Noname manuscript No.
(will be inserted by the editor)

Putting the User in the Loop for Image-Based Modeling

Adarsh Kowdle · Yao-Jen Chang · Andrew Gallagher · Dhruv Batra · Tsuhan Chen

Received: date / Accepted: date

Abstract We refer to the task of recovering the 3D structure of an object or a scene using 2D images as image-based modeling. In this paper, we formulate the task of recovering the 3D structure as a discrete optimization problem solved via energy minimization. In this standard framework of a Markov Random Field (MRF) defined over the image we present algorithms that allow the user to intuitively interact with the algorithm. We introduce an algorithm where the user guides the process of image-based modeling to find and model the object of interest by manually interacting with the nodes of the graph. We develop end user applications using this algorithm that allow object of interest 3D modeling on a mobile device and 3D printing of the object of interest. We also propose an alternate active learning algorithm that guides the user input. An initial attempt is made at reconstructing the scene without supervision. Given the reconstruction, an active learning algorithm uses intuitive cues to quantify the uncertainty of the algorithm and suggest regions, querying the user to provide support for the uncertain regions via simple scribbles. These constraints are used to update the unary and the pairwise energies that, when solved, lead to better reconstructions. We show through machine experiments and a user study that the proposed approach intelligently queries the users for constraints, and users achieve better reconstructions of the scene faster, especially for scenes with textureless surfaces lacking strong textural or structural cues that algorithms typically require.

Keywords Image-based modeling · Interactive 3D reconstruction · Active-learning · Energy minimization

A. Kowdle
Cornell University
E-mail: apk64@cornell.edu

Y. J. Chang
Siemens Corporate Research
E-mail: yao-jen.chang@siemens.com

A. Gallagher
Cornell University
E-mail: acg226@cornell.edu

D. Batra
Virginia Tech
E-mail: dbatra@vt.edu

T. Chen
Cornell University
E-mail: tsuhan@ece.cornell.edu

1 Introduction

Image-based modeling, the recovery of 3D structure of a scene using 2D images, is an active research topic in the computer vision community. There has been significant success with automatic algorithms (Snavely et al, 2006; Furukawa and Ponce, 2009; Pollefeys et al, 2008, 2004; Sinha et al, 2009; Micusík and Kosecká, 2010; Furukawa et al, 2010; Goesele et al, 2007; Gallup et al, 2010). However, when only a few images are available, these automatic algorithms fail to produce a dense reconstruction, leaving holes in case of scene irregularities such as textureless and specular surfaces. While planar approximations to the scene (Sinha et al, 2009; Furukawa et al, 2009; Micusík and Kosecká, 2010; Gallup et al, 2010) help obtain more visually pleasing reconstructions, the cues in a number of scenes are not sufficient to hypothesize a good model. In particular, textureless and specular surfaces and a lack of geometric cues such as lines hinders their performance.

On the other hand, when humans look at a scene they much better discern the geometric structure behind the photons. This forms the basis for interactive algorithms. We illustrate the idea of putting the user in the loop using Fig. 1. The computational engine is the workhorse that uses the constraints provided by the user (*i.e.*, oracle) and recovers

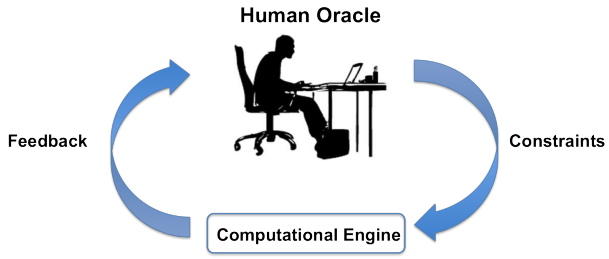


Fig. 1: Overview: Basic ingredients to put the user in the loop for image-based modeling.

the 3D structure of the scene (or object of interest). The computational engine provides feedback in the form of the current solution to the problem or explicit feedback to query the user for additional constraints, placing the user in a closed loop with the computational engine.

A number of works formulate the task of image-based modeling as a discrete labeling problem. For example, shape from silhouette algorithms formulate a binary labeling problem to separate the object from the background (Baumgart, 1974; Lee et al, 2007; Campbell et al, 2007), a number of stereo matching algorithms treat the pixel disparities as discrete labels and formulate a labeling problem to recover the depth of the scene (Scharstein and Szeliski, 2002), piecewise planar stereo works discretize the scene into a finite number of 3D planes and treat these planes as label set (Sinha et al, 2009; Micusík and Kosecká, 2010; Furukawa et al, 2009; Gallup et al, 2010). Motivated by these works, we consider the task of putting the user into the loop. A common framework of all these works is the construction of a Markov Random Field (MRF) over the image illustrated in Fig. 2.

We propose algorithms that use superpixels extracted from the image as the labeling sites[1] *i.e.*, nodes in the MRF, with edges to all adjacent superpixels. User-provided input is incorporated into the node and edge terms of the model as constraints. First, we propose an algorithm where the user provides constraints on the nodes of the graph (Section 4). In this algorithm, the user initializes the loop by providing annotations on the nodes. These node constraints allow the user to define the label space for the problem and modulate the unary term in the energy function. In the second algorithm, an automatic piecewise planar reconstruction algorithm first tries to reconstruct the scene initiating the loop. Given the reconstruction, we propose an active-learning algorithm that uses intuitive cues to quantify the uncertainty of the algorithm. The algorithm then queries the user to provide support for the uncertain regions via edge constraints on the pairwise term that lead to better reconstructions (Section 5). We will discuss the details of the algorithm and applications in the following sections.

[1] Superpixels are used to help reduce computational complexity

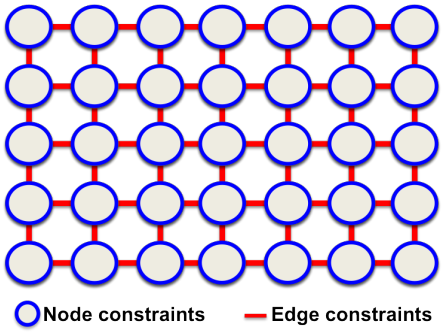


Fig. 2: Discrete labeling problem: The user constraints are provided over the nodes and edges of the graph to help modulate the unary term and the pairwise term, respectively. Note that the grid graph here is only for illustration. We use an irregular graph over superpixels in our work.

Contributions. Our primary contributions are:

- We propose a framework for putting the user in the loop for image-based modeling, formulating it as a discrete labeling problem. We develop two formulations, one where the user guides the algorithm via interactions, and second a novel active-learning formulation where the user is guided by the algorithm.
- We show that we can leverage the user input via very simple interactions in the form of scribbles, which are intuitive for any user to follow.
- We demonstrate through user studies and machine experiments that our proposed algorithm achieves impressive 3D reconstructions and show that the active-learning successfully guidea the user towards improved reconstructions.

Organization. The rest of this paper is organized as follows: Section 2 discusses related work; Section 3 describes the preprocessing performed on the images in the proposed algorithms; Section 4 presents our approach where the user provides node constraints to initiate the loop and reconstruct the object of interest; Section 5 describes our approach of active-learning where the automatic computational engine initiates the loop and guides the user towards where it needs help; Section 6 discusses some applications including performing object of interest 3D modeling on a mobile device; Finally, Section 7 concludes the paper with discussions.

Preliminary versions of the object of interest 3D modeling, interactive piecewise planar 3D reconstruction and active learning for piecewise planar 3D reconstruction appeared as papers (Kowdle et al, 2010), (Kowdle et al, 2011a) and (Kowdle et al, 2011b), respectively. This article brings the above works together in the bigger picture of putting the user in the loop for image-based modeling in a unified discrete labeling framework. Additional results that demonstrate the impact of the active learning approach is shown in Section

5.4.4. In addition, in Section 6 we describe some applications including performing object of interest 3D modeling on a mobile device and 3D printing of the object of interest, using our proposed algorithm. The datasets used in each of our works are publicly available.

2 Related Work

Automatic algorithms. 3D reconstruction from multiple images is an active research topic in the computer vision community. While some 3D reconstruction works (Pollefeys et al, 2008, 2004) are geared towards video, some (Snavely et al, 2006; Goesele et al, 2007) are geared towards unordered photo collections on the internet. Most require a large photo collection. When the number of input images is small, the automatic algorithms fail to produce a dense reconstruction. A survey of multiview stereo methods has been provided by Seitz et al (2006). With a small set of images the reconstruction is incomplete, leaving holes on textureless and specular surfaces. Planar approximations to the scene (Sinha et al, 2009; Furukawa et al, 2009; Micusík and Kosecká, 2010; Gallup et al, 2010; Kowdle et al, 2012b) help obtain more visually pleasing reconstructions. However, these algorithms use image features such as strong edges and lines, which may be absent in textureless surfaces, motivating interactive algorithms.

Interactive algorithms: user driven. A typical approach to obtain the 3D model of a non-planar object is to capture images of the object in a controlled environment like a multi-camera studio with mono-color screen where background subtraction is a well structured problem, and use a shape-from-silhouette algorithm (Szeliski, 1993; Fang et al, 2003; Chen et al, 2008; Forbes et al, 2006) to render the 3D model. Although these techniques have produced promising results in these constrained settings, this is a tedious process, and in some cases not an option (for example, immovable objects like a statue, historically or culturally significant artifacts). However, a more realistic approach is to capture images of the object in its natural environment and directly estimation the 3D structure from these natural images. The images captured in this case would typically have cluttered backgrounds, which is known to be problematic for background subtraction algorithms. There have been many interactive 3D reconstruction algorithms that uses a piecewise planar representation of the scene (Debevec et al, 1996; Criminisi et al, 1999; Sturm and Maybank, 1999; Bartoli, 2007; Hengel et al, 2007; Sinha et al, 2008; Sketchup, 2000). The user interactions required range from providing feature correspondence, to providing plane boundaries and line models of the scene. Debevec et al (1996) proposed an algorithm to reconstruct man-made architectures by marking the edges in the structure and by exploiting symmetry in man-made structures. Hengel et al (2007) and Sinha et al (2008) require the user to provide a detailed line model of the object or mark all the 2D plane polygons in the scene, respectively; and reconstruct the scene by incorporating geometric information from structure-from-motion. Srivastava et al (2009) used scribbles as input to help improve the 3D reconstruction obtained from a single image. In the first algorithm proposed in this paper, we leverage the user input to provide node constraints in the MRF formulation. We propose interactive algorithms driven by the user via simple scribbles that are used to reconstruct non-planar objects, planar scenes, and even render non-planar objects as part of a planar scene.

Interactive algorithms: active-learning. Active learning is a well established subfield of machine learning, which has been shown to benefit a number of computer vision applications such as object categorization (Kapoor et al, 2007), image retrieval (Gosselin and Cord, 2008; Zhou and Huang, 2003), video classification (Yan et al, 2003), dataset annotation (Collins et al, 2008), and interactive co-segmentation (Batra et al, 2011); maximizing the knowledge gain while valuing the user effort (Vijayanarasimhan et al, 2010). However, such an algorithm has not been proposed for image based modeling. Batra et al (2011) proposed an approach for interactive co-segmentation where, starting from the user interactions (scribbles) to identify the object of interest, the algorithm exploits a number of cues using the scribbles, and identifies informative regions to request the user for more interactions. Interactive 3D reconstruction, however, is not a trivial extension of this binary problem to multi-class segmentation. Rich information is already embedded in multiple images of a scene, which an automatic algorithm can fully utilize. However, the automatic algorithms fall short where texture or geometry cues cannot be easily identified from the images. Therefore, we formulate interactive 3D reconstruction as an error-correction and learning problem, where active-learning identifies uncertain regions, requests the user to provide geometric cues, and adapts the algorithm for the specific scene based on the user inputs.

3 Algorithm: Pre-processing

In this paper we work with multiple images of a scene captured from different viewpoints. We perform the following pre-processing steps. We first run structure from motion (SfM) using the algorithm by Snavely et al (2006) on the multiview images to recover the camera projection matrices for all the views, a sparse 3D point cloud and the set of the points visible by each camera. We construct a graph, $G = (V, E)$, over the superpixels[2], with edges between adjacent superpixels

[2] We use mean-shift segmentation (Comaniciu and Meer, 2002) to break an image to about thousand superpixels.

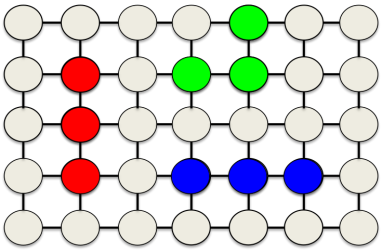


Fig. 3: Node constraints. The colored nodes in the graph represent the node constraints provided by the user via multiple colored scribbles to indicate the different labels. More description about the node constraints and incorporating them into the algorithm is in Section 4.

to formulate our discrete labeling problem. The graph is a planar graph, but not typically a grid graph we use for illustration in Fig. 2. We have developed a Java-based user interface, which we have made publicly available (Tang et al, 2009). This interface is used in all our interactive algorithms.

In the following sections, we will first describe the algorithm that is initiated by the user via node constraints to obtain the final 3D reconstruction via scene co-segmentation. We will then describe the algorithm that is driven by the computational engine and guides the user constraints via an active-learning algorithm.

4 User initiates interactive 3D modeling

In our first approach, the user provides annotation on the nodes of the graph as shown in Fig. 3 and guides the process by initiating the algorithm. The user is first displayed the image collection. The user selects an image and provides scribbles on the image with different colors indicating the label space for the segmentation algorithm. In our work, the labels either indicate the planar surfaces in the scene or just indicate the object of interest and the background. Given these scribbles on the nodes of the graph the problem formulation is similar to the problem of multi-class segmentation (Batra et al, 2011; Vicente et al, 2011). In the following sections, we will first establish notation for the discrete labeling problem and then describe how we incorporate the user interactions to aide image-based modeling.

4.1 Energy minimization

Let the set of images be $\mathcal{X}$. Consider an image-scribble pair $D = \{X, S\}$, where the image X chosen by the user is represented as a collection of n nodes (superpixels) to be labeled, $X = \{X_1, X_2, \ldots, X_n\}$. The user provides a set of scribbles S on the image with multiple labels (suppose that the user defines L labels in the scene), which is represented as the partial set of labels for these nodes $S = \{S_1, S_2, \ldots, S_n\}$ where, $S_i = \{0, 1, \ldots, L-1\}$. Using these labeled nodes (node constraints), we learn an appearance model A described below. We define an energy function over the image as:

$$E(X : A) = \sum_{i \in V} E_i(X_i : A) + \lambda \sum_{(i,j) \in E} E_{ij}\left(X_i, X_j\right), \quad (1)$$

where the first term (unary term) indicates the cost of assigning a node to one of the labels, while the second term (pairwise term) is used for penalizing label disagreement between neighbors. The colon (:) in the equation indicates that the term is dependent on the learnt appearance model.

Unary Term. The unary term is modeled via the node constraints provided by the user. Given the node constraints we learn the appearance model, which consists of a Gaussian Mixture Model for each of the L labels defined by the user i.e, $A = \{\text{GMM}_0, \ldots, \text{GMM}_{L-1}\}$. Specifically, we use color features (Lab space) extracted from superpixels on the labeled nodes and fit GMMs for the corresponding classes. We use MDL to estimate the right number of components to use to describe the data (allowing a maximum of 10 Gaussian components). The unary term for all nodes are then defined as the negative log-likelihood of the features given the class model. We set the unary term of the superpixels labeled by the user to $-\infty$ (a large negative value) as hard constraints in the energy minimization.

Pairwise Term. We use the commonly used contrast sensitive Potts model to model the pairwise term,

$$E_{ij}(X_i, X_j) = \text{I}\left(X_i \neq X_j\right) \exp(-d_{ij}), \quad (2)$$

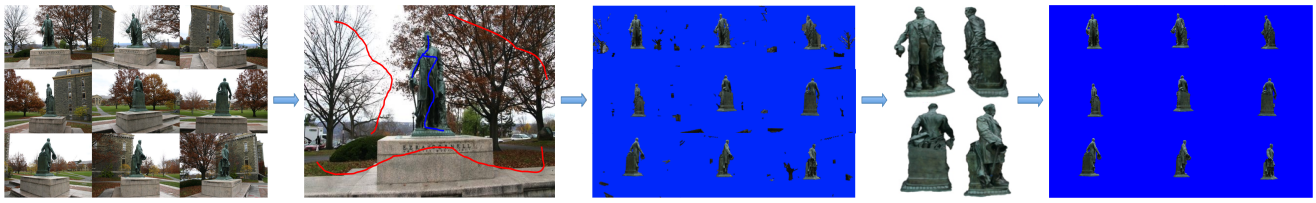
where $\text{I}(\cdot)$ is an indicator function that is 1(0) if the input argument is true(false), d_{ij} is the normalized distance between mean color of the superpixels i and j.

Finally, we use graph-cuts (with α-expansion) to compute the MAP labels for all superpixels (Bagon, 2006; Boykov and Kolmogorov, 2004; Boykov et al, 2001; Kolmogorov and Zabih, 2004). The parameter λ was empirically chosen using one of the datasets and fixed for all scenes[3]. This was found to work well in practice. Given, the above formulation of the energy minimization problem we discuss below how these constraints allow the user to reconstruct non-planar objects, planar scenes and even render non-planar objects as part of the planar scene.

4.2 Reconstructing non-planar objects

Consider reconstructing a non-planar object of interest such as a statue as shown in Fig. 4a. A popular approach to obtain the 3D model is shape from silhouette *i.e.*, obtain the

[3] The parameter λ is set to 0.5



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

Fig. 4: Object of interest 3D modeling. (a) Input multiview images of the object of interest (statue); (b) User node constraints that are used as hard constraints to learn the appearance models a perform the co-segmentation; (c) Resulting co-segmentation that has some inaccurate labeling in the background; (d) Shape from silhouette reconstructs the 3D model by the finding the volume of intersection given the camera parameters; (e) Projecting the 3D model back into each image allows fixing the segmentation errors that existed earlier.

silhouette of the object from multiple viewpoints and infer the volume of intersection (visual hull). We wish to avoid the tedious (sometime impossible) process of taking the object of interest into a controlled setup such as an studio with chroma-keying setup. We instead capture images of the object of interest from multiple views and get the user into the loop to obtain the silhouettes.

The user provides node constraints by providing scribbles of two colors (two labels) on one image to indicate the object of interest and the background as shown in Fig. 4b. The task is setup as a binary labeling problem as described above in Section 4.1 with $L = 2$. While the user may chose only one image to provide the node constraints, we use the idea that the images are tied together through the shared appearance models (A) learnt using the user inputs. The shared appearance models help formulate the energy function described in Eqn 1 for each image. Using graph-cuts we obtain the cosegmentation of the object in each view as shown in Fig. 4c. More details about the two class co-segmentation formulation and an extension to intelligently guide the user input is available in Batra et al (2011).

Note that the segmentation is noisy with small regions in the background that share similar appearance to the object of interest incorrectly labeled. While one approach to fix this is to modulate the smoothness parameter (β), this is a sensitive parameter to tune since it can result in regions of the object of interest being incorrectly labeled as background. We instead use the 3D geometry to help fix errors. Using the camera parameters recovered in the preprocessing stage (Section 3) and the co-segmentation as the silhouettes of the object we use shape from silhouette (Chen et al, 2008) to recover the 3D model of the object of interest shown in Fig. 4d. The volume of intersection recovered eliminates the sparse errors in the background. We now project the 3D model back into each view to fix the errors and obtain a clean co-segmentation of the object of interest Fig. 4d. We refer the reader to Kowdle et al (2010) for more results, and comparisons.

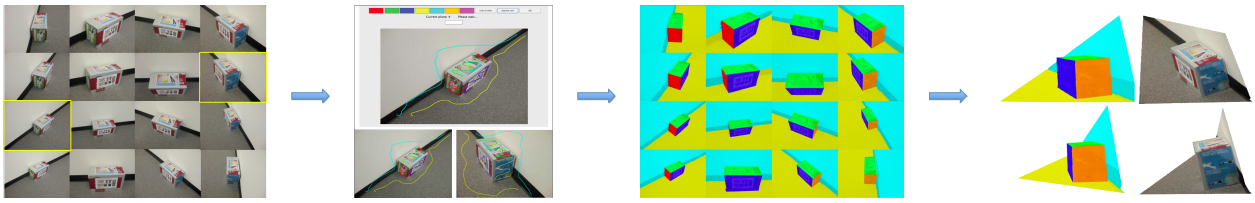
4.3 Reconstructing planar scenes

We have considered reconstructing non-planar objects in the previous section, we now consider obtaining piecewise planar reconstructions of the scene. We use the interface to display the multiview image collection to the user. The user selects an image and provides scribbles on the image with different colors indicating different planar surfaces as shown in Fig. 5b. In the context of the formulation described in Section 4.1, the user provides node constraints for L planar surfaces, the algorithm learns the appearance model to describe each surface and sets up the energy function solved via graph-cuts (with α-expansion). The result segments the image into the different surfaces labeled by the user as shown in Fig. 6a; we call this *scene segmentation*.

4.3.1 Scene segmentation to 3D geometry

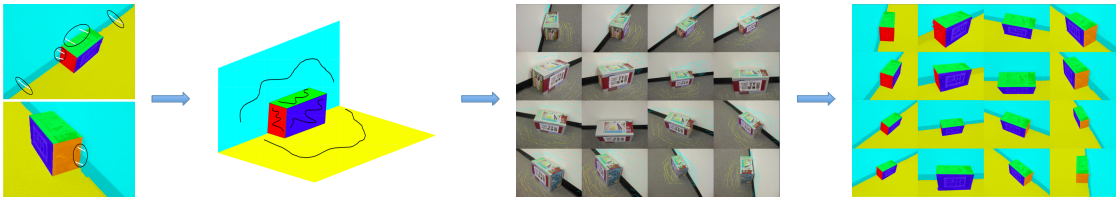
Using SfM we have a sparse 3D point cloud and the 2D feature correspondence across the images for this point cloud. We therefore know the subset of 3D feature points seen from the current view (scribbled image). This information helps transfer the labels from the 2D scene segmentation to the 3D points, based on which scene segment the 3D points project onto. We now use RANSAC-based plane-fitting on the labeled 3D points to estimate the plane parameters enforcing that the plane normal points outwards *i.e.*, towards the camera looking at the scene.

We note that there may be featureless surfaces like the wall in the scene, which lacks enough 3D point support to be reconstructed. The algorithm then prompts the user for some simple additional interactions to indicate the edges shared by this surface with the other surfaces in the scene by easily scribbling two lines across the edge shared as shown in black ellipses in Fig. 6a. We obtain an estimate of the plane parameter by enforcing that the boundary points correspond to the 2D projection of the line of intersection of the connecting 3D planes, thus, resulting in globally optimal plane parameters. However, if the featureless surface shares just



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

Fig. 5: Interactive piecewise planar 3D reconstruction: (a) Input images (image selected by user shown in yellow box); (b) User interactions to indicate the surfaces in the scene; (c) Scene co-segmentation of all images by using the idea of 3D scribbles to propagate scene geometry; (d) Some sample novel views of the reconstruction of the scene, with and without texture.



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

Fig. 6: Scene co-segmentation: (a) Scene segmentation with user interaction indicating connected planes (white scribbles in black ellipses); (b) 3D scribbles inferred from the segmentation; (c) 3D scribbles warped onto the other images to propagate scene geometry (Note: scribbles have been increased to improve visibility; the scribbles used for the results are in Fig. 5b); (d) Scene co-segmentation.

one edge with another plane, we make perpendicularity assumptions for that surface to choose the most probable plane amongst the infinite planes which shares that edge. This assumption has been shown to work well (Hoiem et al, 2005) and would be the best possible estimate, given the support.

4.3.2 3D scribbles and scene co-segmentation

Our goal is to obtain a co-segmentation of the planar surfaces in each of the images. Co-segmentation of the multiple surfaces in the scene is not as trivial as the binary image co-segmentation since, it is hard to define features discriminative between the geometric surfaces. However, when a user provides scribbles on an image, they are doing so based on their perception of the geometry of the scene, *i.e.*, they are not just indicating surfaces and objects in *that* image but, are giving us cues about the 3D scene geometry common across all the images. This is the common thread between the images we exploit to perform the co-segmentation.

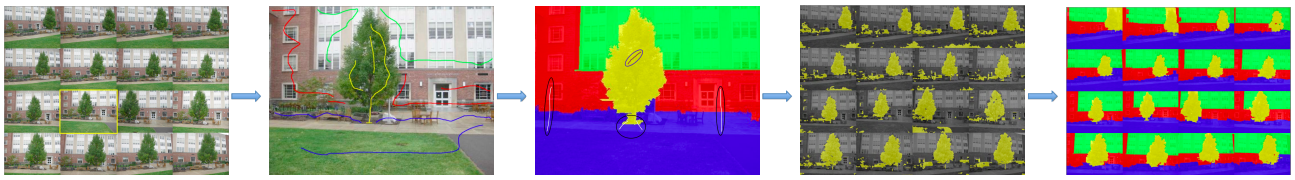
3D scribbles. Using the estimated plane parameters and the camera projection matrix of the scribbled image, we develop the idea of *3D scribbles*. Let the projection matrix of camera i be defined as $M_i = K_i R_i (I - C_i)$ where, K_i is the intrinsic matrix, R_i is the rotation matrix and C_i is the camera center in the world co-ordinate system. Consider, a 2D scribble point $s_{1,j}$ seen from the first camera, on a segment which corresponds to the plane l parameterized by $[\hat{n}_l \; d_l]$ where, $\hat{n}_l$ is the plane normal and d_l is the plane constant. The projection of this scribble point on another image seen from the second camera ($s_{2,j}$) is given by,

$$s_{2,j} = K_2 R_2 \left(\left(\frac{(-d_l - \hat{n}_l.C_1)}{\hat{n}_l.([K_1 R_1]^{-1} s_{1,j})} [K_1 R_1]^{-1} s_{1,j} + C_1 \right) - C_2 \right)$$

We take care to avoid warping the scribbles onto occluded planes by using the scene geometry and camera pose. For example, we consider the warped scribbles only on the planes visible from a particular view.

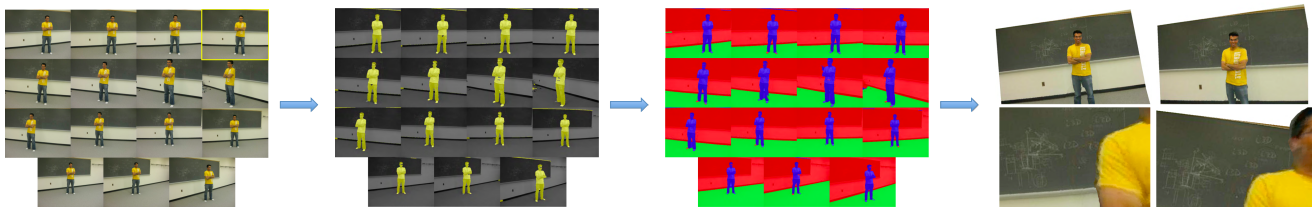
Scene co-segmentation. The resulting scribbles on the images are as shown in Fig. 6c. Using these scribbles as node constraints on all the images, we extend the energy minimization based multi-class labeling described in Section 4.1 to all the images thereby achieving co-segmentation of all the images into the multiple planar surfaces Fig. 6d.

We use the back-projection algorithm to evaluate the point of intersection of a ray from the camera center through every pixel on the image plane, and the estimated 3D surface. Using these 3D points, we generate a mesh for the scene with the corresponding image texture and render a texture mapped planar reconstruction of the scene as shown in Fig. 5d, enabling pleasing fly-throughs.



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

Fig. 7: Outdoor scene with occluding non-planar object: (a) Input images (image selected by user shown in yellow box); (b) User interactions; (c) Resulting scene segmentation with the additional interactions to indicate surface connectedness (white scribbles shown in black circles) and non-planar objects (magenta scribble shown in blue scribble); (d) Object co-segmentation (foreground non-planar object in yellow); (e) Scene co-segmentation by using 3D scribbles to propagate scene geometry.



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

Fig. 8: Indoor scene with occluding non-planar object: (a) Input images (image selected by user shown in yellow box); (b) Non-planar object co-segmentation; (c) Final scene co-segmentation; (d) Novel views of the reconstruction with volumetric rendering of the person.

4.3.3 Rendering non-planar objects in planar scenes

The algorithm thus far renders a planar reconstruction of the scene. In case of non-planar objects in the scene, we get an input from the user to indicate these objects, as shown in the blue ellipse in Fig. 7c. This tells the algorithm which surface and node constraints correspond to the non-planar object. Note that recent automatic approaches (Gallup et al, 2010; Lafarge et al, 2010) can also be used to identify non-planar regions. We estimate an approximate planar proxy for the object, which helps position the object as part of the rendered scene. We then use the algorithm described in Section 4.2 to obtain a visual hull of the non-planar object, which is rendered as part of the scene using an independent mesh. We note that one can also use recent unsupervised algorithms to obtain a co-segmentation of the foreground object (Kowdle et al, 2012b) albeit with some user input to indicate the object of interest in case of multiple foreground objects in the scene.

The scene co-segmentation allows us to create a composite texture map for the scene covering up holes due to the occluding non-planar object as shown in Fig. 9a. The algorithm renders the non-planar objects as part of the planar scene as we show with the tree in the outdoor scene in Fig. 9b and the person in the indoor scene in Fig. 8. Once the algorithm generates the 3D reconstruction, the user can provide more scribbles to indicate new or previously occluded planes, and improve the result, thus closing the loop on our interactive 3D reconstruction algorithm that is initiated by the user via node constraints. Please see video summary[4] with fly-through of the 3D reconstructions. More results and comparisons are available in Kowdle et al (2011a).

(a) (b)

Fig. 9: Non-planar objects: (a) Composite texture map for the scene (top) allows covering up holes due to occlusions (ellipse); (b) Novel views of the reconstruction with a volumetric model of the tree.

5 Computational engine guides the user

In our alternate algorithm, we intend to accept edge constraints from the user. Therefore, we need a smart computational engine that can automatically estimate the 3D structure of the scene and accept the user input across edges when, and where needed. We do so using an active-learning

[4] *http://chenlab.ece.cornell.edu/projects/Interactive_3D*

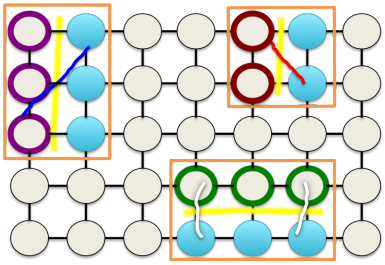


Fig. 11: Edge constraints. The cyan nodes illustrate the nodes the computational engine is uncertain about. The computation engine guides the user to provide support constraints for the uncertain nodes. The blue, white and red scribbles across the yellow edges in the graph illustrate the edge constraints provided by the user to provide support for the cyan nodes (guided by the computational engine). The nodes with the colored border illustrate the confident superpixels within each box that provide support for the cyan nodes. More description about the edge constraints and incorporating them into the framework is in Section 5.

algorithm. We refer to Fig. 1 and consider the ingredients for an active-learning algorithm in the context of image-based modeling. The integral components are: an automatic 3D reconstruction algorithm (computational engine); an approach to quantify the uncertainty of the algorithm and sample the most informative queries for user feedback; the human oracle who provides suitable interactions in response to the query; and lastly, an approach to seamlessly incorporate the feedback from the user into the algorithm. We describe each of the above aspects with respect to our algorithm in detail in the following sections.

5.1 Automatic 3D reconstruction algorithm

We develop a piecewise planar 3D reconstruction algorithm described below using successful ideas from recent works (Sinha et al, 2009; Furukawa et al, 2009; Micusík and Kosecká, 2010; Gallup et al, 2010).

5.1.1 Dense plane hypothesis generation

We use patch-based multiview stereo (PMVS) by Furukawa and Ponce (2009) as a pre-processing step, which compared to the sparse point cloud from SfM (Snavely et al, 2006), provides a much denser set of points that span the scene. Similar to Sinha et al (2009), we hypothesize dominant planes by analyzing the distribution of depths of the 3D points along each hypothesized normal (using the estimated vanishing directions). We break down an image into superpixels[5] and use the assumption that every superpixel would lie on a planar surface (Micusík and Kosecká, 2010; Saxena et al, 2009). Using these superpixels, we hypothesize additional planes by fitting planes to 3D points that project onto the same superpixel. In practice, we observe that this allows us to add new planes not hypothesized before as their normals are different from the dominant normal directions.

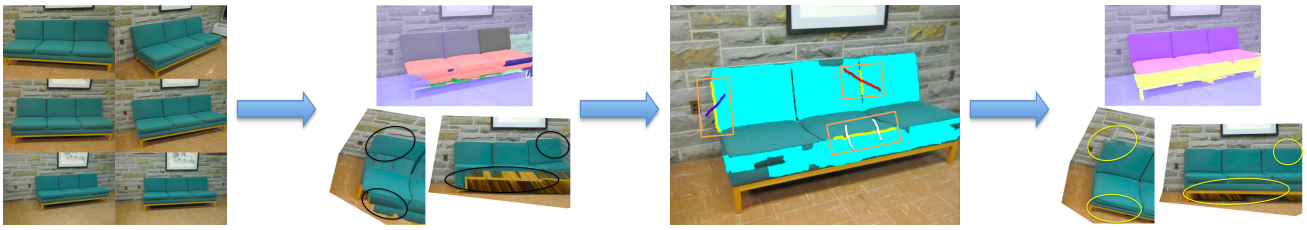
[5] We use graph based segmentation (Felzenszwalb and Huttenlocher, 2004) to break each image down to about 400 superpixels.

5.1.2 Energy minimization

We will use this section to establish notation and describe the automatic piecewise planar reconstruction algorithm, however the main contribution of this work is in Section 5.2 where we exploit the uncertainty of the algorithm. The dense plane hypothesis stage results in P (about sixty) planes. These hypothesized planes serve as the set of discrete labels, which changes the piecewise planar reconstruction problem to a multi-label segmentation problem, formulated as an energy minimization problem over the superpixels. The formulation is similar to that described in Section 4.1, with the discrete label space $\{0, 1, \ldots, P-1\}$. The unary and pairwise term for the automatic piecewise planar stereo algorithm is defined below.

Unary term. For a particular view, we compute homographies for each plane to warp the other images to that view. We use normalized cross-correlation (NCC) to quantify the warp error. We refer the reader to Sinha et al (2009) for more details. We compute the NCC using the superpixel as support at each pixel as opposed to a constant window. We also compute a color term that measures the mean color difference of each superpixel between the original and the warped image. We use a weighted combination of the two normalized terms as the unary term with the weights tuned by observing the performance on one of the datasets.

Pairwise term: Co-planar classifier. We introduce an *adaptive* co-planar classifier to model the pairwise term. We learn a classifier that given a pair of adjacent superpixels returns a score representing the co-planarity of the superpixels. We use the geometric context dataset by Hoiem et al (2007) (with seven ground truth geometric labels). Adjacent superpixels with the same geometric label are used as positive data points while pairs with different labels, are used as negative data points. We note that adjacent superpixels lying on occluding 'parallel' planes would be bad data points, but, in practice this does not hinder the performance. We use *relative* features (Hoiem et al, 2007) for each pair of superpixels as the feature vector for each data point and learn a logistic regression model. This model is continuously updated by the active-learning algorithm. We note that one can also use laser image data to learn a co-planar classifier by fitting planes to the laser data to obtain the samples needed (Sax-



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

Fig. 10: (a) shows a set of multiview images of a scene; (b) shows the result of the automatic algorithm, the plane labeling shown on the top indicates the inaccurate labeling, the novel views of the 3D model are shown at the bottom with black circles showing the errors. (c) the proposed active-learning algorithm quantifies the uncertainty of the algorithm and detects the uncertain regions (in cyan), the uncertainty boxes (in orange) with the highlighted edges (in yellow) are used to query the user for support, the user provides any of three types of interactions within each box via simple scribbles across the highlighted edge, coplanar scribbles (red), not-coplanar scribbles (white) or not-connected scribbles (blue) as shown; (d) shows the result of the algorithm after incorporating the information provided by the user, plane labeling on top shows the improved labeling, the improved reconstruction is shown below through novel viewpoints with yellow circles illustrating the corrected geometry.

ena et al, 2009). We use a contrast sensitive Pott's Model to model the pairwise term.

$$E_{ij}\left(X_i, X_j\right) = I(X_i \neq X_j)\; exp\left(-d_{ij}\right) \tag{3}$$

The pairwise term when adjacent superpixels take different labels should be high when the contrast d_{ij} is low or when the superpixels are likely to be co-planar and high otherwise. Thus, given a pair of adjacent superpixels, using the learnt co-planar classifier, we obtain a score that represents the likelihood of this pair being co-planar. This score is used to model the contrast d_{ij} (1 - similarity score) in the contrast sensitive Pott's model.

We again use graph-cuts (with α-expansion) to compute the MAP labels for all superpixels (Bagon, 2006; Boykov and Kolmogorov, 2004; Boykov et al, 2001; Kolmogorov and Zabih, 2004). This allows us to automatically obtain the piecewise planar reconstruction of the scene. At this stage the algorithm has used the observed multiview stereo cues to obtain the piecewise planar reconstruction albeit with errors as shown in Fig. 10b. The parameters were empirically chosen using one of the datasets and fixed for all scenes. This was found to work well in practice. We explain below our active-learning algorithm to fix the errors in the reconstruction by putting the user into the loop to provide edge constraints deriving support from the current 3D reconstruction of the scene.

5.2 What is the uncertainty?

An important aspect of an active-learning algorithm is to identify the uncertainty of the algorithm. Intuitively, since our algorithm follows an energy minimization framework to solve the multilabel problem over the graph of superpixels, we quantify the uncertainty of the algorithm with respect to the uncertainty in labeling the superpixels. At a high level, we evaluate the uncertainty of a superpixel in terms of *confidence* and *ambiguity*.

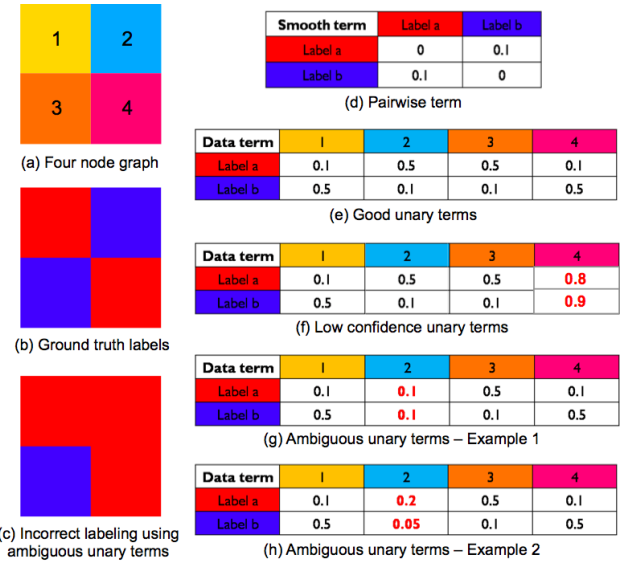


Fig. 12: Synthetic example to illustrate the uncertaintly of the algorithm (Best viewed in color). Details in Section 5.2.

Synthetic example. We explain our intuition through a small synthetic example. Consider, a four node graph with their 4-connected neighborhood as shown in Fig. 12a. Let us suppose the ground truth labeling consists of two labels as shown in Fig. 12b. Now, Table 12d shows the pairwise term and Table 12e shows an instance of unary terms which gives the

ground truth labeling. Note that the unary terms are energies so the lower the value the more affinity to the label. Also note that the unary terms for only the two relevant labels are shown, assume that the energies for the other labels are high and hence not relevant. In Table 12f we observe that while the energies of the two labels for node 4 reflect that it has a preference for Label a, both the energies are very large (shown in red) indicating a low confidence in the decision. In Table 12g we observe that two labels have low energies indicating the ambiguity (shown in red) making the correct decision ambiguous. Similarly, in 12h the unary terms (shown in red) indicate that the node 2 should take the label b but incorporating the pairwise term causes the final labeling to be erroneous as in Fig. 12c. We need an approach to label these nodes as uncertain. We note that entropy of the unary terms (say the ratio of the unary terms for these two labels) would help in case of Table 12g while the entropy in case of Table 12h would be low. We therefore need to incorporate the effect of pairwise to determine ambiguity. We explain how we identify these contributors to the uncertainty in detail below.

5.2.1 Confidence

Confidence quantifies how confident the algorithm is to assign a particular plane hypothesis to the superpixel. Low confidence superpixels represent high uncertainty regions, for example, occlusions. We obtain these regions via the energy minimization framework. Motivated by the multi-view stereo work by Campbell et al (2008), we add an additional label to our set of discrete labels and refer to it as the *unknown* label. For every superpixel, X_i where $i \in V(\textit{all superpixels})$, the unary term $E_i(X_i)$ for the *unknown* label is set at a constant penalty. Intuitively, this penalty is large enough so it does not affect the unary terms of the more confident superpixels while low enough so that low confidence superpixels are separated out. We use the median of all the unary terms, which serves as a safe unary term value in practice for the *unknown* label. As opposed to using a simple threshold on the unary terms to determine low confidence regions, this approach gives the pairwise term an opportunity to try to derive support,when possible, for the low confidence superpixels from their neighbors. The superpixels that take the *unknown* label after the minimization are called *uncertain* superpixels.

5.2.2 Ambiguity

Ambiguity quantifies the uncertainty of the algorithm between different plane hypotheses. Superpixels that are ambiguous about multiple plane hypotheses represent high uncertainty regions, for example, textureless surfaces, specular surfaces, inaccurate plane hypotheses, etc. One approach to determine ambiguous data points in a multi-class labeling problem would be to analyze the unary terms, using the idea that the entropy of the unary terms of ambiguous data points would be high (Jain and Kapoor, 2009). However, the entropy in the unary terms is not sufficient to capture *all* the ambiguity because the effects of the pairwise term are ignored. We thus evaluate the ambiguity by determining the ambiguity of resulting MAP labeling after incorporating the effect of the pairwise. We do so by using the *Graph-cut uncertainty* similar to Batra et al (2011), as explained below.

Let the minimum energy $E(X)$ for the graph $G = (V, E)$ be E_{min}. Given the complete set of plane hypotheses (L labels), suppose that for a superpixel X_i the minimum energy label is l_i. We flip the label of superpixel X_i from l_i to one of the the other labels l_j in L and recompute the energy, $E_{i\rightarrow j}$ of the labeling. At each such flip stage, we compute the absolute difference between the minimum energy (E_{min}) and flip energy ($E_{i\rightarrow j}$),

$$E(X_i)_{(\Delta[i\rightarrow j])} = |(E_{min} - E_{i\rightarrow j})| \tag{4}$$

The ambiguity for every superpixel is computed by measuring the minimum of all such flip energy differences,

$$E(X_i)_{ambig} = \min_{j \in L \setminus i} E(X_i)_{(\Delta[i\rightarrow j])} \tag{5}$$

The intuition behind this is simple. If the algorithm does not have high ambiguity about assigning a particular plane hypothesis to a superpixel, the ambiguity energy difference, $E(X_i)_{ambig}$ should be high. However, if this value is low, it amounts to ambiguity between different plane hypotheses and hence uncertainty. We normalize the ambiguity energy differences and threshold that at 95% to obtain the top 5% of ambiguous superpixels. These are again called *uncertain* superpixels. We note that min-marginals by Kohli and Torr (2008) could also be used to capture ambiguity albeit it is computationally very intensive. While our proposed approach helps us obtain an estimate of uncertainty in an inexpensive way we note that the MRF could be solved by probabilistic methods thereby giving us direct measures of uncertainties. This is however not explored in this manuscript.

5.2.3 Region level uncertainty

In addition to the superpixel level uncertainty, we determine *region level* uncertainty. We determine regions (groups of superpixels) that take a particular independent plane label but have no support from the 3D point cloud, i.e. none of the 3D points project onto the region, and label them as uncertain. The intuition here is that, a set of superpixels with no support from the 3D points, taking their own independent plane label amounts to uncertainty.

5.2.4 Quantifying uncertainty

Grouping the uncertain superpixels to uncertain regions, we first identify and highlight all the boundary or support edges where user interaction might be needed. To ease the interactive process, we draw a box (uncertainty box) centered at this edge, scaled to be the larger of a minimum predefined size or two standard deviations of the edge size. Our active-learning algorithm then queries the user with the regions with the highest uncertainty or information gain. We thus need a metric to quantify the uncertainty of each box.

Consider *n* normals that span all the planes in the scene (from the initial plane hypothesis step). Superpixels are organized in increasing order of cost, based on the lowest cost the superpixel pays for adopting a particular normal (e.g. $C1_s, C2_s, \ldots, Cn_s$). This gives an indication about how certain it is about taking a particular normal. For a low uncertainty region, the value $C1_s$ would be considerably lower than the next best normal, i.e. $C2_s$.

Consider any orange uncertainty box shown in Fig. 13. Let R_i indicate the region under a box i that represents the set of all superpixels part of the uncertain region under the box i.e., the cyan region. Let $R_{i,support}$ indicate the region under box i *not* part of the uncertain region under the box i.e., the non-cyan region. Let $coplanarity(e)$ represent the score of the co-planar classifier for an edge e between two superpixels, and E_i indicate the set of all edges under a box i. The uncertainty is quantified through four terms: Cost ambiguity of the region in the box (A), Confidence of the support region in the box (F), Graph-cut uncertainty (GCU), and Co-planar classifier uncertainty (CoP).

$$A_i = \max_{s \in R_i} \frac{C1_s}{C2_s} \tag{6}$$

$$F_i = \max_{s \in R_{i,support}} (1 - C1_s) \tag{7}$$

$$GCU_i = \min_{s \in R_i} E(X_s)_{ambig} \tag{8}$$

$$CoP_i = \max_{e \in E_i} coplanarity(e) \tag{9}$$

Our final uncertainly score for each box i, is the sum of each of the component uncertainties defined in Eqn (6)-(9), using an equal weighting for each term as a fair setting. In practice, equal weights work well, as we show in Section 5.4. It is worth pointing out that with more training data we can learn these weights via cross validation however, since we use normalized uncertainty components equal weights worked well in practice. We rank the boxes according to this score and query the user with the top three uncertainty boxes for some support. We note that we can achieve a steady improvement by querying the user with only *one* most uncertain box instead of the top three, however, this would need additional iterations of the algorithm, requiring additional user interactions and incurring processing overhead.

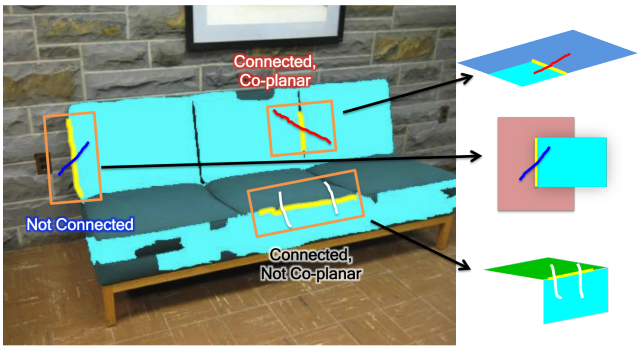


Fig. 13: The user can provide three types of interactions to indicate coplanar regions (red), not-coplanar regions (white) and not-connected regions (blue) across the highlighted edge (yellow) within each uncertainty box (orange), to provide support for the uncertain regions (cyan).

5.3 Putting the user in the loop

In our active-learning framework, given the uncertainty boxes, we wish to obtain user interactions in the form of support for the uncertain regions or edge constraints and incorporate this feedback into the algorithm to improve the reconstruction. The user (oracle) provides one of three scribble based interactions described below, within each box as shown in Fig. 13.

Connected and co-planar regions. When the edge highlighted in the uncertainty box is an edge between connected and co-planar regions, i.e. *same plane*, the user provides a scribble as support across the edge to indicate co-planarity, shown as the red scribble (Fig. 13). We use this additional information to improve the support for the uncertain superpixels. This is done by adding long-range edges (non adjacent nodes) between the nodes (superpixels) scribbled on by the user to allow the algorithm to propagate the confident label to the uncertain superpixels.

Connected but not co-planar regions. In case the highlighted edge is an edge between connected but not co-planar regions, i.e. *different planes*, the algorithm would need cues about the edge shared between these two regions in order to hypothesize a good plane for the uncertain region. We do so by allowing the user to use two white scribbles across the edge to indicate the edge segment shared by the planes (Fig. 13). The edge constraint from the user is first used to break edges of the graph to avoid inaccurate labeling. In addition,

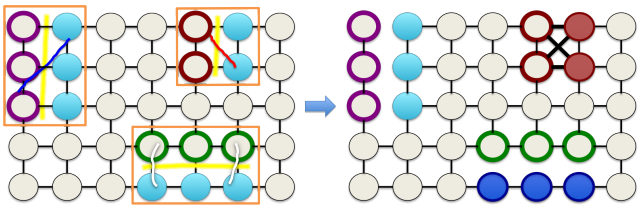


Fig. 14: Incorporating the user constraints to update the structure of the graph. Note how the red scribble (connected and co-planar) adds more edges and strengthens edges; blue scribble (not-connected) breaks edges in the graph; white scribble (connected and not co-planar) breaks edges and hypothesizes a new planar surface for the uncertain region (blue nodes).

using the confident region we obtain the positions of these edge points in 3D. Given this information and the hypothesized normals (Section 5.1.1), we use a RANSAC based approach to find the best fit plane through the 3D edge marked by the user. We add this new plane hypothesis and estimate the corresponding unary term as described in Section 5.1.2, adding hard constraints to ensure that the uncertain superpixels choose this new plane. This is therefore both an edge and a node constraint.

Not connected regions. If the highlighted uncertain edge corresponds to an edge between not connected regions, i.e. *occluding planes*, the user can indicate not-connected regions by using the blue scribble as shown (Fig. 13). We incorporate this information into the algorithm by breaking edges between these superpixels in our graph, thereby hindering these regions from taking the same plane.

Submodularity. In this work the discrete optimization method we use is Graph Cuts, which requires sub-modularity. Our discrete labeling formulation is sub-modular since the pairwise term uses a contrast sensitive Potts model. The three user-constraints described above maintain sub-modularity. The connected and co-planar scribbles (red scribbles) modulate the pairwise term while maintaining sub-modularity. The connected but not co-planar scribbles (white scribbles) and the not connected scribbles (blue scribbles) are similar in that they both lead to breaking edges in the graph. This however changes the structure of the graph while still maintaining sub-modularity since the pairwise terms are still a contrast sensitive Potts model. Potts model encourages smoothness in the labeling but in some cases we know two regions cannot take the same label. For instance in case of the not connected scribble we know that they are not-connected but we do not know what the pairwise relationship between them is. The idea of breaking the edges is to allow for the uncertain regions to not-derive support from the other region. This however does not affect the other edges therefore maintaining the sub-modularity.

We incorporate all the constraints provided by the user and suitably reformulate the graph over superpixels as illustrated in Fig. 14. The connected and co-planar scribble (red) adds more edges to the graph and strengthens the edges to encourage that the uncertain nodes choose the neighboring confident node label. The not-connected scribble (blue) breaks edges in the graph ensuring that information is not passed between the nodes. Lastly, the connected and not co-planar scribble (white) breaks edges in the graph to avoid inaccurate label smoothness across those edges. A new planar surface is then hypothesized for the uncertain region by using the uncertain edge as the 2D projection of the line of intersection of the 3D planes (similar to Section 4.3.1). In addition to modifying the graph, these constraints provide more information about the co-planar regions of the scene, which are used as additional samples to update the co-planar classifier. This updates the pairwise term, which makes the co-planar classifier scene specific. Using the energy minimization framework (Section 5.1.2) on this updated graph, we again obtain the MAP labels for the superpixels, which gracefully propagates the additional information given by the user. The process of obtaining uncertain regions, quantifying uncertainty, querying the user for support, and then updating the algorithm with the additional information is repeated using the new result, closing the loop on the active-learning algorithm.

5.4 Experiments and Results

In this section, we describe the datasets, the evaluation metric we use, and we discuss experiments to *quantitatively* evaluate the performance of the proposed active learning approach via machine experiments and a user study. We also discuss qualitative improvements in the reconstructions.

5.4.1 Datasets

We collect images spanning six scenes (each with about ten images) that lack geometric cues such as lines essential to the automatic algorithm and, include textureless surfaces or specular surfaces that hinder the performance of the automatic algorithm. We also use two standard datasets that have been used in prior automatic works (Sinha et al, 2009). We make all the datasets used in our works (Kowdle et al, 2010, 2011a,b) publicly available[6].

5.4.2 Ground truth

We note that recent work address the task of evaluating interactive algorithms (Kohli et al, 2012; McGuinness and O'Connor, 2012). We do not have access to the ground-truth depth of

[6] *http://chenlab.ece.cornell.edu/projects/ActiveLearningFor3D*

these natural scenes however, in order to quantitatively evaluate the performance of the proposed active-learning algorithm, we manually label pixel-wise ground truth segmentation of the planes for all the datasets. In addition, to capture some 3D information we manually label ground truth normals for each segmented region. The ground truth pixel-wise segmentation along with their ground truth normals serves as a good quantitative indicator of the performance of the algorithm. Given the algorithm's result, we map each ground truth region to the largest label in that region in the algorithm's result, which agrees with the ground truth normal. Using these mapped labels we compute the pixel-wise labeling accuracy for each of the ground truth regions and compute the average accuracy across the datasets. We note that this metric can lead to inaccuracies in case of occluding parallel planes, however, it serves as a good metric to determine the relative performance in our experiments.

5.4.3 Machine experiments

In order to perform an exhaustive set of experiments to evaluate the various design choices, we develop a mechanism to generate *synthetic interactions*, which mimic the human user. For every uncertainty box queried by the algorithm, using the ground truth segmentation, normals, and the occlusion boundaries (manually labeled), we provide one of three interactions described in Section 5.3. We note that an iteration in our experiments refers to providing the interactions in any three distinct locations (e.g. within the three uncertainty boxes in the active-learning experiment).

Performance of active learning. We evaluate the performance of the proposed active-learning algorithm against ground truth sampling (an upper bound) and a random sampling experiment as shown in Fig. 15.

In the ground truth sampling experiment (black curve), at each iteration, we compute a 2D error map using the algorithm's output and the 'ground truth'. The machine interactions are then aimed to provide support to these error regions, beginning from the largest error region, in the order of decreasing size. This is a good upper bound since at each iteration we aim to achieve the best improvement by directly correcting the errors. The active-learning experiment (blue curve) evaluates the performance of the proposed algorithm in which, the machine interactions are guided by the uncertainty boxes indicated by the active-learning algorithm. In the random sampling experiment (red), we do not use the proposed active-learning algorithm to choose the uncertain regions, but instead randomly sample the uncertainty boxes along the segmentation boundaries.

We see from Fig. 15 that the proposed active-learning algorithm performs much better than random sampling and, in addition, performs respectably when compared to the upper bound, given that it does not have the luxury to access ground truth while querying interactions. We also note that it can achieve the peak performance achieved by the random sampling at the end of more than twenty iterations in as few as four iterations.

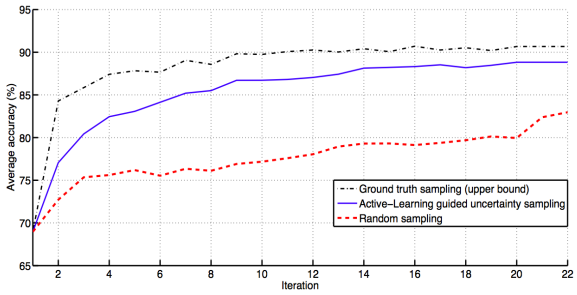


Fig. 15: Machine experiments: Our proposed active-learning algorithm performs significantly better than random sampling and performs respectably compared to ground truth sampling.

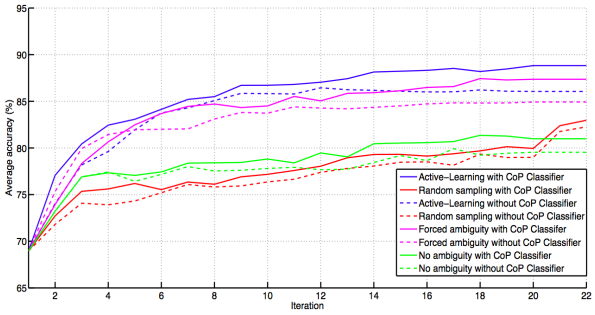


Fig. 16: Machine experiments: The performance using the various design choices shows that the proposed active-learning algorithm with the chosen design obtains the best results, validating our choices. The dashed curves indicate the performance without the adaptive co-planar classifier with the same design choices as the solid curve of the same color. (Section 5.4.3).

Evaluating algorithm design choices. We evaluate the design choices we incorporated into the proposed active-learning algorithm. We first evaluate the effectiveness of incorporating 'ambiguity' to describe uncertainty. The solid green curve in Fig. 16 shows the performance of the algorithm when we ignore ambiguity and only rely on the confidence measure. In comparison with our active-learning curve (solid blue), we see that when the algorithm quantifies only the low confidence regions as uncertain, it fails to capture several critical uncertain regions, leading to a very slow and minimal improvement in performance.

In our algorithm, we use graph-cut uncertainty to capture ambiguity. We evaluate this choice observing the performance when we use the entropy of the data terms to di-

rectly detect the ambiguous regions, forced ambiguity curve (solid magenta) in Fig. 16. This firstly strengthens the importance of ambiguity on comparing with the no ambiguity green curve and in comparison with the active-learning curve (solid blue) shows that graph-cut uncertainty captures relevant regions which are missed by the forced ambiguity.

Lastly, we evaluate the adaptive co-planar classifier. In Fig. 16, comparing the solid curves with the corresponding dashed curves shows that using the adaptive co-planar classifier (CoP) gives steady improvement in performance in all the experiments.

5.4.4 User study

We perform a user study with ten users and three experiments to evaluate the performance of the algorithm. Fig. 17 shows the performance of the users. We restrict the number of iterations to reduce the effort of the users. The first experiment is the random interactions experiment, in which we show the user the segmentation boundaries from the algorithm, however, with no indication about which regions are erroneous, as shown in Fig. 19a. The user was instructed to provide three distinct interactions across any edge by observing the segmentations, with the only cue that each segmented region corresponds to a planar surface according to the algorithm. The *red* curve in Fig. 17 shows the performance of the users. We observe that the human user performs better than the machine with the random interactions experiment because the human user has an implicit notion of the 3D structure of the scene. The annotations from the user are therefore more meaningful.

The second experiment is the exhaustive examination experiment. Here, in addition to the segmentation boundaries, we color code the normals of each segment as shown in Fig. 19b. The user was again instructed to provide three distinct interactions across any edge by observing the errors in the segmentations, with the normals guiding them towards erroneous regions. This leads to much better performance as seen by the *black* curve in Fig. 17.

The last experiment evaluates the proposed active-learning algorithm. We show the user the uncertain regions detected by the algorithm in cyan. We highlight the uncertain edge in yellow, and draw three orange boxes to query the user for interactions, as shown in Fig. 19c. The user was instructed to follow these orange boxes and provide interactions across the edges to provide support for the cyan regions. The *blue* curve in Fig. 17 shows the performance. We observe that the active-learning algorithm performs much better than random interactions and performs at par with the exhaustive examination, indicating that the algorithm effectively guides the user towards relevant uncertain regions.

We compare the time taken by a user guided by the proposed active learning algorithm vs. an unguided user. We plot the average accuracy across the time taken in Fig. 18. The proposed active-learning algorithm achieves better performance and significantly faster (almost 2x speed up).

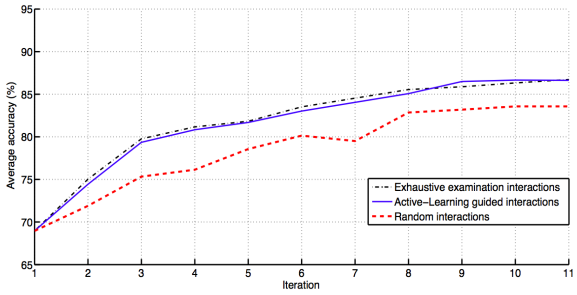


Fig. 17: User study: The proposed active-learning algorithm not only out performs random interactions, but performs at par with exhaustive examination in significantly lower time (Section 5.4.4).

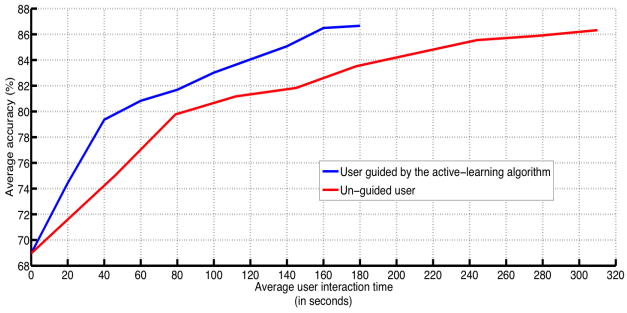


Fig. 18: User study - time: The proposed active-learning algorithm achieves better performance and significantly faster (Section 5.4.4).

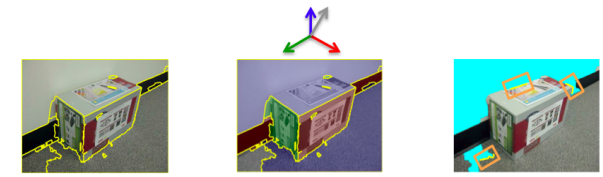


Fig. 19: The three different user experiments conducted to evaluate the proposed algorithm (Section 5.4.4).

5.4.5 Qualitative analysis

In Fig. 20, we show improvements in quality of the labeling and the 3D reconstructions as a result of incorporating the user interactions using the proposed algorithm[7].

Row 1 shows the improved reconstructions in presence of homogeneous surfaces like the wall and ground; Row 3 shows the improved result in case of an occluding ob-

[7] *http://chenlab.ece.cornell.edu/projects/ActiveLearningFor3D*

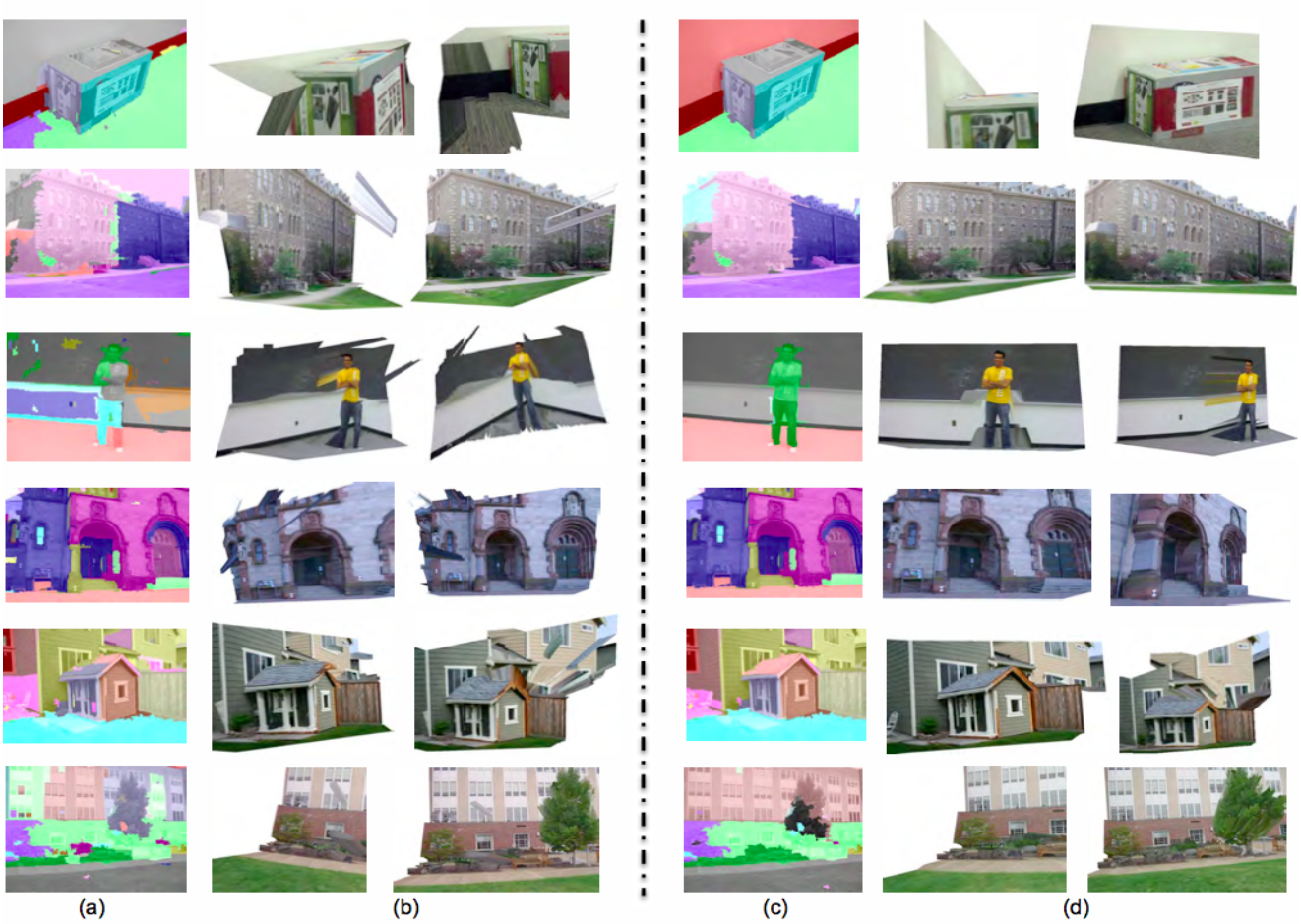


Fig. 20: Qualitative results: (a) and (b) show the plane labeling and, novel views of 3D reconstruction from the automatic algorithm respectively; (c) and (d) shows the improved results using the active-learning algorithm respectively.

ject (planar approximation) and homogeneous background. Rows 4 and 5 show the output of the algorithm on public datasets used in prior work (Sinha et al, 2009). These are datasets in which the algorithm has strong cues to automatically reconstruct the scene requiring minimal user interactions. These show that our automatic algorithm is not sub-optimal.

Relying on superpixels can hinder the performance in some cases. Note, for example, the error near the legs in row 3 due to a narrow superpixel leak. Row 6 demonstrates a failure case of the algorithm. In this example, there was a superpixel that leaks from the top of the tree onto the building. Since the uncertain edge we show the user always follows the superpixel boundaries, superpixel leaks can affect the performance. In this case, when queried, the user would always mark the regions as co-planar, resulting in a part of the tree labeled as part of the building behind it. However, we note that the proposed algorithm still performs significantly better than the automatic algorithm.

Comparison. We qualitatively compare the performance of a user guided by the proposed active-learning algorithm with the performance of an unguided user in Fig. 21. The initial reconstruction of the scene has errors that are partially fixed after 8 iterations by constraints provided by the unguided user. In comparison, the user guided by the proposed active-learning algorithm achieves a much more accurate reconstruction twice as fast, after only 5 iterations.

6 Applications

If there is one thing the growing popularity of immersive virtual environments (like Second-Life® with 6.1 Million members) and gaming environments (like Project Natal®) has taught us – it is that people crave personalization. For example, gamers want to be able to 'scan' and use their own gear (such as skateboards) in a skateboarding game; people want to be able to take something from the real world (such as a statue, or your house) into the virtual environment. We use the proposed idea of putting the user in the loop to de-

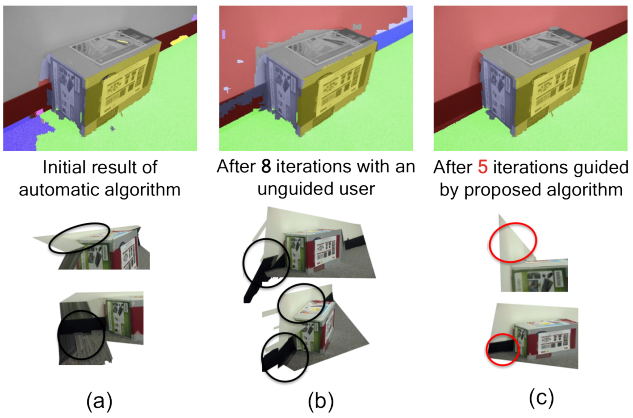


Fig. 21: Qualitative comparison: (a) The initial reconstruction of the scene, with errors shown in black ellipses; (b) The result after an unguided user provide interactions for 8 iterations, where errors still exist as shown in black ellipses; (c) In comparison, the user guided by the proposed active-learning algorithm achieves accurate reconstruction after only 5 iterations. Errors fixed are shown in red ellipses.

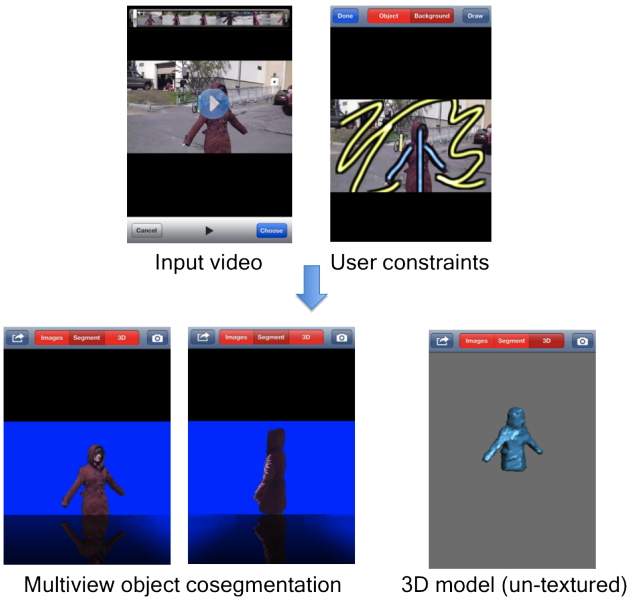


Fig. 22: iModel: Object of interest 3D modeling on a mobile device.

Fig. 23: Physical 3D printout of the object of interest obtained using the proposed algorithm. The top row shows the set of multiview images of the object of interest used to obtain the 3D printout below.

velop an easy approach to obtain a 3D model of their object of interest. In particular, driven by the ubiquitous spread of mobile devices with touch-screen interfaces, we develop a mobile application to perform this task.

We give an overview of our mobile application (Kowdle et al, 2012a) in Fig. 22. The application uses a client-server setup and is developed for iOS devices. The client (the user) captures a video of the object of interest by walking around the object. This video is then sent to the server that samples frames from the video, starts running structure-from-motion to extract the camera parameters and sends the sampled frames back to the client. The user is now allowed to flip through the images, select any image and provide user interactions via the touch-screen to indicate the object of interest and the background via scribbles. These scribbles are then sent to the server, which performs interactive co-segmentation, to perform shape-from-silhouette. The co-segmentation of the object of interest from each view and the 3D model of the object are now sent back to the client; which the user can visualize. Please refer our website for a demo video of the application[8].

While augmented reality is a well established application that allows for virtually placing novel objects in a scene, an interesting application of the reconstructed 3D model follows the recent trend in 3D printing. We use the reconstructed model to obtain a physical 3D printout of the object[9] as shown in Fig. 23, allowing for an interesting application to obtain physical 3D models from images of the object captured in it's natural environment.

7 Conclusions and future work

In this paper, we have proposed a framework to put the user in the loop for image-based modeling. Motivated by the recent success in discrete labeling formulation for image-based modeling we have leveraged the user input as node and edge constraints for the underlying Markov Random Field. We have considered algorithms where the user initiates the al-

[8] *http://chenlab.ece.cornell.edu/projects/iModel*

[9] The 3D printouts were obtained using the online service *http://www.shapeways.com*

gorithm to indicate the object of interest, allowing for reconstructing non-planar objects and planar scenes. We proposed a novel active-learning algorithm for piecewise planar 3D reconstruction where the computational engine guides the user constraints. The algorithm tries to reconstruct the scene automatically, quantifies uncertainty, and asks the user to provide support for the most uncertain regions via simple and intuitive interactions (coplanar, not-coplanar, and not-connected scribbles). The algorithm incorporates these constraints to obtain better reconstructions, thus closing the loop on the interactive algorithm. We show through a user study and machine experiments that the proposed algorithm not only improves the reconstruction, but does so in significantly lower time than exhaustive examination by the user. In addition, we have demonstrated some end user applications including object of interest 3D modeling on a mobile device and 3D printing an object of interest.

Future work. We believe that the proposed idea of active-learning for putting the user in the loop has a lot of potential beyond piecewise planar reconstructions. The framework of guiding the user to provide feedback to obtain better reconstructions can, not only be extended to multi-view stereo approaches (in which, other forms of interactions can aid dense *surface* reconstruction), but can also be used with works trying to obtain a 3D reconstruction from a single image, which has an inherent learning framework. The active-learning framework incorporates the positive aspects of both the automatic as well as the interactive algorithms, using the scene specific user inputs when and where needed, to render improved reconstructions. We note that the algorithms proposed in this paper explore putting the user in the loop for image-based modeling after the images have been captured *i.e.*, *post capture*. We can also explore how we can put the user in the loop *at capture time*. For example, guide the user at capture time, and observe the physical interactions between the user (or say a robot operated by the user) and the scene at capture time to recover the 3D structure of the scene. These are interesting future directions for putting the user in the loop for image-based modeling.

Acknowledgements The authors thank Anandram Sundar for the data annotation.

References

Bagon S (2006) Matlab wrapper for graph cut. `http://www.wisdom.weizmann.ac.il/~bagon`

Bartoli A (2007) A random sampling strategy for piecewise planar scene segmentation. CVIU 105(1):42–59

Batra D, Kowdle A, Parikh D, Luo J, Chen T (2011) Interactively co-segmenting topically related images with intelligent scribble guidance. IJCV 93(3):273–292

Baumgart BG (1974) Geometric modeling for computer vision. PhD thesis, Stanford University

Boykov Y, Kolmogorov V (2004) An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. PAMI 26(9):1124–1137

Boykov Y, Veksler O, Zabih R (2001) Efficient approximate energy minimization via graph cuts. PAMI 20(12):1222–1239

Campbell N, Vogiatzis G, Hernndez C, Cipolla R (2007) Automatic 3d object segmentation in multiple views using volumetric graph-cuts. In: BMVC

Campbell ND, Vogiatzis G, Hernández C, Cipolla R (2008) Using multiple hypotheses to improve depth-maps for multi-view stereo. In: ECCV

Chen Z, Chou HL, Chen WC (2008) A performance controllable octree construction method. In: ICPR

Collins B, Deng J, Li K, Fei-Fei L (2008) Towards scalable dataset construction: An active learning approach. In: ECCV

Comaniciu D, Meer P (2002) Mean shift: a robust approach toward feature space analysis. PAMI 24(5):603–619

Criminisi A, Reid ID, Zisserman A (1999) Single view metrology. In: ICCV

Debevec P, Taylor C, Malik J (1996) Modeling and rendering architecture from photographs: A hybrid geometry- and image-based approach. In: SIGGRAPH

Fang YH, Chou HL, Chen Z (2003) 3d shape recovery of complex objects from multiple silhouette images. Pattern Recogn Lett 24(9-10):1279–1293

Felzenszwalb PF, Huttenlocher DP (2004) Efficient graph-based image segmentation. IJCV 59(2):167–181

Forbes K, Nicolls F, de Jager G, Voigt A (2006) Shape-from-silhouette with two mirrors and an uncalibrated camera. In: ECCV, pp 165–178

Furukawa Y, Ponce J (2009) Accurate, dense, and robust multi-view stereopsis. PAMI

Furukawa Y, Curless B, Seitz S, Szeliski R (2009) Reconstructing building interiors from images. In: ICCV

Furukawa Y, Curless B, Seitz SM, Szeliski R (2010) Towards internet-scale multi-view stereo. In: CVPR

Gallup D, Frahm J, Pollefeys M (2010) Piecewise planar and non-planar stereo for urban scene reconstruction. In: CVPR

Goesele M, Snavely N, Curless B, Hoppe H, Seitz SM (2007) Multi-view stereo for community photo collections. In: ICCV

Gosselin PH, Cord M (2008) Active learning methods for interactive image retrieval. IEEE Trans on Image Processing 17(7):1200–1211

Hengel A, Dick AR, ThormŁhlen T, Ward B, Torr PHS (2007) Videotrace: rapid interactive scene modelling from video. ACM Trans Graph 26(3):86

Hoiem D, Efros A, Hebert M (2005) Automatic photo pop-up. In: SIGGRAPH

Hoiem D, Efros AA, Hebert M (2007) Recovering surface layout from an image. IJCV 75(1)

Jain P, Kapoor A (2009) Active learning for large multi-class problems. In: CVPR, pp 762–769

Kapoor A, Grauman K, Urtasun R, Darrell T (2007) Active learning with gaussian processes for object categorization. In: ICCV

Kohli P, Torr PHS (2008) Measuring uncertainty in graph cut solutions. CVIU 112(1):30–38

Kohli P, Nickisch H, Rother C, Rhemann C (2012) User-centric learning and evaluation of interactive segmentation systems. In: IJCV

Kolmogorov V, Zabih R (2004) What energy functions can be minimized via graph cuts? PAMI 26(2):147–159

Kowdle A, Batra D, Chen W, Chen T (2010) iModel: Interactive co-segmentation for object of interest 3d modeling. In: ECCV - RMLE Workshop

Kowdle A, Chang Y, Batra D, Chen T (2011a) Scribble based interactive 3d reconstruction via scene cosegmentation. In: ICIP

Kowdle A, Chang Y, Gallagher A, Chen T (2011b) Active learning for piecewise planar 3d reconstruction. In: CVPR

Kowdle A, Liu H, Hsu S, Lew J, Puri C, Batra D, Chen T (2012a) iModel: Object of interest 3d modeling via interactive co-segmentation on a mobile device. In: Demo session at CVPR

Kowdle A, Sinha S, Szeliski R (2012b) Multiple view object cosegmentation using appearance and stereo cues. In: ECCV

Lafarge F, Keriven R, Brédif M, Hiep V (2010) Hybrid multi-view reconstruction by jump-diffusion. In: CVPR

Lee W, Woo W, Boyer E (2007) Identifying foreground from multiple images. In: ACCV

McGuinness K, O'Connor NE (2012) Toward automated evaluation of interactive segmentation. In: CVIU

Micusík B, Kosecká J (2010) Multi-view superpixel stereo in urban environments. IJCV 89(1):106–119

Pollefeys M, Van Gool L, Vergauwen M, Verbiest F, Cornelis K, Tops J, Koch R (2004) Visual modeling with a hand-held camera. IJCV V59(3):207–232

Pollefeys M, Nistr D, Frahm J, Akbarzadeh A, Mordohai P, Clipp B, Engels C, Gallup D, Kim S, Merrell P, Salmi C, Sinha S, Talton B, Wang L, Yang Q, Stewnius H, Yang R, Welch G, Towles H (2008) Detailed real-time urban 3d reconstruction from video. IJCV 78(2-3):143–167

Saxena A, Sun M, Ng AY (2009) Make3d: Learning 3d scene structure from a single still image. PAMI 31(5):824–840

Scharstein D, Szeliski R (2002) A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. IJCV 47(1-3):7–42

Seitz SM, Curless B, Diebel J, Scharstein D, Szeliski R (2006) A comparison and evaluation of multi-view stereo reconstruction algorithms. In: CVPR

Sinha S, Steedly D, Szeliski R, Agrawala M, Pollefeys M (2008) Interactive 3d architectural modeling from unordered photo collections. SIGGRAPH Asia

Sinha S, Steedly D, Szeliski R (2009) Piecewise planar stereo for image-based rendering. In: ICCV

Sketchup (2000) Google sketchup: `http://sketchup.google.com/`

Snavely N, Seitz S, Szeliski R (2006) Photo tourism: Exploring photo collections in 3d. In: SIGGRAPH

Srivastava S, Saxena A, Theobalt C, Thrun S, Ng AY (2009) i23 - rapid interactive 3d reconstruction from a single image. In: Vision, Modeling and Visualization

Sturm PF, Maybank SJ (1999) A method for interactive 3d reconstruction of piecewise planar objects from single images. In: BMVC

Szeliski R (1993) Rapid octree construction from image sequences. CVGIP: Image Understanding 58(1):23–32

Tang K, Kowdle A, Batra D, Chen T (2009) iScribble, `http://chenlab.ece.cornell.edu/projects/iScribble/iScribble.html`

Vicente S, Rother C, Kolmogorov V (2011) Object cosegmentation. In: CVPR

Vijayanarasimhan S, Jain P, Grauman K (2010) Far-sighted active learning on a budget for image and video recognition. In: CVPR

Yan R, Yang J, Hauptmann A (2003) Automatically labeling video data using multi-class active learning. In: ICCV

Zhou XS, Huang TS (2003) Relevance feedback in image retrieval: A comprehensive review. Multimedia Systems 8(6):536–544