Make3D: Learning 3D Scene Structure from a Single Still Image

Ashutosh Saxena, Min Sun and Andrew Y. Ng

***Abstract*— We consider the problem of estimating detailed 3-d structure from a single still image of an unstructured environment. Our goal is to create 3-d models which are both quantitatively accurate as well as visually pleasing.**

For each small homogeneous patch in the image, we use a Markov Random Field (MRF) to infer a set of "plane parameters" that capture both the 3-d location and 3-d orientation of the patch. The MRF, trained via supervised learning, models both image depth cues as well as the relationships between different parts of the image. Other than assuming that the environment is made up of a number of small planes, our model makes no explicit assumptions about the structure of the scene; this enables the algorithm to capture much more detailed 3-d structure than does prior art, and also give a much richer experience in the 3-d flythroughs created using image-based rendering, even for scenes with significant non-vertical structure.

Using this approach, we have created qualitatively correct 3-d models for 64.9% of 588 images downloaded from the internet. We have also extended our model to produce large scale 3d models from a few images.[1]

***Index Terms*— Machine learning, Monocular vision, Learning depth, Vision and Scene Understanding, Scene Analysis: Depth cues.**

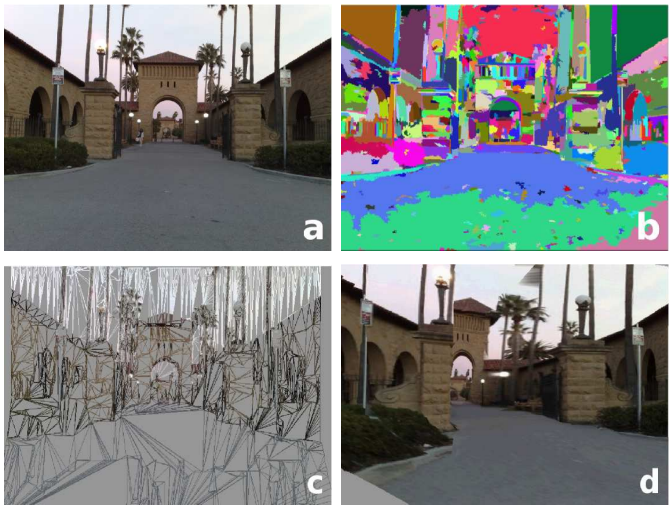


Fig. 1. (a) An original image. (b) Oversegmentation of the image to obtain "superpixels". (c) The 3-d model predicted by the algorithm. (d) A screenshot of the textured 3-d model.

I. Introduction

Upon seeing an image such as Fig. 1a, a human has no difficulty understanding its 3-d structure (Fig. 1c,d). However, inferring such 3-d structure remains extremely challenging for current computer vision systems. Indeed, in a narrow mathematical sense, it is impossible to recover 3-d depth from a single image, since we can never know if it is a picture of a painting (in which case the depth is flat) or if it is a picture of an actual 3-d environment. Yet in practice people perceive depth remarkably well given just one image; we would like our computers to have a similar sense of depths in a scene.

Understanding 3-d structure is a fundamental problem of computer vision. For the specific problem of 3-d reconstruction, most prior work has focused on stereovision [4], structure from motion [5], and other methods that require two (or more) images. These geometric algorithms rely on triangulation to estimate depths. However, algorithms relying only on geometry often end up ignoring the numerous additional *monocular* cues that can also be used to obtain rich 3-d information. In recent work, [6]–[9] exploited some of these cues to obtain some 3-d information. Saxena, Chung and Ng [6] presented an algorithm for predicting depths from monocular image features. [7] used monocular depth perception to drive a remote-controlled car autonomously. [8], [9] built models using a strong assumptions that the scene consists of ground/horizontal planes and vertical walls (and possibly sky); these methods therefore do not apply to the many scenes that are not made up only of vertical surfaces standing on a horizontal floor. Some examples include images of mountains, trees (e.g., Fig. 15b and 13d), staircases (e.g., Fig. 15a), arches (e.g., Fig. 11a and 15k), rooftops (e.g., Fig. 15m), etc. that often have much richer 3-d structure.

In this paper, our goal is to infer 3-d models that are both quantitatively accurate as well as visually pleasing. We use the insight that most 3-d scenes can be segmented into many small, approximately planar surfaces. (Indeed, modern computer graphics using OpenGL or DirectX models extremely complex scenes this way, using triangular facets to model even very complex shapes.) Our algorithm begins by taking an image, and attempting to segment it into many such small planar surfaces. Using a superpixel segmentation algorithm, [10] we find an over-segmentation of the image that divides it into many small regions (superpixels). An example of such a segmentation is shown in Fig. 1b. Because we use an over-segmentation, planar surfaces in the world may be broken up into many superpixels; however, each superpixel is likely to (at least approximately) lie entirely on only one planar surface.

For each superpixel, our algorithm then tries to infer the 3-d position and orientation of the 3-d surface that it came from. This 3-d surface is not restricted to just vertical and horizontal directions, but can be oriented in any direction. Inferring 3-d position from a single image is non-trivial, and humans do it using many different visual depth cues, such as texture (e.g., grass has a very different texture when viewed close up than when viewed far away); color (e.g., green patches are more likely to be grass on

Ashutosh Saxena, Min Sun and Andrew Y. Ng are with Computer Science Department, Stanford University, Stanford, CA 94305. Email: {asaxena,aliensun,ang}@cs.stanford.edu.

[1]Parts of this work were presented in [1], [2] and [3].

the ground; blue patches are more likely to be sky). Our algorithm uses supervised learning to learn how different visual cues like these are associated with different depths. Our learning algorithm uses a Markov random field model, which is also able to take into account constraints on the relative depths of nearby superpixels. For example, it recognizes that two adjacent image patches are more likely to be at the same depth, or to be even co-planar, than being very far apart.

Having inferred the 3-d position of each superpixel, we can now build a 3-d mesh model of a scene (Fig. 1c). We then texture-map the original image onto it to build a textured 3-d model (Fig. 1d) that we can fly through and view at different angles.

Other than assuming that the 3-d structure is made up of a number of small planes, we make no explicit assumptions about the structure of the scene. This allows our approach to generalize well, even to scenes with significantly richer structure than only vertical surfaces standing on a horizontal ground, such as mountains, trees, etc. Our algorithm was able to automatically infer 3-d models that were both qualitatively correct and visually pleasing for 64.9% of 588 test images downloaded from the internet. We further show that our algorithm predicts quantitatively more accurate depths than both previous work.

Extending these ideas, we also consider the problem of creating 3-d models of large novel environments, given only a small, sparse, set of images. In this setting, some parts of the scene may be visible in multiple images, so that triangulation cues (structure from motion) can be used to help reconstruct them; but larger parts of the scene may be visible only in one image. We extend our model to seamlessly combine triangulation cues and monocular image cues. This allows us to build full, photo-realistic 3-d models of larger scenes. Finally, we also demonstrate how we can incorporate object recognition information into our model. For example, if we detect a standing person, we know that people usually stand on the floor and thus their feet must be at ground-level. Knowing approximately how tall people are also helps us to infer their depth (distance) from the camera; for example, a person who is 50 pixels tall in the image is likely about twice as far as one who is 100 pixels tall. (This is also reminiscent of [11], who used a car and pedestrian detector and the known size of cars/pedestrians to estimate the position of the horizon.)

The rest of this paper is organized as follows. Section II discusses the prior work. Section III describes the intuitions we draw from human vision. Section IV describes the representation we choose for the 3-d model. Section V describes our probabilistic models, and Section VI describes the features used. Section VII describes the experiments we performed to test our models. Section VIII extends our model to the case of building large 3-d models from sparse views. Section IX demonstrates how information from object recognizers can be incorporated into our models for 3-d reconstruction, and Section X concludes.

II. Prior Work

For a few specific settings, several authors have developed methods for depth estimation from a single image. Examples include shape-from-shading [12], [13] and shape-from-texture [14], [15]; however, these methods are difficult to apply to surfaces that do not have fairly uniform color and texture. Nagai et al. [16] used Hidden Markov Models to performing surface reconstruction from single images for known, fixed objects such as hands and faces. Hassner and Basri [17] used an example-based approach to estimate depth of an object from a known object class. Han and Zhu [18] performed 3-d reconstruction for known specific classes of objects placed in untextured areas. Criminisi, Reid and Zisserman [19] provided an interactive method for computing 3-d geometry, where the user can specify the object segmentation, 3-d coordinates of some points, and reference height of an object. Torralba and Oliva [20] studied the relationship between the Fourier spectrum of an image and its mean depth.

In recent work, Saxena, Chung and Ng (SCN) [6], [21] presented an algorithm for predicting depth from monocular image features; this algorithm was also successfully applied for improving the performance of stereovision [22]. Michels, Saxena and Ng [7] also used monocular depth perception and reinforcement learning to drive a remote-controlled car autonomously in unstructured environments. Delage, Lee and Ng (DLN) [8], [23] and Hoiem, Efros and Hebert (HEH) [9] assumed that the environment is made of a flat ground with vertical walls. DLN considered indoor images, while HEH considered outdoor scenes. They classified the image into horizontal/ground and vertical regions (also possibly sky) to produce a simple "pop-up" type fly-through from an image.

Our approach uses a Markov Random Field (MRF) to model monocular cues and the relations between various parts of the image. MRFs are a workhorse of machine learning, and have been applied to various problems in which local features were insufficient and more contextual information had to be used. Examples include stereovision [4], [22], image segmentation [10], and object classification [24].

There is also ample prior work in 3-d reconstruction from multiple images, as in stereovision and structure from motion. It is impossible for us to do this literature justice here, but recent surveys include [4] and [25], and we discuss this work further in Section VIII.

III. Visual Cues for Scene Understanding

Images are formed by a projection of the 3-d scene onto two dimensions. Thus, given only a single image, the true 3-d structure is ambiguous, in that an image might represent an infinite number of 3-d structures. However, not all of these possible 3-d structures are equally likely. The environment we live in is reasonably structured, and thus humans are usually able to infer a (nearly) correct 3-d structure, using prior experience.

Given a single image, humans use a variety of monocular cues to infer the 3-d structure of the scene. Some of these cues are based on local properties of the image, such as texture variations and gradients, color, haze, and defocus [6], [26], [27]. For example, the texture of surfaces appears different when viewed at different distances or orientations. A tiled floor with parallel lines will also appear to have tilted lines in an image, such that distant regions will have larger variations in the line orientations, and nearby regions will have smaller variations in line orientations. Similarly, a grass field when viewed at different orientations/distances will appear different. We will capture some of these cues in our model. However, we note that local image cues alone are usually insufficient to infer the 3-d structure. For example, both blue sky and a blue object would give similar local features; hence it is difficult to estimate depths from local features alone.

Fig. 2. (Left) An image of a scene. (Right) Oversegmented image. Each small segment (superpixel) lies on a plane in the 3d world. (*Best viewed in color.*)

The ability of humans to "integrate information" over space, i.e. understand the relation between different parts of the image, is crucial to understanding the scene's 3-d structure. [27, chap. 11] For example, even if part of an image is a homogeneous, featureless, gray patch, one is often able to infer its depth by looking at nearby portions of the image, so as to recognize whether this patch is part of a sidewalk, a wall, etc. Therefore, in our model we will also capture relations between different parts of the image.

Humans recognize many visual cues, such that a particular shape may be a building, that the sky is blue, that grass is green, that trees grow above the ground and have leaves on top of them, and so on. In our model, both the relation of monocular cues to the 3-d structure, as well as relations between various parts of the image, will be learned using supervised learning. Specifically, our model will be trained to estimate depths using a training set in which the ground-truth depths were collected using a laser scanner.

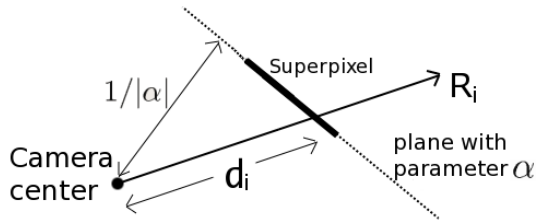


Fig. 3. A 2-d illustration to explain the plane parameter α and rays R from the camera.

IV. Representation

Our goal is to create a full photo-realistic 3-d model from an image. Following most work on 3-d models in computer graphics and other related fields, we will use a polygonal mesh representation of the 3-d model, in which we assume the world is made of a set of small planes.[2] In detail, given an image of the scene, we first find small homogeneous regions in the image, called "Superpixels" [10]. Each such region represents a coherent region in the scene with all the pixels having similar properties. (See Fig. 2.) Our basic unit of representation will be these small planes in the world, and our goal is to infer the location and orientation of each one.

[2]This assumption is reasonably accurate for most artificial structures, such as buildings. Some natural structures such as trees could perhaps be better represented by a cylinder. However, since our models are quite detailed, e.g., about 2000 planes for a small scene, the planar assumption works quite well in practice.

Fig. 4. (Left) Original image. (Right) Superpixels overlaid with an illustration of the Markov Random Field (MRF). The MRF models the relations (shown by the edges) between neighboring superpixels. (Only a subset of nodes and edges shown.)

More formally, we parametrize both the 3-d location and orientation of the infinite plane on which a superpixel lies by using a set of plane parameters $\alpha \in \mathbb{R}^3$. (Fig. 3) (Any point $q \in \mathbb{R}^3$ lying on the plane with parameters α satisfies $\alpha^T q = 1$.) The value $1/|\alpha|$ is the distance from the camera center to the closest point on the plane, and the normal vector $\hat{\alpha} = \frac{\alpha}{|\alpha|}$ gives the orientation of the plane. If R_i is the unit vector (also called the ray R_i) from the camera center to a point i lying on a plane with parameters α, then $d_i = 1/R_i^T \alpha$ is the distance of point i from the camera center.

V. Probabilistic Model

It is difficult to infer 3-d information of a region from local cues alone (see Section III), and one needs to infer the 3-d information of a region in relation to the 3-d information of other regions.

In our MRF model, we try to capture the following properties of the images:

- **Image Features and depth**: The image features of a superpixel bear some relation to the depth (and orientation) of the superpixel.
- **Connected structure**: Except in case of occlusion, neighboring superpixels are more likely to be connected to each other.
- **Co-planar structure**: Neighboring superpixels are more likely to belong to the same plane, if they have similar features and if there are no edges between them.
- **Co-linearity**: Long straight lines in the image plane are more likely to be straight lines in the 3-d model. For example, edges of buildings, sidewalk, windows.

Note that no single one of these four properties is enough, by itself, to predict the 3-d structure. For example, in some cases, local image features are not strong indicators of the depth (and orientation) (e.g., a patch on a blank feature-less wall). Thus, our approach will combine these properties in an MRF, in a way that depends on our "confidence" in each of these properties. Here, the "confidence" is itself estimated from local image cues, and will vary from region to region in the image.

Our MRF is composed of five types of nodes. The input to the MRF occurs through two variables, labeled x and ϵ. These variables correspond to features computed from the image pixels (see Section VI for details.) and are always observed; thus the MRF is conditioned on these variables. The variables ν

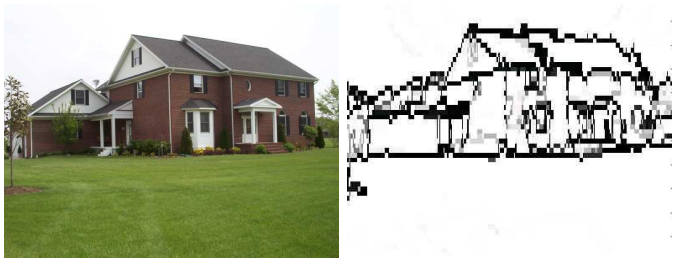


Fig. 5. (Left) An image of a scene. (Right) Inferred "soft" values of $y_{ij} \in [0, 1]$. ($y_{ij} = 0$ indicates an occlusion boundary/fold, and is shown in black.) Note that even with the inferred y_{ij} being not completely accurate, the plane parameter MRF will be able to infer "correct" 3-d models.

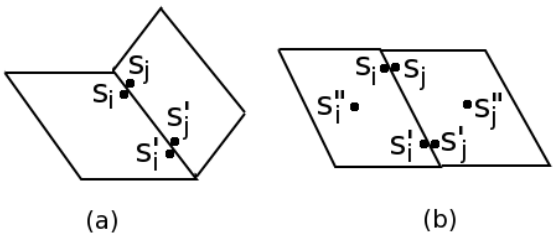


Fig. 6. Illustration explaining effect of the choice of s_i and s_j on enforcing (a) Connected structure and (b) Co-planarity.

indicate our degree of confidence in a depth estimate obtained only from local image features. The variables y indicate the presence or absence of occlusion boundaries and folds in the image. These variables are used to selectively enforce coplanarity and connectivity between superpixels. Finally, the variables α are the plane parameters that are inferred using the MRF, which we call "Plane Parameter MRF."[3]

Occlusion Boundaries and Folds: We use the variables $y_{ij} \in \{0, 1\}$ to indicate whether an "edgel" (the edge between two neighboring superpixels) is an occlusion boundary/fold or not. The inference of these boundaries is typically not completely accurate; therefore we will infer *soft* values for y_{ij}. (See Fig. 5.) More formally, for an edgel between two superpixels i and j, $y_{ij} = 0$ indicates an occlusion boundary/fold, and $y_{ij} = 1$ indicates none (i.e., a planar surface).

In many cases, strong image gradients do not correspond to the occlusion boundary/fold, e.g., a shadow of a building falling on a ground surface may create an edge between the part with a shadow and the one without. An edge detector that relies just on these local image gradients would mistakenly produce an edge. However, there are other visual cues beyond local image gradients that better indicate whether two planes are connected/coplanar or not. Using learning to combine a number of such visual features makes the inference more accurate. In [28], Martin, Fowlkes and Malik used local brightness, color and texture for learning segmentation boundaries. Here, our goal is to learn occlusion boundaries and folds. In detail, we model y_{ij} using a logistic response as $P(y_{ij} = 1|\epsilon_{ij}; \psi) = 1/(1 + \exp(-\psi^T \epsilon_{ij}))$. where, ϵ_{ij} are features of the superpixels i and j (Section VI-B), and ψ are the parameters of the model. During inference, we will use a mean field-like approximation, where we replace y_{ij} with its mean value under the logistic model.

Now, we will describe how we model the distribution of the plane parameters α, conditioned on y.

Fractional depth error: For 3-d reconstruction, the fractional (or relative) error in depths is most meaningful; it is used in structure for motion, stereo reconstruction, etc. [4], [29] For ground-truth depth d, and estimated depth $\hat{d}$, fractional error is defined as $(\hat{d} - d)/d = \hat{d}/d - 1$. Therefore, we will be penalizing fractional errors in our MRF.

MRF Model: To capture the relation between the plane parameters and the image features, and other properties such as co-planarity, connectedness and co-linearity, we formulate our MRF as

$$P(\alpha|X, \nu, y, R; \theta) = \frac{1}{Z} \prod_i f_1(\alpha_i|X_i, \nu_i, R_i; \theta) \prod_{i,j} f_2(\alpha_i, \alpha_j|y_{ij}, R_i, R_j) \quad (1)$$

where, α_i is the plane parameter of the superpixel i. For a total of S_i points in the superpixel i, we use x_{i,s_i} to denote the features for point s_i in the superpixel i. $X_i = \{x_{i,s_i} \in \mathbb{R}^{524} : s_i = 1, ..., S_i\}$ are the features for the superpixel i. (Section VI-A) Similarly, $R_i = \{R_{i,s_i} : s_i = 1, ..., S_i\}$ is the set of rays for superpixel i.[4] ν is the "confidence" in how good the (local) image features are in predicting depth (more details later).

The first term $f_1(\cdot)$ models the plane parameters as a function of the image features x_{i,s_i}. We have $R^T_{i,s_i}\alpha_i = 1/d_{i,s_i}$ (where R_{i,s_i} is the ray that connects the camera to the 3-d location of point s_i), and if the estimated depth $\hat{d}_{i,s_i} = x^T_{i,s_i}\theta_r$, then the fractional error would be

$$\frac{\hat{d}_{i,s_i} - d_{i,s_i}}{d_{i,s_i}} = \frac{1}{d_{i,s_i}}(\hat{d}_{i,s_i}) - 1 = R^T_{i,s_i}\alpha_i(x^T_{i,s_i}\theta_r) - 1$$

Therefore, to minimize the aggregate fractional error over all the points in the superpixel, we model the relation between the plane parameters and the image features as

$$f_1(\alpha_i|X_i, \nu_i, R_i; \theta) = \exp\left(-\sum_{s_i=1}^{S_i} \nu_{i,s_i} \left|R^T_{i,s_i}\alpha_i(x^T_{i,s_i}\theta_r) - 1\right|\right) \quad (2)$$

The parameters of this model are $\theta_r \in \mathbb{R}^{524}$. We use different parameters (θ_r) for rows $r = 1, ..., 11$ in the image, because the images we consider are roughly aligned upwards (i.e., the direction of gravity is roughly downwards in the image), and thus it allows our algorithm to learn some regularities in the images—that different rows of the image have different statistical properties. E.g., a blue superpixel might be more likely to be sky if it is in the upper part of image, or water if it is in the lower part of the image, or that in the images of environments available on the internet, the horizon is more likely to be in the middle one-third of the image. (In our experiments, we obtained very similar results using a number of rows ranging from 5 to 55.) Here, $\nu_i = \{\nu_{i,s_i} : s_i = 1, ..., S_i\}$ indicates the confidence

[3]For comparison, we also present an MRF that only models the 3-d location of the points in the image ("Point-wise MRF," see Appendix).

[4]The rays are obtained by making a reasonable guess on the camera intrinsic parameters—that the image center is the origin and the pixel-aspect-ratio is one—unless known otherwise from the image headers.

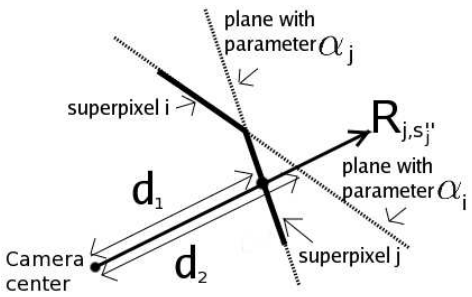
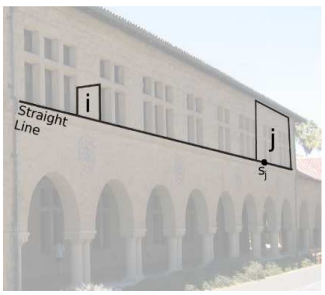
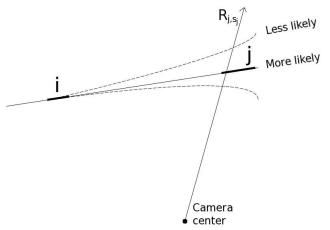


Fig. 7. A 2-d illustration to explain the co-planarity term. The distance of the point s_j on superpixel j to the plane on which superpixel i lies along the ray R_{j,s_j}" is given by $d_1 - d_2$.



(a) 2-d image



(b) 3-d world, top view

Fig. 8. Co-linearity. (a) Two superpixels i and j lying on a straight line in the 2-d image, (b) An illustration showing that a long straight line in the image plane is more likely to be a straight line in 3-d.

of the features in predicting the depth $\hat{d}_{i,s_i}$ at point s_i.[5] If the local image features were not strong enough to predict depth for point s_i, then $\nu_{i,s_i} = 0$ turns off the effect of the term $\left|R_{i,s_i}^T \alpha_i (x_{i,s_i}^T \theta_r) - 1\right|$.

The second term $f_2(\cdot)$ models the relation between the plane parameters of two superpixels i and j. It uses pairs of points s_i and s_j to do so:

$$f_2(\cdot) = \textstyle\prod_{\{s_i,s_j\} \in N} h_{s_i,s_j}(\cdot) \tag{3}$$

We will capture co-planarity, connectedness and co-linearity, by different choices of $h(\cdot)$ and $\{s_i, s_j\}$.

Connected structure: We enforce this constraint by choosing s_i and s_j to be on the boundary of the superpixels i and j. As shown in Fig. 6a, penalizing the distance between two such points ensures that they remain fully connected. The relative (fractional) distance between points s_i and s_j is penalized by

$$h_{s_i,s_j}(\alpha_i, \alpha_j, y_{ij}, R_i, R_j) = \exp\left(-y_{ij} |(R_{i,s_i}^T \alpha_i - R_{j,s_j}^T \alpha_j)\hat{d}|\right) \tag{4}$$

In detail, $R_{i,s_i}^T \alpha_i = 1/d_{i,s_i}$ and $R_{j,s_j}^T \alpha_j = 1/d_{j,s_j}$; therefore, the term $(R_{i,s_i}^T \alpha_i - R_{j,s_j}^T \alpha_j)\hat{d}$ gives the fractional distance $|(d_{i,s_i} - d_{j,s_j})/\sqrt{d_{i,s_i} d_{j,s_j}}|$ for $\hat{d} = \sqrt{\hat{d}_{s_i} \hat{d}_{s_j}}$. Note that in case of occlusion, the variables $y_{ij} = 0$, and hence the two superpixels will not be forced to be connected.

Co-planarity: We enforce the co-planar structure by choosing a third pair of points s_i'' and s_j'' in the center of each superpixel along with ones on the boundary. (Fig. 6b) To enforce co-planarity, we penalize the relative (fractional) distance of point s_j'' from the plane in which superpixel i lies, along the ray $R_{j,s_j''}$ (See Fig. 7).

$$h_{s_j''}(\alpha_i, \alpha_j, y_{ij}, R_{j,s_j''}) = \exp\left(-y_{ij} |(R_{j,s_j''}^T \alpha_i - R_{j,s_j''}^T \alpha_j)\hat{d}_{s_j''}|\right) \tag{5}$$

with $h_{s_i'',s_j''}(\cdot) = h_{s_i''}(\cdot) h_{s_j''}(\cdot)$. Note that if the two superpixels are coplanar, then $h_{s_i'',s_j''} = 1$. To enforce co-planarity between two distant planes that are not connected, we can choose three such points and use the above penalty.

Co-linearity: Consider two superpixels i and j lying on a long straight line in a 2-d image (Fig. 8a). There are an infinite number of curves that would project to a straight line in the image plane; however, a straight line in the image plane is more likely to be a straight one in 3-d as well (Fig. 8b). In our model, therefore, we will penalize the relative (fractional) distance of a point (such as s_j) from the ideal straight line.

In detail, consider two superpixels i and j that lie on planes parameterized by α_i and α_j respectively in 3-d, and that lie on a straight line in the 2-d image. For a point s_j lying on superpixel j, we will penalize its (fractional) distance along the ray R_{j,s_j} from the 3-d straight line passing through superpixel i. I.e.,

$$h_{s_j}(\alpha_i, \alpha_j, y_{ij}, R_{j,s_j}) = \exp\left(-y_{ij} |(R_{j,s_j}^T \alpha_i - R_{j,s_j}^T \alpha_j)\hat{d}|\right) \tag{6}$$

with $h_{s_i,s_j}(\cdot) = h_{s_i}(\cdot) h_{s_j}(\cdot)$. In detail, $R_{j,s_j}^T \alpha_j = 1/d_{j,s_j}$ and $R_{j,s_j}^T \alpha_i = 1/d'_{j,s_j}$; therefore, the term $(R_{j,s_j}^T \alpha_i - R_{j,s_j}^T \alpha_j)\hat{d}$ gives the fractional distance $\left|(d_{j,s_j} - d'_{j,s_j})/\sqrt{d_{j,s_j} d'_{j,s_j}}\right|$ for $\hat{d} = \sqrt{\hat{d}_{j,s_j} \hat{d}'_{j,s_j}}$. The "confidence" y_{ij} depends on the length of the line and its curvature—a long straight line in 2-d is more likely to be a straight line in 3-d.

Parameter Learning and MAP Inference: Exact parameter learning of the model is intractable; therefore, we use Multi-Conditional Learning (MCL) for approximate learning, where the graphical model is approximated by a product of several marginal conditional likelihoods [30], [31]. In particular, we estimate the θ_r parameters efficiently by solving a Linear Program (LP). (See Appendix for more details.)

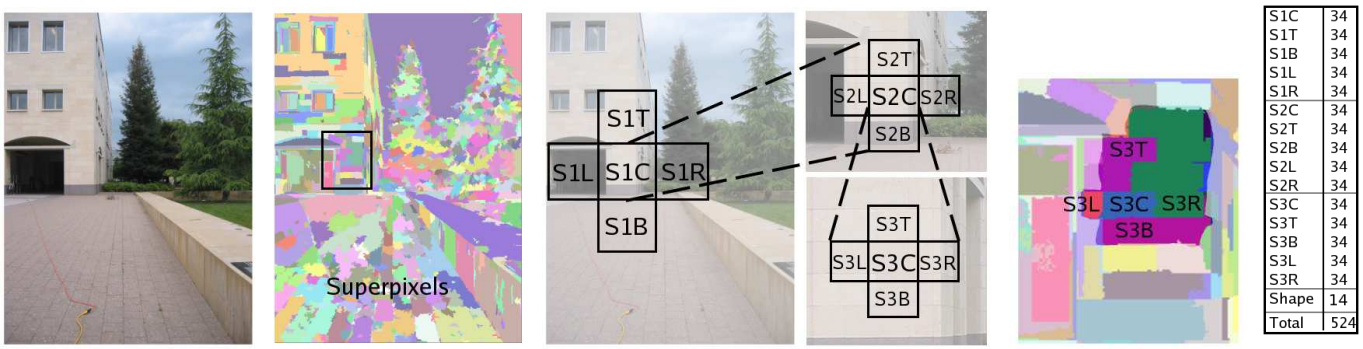
MAP inference of the plane parameters α, i.e., maximizing the conditional likelihood $P(\alpha|X, \nu, y, R; \theta)$, is efficiently performed by solving a LP. We implemented an efficient method that uses the sparsity in our problem, so that inference can be performed in about 4-5 seconds for an image having about 2000 superpixels on a single-core Intel 3.40GHz CPU with 2 GB RAM. (See Appendix for more details.)

VI. Features

For each superpixel, we compute a battery of features to capture some of the monocular cues discussed in Section III. We also compute features to predict meaningful boundaries in the images, such as occlusion and folds. We rely on a large number of different types of features to make our algorithm more robust and to make it generalize even to images that are very different from the training set.

[5]The variable ν_{i,s_i} is an indicator of how good the image features are in predicting depth for point s_i in superpixel i. We learn ν_{i,s_i} from the monocular image features, by estimating the expected value of $|d_i - x_i^T \theta_r|/d_i$ as $\phi_r^T x_i$ with logistic response, with ϕ_r as the parameters of the model, features x_i and d_i as ground-truth depths.

Fig. 9. The convolutional filters used for texture energies and gradients. The first 9 are 3x3 Laws' masks. The last 6 are the oriented edge detectors at 30^0. The first nine Law's masks do local averaging, edge detection and spot detection. The 15 Laws' mask are applied to the image to the Y channel of the image. We apply only the first averaging filter to the color channels Cb and Cr; thus obtain 17 filter responses, for each of which we calculate energy and kurtosis to obtain 34 features of each patch.



(a) (b) (c) (d) (e)

Fig. 10. The feature vector. (a) The original image, (b) Superpixels for the image, (c) An illustration showing the location of the neighbors of superpixel S3C at multiple scales, (d) Actual neighboring superpixels of S3C at the finest scale, (e) Features from each neighboring superpixel along with the superpixel-shape features give a total of 524 features for the superpixel S3C. (*Best viewed in color.*)

A. Monocular Image Features

For each superpixel at location i, we compute both texture-based summary statistic features and superpixel shape and location based features. Similar to SCN, we use the output of 17 filters (9 Laws masks, 2 color channels in YCbCr space and 6 oriented edges, see Fig. 10). These are commonly used filters that capture the texture of a 3x3 patch and the edges at various orientations. The filters outputs $F_n(x, y)$, $n = 1, ..., 17$ are incorporated into $E_i(n) = \sum_{(x,y)\in S_i} |I(x, y) * F_n(x, y)|^k$, where k = 2,4 gives the energy and kurtosis respectively. This gives a total of 34 values for each superpixel. We compute features for each superpixel to improve performance over SCN, who computed them only for fixed rectangular patches. Our superpixel shape and location based features (14, computed only for the superpixel) included the shape and location based features in Section 2.2 of [9], and also the eccentricity of the superpixel. (See Fig. 10.)

We attempt to capture more "contextual" information by also including features from neighboring superpixels (we pick the largest four in our experiments), and at multiple spatial scales (three in our experiments). (See Fig. 10.) The features, therefore, contain information from a larger portion of the image, and thus are more expressive than just local features. This makes the feature vector x_i of a superpixel $34 * (4 + 1) * 3 + 14 = 524$ dimensional.

B. Features for Boundaries

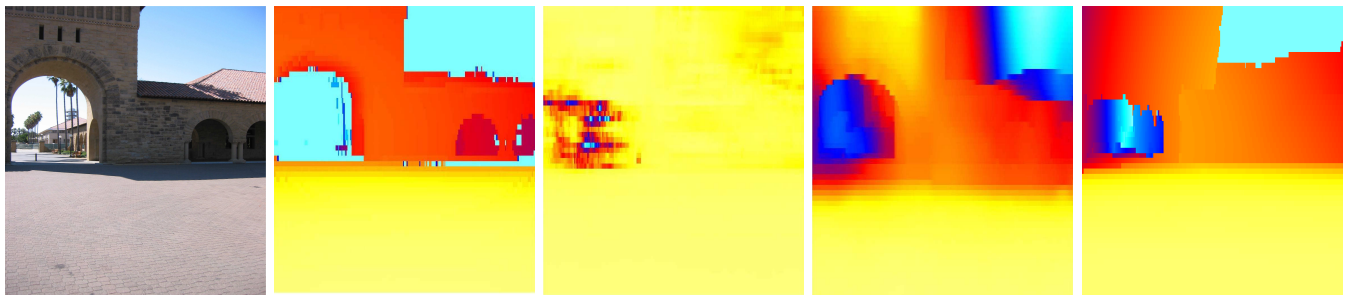
Another strong cue for 3-d structure perception is boundary information. If two neighboring superpixels of an image display different features, humans would often perceive them to be parts of different objects; therefore an edge between two superpixels with distinctly different features, is a candidate for a occlusion boundary or a fold. To compute the features ϵ_{ij} between superpixels i and j, we first generate 14 different segmentations for each image for 2 different scales for 7 different properties based on textures, color, and edges. We modified [10] to create segmentations based on these properties. Each element of our 14 dimensional feature vector ϵ_{ij} is then an indicator if two superpixels i and j lie in the same segmentation. For example, if two superpixels belong to the same segments in all the 14 segmentations then it is more likely that they are coplanar or connected. Relying on multiple segmentation hypotheses instead of one makes the detection of boundaries more robust. The features ϵ_{ij} are the input to the classifier for the occlusion boundaries and folds.

VII. Experiments

A. Data collection

We used a custom-built 3-D scanner to collect images (e.g., Fig. 11a) and their corresponding depthmaps using lasers (e.g., Fig. 11b). We collected a total of 534 images+depthmaps, with an image resolution of 2272x1704 and a depthmap resolution of 55x305, and used 400 for training our model. These images were collected during daytime in a diverse set of urban and natural areas in the city of Palo Alto and its surrounding regions.

We tested our model on rest of the 134 images (collected using our 3-d scanner), and also on 588 internet images. The internet images were collected by issuing keywords on Google image search. To collect data and to perform the evaluation of the algorithms in a completely unbiased manner, a person *not* associated with the project was asked to collect images of environments (greater than 800x600 size). The person chose the following keywords to collect the images: campus, garden, park, house, building, college, university, church, castle, court, square, lake, temple, scene. The images thus collected were from places from all over the world, and contained environments that were significantly different from the training set, e.g. hills, lakes, night scenes, etc. The person chose only those images which were of "environments," i.e. she removed images of the geometrical



(a) (b) (c) (d) (e)

Fig. 11. (a) Original Image, (b) Ground truth depthmap, (c) Depth from image features only, (d) Point-wise MRF, (e) Plane parameter MRF. (*Best viewed in color.*)

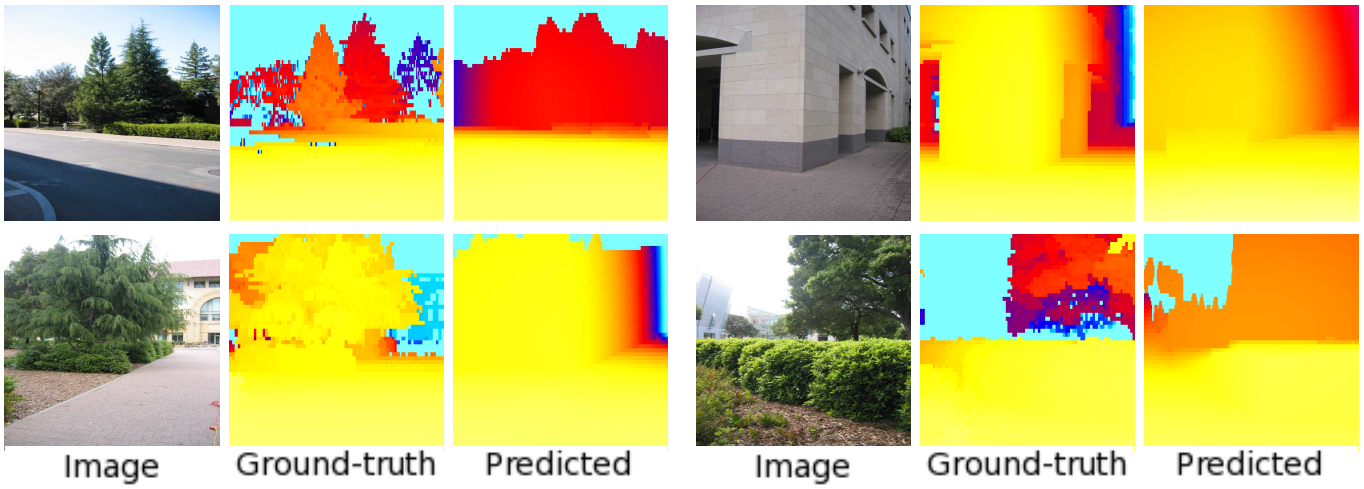


Fig. 12. Typical depthmaps predicted by our algorithm on hold-out test set, collected using the laser-scanner. (*Best viewed in color.*)

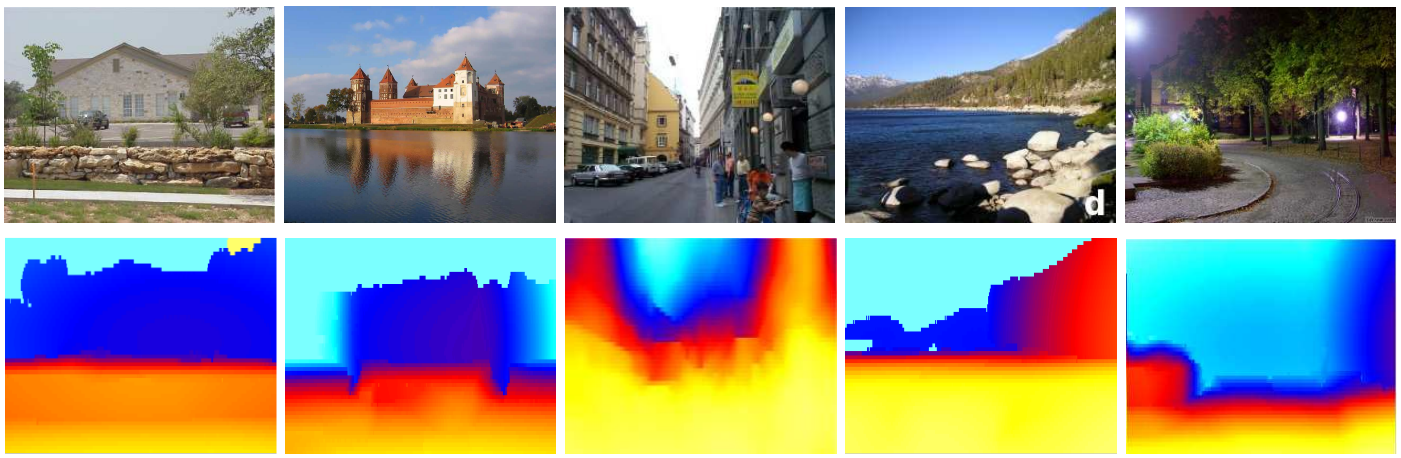


Fig. 13. Typical results from our algorithm. (Top row) Original images, (Bottom row) depthmaps (shown in log scale, yellow is closest, followed by red and then blue) generated from the images using our plane parameter MRF. (*Best viewed in color.*)

figure 'square' when searching for keyword 'square'; no other pre-filtering was done on the data.

In addition, we manually labeled 50 images with 'ground-truth' boundaries to learn the parameters for occlusion boundaries and folds.

B. Results and Discussion

We performed an extensive evaluation of our algorithm on 588 internet test images, and 134 test images collected using the laser scanner.

In Table I, we compare the following algorithms:
(a) Baseline: Both for pointwise MRF (Baseline-1) and plane parameter MRF (Baseline-2). The Baseline MRF is trained without any image features, and thus reflects a "prior" depthmap of sorts.
(b) Our Point-wise MRF: with and without constraints (connectivity, co-planar and co-linearity).
(c) Our Plane Parameter MRF (PP-MRF): without any constraint, with co-planar constraint only, and the full model.
(d) Saxena et al. (SCN), [6], [21] applicable for quantitative errors only.

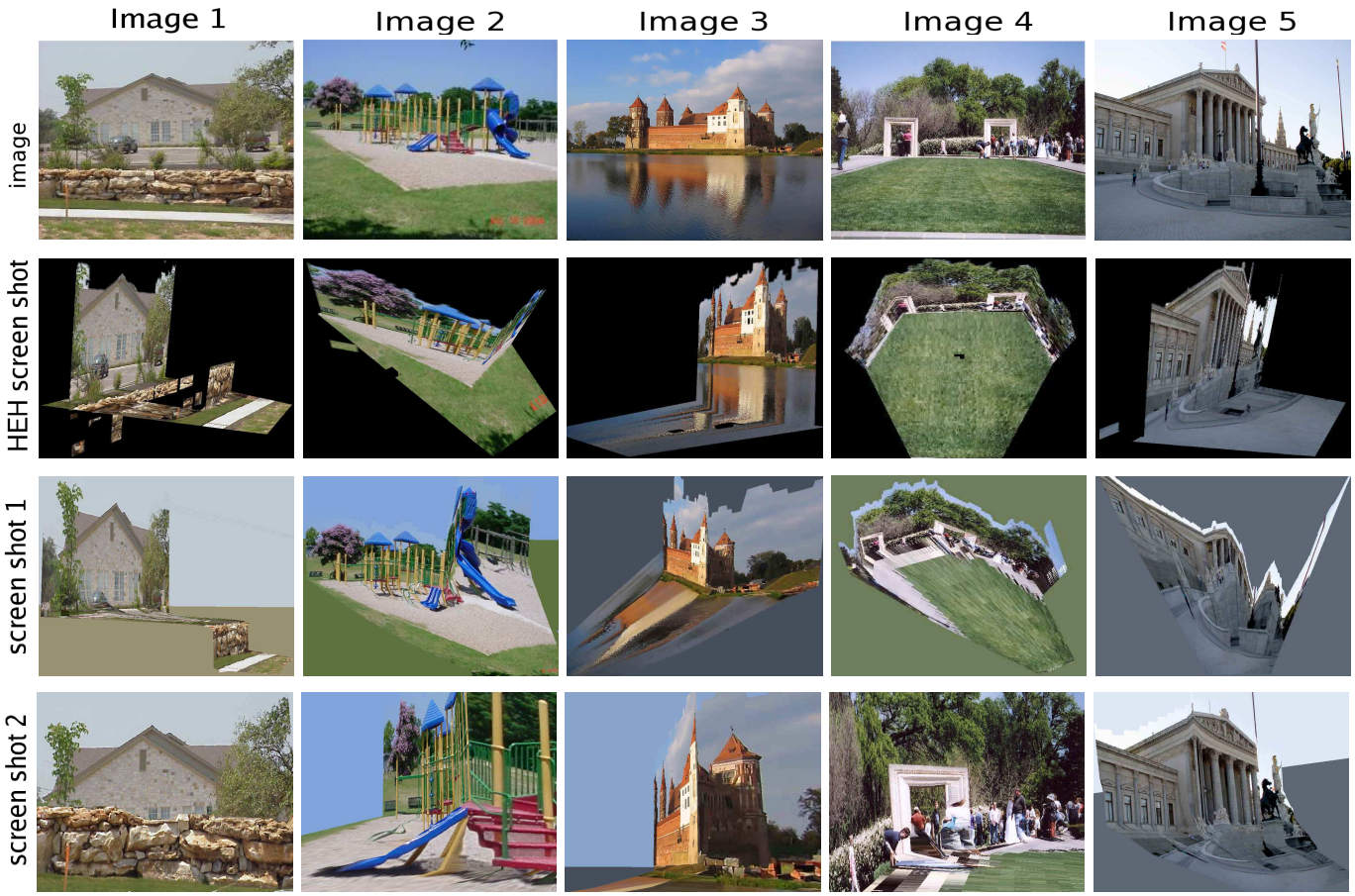


Fig. 14. Typical results from HEH and our algorithm. **Row 1**: Original Image. **Row 2**: 3-d model generated by HEH, **Row 3 and 4**: 3-d model generated by our algorithm. (Note that the screenshots cannot be simply obtained from the original image by an affine transformation.) In **image 1**, HEH makes mistakes in some parts of the foreground rock, while our algorithm predicts the correct model; with the rock occluding the house, giving a novel view. In **image 2**, HEH algorithm detects a wrong ground-vertical boundary; while our algorithm not only finds the correct ground, but also captures a lot of non-vertical structure, such as the blue slide. In **image 3**, HEH is confused by the reflection; while our algorithm produces a correct 3-d model. In **image 4**, HEH and our algorithm produce roughly equivalent results—HEH is a bit more visually pleasing and our model is a bit more detailed. In **image 5**, both HEH and our algorithm fail; HEH just predict one vertical plane at a incorrect location. Our algorithm predicts correct depths of the pole and the horse, but is unable to detect their boundary; hence making it qualitatively incorrect.

TABLE I
RESULTS: QUANTITATIVE COMPARISON OF VARIOUS METHODS.

METHOD	CORRECT (%)	% PLANES CORRECT	$\log_{10}$	REL
SCN	NA	NA	0.198	0.530
HEH	33.1%	50.3%	0.320	1.423
BASELINE-1	0%	NA	0.300	0.698
NO PRIORS	0%	NA	0.170	0.447
POINT-WISE MRF	23%	NA	**0.149**	0.458
BASELINE-2	0%	0%	0.334	0.516
NO PRIORS	0%	0%	0.205	0.392
CO-PLANAR	45.7%	57.1%	0.191	0.373
PP-MRF	**64.9%**	**71.2%**	0.187	**0.370**

(e) Hoiem et al. (HEH) [9]. For fairness, we scale and shift their depthmaps before computing the errors to match the global scale of our test images. Without the scaling and shifting, their error is much higher (7.533 for relative depth error).

We compare the algorithms on the following metrics: (a) % of models qualitatively correct, (b) % of major planes correctly identified,[6] (c) Depth error $|\log d - \log \hat{d}|$ on a log-10 scale, averaged over all pixels in the hold-out test set, (d) Average relative depth error $\frac{|d-\hat{d}|}{d}$. (We give these two numerical errors on only the 134 test images that we collected, because ground-truth laser depths are not available for internet images.)

Table I shows that both of our models (Point-wise MRF and Plane Parameter MRF) outperform the other algorithms in quantitative accuracy in depth prediction. Plane Parameter MRF gives better relative depth accuracy and produces sharper depthmaps (Fig. 11, 12 and 13). Table I also shows that by capturing the image properties of connected structure, co-planarity and co-linearity, the models produced by the algorithm become significantly better. In addition to reducing quantitative errors, PP-MRF does indeed produce significantly better 3-d models. When producing 3-d flythroughs, even a small number of erroneous planes make the 3-d model visually unacceptable, even though

[6]For the first two metrics, we define a model as correct when for 70% of the major planes in the image (major planes occupy more than 15% of the area), the plane is in correct relationship with its nearest neighbors (i.e., the relative orientation of the planes is within 30 degrees). Note that changing the numbers, such as 70% to 50% or 90%, 15% to 10% or 30%, and 30 degrees to 20 or 45 degrees, gave similar trends in the results.

TABLE II
PERCENTAGE OF IMAGES FOR WHICH HEH IS BETTER, OUR PP-MRF IS BETTER, OR IT IS A TIE.

ALGORITHM	%BETTER
TIE	15.8%
HEH	22.1%
PP-MRF	**62.1%**

the quantitative numbers may still show small errors.

Our algorithm gives qualitatively correct models for 64.9% of images as compared to 33.1% by HEH. The qualitative evaluation was performed by a person not associated with the project following the guidelines in Footnote 6. Delage, Lee and Ng [8] and HEH generate a popup effect by folding the images at "ground-vertical" boundaries—an assumption which is not true for a significant number of images; therefore, their method fails in those images. Some typical examples of the 3-d models are shown in Fig. 14. (Note that all the *test* cases shown in Fig. 1, 13, 14 and 15 are from the dataset downloaded from the internet, except Fig. 15a which is from the laser-test dataset.) These examples also show that our models are often more detailed, in that they are often able to model the scene with a multitude (over a hundred) of planes.

We performed a further comparison. Even when both algorithms are evaluated as qualitatively correct on an image, one result could still be superior. Therefore, we asked the person to compare the two methods, and decide which one is better, or is a tie.[7] Table II shows that our algorithm outputs the better model in 62.1% of the cases, while HEH outputs better model in 22.1% cases (tied in the rest).

Full documentation describing the details of the unbiased human judgment process, along with the 3-d flythroughs produced by our algorithm, is available online at:

http://make3d.stanford.edu/research

Some of our models, e.g. in Fig. 15j, have cosmetic defects—e.g. stretched texture; better texture rendering techniques would make the models more visually pleasing. In some cases, a small mistake (e.g., one person being detected as far-away in Fig. 15h, and the banner being bent in Fig. 15k) makes the model look bad, and hence be evaluated as "incorrect."

Finally, in a large-scale web experiment, we allowed users to upload their photos on the internet, and view a 3-d flythrough produced from their image by our algorithm. About 23846 unique users uploaded (and rated) about 26228 images.[8] Users rated 48.1% of the models as good. If we consider the images of scenes only, i.e., exclude images such as company logos, cartoon characters, closeups of objects, etc., then this percentage was 57.3%. We have made the following website available for downloading datasets/code, and for converting an image to a 3-d model/flythrough:

http://make3d.stanford.edu

Our algorithm, trained on images taken in daylight around the city of Palo Alto, was able to predict qualitatively correct 3-d models for a large variety of environments—for example, ones that have hills or lakes, ones taken at night, and even paintings. (See Fig. 15 and the website.) We believe, based on our experiments with varying the number of training examples (not reported here), that having a larger and more diverse set of training images would improve the algorithm significantly.

VIII. LARGER 3-D MODELS FROM MULTIPLE IMAGES

A 3-d model built from a single image will almost invariably be an incomplete model of the scene, because many portions of the scene will be missing or occluded. In this section, we will use both the monocular cues and multi-view triangulation cues to create better and larger 3-d models.

Given a sparse set of images of a scene, it is sometimes possible to construct a 3-d model using techniques such as structure from motion (SFM) [5], [32], which start by taking two or more photographs, then find correspondences between the images, and finally use triangulation to obtain 3-d locations of the points. If the images are taken from nearby cameras (i.e., if the baseline distance is small), then these methods often suffer from large triangulation errors for points far-away from the camera.[9] If, conversely, one chooses images taken far apart, then often the change of viewpoint causes the images to become very different, so that finding correspondences becomes difficult, sometimes leading to spurious or missed correspondences. (Worse, the large baseline also means that there may be little overlap between the images, so that few correspondences may even exist.) These difficulties make purely geometric 3-d reconstruction algorithms fail in many cases, specifically when given only a small set of images.

However, when tens of thousands of pictures are available—for example, for frequently-photographed tourist attractions such as national monuments—one can use the information present in *many* views to reliably discard images that have only few correspondence matches. Doing so, one can use only a small subset of the images available (∼15%), and still obtain a "3-d point cloud" for points that were matched using SFM. This approach has been very successfully applied to famous buildings such as the Notre Dame; the computational cost of this algorithm was significant, and required about a week on a cluster of computers [33].

The reason that many geometric "triangulation-based" methods sometimes fail (especially when only a few images of a scene are available) is that they do not make use of the information present in a single image. Therefore, we will extend our MRF model to seamlessly combine triangulation cues and monocular image cues to build a full photo-realistic 3-d model of the scene. Using monocular cues will also help us build 3-d model of the parts that are visible only in one view.

[7]To compare the algorithms, the person was asked to count the number of errors made by each algorithm. We define an error when a major plane in the image (occupying more than 15% area in the image) is in wrong location with respect to its neighbors, or if the orientation of the plane is more than 30 degrees wrong. For example, if HEH fold the image at incorrect place (see Fig. 14, image 2), then it is counted as an error. Similarly, if we predict top of a building as far and the bottom part of building near, making the building tilted—it would count as an error.

[8]No restrictions were placed on the type of images that users can upload. Users can rate the models as good (thumbs-up) or bad (thumbs-down).

[9]I.e., the depth estimates will tend to be inaccurate for objects at large distances, because even small errors in triangulation will result in large errors in depth.

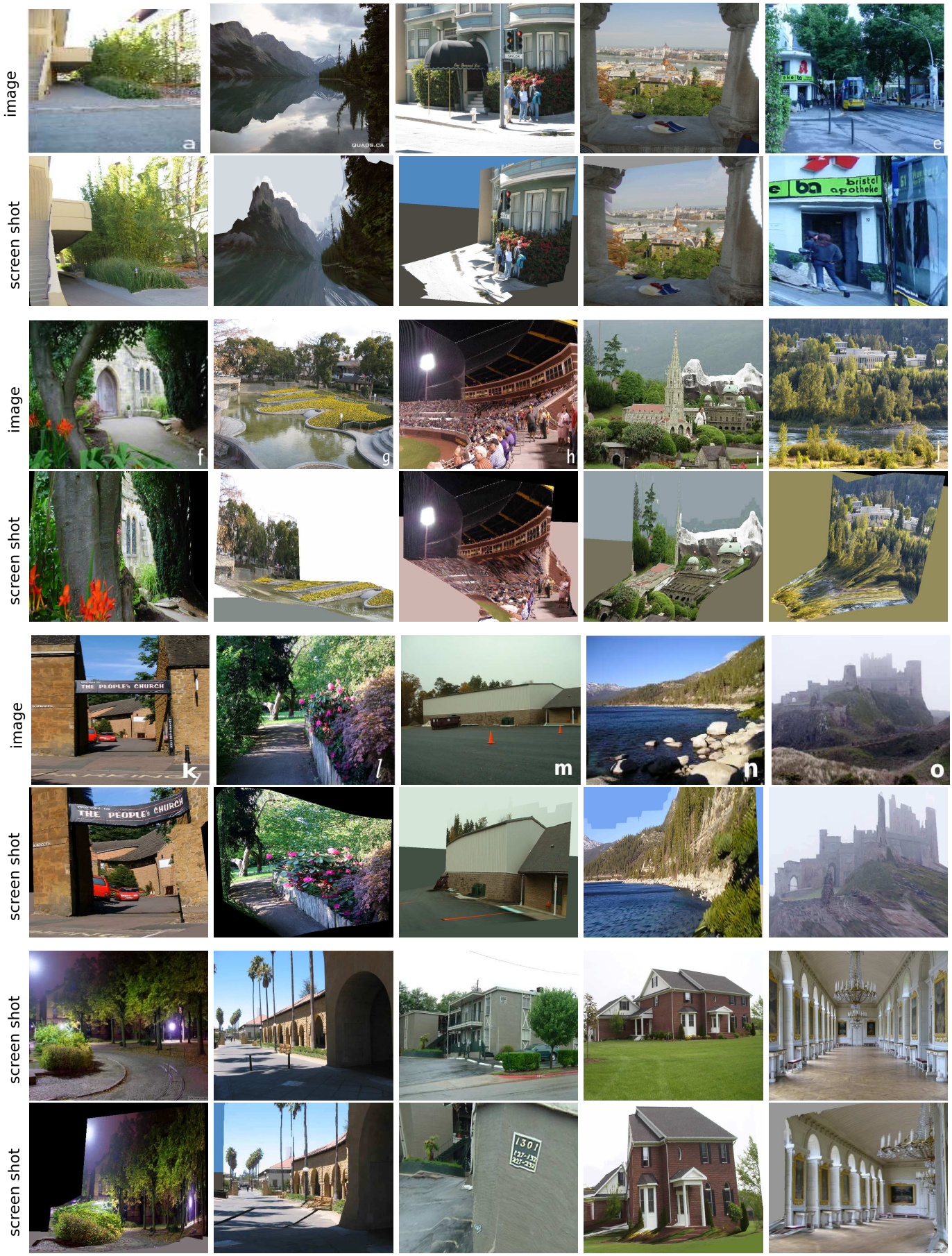


Fig. 15. Typical results from our algorithm. Original image (top), and a screenshot of the 3-d flythrough generated from the image (bottom of the image). The 11 images (a-g,l-t) were evaluated as "correct" and the 4 (h-k) were evaluated as "incorrect."

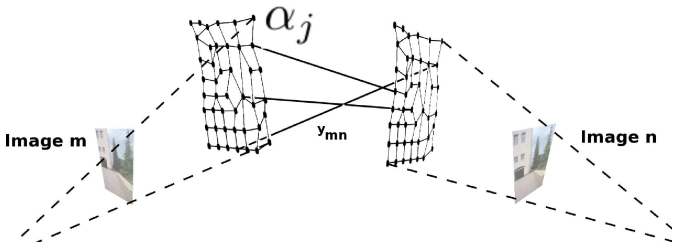


Fig. 16. An illustration of the Markov Random Field (MRF) for inferring 3-d structure. (Only a subset of edges and scales shown.)

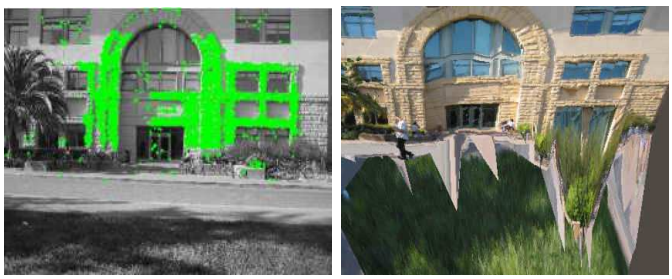


Fig. 17. An image showing a few matches (left), and the resulting 3-d model (right) without estimating the variables y for confidence in the 3-d matching. The noisy 3-d matches reduce the quality of the model. (Note the cones erroneously projecting out from the wall.)

A. Representation

Given two small plane (superpixel) segmentations of two images, there is no guarantee that the two segmentations are "consistent," in the sense of the small planes (on a specific object) in one image having a one-to-one correspondence to the planes in the second image of the same object. Thus, at first blush it appears non-trivial to build a 3-d model using these segmentations, since it is impossible to associate the planes in one image to those in another. We address this problem by using our MRF to reason simultaneously about the position and orientation of every plane in every image. If two planes lie on the same object, then the MRF will (hopefully) infer that they have exactly the same 3-d position. More formally, in our model, the plane parameters α_i^n of each small i^{th} plane in the n^{th} image are represented by a node in our Markov Random Field (MRF). Because our model uses L_1 penalty terms, our algorithm will be able to infer models for which $\alpha_i^n = \alpha_j^m$, which results in the two planes exactly overlapping each other.

B. Probabilistic Model

In addition to the image features/depth, co-planarity, connected structure, and co-linearity properties, we will also consider the depths obtained from triangulation (SFM)—the depth of the point is more likely to be close to the triangulated depth. Similar to the probabilistic model for 3-d model from a single image, most of these cues are noisy indicators of depth; therefore our MRF model will also reason about our "confidence" in each of them, using latent variables y_T (Section VIII-C).

Let $Q^n = [\text{Rotation}, \quad \text{Translation}] \in \mathbb{R}^{3\times 4}$ (technically SE(3)) be the camera pose when image n was taken (w.r.t. a fixed reference, such as the camera pose of the first image), and let d_T be the depths obtained by triangulation (see Section VIII-C). We formulate our MRF as

$$P(\alpha|X,Y,d_T;\theta) \propto \prod_n f_1(\alpha^n|X^n,\nu^n,R^n,Q^n;\theta^n) \prod_n f_2(\alpha^n|y^n,R^n,Q^n) \prod_n f_3(\alpha^n|d_T^n,y_T^n,R^n,Q^n) \quad (7)$$

where, the superscript n is an index over the images, For an image n, α_i^n is the plane parameter of superpixel i in image n. Sometimes, we will drop the superscript for brevity, and write α in place of α^n when it is clear that we are referring to a particular image.

The first term $f_1(\cdot)$ and the second term $f_2(\cdot)$ capture the monocular properties, and are same as in Eq. 1. We use $f_3(\cdot)$ to model the errors in the triangulated depths, and penalize the (fractional) error in the triangulated depths d_{Ti} and $d_i = 1/(R_i^T\alpha_i)$. For K^n points for which the triangulated depths are available, we therefore have

$$f_3(\alpha|d_T,y_T,R,Q) \propto \prod_{i=1}^{K^n} \exp\left(-y_{Ti}\left|d_{Ti}{R_i}^T\alpha_i - 1\right|\right). \quad (8)$$

This term places a "soft" constraint on a point in the plane to have its depth equal to its triangulated depth.

MAP Inference: For MAP inference of the plane parameters, we need to maximize the conditional log-likelihood $\log P(\alpha|X,Y,d_T;\theta)$. All the terms in Eq. 7 are L_1 norm of a linear function of α; therefore MAP inference is efficiently solved using a Linear Program (LP).

C. Triangulation Matches

In this section, we will describe how we obtained the correspondences across images, the triangulated depths d_T and the "confidences" y_T in the $f_3(\cdot)$ term in Section VIII-B.

We start by computing 128 SURF features [34], and then calculate matches based on the Euclidean distances between the features found. Then to compute the camera poses $Q = [\text{Rotation}, \quad \text{Translation}] \in \mathbb{R}^{3\times 4}$ and the depths d_T of the points matched, we use bundle adjustment [35] followed by using monocular approximate depths to remove the scale ambiguity. However, many of these 3-d correspondences are noisy; for example, local structures are often repeated across an image (e.g., Fig. 17, 19 and 21).[10] Therefore, we also model the "confidence" y_{Ti} in the i^{th} match by using logistic regression to estimate the probability $P(y_{Ti} = 1)$ of the match being correct. For this, we use neighboring 3-d matches as a cue. For example, a group of spatially consistent 3-d matches is more likely to be correct than

[10]Increasingly many cameras and camera-phones come equipped with GPS, and sometimes also accelerometers (which measure gravity/orientation). Many photo-sharing sites also offer geo-tagging (where a user can specify the longitude and latitude at which an image was taken). Therefore, we could also use such geo-tags (together with a rough user-specified estimate of camera orientation), together with monocular cues, to improve the performance of correspondence algorithms. In detail, we compute the approximate depths of the points using monocular image features as $\hat{d} = x^T\theta$; this requires only computing a dot product and hence is fast. Now, for each point in an image B for which we are trying to find a correspondence in image A, typically we would search in a band around the corresponding epipolar line in image A. However, given an approximate depth estimated from from monocular cues, we can limit the search to a rectangular window that comprises only a subset of this band. (See Fig. 18.) This would reduce the time required for matching, and also improve the accuracy significantly when there are repeated structures in the scene. (See [2] for more details.)

a single isolated 3-d match. We capture this by using a feature vector that counts the number of matches found in the present superpixel and in larger surrounding regions (i.e., at multiple spatial scales), as well as measures the relative quality between the best and second best match.

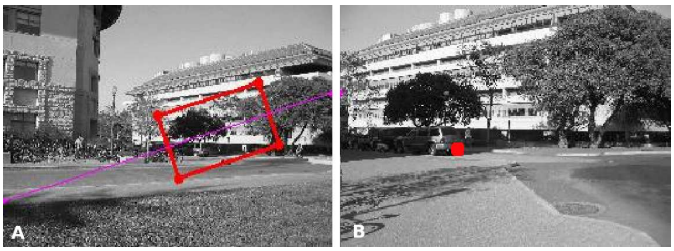


Fig. 18. Approximate monocular depth estimates help to limit the search area for finding correspondences. For a point (shown as a red dot) in image B, the corresponding region to search in image A is now a rectangle (shown in red) instead of a band around its epipolar line (shown in blue) in image A.

D. Phantom Planes

This cue enforces occlusion constraints across multiple cameras. Concretely, each small plane (superpixel) comes from an image taken by a specific camera. Therefore, there must be an unoccluded view between the camera and the 3-d position of that small plane—i.e., the small plane must be visible from the camera location where its picture was taken, and it is not plausible for any other small plane (one from a different image) to have a 3-d position that occludes this view. This cue is important because often the connected structure terms, which informally try to "tie" points in two small planes together, will result in models that are inconsistent with this occlusion constraint, and result in what we call "phantom planes"—i.e., planes that are not visible from the camera that photographed it. We penalize the distance between the offending phantom plane and the plane that occludes its view from the camera by finding additional correspondences. This tends to make the two planes lie in exactly the same location (i.e., have the same plane parameter), which eliminates the phantom/occlusion problem.

E. Experiments

In this experiment, we create a photo-realistic 3-d model of a scene given only a few images (with unknown location/pose), even ones taken from very different viewpoints or with little overlap. Fig. 19, 20, 21 and 22 show snapshots of some 3-d models created by our algorithm. Using monocular cues, our algorithm is able to create full 3-d models even when large portions of the images have no overlap (Fig. 19, 20 and 21). In Fig. 19, monocular predictions (not shown) from a single image gave approximate 3-d models that failed to capture the arch structure in the images. However, using both monocular and triangulation cues, we were able to capture this 3-d arch structure. The models are available at:

http://make3d.stanford.edu/research

IX. Incorporating Object Information

In this section, we will demonstrate how our model can also incorporate other information that might be available, for example, from object recognizers. In prior work, Sudderth et al. [36] showed that knowledge of objects could be used to get crude depth estimates, and Hoiem et al. [11] used knowledge of objects and their location to improve the estimate of the horizon. In addition to estimating the horizon, the knowledge of objects and their location in the scene give strong cues regarding the 3-d structure of the scene. For example, that a person is more likely to be on top of the ground, rather than under it, places certain restrictions on the 3-d models that could be valid for a given image.

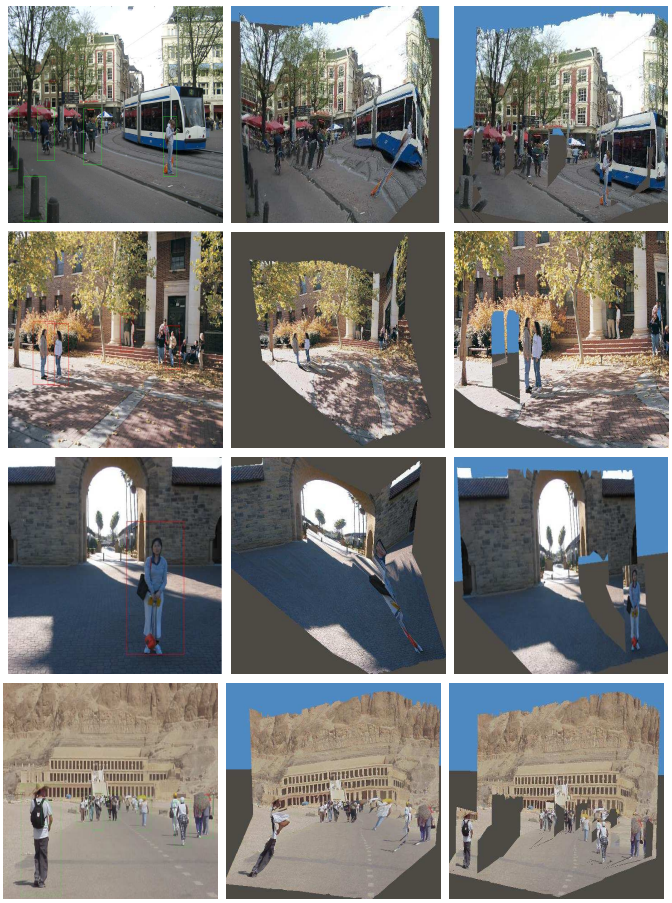
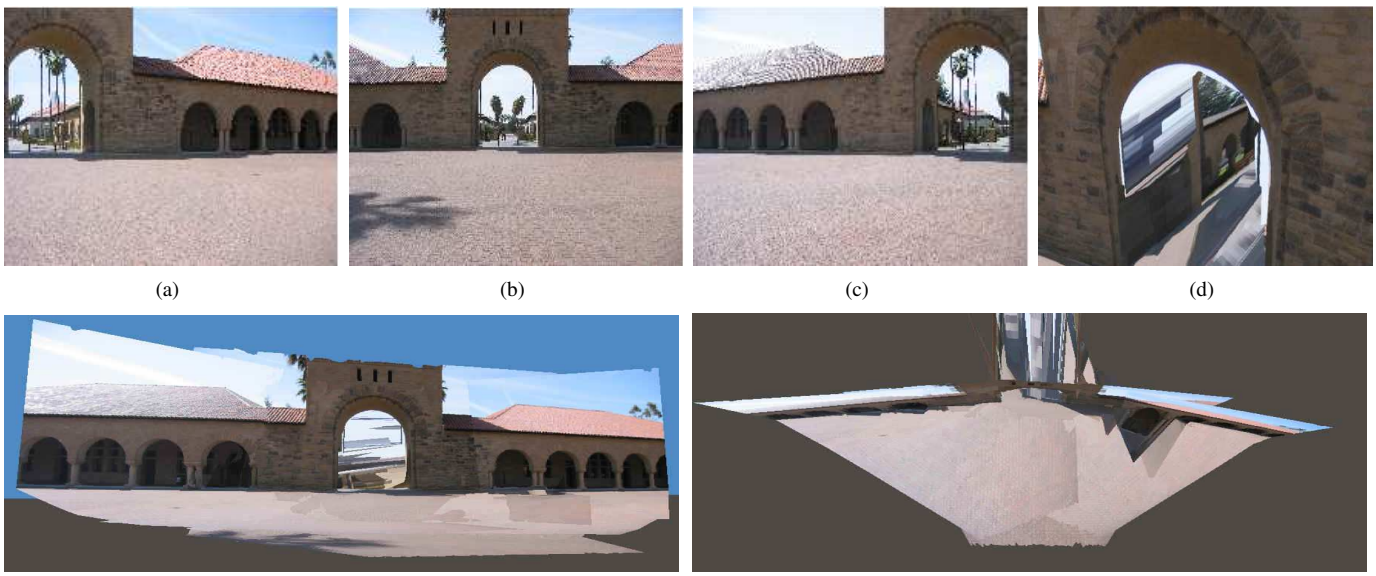


Fig. 23. (Left) Original Images, (Middle) Snapshot of the 3-d model without using object information, (Right) Snapshot of the 3-d model that uses object information.

Here we give some examples of such cues that arise when information about objects is available, and describe how we can encode them in our MRF:

(a) *"Object A is on top of object B"*
This constraint could be encoded by restricting the points $s_i \in \mathbb{R}^3$ on object A to be on top of the points $s_j \in \mathbb{R}^3$ on object B, i.e., $s_i^T \hat{z} \geq s_j^T \hat{z}$ (if $\hat{z}$ denotes the "up" vector). In practice, we actually use a probabilistic version of this constraint. We represent this inequality in plane-parameter space ($s_i = R_i d_i = R_i/(\alpha_i^T R_i)$). To penalize the fractional error $\xi = \left(R_i^T \hat{z} R_j^T \alpha_j - R_j^T \hat{z} R_i \alpha_i\right) \hat{d}$ (the constraint corresponds to $\xi \geq 0$), we choose an MRF potential $h_{s_i,s_j}(.) = \exp\left(-y_{ij}\left(\xi + |\xi|\right)\right)$, where y_{ij} represents the uncertainty in the object recognizer output. Note that for $y_{ij} \to \infty$ (corresponding to certainty in the object recognizer), this becomes a "hard" constraint $R_i^T \hat{z}/(\alpha_i^T R_i) \geq R_j^T \hat{z}/(\alpha_j^T R_j)$.

In fact, we can also encode other similar spatial-relations by choosing the vector $\hat{z}$ appropriately. For example, a constraint *"Object A is in front of Object B"* can be encoded by choosing



(a) (b) (c) (d)

(e) (f)

Fig. 19. (a,b,c) Three original images from different viewpoints; (d,e,f) Snapshots of the 3-d model predicted by our algorithm. (f) shows a top-down view; the top part of the figure shows portions of the ground correctly modeled as lying either within or beyond the arch.

(a) (b) (c) (d)

Fig. 20. (a,b) Two original images with only a little overlap, taken from the same camera location. (c,d) Snapshots from our inferred 3-d model.

(a) (b) (c) (d)

Fig. 21. (a,b) Two original images with many repeated structures; (c,d) Snapshots of the 3-d model predicted by our algorithm.

$\hat{z}$ to be the ray from the camera to the object.

(b) *"Object A is attached to Object B"*

For example, if the ground-plane is known from a recognizer, then many objects would be more likely to be "attached" to the ground plane. We easily encode this by using our connected-structure constraint.

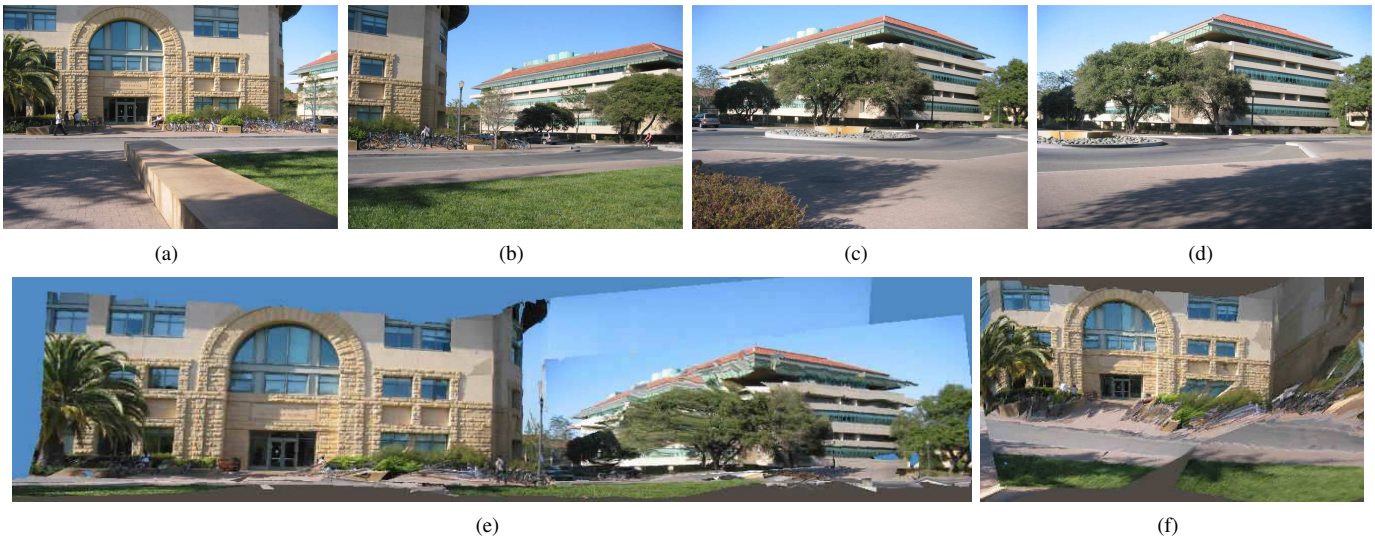
(c) *Known plane orientation*

If orientation of a plane is roughly known, e.g. that a person is more likely to be "vertical", then it can be easily encoded by adding to Eq. 1 a term $f(\alpha_i) = \exp\left(-w_i|\alpha_i^T \hat{z}|\right)$; here, w_i represents the confidence, and $\hat{z}$ represents the up vector.

We implemented a recognizer (based on the features described in Section VI) for ground-plane, and used the Dalal-Triggs Detector [37] to detect pedestrians. For these objects, we encoded the (a), (b) and (c) constraints described above. Fig. 23 shows that using the pedestrian and ground detector improves the accuracy of the 3-d model. Also note that using "soft" constraints in the MRF (Section IX), instead of "hard" constraints, helps in estimating correct 3-d models even if the object recognizer makes a mistake.

X. Conclusions

We presented an algorithm for inferring detailed 3-d structure from a single still image. Compared to previous approaches, our algorithm creates detailed 3-d models which are both quantitatively more accurate and visually more pleasing. Our approach



(a) (b) (c) (d)

(e) (f)

Fig. 22. (a,b,c,d) Four original images; (e,f) Two snapshots shown from a larger 3-d model created using our algorithm.

begins by over-segmenting the image into many small homogeneous regions called "superpixels" and uses an MRF to infer the 3-d position and orientation of each. Other than assuming that the environment is made of a number of small planes, we do not make any explicit assumptions about the structure of the scene, such as the assumption by Delage et al. [8] and Hoiem et al. [9] that the scene comprises vertical surfaces standing on a horizontal floor. This allows our model to generalize well, even to scenes with significant non-vertical structure. Our algorithm gave significantly better results than prior art; both in terms of quantitative accuracies in predicting depth and in terms of fraction of qualitatively correct models. Finally, we extended these ideas to building 3-d models using a sparse set of images, and showed how to incorporate object recognition information into our method.

The problem of depth perception is fundamental to computer vision, one that has enjoyed the attention of many researchers and seen significant progress in the last few decades. However, the vast majority of this work, such as stereopsis, has used multiple image geometric cues to infer depth. In contrast, single-image cues offer a largely orthogonal source of information, one that has heretofore been relatively underexploited. Given that depth and shape perception appears to be an important building block for many other applications, such as object recognition [11], [38], grasping [39], navigation [7], image compositing [40], and video retrieval [41], we believe that monocular depth perception has the potential to improve all of these applications, particularly in settings where only a single image of a scene is available.

ACKNOWLEDGMENTS

We thank Rajiv Agarwal and Jamie Schulte for help in collecting data. We also thank Jeff Michels, Olga Russakovsky and Sebastian Thrun for helpful discussions. This work was supported by the National Science Foundation under award CNS-0551737, by the Office of Naval Research under MURI N000140710747, and by Pixblitz Studios.

APPENDIX

A.1 Parameter Learning

Since exact parameter learning based on conditional likelihood for the Laplacian models is intractable, we use Multi-Conditional Learning (MCL) [30], [31] to divide the learning problem into smaller learning problems for each of the individual densities. MCL is a framework for optimizing graphical models based on a product of several marginal conditional likelihoods each relying on common sets of parameters from an underlying joint model and predicting different subsets of variables conditioned on other subsets.

In detail, we will first focus on learning θ_r given the ground-truth depths d (obtained from our 3-d laser scanner, see Section VII-A) and the value of y_{ij} and ν_{i,s_i}. For this, we maximize the conditional pseudo log-likelihood $\log P(\alpha|X,\nu,y,R;\theta_r)$ as

$$\theta_r^* = \arg\max_{\theta_r} \sum_i \log f_1(\alpha_i|X_i,\nu_i,R_i;\theta_r) + \sum_{i,j} \log f_2(\alpha_i,\alpha_j|y_{ij},R_i,R_j)$$

Now, from Eq. 1 note that $f_2(\cdot)$ does not depend on θ_r; therefore the learning problem simplifies to minimizing the L_1 norm, i.e., $\theta_r^* = \arg\min_{\theta_r} \sum_i \sum_{s_i=1}^{S_i} \nu_{i,s_i} \left| \frac{1}{d_{i,s_i}} (x_{i,s_i}^T \theta_r) - 1 \right|$.

In the next step, we learn the parameters ϕ of the logistic regression model for estimating ν in footnote 5. Parameters of a logistic regression model can be estimated by maximizing the conditional log-likelihood. [42] Now, the parameters ψ of the logistic regression model $P(y_{ij}|\epsilon_{ij};\psi)$ for occlusion boundaries and folds are similarly estimated using the hand-labeled ground-truth ground-truth training data by maximizing its conditional log-likelihood.

A.2 MAP Inference

When given a new test-set image, we find the MAP estimate of the plane parameters α by maximizing the conditional log-likelihood $\log P(\alpha|X,\nu,Y,R;\theta_r)$. Note that we solve for α as a continuous variable optimization problem, which is unlike many other techniques where discrete optimization is more popular, e.g., [4]. From Eq. 1, we have

$$\alpha^* = \arg\max_{\alpha} \log P(\alpha|X,\nu,y,R;\theta_r)$$
$$= \arg\max_{\alpha} \log \frac{1}{Z} \prod_i f_1(\alpha_i|X_i,\nu_i,R_i;\theta_r) \prod_{i,j} f_2(\alpha_i,\alpha_j|y_{ij},R_i,R_j)$$

Note that the partition function Z does not depend on α. Therefore, from Eq. 2, 4 and 5 and for $\hat{d} = x^T\theta_r$, we have

$$= \arg\min_\alpha \sum_{i=1}^{K} \Big(\sum_{s_i=1}^{S_i} \nu_{i,s_i} \left| (R_{i,s_i}^T \alpha_i)\hat{d}_{i,s_i} - 1 \right| + \sum_{j \in N(i)} \sum_{s_i, s_j \in B_{ij}} y_{ij} \left| (R_{i,s_i}^T \alpha_i - R_{j,s_j}^T \alpha_j)\hat{d}_{s_i,s_j} \right| + \sum_{j \in N(i)} \sum_{s_j \in C_j} y_{ij} \left| (R_{j,s_j}^T \alpha_i - R_{j,s_j}^T \alpha_j)\hat{d}_{s_j} \right| \Big)$$

where K is the number of superpixels in each image; $N(i)$ is the set of "neighboring" superpixels—one whose relations are modeled—of superpixel i; B_{ij} is the set of pair of points on the boundary of superpixel i and j that model connectivity; C_j is the center point of superpixel j that model co-linearity and co-planarity; and $\hat{d}_{s_i,s_j} = \sqrt{\hat{d}_{s_i}\hat{d}_{s_j}}$. Note that each of terms is a L_1 norm of a linear function of α; therefore, this is a L_1 norm minimization problem, [43, chap. 6.1.1] and can be compactly written as

$$\arg\min_x \|Ax - b\|_1 + \|Bx\|_1 + \|Cx\|_1$$

where $x \in \mathbb{R}^{3K\times 1}$ is a column vector formed by rearranging the three x-y-z components of $\alpha_i \in \mathbb{R}^3$ as $x_{3i-2} = \alpha_{ix}$, $x_{3i-1} = \alpha_{iy}$ and $x_{3i} = \alpha_{iz}$; A is a block diagonal matrix such that $A\Big[(\sum_{l=1}^{i-1} S_l) + s_i, \quad (3i-2):3i\Big] = R_{i,s_i}^T \hat{d}_{i,s_i} \nu_{i,s_i}$and $b_1 \in \mathbb{R}^{3K\times 1}$ is a column vector formed from ν_{i,s_i}. B and C are all block diagonal matrices composed of rays R, $\hat{d}$ and y; they represent the cross terms modeling the connected structure, co-planarity and co-linearity properties.

In general, finding the global optimum in a loopy MRF is difficult. However in our case, the minimization problem is an Linear Program (LP), and therefore can be solved exactly using any linear programming solver. (In fact, any greedy method including a loopy belief propagation would reach the global minima.) For fast inference, we implemented our own optimization method, one that captures the sparsity pattern in our problem, and by approximating the L_1 norm with a smooth function:

$$\|x\|_1 \cong \Upsilon_\beta(x) = \tfrac{1}{\beta}\big[\log(1+\exp(-\beta x)) + \log(1+\exp(\beta x))\big]$$

Note that $\|x\|_1 = \lim_{\beta\to\infty} \|x\|_\beta$, and the approximation can be made arbitrarily close by increasing β during steps of the optimization. Then we wrote a customized Newton method based solver that computes the Hessian efficiently by utilizing the sparsity. [43]

B. Point-wise MRF

For comparison, we present another MRF, in which we use points in the image as basic unit, instead of the superpixels; and infer only their 3-d location. The nodes in this MRF are a dense grid of points in the image, where the value of each node represents its depth. The depths in this model are in log scale to emphasize fractional (relative) errors in depth. Unlike SCN's fixed rectangular grid, we use a deformable grid, aligned with structures in the image such as lines and corners to improve performance. Further, in addition to using the connected structure property (as in SCN), our model also captures co-planarity and co-linearity. Finally, we use logistic response to identify occlusion and folds, whereas SCN learned the variances.

We formulate our MRF as

$$P(d|X,Y,R;\theta) = \frac{1}{Z}\prod_i f_1(d_i|x_i,y_i;\theta) \prod_{i,j\in N} f_2(d_i,d_j|y_{ij},R_i,R_j) \prod_{i,j,k\in N} f_3(d_i,d_j,d_k|y_{ijk},R_i,R_j,R_k)$$

where, $d_i \in \mathbb{R}$ is the depth (in log scale) at a point i. x_i are the image features at point i. The first term $f_1(.)$ models the relation between depths and the image features as $f_1(d_i|x_i,y_i;\theta) = \exp\Big(-y_i|d_i - x_i^T\theta_{r(i)}|\Big)$. The second term $f_2(\cdot)$ models connected structure by penalizing differences in the depths of neighboring points as $f_2(d_i,d_j|y_{ij},R_i,R_j) = \exp\big(-y_{ij}||(R_i d_i - R_j d_j)||_1\big)$. The third term $f_3(\cdot)$ depends on three points i,j and k, and models co-planarity and co-linearity. For modeling co-linearity, we choose three points q_i, q_j, and q_k lying on a straight line, and penalize the curvature of the line:

$$f_3(d_i,d_j,d_k|y_{ijk},R_i,R_j,R_k) = \exp\big(-y_{ijk}||R_j d_j - 2R_i d_i + R_k d_k||_1\big)$$

where $y_{ijk} = (y_{ij}+y_{jk}+y_{ik})/3$. Here, the "confidence" term y_{ij} is similar to the one described for Plane Parameter MRF; except in cases when the points do not cross an edgel (because nodes in this MRF are a dense grid), when we set y_{ij} to zero.

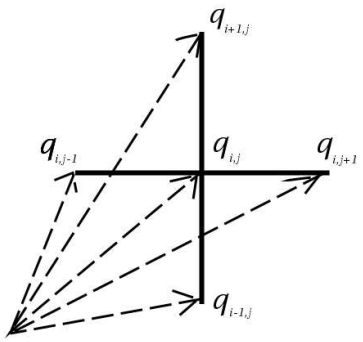


Fig. 24. Enforcing local co-planarity by using five points.

We also enforce co-planarity by penalizing two terms $h(d_{i,j-1}, d_{i,j}, d_{i,j+1}, y_{i,(j-1):(j+1)}, R_{i,j-1}, R_{i,j}, R_{i,j+1})$, and $h(d_{i-1,j}, d_{i,j}, d_{i+1,j}, y_{(i-1):(i+1),j}, R_{i-1,j}, R_{i,j}, R_{i+1,j})$. Each term enforces the two sets of three points to lie on the same line in 3-d; therefore in effect enforcing five points $q_{i-1,j}$, $q_{i,j}$, $q_{i+1,j}$, $q_{i,j-1}$, and $q_{i,j+1}$ lie on the same plane in 3-d. (See Fig. 24.)

Parameter learning is done similar to the one in Plane Parameter MRF. MAP inference of depths, i.e. maximizing $\log P(d|X,Y,R;\theta)$ is performed by solving a linear program (LP). However, the size of LP in this MRF is larger than in the Plane Parameter MRF.

REFERENCES

[1] A. Saxena, M. Sun, and A. Y. Ng, "Learning 3-d scene structure from a single still image," in *ICCV workshop on 3D Representation for Recognition (3dRR-07)*, 2007.

[2] ——, "3-d reconstruction from sparse views using monocular vision," in *ICCV workshop on Virtual Representations and Modeling of Large-scale environments (VRML)*, 2007.

[3] ——, "Make3d: Depth perception from a single still image," in *AAAI*, 2008.

[4] D. Scharstein and R. Szeliski, "A taxonomy and evaluation of dense two-frame stereo correspondence algorithms," *International Journal of Computer Vision (IJCV)*, vol. 47, 2002.

[5] D. A. Forsyth and J. Ponce, *Computer Vision : A Modern Approach*. Prentice Hall, 2003.

[6] A. Saxena, S. H. Chung, and A. Y. Ng, "Learning depth from single monocular images," in *Neural Information Processing Systems (NIPS) 18*, 2005.

[7] J. Michels, A. Saxena, and A. Y. Ng, "High speed obstacle avoidance using monocular vision and reinforcement learning," in *International Conference on Machine Learning (ICML)*, 2005.

[8] E. Delage, H. Lee, and A. Y. Ng, "A dynamic bayesian network model for autonomous 3d reconstruction from a single indoor image," in *Computer Vision and Pattern Recognition (CVPR)*, 2006.

[9] D. Hoiem, A. Efros, and M. Herbert, "Geometric context from a single image," in *International Conference on Computer Vision (ICCV)*, 2005.

[10] P. Felzenszwalb and D. Huttenlocher, "Efficient graph-based image segmentation," *IJCV*, vol. 59, 2004.

[11] D. Hoiem, A. Efros, and M. Hebert, "Putting objects in perspective," in *Computer Vision and Pattern Recognition (CVPR)*, 2006.

[12] R. Zhang, P. Tsai, J. Cryer, and M. Shah, "Shape from shading: A survey," *IEEE Trans Pattern Analysis & Machine Intelligence (IEEE-PAMI)*, vol. 21, pp. 690–706, 1999.

[13] A. Maki, M. Watanabe, and C. Wiles, "Geotensity: Combining motion and lighting for 3d surface reconstruction," *International Journal of Computer Vision (IJCV)*, vol. 48, no. 2, pp. 75–90, 2002.

[14] J. Malik and R. Rosenholtz, "Computing local surface orientation and shape from texture for curved surfaces," *International Journal of Computer Vision (IJCV)*, vol. 23, no. 2, pp. 149–168, 1997.

[15] T. Lindeberg and J. Garding, "Shape from texture from a multi-scale perspective," 1993.

[16] T. Nagai, T. Naruse, M. Ikehara, and A. Kurematsu, "Hmm-based surface reconstruction from single images," in *Proc IEEE International Conf Image Processing (ICIP)*, vol. 2, 2002.

[17] T. Hassner and R. Basri, "Example based 3d reconstruction from single 2d images," in *CVPR workshop on Beyond Patches*, 2006.

[18] F. Han and S.-C. Zhu, "Bayesian reconstruction of 3d shapes and scenes from a single image," in *ICCV Workshop Higher-Level Knowledge in 3D Modeling Motion Analysis*, 2003.

[19] A. Criminisi, I. Reid, and A. Zisserman, "Single view metrology," *International Journal of Computer Vision (IJCV)*, vol. 40, pp. 123–148, 2000.

[20] A. Torralba and A. Oliva, "Depth estimation from image structure," *IEEE Trans Pattern Analysis and Machine Intelligence (PAMI)*, vol. 24, no. 9, pp. 1–13, 2002.

[21] A. Saxena, S. H. Chung, and A. Y. Ng, "3-D depth reconstruction from a single still image," *International Journal of Computer Vision (IJCV)*, 2007.

[22] A. Saxena, J. Schulte, and A. Y. Ng, "Depth estimation using monocular and stereo cues," in *International Joint Conference on Artificial Intelligence (IJCAI)*, 2007.

[23] E. Delage, H. Lee, and A. Ng, "Automatic single-image 3d reconstructions of indoor manhattan world scenes," in *International Symposium on Robotics Research (ISRR)*, 2005.

[24] K. Murphy, A. Torralba, and W. Freeman, "Using the forest to see the trees: A graphical model relating features, objects, and scenes," in *Neural Information Processing Systems (NIPS) 16*, 2003.

[25] Y. Lu, J. Zhang, Q. Wu, and Z. Li, "A survey of motion-parallax-based 3-d reconstruction algorithms," *IEEE Tran on Systems, Man and Cybernetics, Part C*, vol. 34, pp. 532–548, 2004.

[26] J. Loomis, "Looking down is looking up," *Nature News and Views*, vol. 414, pp. 155–156, 2001.

[27] B. A. Wandell, *Foundations of Vision*. Sunderland, MA: Sinauer Associates, 1995.

[28] D. R. Martin, C. C. Fowlkes, and J. Malik, "Learning to detect natural image boundaries using local brightness, color and texture cues," *IEEE Trans Pattern Analysis and Machine Intelligence*, vol. 26, 2004.

[29] R. Koch, M. Pollefeys, and L. V. Gool, "Multi viewpoint stereo from uncalibrated video sequences," in *European Conference on Computer Vision (ECCV)*, 1998.

[30] M. K. Chris Paul, Xuerui Wang and A. McCallum, "Multi-conditional learning for joint probability models with latent variables," in *NIPS Workshop Advances Structured Learning Text and Speech Processing*, 2006.

[31] A. McCallum, C. Pal, G. Druck, and X. Wang, "Multi-conditional learning: generative/discriminative training for clustering and classification," in *AAAI*, 2006.

[32] M. Pollefeys, "Visual modeling with a hand-held camera," *International Journal of Computer Vision (IJCV)*, vol. 59, 2004.

[33] N. Snavely, S. M. Seitz, and R. Szeliski, "Photo tourism: Exploring photo collections in 3d," *ACM SIGGRAPH*, vol. 25, no. 3, pp. 835–846, 2006.

[34] H. Bay, T. Tuytelaars, and L. V. Gool, "Surf: Speeded up robust features," in *European Conference on Computer Vision (ECCV)*, 2006.

[35] M. Lourakis and A. Argyros, "A generic sparse bundle adjustment c/c++ package based on the levenberg-marquardt algorithm," Foundation for Research and Technology - Hellas, Tech. Rep., 2006.

[36] E. Sudderth, A. Torralba, W. T. Freeman, and A. S. Willsky, "Depth from familiar objects: A hierarchical model for 3d scenes," in *Computer Vision and Pattern Recognition (CVPR)*, 2006.

[37] N. Dalai and B. Triggs, "Histogram of oriented gradients for human detection," in *Computer Vision and Pattern Recognition (CVPR)*, 2005.

[38] A. Torralba, "Contextual priming for object detection," *International Journal of Computer Vision*, vol. 53, no. 2, pp. 161–191, 2003.

[39] A. Saxena, J. Driemeyer, J. Kearns, and A. Ng, "Robotic grasping of novel objects," in *Neural Information Processing Systems (NIPS) 19*, 2006.

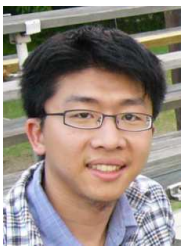
[40] M. Kawakita, K. Iizuka, T. Aida, T. Kurita, and H. Kikuchi, "Real-time three-dimensional video image composition by depth information," in *IEICE Electronics Express*, 2004.

[41] R. Ewerth, M. Schwalb, and B. Freisleben, "Using depth features to retrieve monocular video shots," in *ACM International Conference on Image and Video Retrieval*, 2007.

[42] C. M. Bishop, *Pattern Recognition and Machine Learning*. Springer, 2006.

[43] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge University Press, 2004.

Ashutosh Saxena received his B. Tech. degree in Electrical Engineering from Indian Institute of Technology (IIT) Kanpur, India in 2004. He is currently a PhD student in Electrical Engineering at Stanford University. His research interests include machine learning, robotics perception, and computer vision. He has won best paper awards in 3DRR and IEEE ACE. He was also a recipient of National Talent Scholar award in India.



Min Sun graduated from National Chiao Tung University in Taiwan in 2003 with an Electrical Engineering degree. He received the MS degree from Stanford University in Electrical Engineering department in 2007. He is currently a PhD student in the Vision Lab at the Princeton University. His research interests include object recognition, image understanding, and machine learning. He was also a recipient of W. Michael Blumenthal Family Fund Fellowship.

Andrew Y. Ng received his B.Sc. from Carnegie Mellon University, his M.Sc. from the Massachusetts Institute of Technology, and his Ph.D. from the University of California, Berkeley. He is an Assistant Professor of Computer Science at Stanford University, and his research interests include machine learning, robotic perception and control, and broad-competence AI. His group has won best paper/best student paper awards at ACL, CEAS and 3DRR. He is also a recipient of the Alfred P. Sloan Fellowship.