H(p)) - A_{i(p)}\right)^2 \\ + & \lambda_B \left(d(p)(\beta_B \cdot H(p)) - B_{i(p)}\right)^2 \end{array}\right) \tag{4}$$

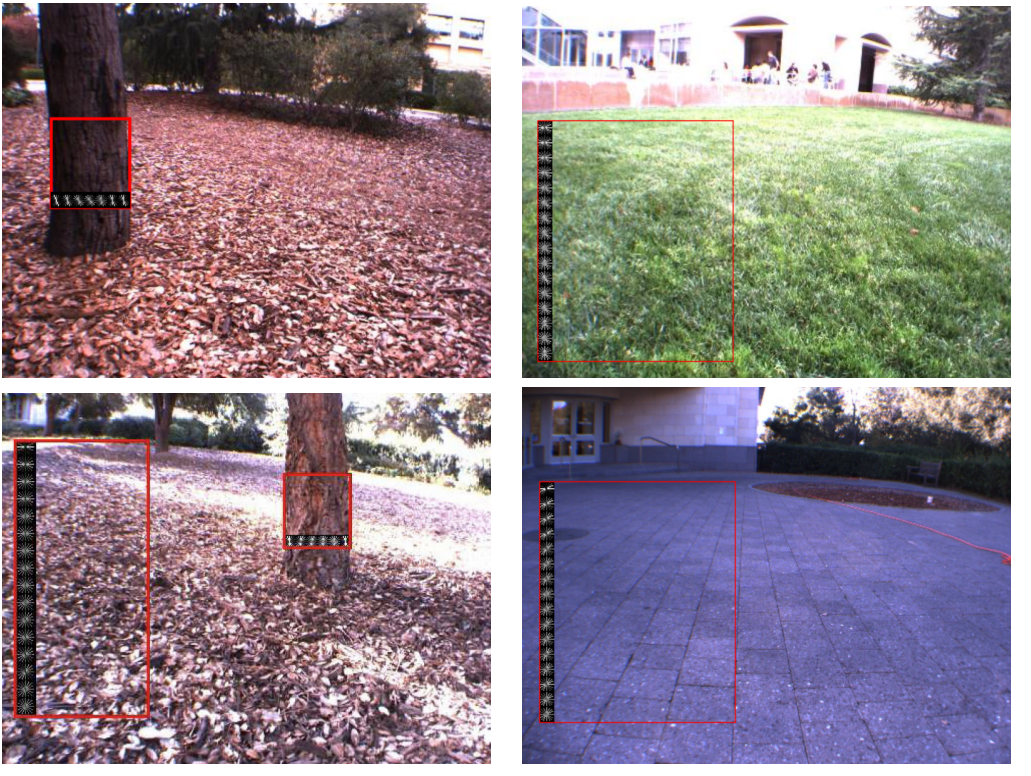


Figure 1: HOG features for image regions. The amount of edge as a function of angle, a HOG feature, is averaged over different vertical and horizontal regions on various images. The different surface orientations of these regions affect the HOG features. We can see the cylindrical structure of tree trunks and the fact that the ground plane becomes more tilted as the distance increases.

2.1 HOG as Surface Orientation Cues

The basic intuition behind HOG as an orientation cue is that as a surface is tilted away from the camera the edges in the direction of the tilt become foreshortened while the edges orthogonal to the tilt are not. This changes the edge orientation distribution and therefore the edge orientation distribution can be used as a cue for surface orientation. This effect is shown in figure 1 where the average HOG feature is shown for various regions of tree trunk, forest floor, grass lawn, and patio tile. The cylindrical shape of the tree trunk is clearly indicated by the warping of the HOG feature as a function of position on the trunk.

We consider a surface patch imaged by a perspective camera. A perspective camera induces the following map from three dimensional coordinates to image plane coordinates.

$$x' = (fx)/z \quad y' = (fy)/z \tag{5}$$

We assume a coordinate system on the surface patch such that each point on the surface patch has coordinates x_s, y_s. The image plane and surface coordinates can be selected so that we have the following map from surface coordinates to three dimensional coordinates where Ψ

is the angle between surface normal and the image plane normal.

$$x = x_s \quad y = y_s \cos\Psi \quad z = z_0 + y_s \sin\Psi \tag{6}$$

To justify the form of the orientation energy (4) we first note that in the same coordinate system as (6) the equation for the surface plane can be written as follows.

$$z = z_0 + y \tan\Psi \tag{7}$$

If we let b be the distance between the foci of the two cameras we have that the disparity d equals bf/z. Multiplying (7) by $bf/(zz_0)$ gives the following where B is the y coefficient in the disparity plane (as in (1).

$$d = d_0 + By' \tag{8}$$

$$B = -\frac{d_0 \tan\Psi}{f} \tag{9}$$

For the pixel p at the center of the image we have $d(p) = d_0$. We handle pixels outside of the center of the image by considering panning the camera to bring the desired point to the center and approximating panning the camera by translating the image. This gives the following general relation between the disparity plane parameter and the angle Ψ between the ray form the camera and the surface normal.

$$B = -\frac{d(p) \tan\Psi}{f} \tag{10}$$

In the orientation energy (4) we interpret $\beta_B \cdot H(p)$ as a predictor of $-(\tan\Psi)/f$ and we multiply by $d(p)$ to get a predictor of B.

3 Hard Conditional EM

We consider a general conditional probability model $P_\beta(y|x)$ over arbitrary variables x and y and defined in terms of a parameter vector β and an arbitrary latent variable z.

$$P(y|x,\beta) = \sum_z P(y,z|x,\beta) \tag{11}$$

In our slanted plane model x is a segmented left image, y is a right image, and z as an assignment of a plane to each segment of x. But in this section we consider the general case defined by (11). Given training data $(x_1,y_1),\ldots(x_N,y_N)$ conditional EM is an algorithm for locally optimizing the parameter vector β so as to maximize the probability of the y values given the x values in the training data.

$$\beta^* = \operatorname*{argmax}_\beta \sum_{i=1}^{N} \ln P(y_i|x_i,\beta) \tag{12}$$

Conditional EM is a straightforward modification of EM and is defined by the following two updates where β is initialized with domain specific heuristics.

$$P_i(z) := P(z|x_i,y_i,\beta) \tag{13}$$

$$\beta := \operatorname*{argmax}_\beta \sum_{i=1}^{N} \mathrm{E}_{z\sim P_i}\left[\ln P(y_i,z_i,|x_i,\beta)\right] \tag{14}$$

Update (13) is called the E step and update (14) is called the M step. Hard EM, also known as Viterbi training, works with the single most likely (hard) value of z rather than the (soft) distribution P_i defined by (13). Hard conditional EM locally optimizes the following version of (12).

$$\beta^* = \operatorname*{argmax}_{\beta} \sum_{i=1}^{N} \max_{z} \ln P(y_i, z | x_i, \beta) \tag{15}$$

Hard conditional EM is defined to be the process of iterating the updates (16) and (17) below which can be interpreted as hard versions of (13) and (14).

$$z_i := \operatorname*{argmax}_{z} P(y_i, z | x_i, \beta) \tag{16}$$

$$\beta := \operatorname*{argmax}_{\beta} \sum_{i=1}^{N} \ln P(y_i, z_i, | x_i, \beta) \tag{17}$$

We will call (16) the hard E step and (17) the hard M step. Updates (16) and (17) are both coordinate ascent steps for the objective defined by (15). However, we refer to (16) and (17) as hard conditional EM rather than simply "coordinate ascent" because of the clear analogy between (15), (16), (17) and (12), (13), (14).

In the case of the slanted plane stereo model, the hard E step (16) is implemented using a stereo inference algorithm which computes z_i by minimizing an energy functional. In this case the parameter vector β is a pair $\beta = (\beta_z, \beta_y)$ where β_z parameterizes $P(z|x)$ and β_y parameterizes $P(y|x,z)$. The inference algorithm is described in section 4. Our implementation of the hard M step relies on a factorization of the probability model into two conditional probability models each of which is defined by an energy functional. Unlike CRFs, we do not require the energy functional to be linear in the model parameters.

$$P(y, z | x, \beta_y, \beta_z) = P(z | x, \beta_z) P(y | x, z, \beta_y) \tag{18}$$

$$P(z | x, \beta_z) = \frac{\exp(-E_z(x, z, \beta_z))}{Z_z(x, \beta_z)} \tag{19}$$

$$Z_z(x, \beta_z) = \sum_{z} \exp(-E_z(x, z, \beta_z))$$

$$P(y | x, z, \beta_y) = \frac{\exp(-E_y(x, y, z, \beta_z))}{Z_y(x, z, \beta_y)} \tag{20}$$

$$Z_y(x, z, \beta_y) = \sum_{y} \exp(-E_y(x, y, z, \beta_y))$$

Given this factorization of the model, the hard M step (17) can be written as the following pair of updates.

$$\beta_z := \operatorname*{argmax}_{\beta_z} \sum_{i} \ln P(z_i | x_i, \beta_z) \tag{21}$$

$$\beta_y := \operatorname*{argmax}_{\beta_y} \sum_{i} \ln P(y_i | x_i, z_i, \beta_y) \tag{22}$$

Let L abbreviate the quantity being maximized in the right hand side of (21) and let $E_i(z)$ abbreviate $E_z(x_i, z_i, \beta)$. We can express the gradient of L as follows.

$$\nabla_{\beta_z} L = \sum_{i=1}^{N} \left(\mathrm{E}_{z \sim P_z(z|x_i,\beta)} \left[\nabla_{\beta_z} E_i(z) \right] - \nabla_{\beta_z} E_i(z_i) \right) \tag{23}$$

A similar equation holds for (22). We can estimate $\nabla_{\beta_z} L$ by sampling z from $P(z|x_i, \beta_z)$ using an MCMC sampling process. We can then optimize (21), and similarly (22), by gradient descent.

For the experiments reported here we use contrastive divergence [7, 12] to sample z rather than a long running MCMC process. In contrastive divergence we initialize z to be z_i and then perform only a few MCMC updates to get a sample of z. Contrastive divergence can be motivated by the observation that if z_i is assumed to be drawn at random form $P(z|x_i, \beta)$ then the expected contrastive divergence update is zero. So as β better fits the pairs (x_i, z_i) one expects the contrastive divergence gradient estimate to tend to zero. Furthermore, because only a few updates are used in the MCMC process, contrastive divergence runs faster and with lower variance than a longer running MCMC process.

4 Inference

Given a pair of images (x, y), and a given segmentation of x, and a given setting of the model parameters β_z and β_y, the inference problem is to find an assignment z of plane parameters to segments so as to minimize the total energy $E(x, z, \beta_z) + E(x, y, z, \beta_y)$. In our experiments we compute a segmentation using the Felzenszwalb-Huttenlocher segmentation algorithm [10]. The energy defines a Markov random field. More specifically, the texture energy and the match energy defines a potential on each segment independently and the smoothness energy defines a potential on pairs of adjacent segments. The energy can be written as follows where i and j range over segments, $N(i)$ is the set of segments bordering i, and z_i is the three dimensional vector of plane parameters (A_i, B_i, C_i) for segment i.

$$E(z) = \sum_i (E_T(z_i) + E_M(z_i)) + \sum_{i, j \in N(i)} E_S(z_i, z_j) \tag{24}$$

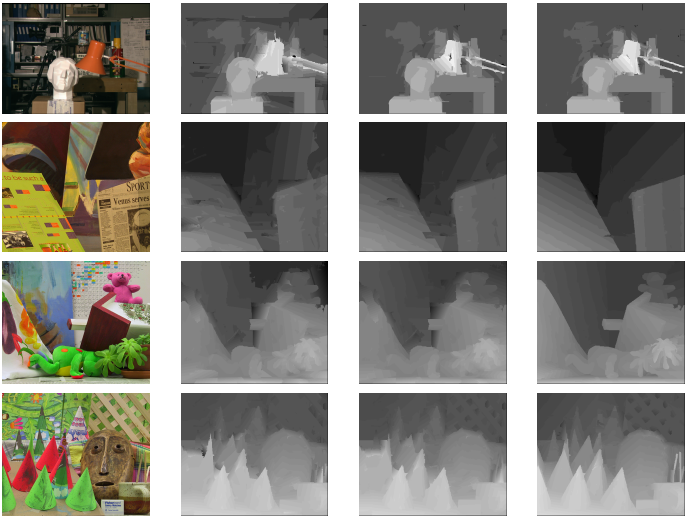
We first initialize the assignment z using methods loosely inspired by [16]. To initialize z we first run Felzenszwalb and Huttenlocher's efficient loopy BP algorithm using a classical stereo energy functional to compute a disparity value for each pixel [9]. We then do a least squares regression to fit a plane to the disparities in each segment. This gives an initial assignment z. Given an initial assignment z we then perform a max product variant of particle belief propagation [13]. More specifically, we iterate the following two steps.

1. Let C_i be a set of candidate values for z_i derived by repeatedly adding random noise to the current value of z_i.
2. Run discrete max-product BP with the finite value set C_i for each node i.
3. Set z_i to be the best value for i found in 2) and repeat.

The iteration can be stopped after a fixed number of iterations or when the energy is no longer reduced.

5 Experimental Results

We implement two training methods in our experiments — supervised and unsupervised. For each of the supervised and unsupervised training methods we train both a version with



Left image Iteration 1 Iteration 3 Iteration 5

Figure 2: Improvement with training on the Middlebury dataset.

texture cues for surface orientation and a version without such cues. In supervised training we set z_i (for each training image) by fitting a plane in each segment to the ground truth disparities for that segment. We then train the model using a contrastive divergence implementation of the hard M step (17) which we describe in more detail below. In supervised training we use only a single setting of z and run one iteration of (17). In unsupervised training we use the same separation into training and test pairs but do not use ground truth on the training pairs. Instead we iterate (16) and (17) six times starting with initial values for the parameters.

We use the inference algorithm described in section 4 to implement the hard E step (16). This uses a form of max-product particle belief propagation. Given an assignment z of a plane to each segment, we propose 15 additional candidate planes for each segment by adding Gaussian noise to the plane specified by z. The plane parameters A and B have units of pixels of disparity per pixel in the image, and hence are dimensionless. Typical values of $|A|$ and $|B|$ are from .1 to 1. In the proposal distribution we use Gaussian noise with a standard deviation of .007 for each of A and B and use a deviation of .1 pixels for C. We perform six rounds of proposing and selecting.

We implement the hard M step (17) by first breaking it down into (21) for training $P(z|x,\beta_z)$ and (22) for training $P(y|x,z,\beta_y)$. The form of the match energy (a simple quadratic energy) allows a closed form solution for (22). We implement (21) by gradient descent using a contrastive divergence approximation of the gradient in (23). We perform 8 gradient descent parameter updates with a constant learning rate. To estimate the expectation in (23) in each parameter update we generate 10 alternative plane assignments using single MCMC stochastic step starting at z and accepting or rejecting Gaussian noise added once to each plane. The MCMC process proposes a new plane for each segment by adding Gaussian noise and then accepting or rejecting that proposal using the standard Metropolis rejection rule.

Table 1 shows the performance of our system on the Middleburry stereo evaluation (version 2). The numbers shown are for unsupervised training with texture features. In this case

Avg.	Tsukuba			Venus			Teddy			Cones			Avg.
Rank	nonocc	all	disc	nonocc	all	disc	nonocc	all	disc	nonocc	all	disc	bad
31.4	3.12	5.22 44	13.9	1.03	1.17 28	11.5	7.08	7.30 3	16.1	6.90	10.7 26	16.0	8.33%

Table 1: Performance on the Middleburry stereo evaluation. The numbers shown are for unsupervised training with texture features.

all four images where used as unsupervised training data (ground truth disparities were not used in training). Figure 2 shows the inferred depth maps for the Middleburry images at various points in the parameter training. The figure shows a clear improvement as the parameters are trained.

	RMS Disparity Error (pixels)	Average Error $\lvert \log_{10} Z - \log_{10} \hat{Z} \rvert$
Saxena et al. [3]		.074
Unsuper., Notexture	1.158	.073
Unsuper., Texture	1.081	.069
Super., Notexture	1.071	.069
Super., Texture	1.001	.063

Table 2: RMS disparity error (in pixels) and average error (average base 10 logarithm of the multiplicative error) on the Stanford stereo pairs for four versions of our systems plus the best reported result from [3] on this data. Each system was either trained using the ground truth depth map (supervised) or trained purely from unlabeled stereo pairs (unsupervised) and either used texture cues (Texture) for surface orientation or did not (Notexture).

We have also run experiments on a set of rectified stereo pairs taken from the Stanford color stereo dataset [1] which has been used to train monocular depth estimation [2, 3, 4]. The images cover different types of outdoor scenes (buildings, grass, forests, trees, bushes, etc.) and some indoor scenes. They were epipolar rectified using a rectification kit from Fusiello et al. [14]. We removed from the dataset all pairs for which the energy value achieved by loopy BP was above a specified threshold. The majority of the eliminated images were cases where the rectification had failed. This left 200 out of an original 250 stereo pairs. Each stereo pair in this dataset is associated with ground truth depth information from a laser range finder. We randomly divide the 200 properly rectified stereo pairs into 180 training pairs and 20 test pairs. Results on this data set for four versions of our system are shown in table 2. Note that the texture information helps improve the performance in both supervised and unsupervised cases.

6 Conclusion

In many applications we would like to be able to build systems that learn from data collected from mechanical devices such as microphones and cameras. Stereo vision provides perhaps the simplest setting in which to study unsupervised learning. We have formulated an

[1] http://ai.stanford.edu/ asaxena/learningdepth/data

approach to unsupervised learning based on maximizing conditional likelihood and demonstrated its use for unsupervised learning of stereo depth with monocular depth cues. Ultimately we are interested in learning highly parameterized sophisticated models including, perhaps, models of surface types, shape from shading, albedo smoothness priors, lighting smoothness priors, and even object pose models. We believe that unsupervised learning based on maximizing conditional likelihood can be scaled to much more sophisticated models than those demonstrated in this paper.

References

[1] J. Aloimonos. Shape from texture. *Biol. Cybern.*, 58(5):345–360, 1988. ISSN 0340-1200. doi: http://dx.doi.org/10.1007/BF00363944.

[2] Andrew Y. Ng Ashutosh Saxena, Sung H. Chung. Learning depth from single monocular images. In *NIPS*, 2005.

[3] Jamie Schulte Ashutosh Saxena and Andrew Y. Ng. Depth estimation using monocular and stereo cues. In *IJCAI*, 2007.

[4] Min Sun Ashutosh Saxena and Andrew Y. Ng. 3-d depth reconstruction from a single still image. *IJCV*, 2007.

[5] Stan Birchfield and Carlo Tomasi. Multiway cut for stereo and motion with slanted surfaces. In *ICCV*, 1999.

[6] A. Blake and C. Marinos. Shape from texture: estimation, isotropy and moments. *Artificial Intelligence*, 45(3):323–80, 1990.

[7] M. A. Carreira-Perpiñán and G.E. Hinton. On contrastive divergence learning. In *10th Int. Workshop on Artificial Intelligence and Statistics (AISTATS 2005)*, 2005.

[8] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *CVPR*, pages I: 886–893, 2005.

[9] Pedro F. Felzenszwalb and Daniel P. Huttenlocher. Efficient belief propagation for early vision. *IJCV*, 70(1), October 2006.

[10] Pedro F. Felzenszwalb and Daniel P. Huttenlocher. Efficiently computing a good segmentation. In *In IEEE Conference on Computer Vision and Pattern Recognition*, pages 98–104, 1998.

[11] JM Hammersley and P Clifford. Markov fields on finite graphs and lattices. Unpublished Manuscript, 1971.

[12] Geoffrey E. Hinton. Training products of experts by minimizing contrastive divergence. *Neural Computation*, 14(8):1771–1800, 2002.

[13] Alexander Ihler and David McAllester. Particle belief propagation. In *AISTATS-09*, 2009.

[14] L. Irsara and A. Fusiello. Quasi-euclidean uncalibrated epipolar rectification. In *Rapporto di Ricerca RR 43/2006, Dipartimento di Informatica - Università di Verona*, 2006.

[15] Ross Kindermann and J. Laurie Snell. *Markov Random Fields and Their Applications*. Americal Mathematical Society, 1980.

[16] A. Klaus, M. Sormann, and K. Karner. Segment-based stereo matching using belief propagation and a self-adapting dissimilarity measure. In *ICPR-06*, 2006.

[17] Dan Kong and Hai Tao. A method for learning matching errors in stereo computation. In *BMVC*, 2004.

[18] John Lafferty, Andrew McCallum, and Fernando Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proc. 18th International Conf. on Machine Learning (ICML)*, pages 282–289. Morgan Kaufmann, San Francisco, CA, 2001.

[19] Stan Z. Li. *Markov Random Field Modeling in Computer Vision*. Springer-Vewrlag, 1995.

[20] Anthony Lobay and D.A. Forsyth. Recovering shape and irradiance maps from rich dense texton fields. In *Proceedings of Computer Vision and Pattern Recognition (CVPR)*, 2004.

[21] J. Malik and R. Rosenholtz. Computing local surface orientation and shape from texture for curved surfaces. *IJCV*, pages 149–168, 1997.

[22] Daniel Scharstein and Chris Pal. Learning conditional random fields for stereo. In *CVPR*, 2007.

[23] A.P. Witkin. Recovering surface shape and orientation from texture. *Artificial Intelligence*, 17:17–45, 1981.

[24] Li Zhang and Steven M. Seitz. Estimating optimal parameters for mrf stereo from a single image pair. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 29(2), 2007. based on "Parameter Estimation for MRF Stereo", CVPR 2005.