IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received February 8, 2020, accepted March 3, 2020. Date of publication xxxx 00, 0000, date of current version xxxx 00, 0000.

Digital Object Identifier 10.1109/ACCESS.2020.2984745

Automatic Dense Annotation for Monocular 3D Scene Understanding

MD ALIMOOR REZA[1], KAI CHEN[1], AKSHAY NAIK[1], DAVID J. CRANDALL[1], AND SOON-HEUNG JUNG[2]

[1]Luddy School of Informatics, Computing, and Engineering, Indiana University, Bloomington, IN 47408, USA
[2]Electronics and Telecommunications Research Institute, Daejeon 34129, South Korea

Corresponding author: Md Alimoor Reza (mdreza@iu.edu)

This work was supported in part by the Electronics and Telecommunications Research Institute (ETRI) grant funded by the Korean Government (Development of Fundamental Technology for Hyper-Realistic Media Space), under Grant 19ZR1100, and in part by the National Science Foundation under Grant CAREER IIS-1253549.

ABSTRACT Deep neural networks have revolutionized many areas of computer vision, but they require notoriously large amounts of labeled training data. For tasks such as semantic segmentation and monocular 3d scene layout estimation, collecting high-quality training data is extremely laborious because dense, pixel-level ground truth is required and must be annotated by hand. In this paper, we present two techniques for significantly reducing the manual annotation effort involved in collecting large training datasets. The tools are designed to allow rapid annotation of entire videos collected by RGBD cameras, thus generating thousands of ground-truth frames to use for training. First, we propose a fully-automatic approach to produce dense pixel-level semantic segmentation maps. The technique uses noisy evidence from pre-trained object detectors and scene layout estimators and incorporates spatial and temporal context in a conditional random field formulation. Second, we propose a semi-automatic technique for dense annotation of 3d geometry, and in particular, the 3d poses of planes in indoor scenes. This technique requires a human to quickly annotate just a handful of keyframes per video, and then uses the camera poses and geometric reasoning to propagate these labels through an entire video sequence. Experimental results indicate that the technique could be used as an alternative or complementary source of training data, allowing large-scale data to be collected with minimal human effort.

INDEX TERMS Scene understanding, 3D reconstruction, semi-supervised learning, computer vision.

I. INTRODUCTION

Understanding the semantic, three-dimensional structure of the visual world is a fundamental problem in computer vision, with innumerable applications ranging from automatic photo retrieval to autonomous vehicles. A particularly difficult problem is to understand scene content from a single image. When a photograph is taken, the projective transformation ''converts'' a 3d scene into a 2d image, throwing away most explicit cues about depths of points in the scene. Reversing this process — understanding three-dimensional scenes from two-dimensional images — is very difficult, and in fact is mathematically ill-posed, because of the inherent ambiguity in the task.

The associate editor coordinating the review of this manuscript and approving it for publication was Thomas Canhao Xu.

However, humans are often able to infer 3d scene structure from 2d photos, including the identity of objects and approximate 3d layout, even when there is significant occlusion between scene elements. To do this, we use a variety of cues including perspective, relative object size and position, shadows, etc., combined with intuition from a lifetime of experience about the world [1]. Encoding this reasoning into an automatic algorithm has been a long-standing goal of computer vision, but has proven difficult: human-level performance requires not just low-level image cues, but also higher-level semantic cues: identifying objects, reasoning about their typical relationships, applying the laws of nature, etc.

Understanding indoor scenes poses particular problems. Indoor spaces have relatively textureless surfaces such as walls, making it difficult to identify distinctive keypoints for matching or analysis. Moreover, in indoor photos the distance between the camera and scene is usually small, exacerbat-

ing problems with perspective distortion and lens artifacts. On the other hand, reconstruction of indoor scenes can benefit from strong prior information about the world: rooms usually consist of prominent planar surfaces (walls, ceiling, floor) that intersect at 90 degree angles, and rooms of specific types usually contain common objects in certain canonical configurations (e.g., living room with couches, kitchen with table and chairs, etc.).

Recently, progress on problems related to scene understanding, including object recognition and 3d scene layout estimation, has accelerated because of the dramatic success of deep learning on many computer vision problems. Although the exact mechanism for this success is not fully understood, one common hypothesis is that modern deep learning models – especially convolutional neural networks – are particularly adept at capturing regularities of the visual structure of the world. In the context of monocular 3d scene reconstruction, for example, this means that deep neural networks trained on large-scale scene datasets can provide powerful models of the ''prior distribution'' of the real visual world, allowing the networks to produce a plausible 3d model despite the inherent ambiguity of the 2d-to-3d problem.

However, the major disadvantage of these techniques is that they require large-scale training data, typically on the order of tens of thousands to millions of images. Worse than the quantity of imagery, though, is the density of labels needed for many tasks. For example, two critical tasks for understanding scenes are semantic segmentation [2]–[4] — identifying meaningful pixel regions in an image and assigning object or material labels to each of them — and estimating the 3d structure of the scene [5]–[9]. Unfortunately, training modern machine learning-based algorithms for either of these problems requires the extremely labor-intensive process of densely annotating an image pixel-by-pixel, typically by hand. This severely restricts the amount of training data that can be collected for these methods, which means that researchers tend to use training datasets that are convenient instead of the ones that are best suited for a particular problem. This, in turn, limits the performance of these algorithms in real-world applications.

In this paper, we explore how to collect large-scale training data with minimal human interaction for these two tasks: semantic segmentation and 3d room layout from single images. Our approach is to develop a novel algorithm and tool that allows people to provide quick, high-level annotations about the content and geometry of an image. We increase annotation speed by several orders of magnitude by doing this annotation on *video clips* instead of single images. Video is an attractive source of data because a single video may have thousands of frames, showing an environment from many different perspectives as the camera moves around the environment. Moreover, because the frames of a video are correlated, hand-labeled ground truth is less onerous to collect, since annotations can be semi-automatically propagated from one frame to the next. The end result is thousands of individual images with high-quality annotations, but with just a small amount of human labor. We assume that we have RGBD data (from a depth camera) to assist this annotation.

This work builds on previous approaches that have asked humans to label a few keyframes, and then automatically propagate these annotations across the entire video [10]–[12]. We first consider how to annotate semantic segmentation maps. Instead of requiring human annotation, we rely on signals from an object detector [13] applied to various object categories (e.g., bed, tv, etc.). To account for the remaining regions that are not explained by the object detectors, we automatically estimate the 3D layout of the scene, which helps to identify background regions. We then introduce a novel energy minimization-based formulation for solving for dense pixel-level annotations over an entire video. This work is based on preliminary results that were presented at IROS 2019 [14].

We then turn to annotate training data for 3d room layout. Unfortunately, this is not just a simple matter of scanning scenes in 3d: there is a fundamental problem with collecting data from range scanners and fitting 3d models to those point clouds, because the data is sparse and errors are simply unavoidable. We propose a semi-automatic method of estimating the 3d wire-frame or skeleton of an indoor scene. The skeletal structure can be represented as a collection of 3d lines that intersect with each other at junctions such as floor-wall and ceiling-wall intersections. From these structures, high-quality training data for our monocular depth estimation model can be produced.

To summarize, we make the following contributions:

- First, we propose a novel method to densely annotate pixels of an indoor scene for semantic segmentation. Our method combines masks from pre-trained object detectors with the estimated indoor scene layout to explain all the pixels in an image including the background (Figure 1). We formulate the pixel-level annotation in a Conditional Random Field (CRF) energy minimization framework, to use the regularities between successive video frames to produce a consistent annotation over the entire video.
- Second, we propose a novel method that allows human annotators to quickly draw the rough 3d structure of an indoor scene in a few keyframes, and then propagates those layouts automatically across video frames. The annotations required in each keyframe are very sparse and easy to provide (e.g., 2d annotations of line segment endpoints).
- Finally, we show that our automatic annotations can be used to train data-hungry Deep Neural Networks.

II. RELATED WORK

Our semi-automatic annotation tools are tested on two crucial tasks for 3d scene understanding: semantic segmentation, and 3d scene layout estimation. We briefly review work on these two applications, as well as general work related to semi-automatic video annotation.

FIGURE 1. We automatically annotate indoor scenes for training semantic segmentation models. Images (left) are automatically annotated (right) based on off-the-shelf object detectors and a 3D room layout estimator.

A. SEMANTIC SEGMENTATION

Before the advent of Deep Convolutional Neural Networks (DCNNs), semantic segmentation was usually performed bottom-up using hand-engineered features [15]. Deep neural networks have since surpassed these earlier approaches with high accuracy. One successful application of an end-to-end trainable convolutional network for semantic segmentation is the Fully Convolution Network (FCN) of Long *et al.* [2]. This idea was further refined by SegNet [3]. To mitigate the cost of pixel-level annotation, Dong *et al.* [16] recently proposed a few-shot semantic segmentation approach that learns a DCNN model from very few annotated images.

Semantic segmentation is closely related to object detection, which identifies objects in an image along with their locations (typically in the form of bounding boxes). SSD [17], YOLO [18], and Mask R-CNN [13] are popular choices. For example, Mask R-CNN [13] detects objects by first creating regions of interest, performing classification on each region and then using per-class non-maximal suppression to avoid duplicate bounding boxes. For our work, we use Mask R-CNN, since it also provides segmentation masks for detected objects.

Of course, a key challenge with these models is how to collect the densely-annotated training data to permit supervised training. Castrejon *et al.* [19] learned a Recurrent Neural Network (RNN) model that could predict the polygonal vertices encompassing an object inside a cropped RGB image. This method includes an interactive tool to correct prediction errors. EasyLabel [20] is a semi-automatic method for annotating objects on the RGB-D table-top setting. LabelFusion [21] is another semi-automatic method for generating large quantities of semantic labels from RGB-D videos. This method receives user annotations on the 3D reconstruction of the environment, which are then used to propagate the labels across the frames in the RGB-D video. Unlike these methods, we propose a fully automatic method for labeling all pixels — covering a range of categories from small objects to large furniture and background — for all the images in an RGB-D video, as well as annotating the 3d scene structure of the indoor scenes.

B. MONOCULAR DEPTH ESTIMATION

As with semantic segmentation, deep learning has revolutionized the study of reconstructing 3d from single RGB frames: in fact, deep learning has arguably breathed new life into a problem that was too difficult for the traditional techniques that had been deployed before. Wang *et al.* [7] propose an end-to-end deep learning architecture that estimates a 3D shape in the form of a triangular mesh from a single color image. Lee *et al.* [22] combine deep learning with inspiration from traditional techniques based on Fourier analysis for single-image depth estimation. Zhao *et al.* [23] propose a simple feed-forward deep neural network that yields low reconstruction errors when reconstructing 3d from a single image of a 2D object. Laina *et al.* [24] introduces another end-to-end trainable deeper neural network with a reverse Huber loss function for depth estimation from a single RGB image.

While those techniques focus on reconstructing single objects, other work has applied deep learning to reconstruct the layout of an entire indoor scene. Mallya *et al.* [8] propose a box-shaped room layout prediction method by using informative edge maps from an RGB image. Im2CAD [9] was inspired by Roberts' classic Block World [25] paper, and attempts to infer a complete 3D interpretation of a scene photo including the layout of the room by exploiting deep neural network features and leveraging a rich database of 3D CAD models to replicate various indoor objects such as table, chair, bed, etc. LayoutNet [26] proposes a generic framework for room layout estimation from a single RGB image. The proposed architecture follows an encoder-decoder architecture that receives 6-channel input (3-channel RGB and 3-channel Manhattan constraint line-segments). Lee *et al.* [27] propose RoomNet, an end-to-end network that maps monocular RGB room images to keypoint-based room structure images. Their model jointly predicts both scene type and room layout in the form of keypoint (i.e., corners of a room) positions.

Huang *et al.* [28] propose FrameNet, a model to learn a canonical frame from a RGB image, where a canonical frame is represented by three orthogonal directions, one along the normal direction and two in the tangent plane. Dasgupta *et al.* [29] propose a novel method called DeLay for room layout estimation from a single monocular RGB image that uses a CNN model to generate an initial belief map, which is then used by an optimization algorithm to predict the final room layout. This model makes a strong Manhattan World assumption — i.e., that the room is cuboid in shape. Liu *et al.* [30] introduce PlaneNet, a Dilated Residual Network (DRN) to predict plane parameters and corresponding segmentation masks. They are able to produce piece-wise planar and semantically meaningful structures from a single RGB image. A major caveat of this work is the assumption that the number of planes in a room is fixed. This hard constraint has been eliminated in the follow-up work called PlaneRCNN [5], where the detection module can detect any arbitrary number of planes present in a scene.

Of course, deep learning is notoriously data-hungry, and so progress in deep learning for 3d reconstruction has required collecting large labeled datasets for training and testing. Yi *et al.* [31] introduce a large-scale 3D shape understanding

benchmark using data and annotations from the ShapeNet 3D object database [32]. Sun *et al.* [6] introduced a large-scale benchmark (Pix3D) of diverse image-shape pairs with pixel-level 2D-3D alignment. Prior datasets typically contained only synthetic data or lacked precise alignment between 2D images and 3D shapes, but Pix3D has better dataset statistics and better performance in quantitative evaluations. Raw images were collected from web search engines and shapes were collected from 3D repositories, and then the labeled keypoints on the 2D images and 3D shapes were used for the alignment.

Other datasets have been collected for whole scenes instead of just objects, but these datasets typically have many images but lower-quality annotations. For example, the Active Vision Dataset (AVD) [33] contains diverse indoor environment types across multiple geographic locations in the United States. Each video was captured by a camera mounted on a robot which was directed to roam through the rooms inside various apartments. SUN3D [34] consists of thousands of RGBD frames captured across various indoor locations in university campuses and dormitories. Only a fraction of these frames have been manually annotated. Our approach can be applied to any of these RGBD video datasets, allowing us to quickly annotate existing video data with rich annotations on scene structure.

C. OTHER RELATED WORK

Our techniques are related to general work on semi-automatic video labeling. Most of these techniques start with manual annotations of a few keyframes, and then propagate those annotations across the remaining frames using cues such as spatial proximity, optical flow, or 3D reconstruction [10]–[12], [34], [35]. Many of these techniques are similar in nature to those used in object tracking.

Another strategy for dealing with limited training data is to generate synthetic data [36], [37], but a caveat is that deep neural networks trained with synthetic data may not perform well when applied on real-world images. Tsutsui *et al.* [38] found that synthetic training images actually hurt the performance of fine-grained object recognition, but creating learned *mixtures* of synthetic and real images was effective. However, the improvement was quite small and it still requires large-scale labeled training data. While these techniques will continue to improve, a more effective approach in the meantime may be to generate annotated data directly from natural images. In this paper, we address this problem and propose an automatic method for generating annotations from frames of video sequences.

III. AUTOMATIC TRAINING DATA ANNOTATION FOR SEMANTIC SEGMENTATION

We address the problem of automatically annotating all the pixels in a frame from an indoor video without any human annotation. Most pixels in an indoor scene belong to one of two broad categories: *object* or *background*. To automatically annotate all the pixels in an image, we need to find labels for these two different categories. Object detectors allow us to incorporate annotation information for the various specific *object* categories, such as "bed," "chair," "tv," etc. But a large fraction of the pixels in an indoor scene consist of *background* categories such as "wall," "ceiling," "floor," "window," etc. In order to annotate the pixels for these *background* categories not explained by an object detector, we resort to 3D layout estimation of the scene. Information from these two complementary sources is fused together by solving an energy minimization problem in a Conditional Random Field (CRF) framework. Figure 2 shows the pipeline of our methodology. We now describe these components in detail.

A. OBJECT DETECTION

Object detection [17], [18] identifies the *objects* present in an image along with their locations in the form of rectangular bounding boxes. To find a coarse segmentation mask of each detected object, we use the object segmentation method of Mask-RCNN [13]. Figure 3 (top row) shows detection results on images from two different scenes in our experiments. Notice that while the object identifications and boundaries are generally accurate, a large fraction of pixels that are in the *background* are not labeled. We find the annotation information for these image pixels by estimating the structural layout of the scene.

B. 3D SCENE LAYOUT ESTIMATION

The approximate structure of a typical indoor scene consists of a set of 3D planes intersecting with each other. Individual components of these planar structures can typically be labeled as "wall," "floor," "ceiling," etc. Finding and identifying these planes is an open research question, of course — it is part of the motivation behind this paper, since we need to collect more high-quality training data to produce better scene layout estimators. To break this chicken-and-egg problem, we used a traditional technique not based on deep learning, and thus less sensitive to shifts in application or context. After experimenting with various of these, we settled on the approach of Taylor *et al.* [39], which estimates the structure of the scene by first finding 3D planes utilizing the depth channel from an RGB-D image, and then assigns labels to each plane based on its estimated normal. The plane aligned to the gravity direction is labeled as "floor," the plane orthogonal to the "floor" is labeled "wall," and the remaining portion of the layout is labeled as "ceiling." Figure 3 shows the estimated scene layout components for two sample images.

C. SUPERPIXELS

An image superpixel is a set of contiguous pixels that share homogeneity in appearance, texture, etc. [40]–[42]. A superpixel generation algorithm partitions the image into a reduced number of segments, thereby speeding up the work of subsequent processing which can process partitions instead of individual pixels. Reza *et al.* [10] generated high-quality

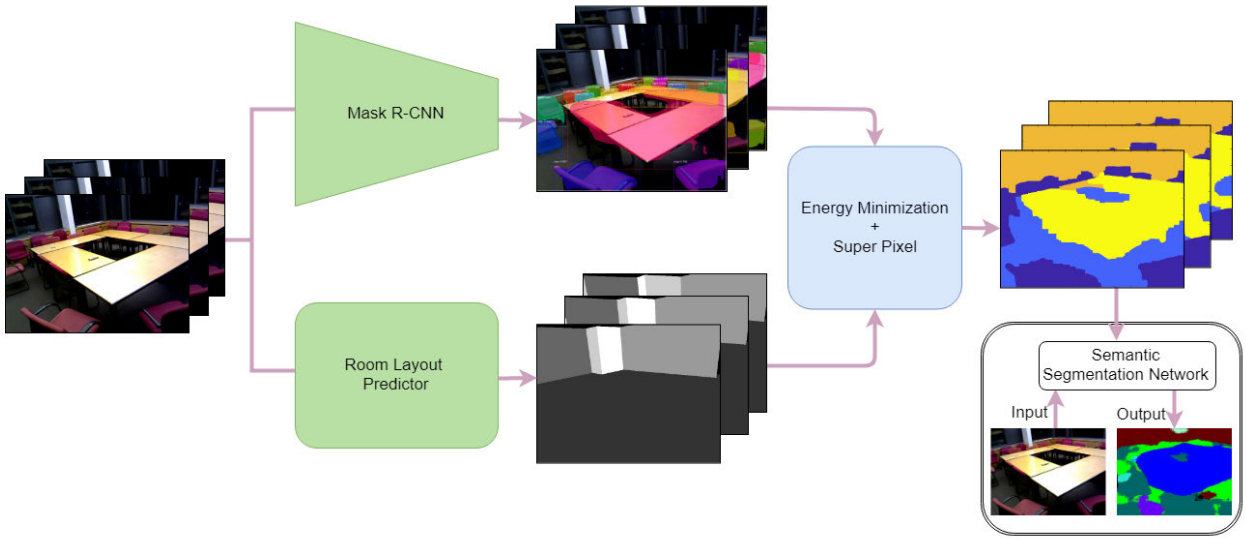


FIGURE 2. For each video frame, we identify candidate object masks using pre-trained object detectors (top branch). The pixels not explained by the detector are estimated from 3d scene layout (bottom branch). This evidence is combined in an energy minimization framework to estimate our final annotation.

FIGURE 3. Sample detection and 3D room layout results from two different scenes: *Studyroom* (Left) and *MIT-32* (Right) from SUN3D [34]. Detector outputs (top) from Mask RCNN [13] provide an initial coarse segmentation around detected objects, while 3D layout estimation (below) explains background categories including "wall," "floor," and "ceiling."

superpixels, but relied on an expensive image-contour generation process that can take several minutes per image. In contrast, we follow a simpler and more efficient alternative, SLIC (Simple Linear Iterative Clustering) [41], which can generate superpixels in less than a second. Figure 4(b) shows superpixel boundaries overlaid on an image from our experiments. We use our superpixels as atomic units to incorporate annotation information from our two complementary sources of evidence, object detection and 3d scene layout estimation.

D. PIXELWISE ANNOTATION

We assume that we are given a video sequence consisting of frames $\{I_1, I_2, \ldots, I_N\}$. For a given unannotated frame I_k, we would like to minimize,

$$E(X_k|I_k, I_{k-1}, I_{k-2}, I_{k-3}) = \sum_{i\in V} \theta_i(x_i; I_k) + \sum_{i\in V} \phi_i(x_i; I_{k-1}, I_{k-2}, I_{k-3}) + \sum_{(i,j)\in\zeta} \psi_{ij}(x_i, x_j; I_k), \quad (1)$$

where $\theta_i(.)$ and $\phi_i(.)$ are the *unary* energy functions and $\psi_{ij}(.)$ is the *pairwise* function. The CRF graph $G = (V, \zeta)$ is defined over the pixels in the image I_k and 4-connected neighbors. We use the 3 frames immediately preceding I_k, namely I_{k-1}, I_{k-2}, and I_{k-3}, and exploit their unaries computed earlier by transferring them into the current frame using optical flow. This ensures temporal smoothness in finding the annotation for the current frame.

UNARY TERMS

From the detector output, we obtain a set of detected *object* masks along with their labels. For the *background* category, the predicted layout mask intersects with almost the entire image. We assign a fixed score to all the pixels that overlap with our various *background* categories (such as "wall," "floor," "ceiling," etc.). Figure 3 shows detection masks in different colors along with their label on the top-left corner of each bounding box. We find the intersection of a mask with a superpixel, and within each superpixel distribute the same score to all the pixels.

More specifically, we compute our first unary term,

$$\theta_i(x_i; I_k) = -f(x_i; I_k), \quad (2)$$

where $f(.)$ is a score for the pixel i computed by the superpixel that engulfs it. For each superpixel, we count the fraction of pixels that overlap with the detection mask of object a_j. As an

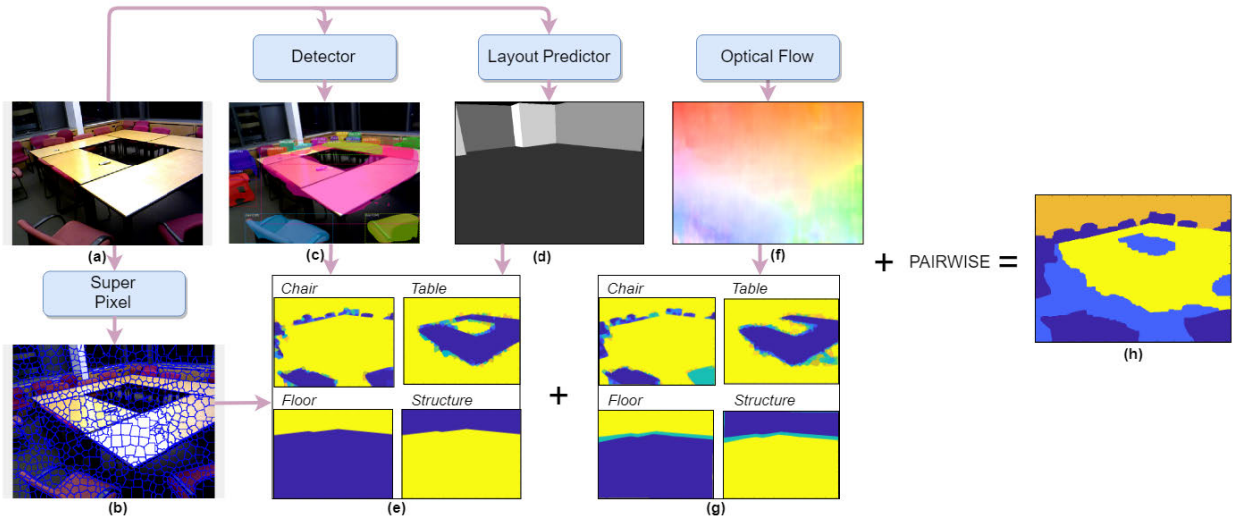


FIGURE 4. Visualization of our energy minimization formulation. (a) For each frame, we (b) identify candidate object segmentation masks from pre-trained object detectors [13]. (d) The remaining pixels are estimated from the layout of the scene [39]. These are combined via energy minimization to estimate our final annotation (h). In addition to a unary term (e) from the current frame, we incorporate a second unary (g) that encodes evidence from previous frames, using optical flow as shown in (f).

FIGURE 5. Detection masks on three successive frames in a video sequence. Notice that the detector fires inconsistently on the same instance of "chair" object category. Our formulation can handle this noise with a unary term $\phi(.)$ that encourages temporal consistency across frames.

example, if a detection mask from the "chair" category completely overlaps with a superpixel, then $f(.)$ assigns a score of 1.0 for "chair" category. Figure 4(b) shows an example of our unary energy term for different annotation categories.

Our second unary term is,

$$\phi_i(x_i; I_{k-1}, I_{k-2}, I_{k-3}) = -g(x_i; I_{k-1}, I_{k-2}, I_{k-3}), \quad (3)$$

where $g(.)$ is another scoring function based on the unary energy terms for the three frames immediately preceding frame I_k, in particular taking the average of the unary energy terms from the frames I_{k-1}, I_{k-2}, and I_{k-3} by transferring them into frame I_k using optical flow. Figure 5 shows a situation that demands this temporal consistency for finding the correct annotation.

PAIRWISE TERM

To encourage smoothness, we adopt a simple Potts model for our pairwise energy function, which penalizes adjacent pixels having different annotations,

$$\psi_{ij}(x_i, x_j; I_k) = \begin{cases} 0, & x_i = x_j \\ b, & x_i \neq x_j, \end{cases} \quad (4)$$

where b was empirically set to 0.5 for all our experiments.

Equation (1) is minimized using Graph Cuts [43] inference. A summary of the steps for finding the automatic annotation for an image is shown in Figure 4.

IV. VIDEO GEOMETRIC LABEL GENERATION

We now turn to generate data for our second problem of key importance in scene understanding: automatic 3d layout estimation from single 2d images. Our approach is to develop a novel algorithm and tool that allows humans to provide quick, high-level annotations about the geometry of an image, and then use those annotations to fit a planar room layout structure to noisy, 3d depth maps. When the RGBD data is a video from a moving camera of a stationary scene, our approach is able to propagate the annotations across time to unlabeled frames, thus reducing the amount of human labor involved by several orders of magnitude. We also assume that the camera poses of the individual video frames are available as a prior, which could be estimated from the standard Structure from Motion (SfM) pipeline [44].

In particular, we propose a semi-automatic method of estimating the 3D layout of indoor scenes, in the form of a wire-frame or skeleton. The skeletal structure can be represented as a collection of 3d lines that intersect with each other at junctions such as floor-wall and ceiling-wall intersections. Our goal is to semi-automatically estimate this wire-frame structure of the indoor scene for all the frames in an RGB-D video, in order to produce high-quality training data for our monocular depth estimation model. We estimate the 3D wire-frame structure of a scene in two stages: (i) Corner point annotation in a few keyframes and ii) Layout estimation for the entire video utilizing the annotated keyframes.

TABLE 1. Quantitative evaluation of our proposed automatic annotation method. First and second items in each entry denote evaluation metrics *average per-class* and *average IoU* respectively. The last column reports mean across the categories in each video (row wise). The bottom row shows the mean across video sequences for each category (column wise)

Video	Bed	Ceiling	Chair	Floor	Furniture	Props	Structure	Table	TV	Mean across category
hotel-umd	81.9 / 60.0	60.6 / 33.6	51.3 / 39.0	56.7 / 37.3	12.9 / 05.6	21.9 / 10.6	66.6 / 59.1	—	54.4 / 52.7	50.8 / 37.2
hv-c5	—	0 / 0	77.6 / 66.9	83.8 / 49.9	0 / 0	64.0 / 05.7	81.7 / 76.4	84.8 / 80.2	—	56.0 / 39.9
studyroom	—	0 / 0	74.8 / 64.8	74.2 / 59.6	36.0 / 31.0	23.9 / 07.5	87.5 / 70.9	48.0 / 45.4	—	49.2 / 39.9
mit-32	—	—	72.4 / 66.0	91.5 / 73.5	—	35.6 / 09.6	69.9 / 62.1	59.6 / 55.4	—	65.9 / 53.3
hv-c6	—	—	77.5 / 68.0	70.2 / 39.7	0 / 0	23.9 / 04.3	87.5/ 84.2	84.4 / 76.8	—	57.2 / 45.5
hv-c8	—	18.5 / 10.9	79.5 / 10.9	95.7 / 68.6	0 / 0	70.5 / 07.1	74.9 / 73.4	77.0 / 74.4	—	59.4 / 44.3
dorm	85.7 / 84.0	49.9 / 42.6	96.8 / 73.4	89.7 / 06.9	14.9 / 08.6	47.1 / 35.5	69.3 / 58.5	47.7 / 44.5	—	62.6 / 44.3
mit-lab	—	0 / 0	99.2 / 75.3	78.2 / 68.9	99.8 / 54.4	18.1 / 15.9	80.5 / 77.0	88.9 / 38.5	—	66.4 / 47.1
Mean across video sequence	**83.8 / 72**	21.5 / 14.5	**78.6 / 58.0**	**80 / 50.6**	23.4 / 14.2	38.1 / 12.0	**77.2 / 70.2**	**70.1 / 59.3**	54.4 / 52.7	—

A. CORNER POINT ANNOTATION IN KEYFRAMES

We need hand-labeled annotations for just a fraction of frames (less than 5 out of a thousand) for an RGB-D video, and we design this annotation process in a way such that they can be collected quickly and easily. We first ask the user to watch a video clip of video collected from a moving camera of an indoor scene, and to identify around 10-12 frames that collectively (roughly) cover all parts of the scene. We then ask the user to annotate each of these frames by clicking on the two endpoints of all visible 2D lines. In particular, we ask the user to (1) annotate horizontal and vertical line segments in 2D image space that are part of the wire-frame skeleton of the scene, and (2) verify that each line, when extended, intersects with another line (vertically or horizontal) that is part of the wire-frame skeleton.

Figure 6 shows a sample annotation of a scene from our experiments. We utilize these partially-annotated keyframes in the subsequent stage to estimate the layout of the entire RGB-D video. Some scenes are heavily cluttered and occluded, hence only a small 2D line segment might be visible in the scene. We address these limitations by inferring the extent of the entire line in the next stage of our layout estimation algorithm.

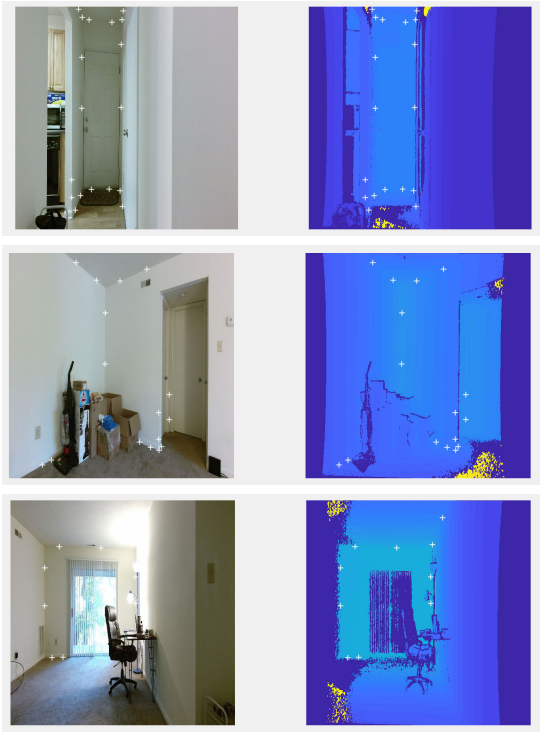


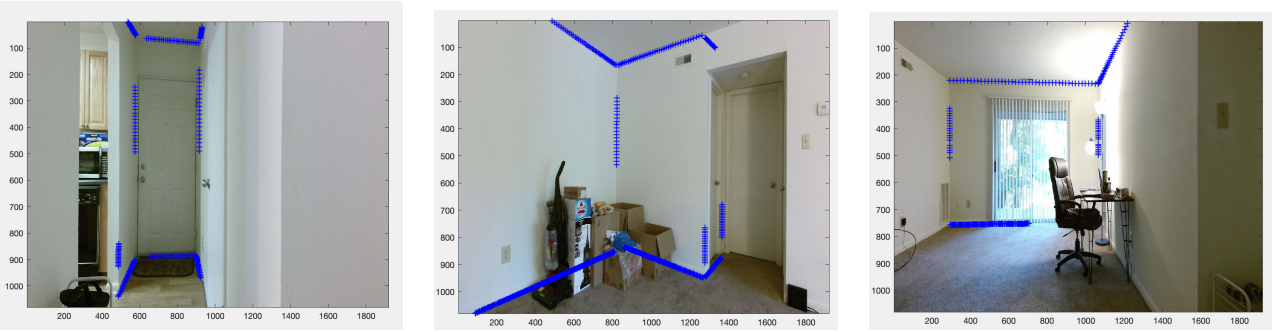
FIGURE 6. Annotated endpoints of small 2D line segments are visualized as white '+' signs in three manually-selected keyframes (best viewed in color). These endpoints are annotated in pairs. Notice these annotations are unlabeled, i.e., it is not known whether a line associated with a pair is *vertical* or *horizontal*. RGB (left) and depth (right) images are shown for each keyframe in each row.

B. CANDIDATE LAYOUT ESTIMATION IN THE KEYFRAMES

We estimate the initial layout on these partially-annotated keyframes by extending the 2D line segments until they reach either the boundary of the image or a visible intersection (e.g., corner point in the room). We then find the 2D line equations associated with each 2D line-segment in image space and then find all pairs of intersections between these 2D lines as our initial hypothesis of the layout. Then for each line segment, we extend it both directions to find the intersection that is closest to the line. Since the depth channel is very noisy and a portion of the space has missing depth information, we need to infer missing 3D points and, as a consequence, our algorithm needs to move back and forth from the 2D line equation to the 3D line equation. Once the layouts are estimated in the keyframes, we transfer these layouts to the rest of the frames using the camera poses and 3D point clouds.

C. LAYOUT PROPAGATION FROM THE CANDIDATE LAYOUTS

The scene layout for each of these keyframes, represented as 3D lines, are projected into unlabeled contiguous frames in an interactive process. The projections of these multiple layouts are typically not aligned due to camera pose estimation errors in the new frame. To address this problem, we first perform some data association steps on individual 3D lines. Then, the final layout is estimated by reasoning on 3D line to line intersections on these resulting data associations.



(a) (b) (c)

FIGURE 7. Initial layout in three keyframes from the 2D point manual annotations. The points corresponds to vertical 3D lines projected in 2D image space.

In more detail, the process for generating the final layout is described in the following several steps.

1) VERTICAL 3D LINE ASSOCIATION

The vertical 3D lines that form the initial layout in all the annotated keyframes are transferred to the target frame for which we need to estimate the scene layout. The 3D points associated with these vertical 3D lines are projected into the current frame's camera coordinate space using the world-to-camera transformation matrix.

To do this, we first find the groups of vertical 3D lines that are close to each other. We construct a graph $G = (V, E)$, with a vertex associated with each 3D line. We add an edge in this graph if one 3D line is reachable from another. More precisely, we compute a form of adjacency matrix in this graph which we refer to as the *reachable matrix,* R^{3d}. Whether a line is reachable from another is decided based on the average distance between the two lines. For example, we compute the distance (in 3D) from 3D line l_i to 3D line l_j and vice versa. If these two distances are within a threshold, we set R^{3d}_{ij} to 1, and otherwise set it to 0.

Once the reachable matrix R^{3d} is computed, we find the connected components on the graph G. Each component represents the set of 3D lines that corresponds to the same 3D line coming from the annotated keyframes. We accumulate all the 3D points associated with each 3D line in a group, and then estimate a single 3D line from these accumulated 3D points using a RANSAC-based line fitting algorithm. The same process is repeated for all the components in the graph G. This data association step allows us to find a final set of 3D vertical lines, which is subsequently used to form the final layout.

2) HORIZONTAL 3D LINE ASSOCIATION

Similar to the vertical case, the horizontal 3D lines that form the initial layout in all the annotated keyframes are also transferred to the current frame. The 3D points associated with these horizontal 3D lines are projected into the current frame's camera coordinate space using the world-to-camera transformation matrix.

We then adopted a similar data association approach in finding the final set of horizontal 3D lines. Vertical lines, in general, follow an orientation that is towards the gravity direction in an indoor scene, e.g., vertical edges of a door, vertical edges of a window, edges in between two wall intersections, etc. Unlike the vertical 3D lines, the horizontal 3D lines are oriented in several directions. We split the horizontal 3D lines data association in two steps instead of directly associating them. First, we separate the horizontal 3D lines based on their orientation so that those 3D lines that are oriented towards the same direction are grouped together. Second, we take all the horizontal 3D lines in each orientation-group in turn, and then follow a 3D distance (from one line to another) based association as used in the vertical 3D line data association.

To illustrate the process, assume that the horizontal 3D lines are oriented towards either the *Z-axis* or the *X-axis* in the current frame's camera coordinate space. Our algorithm separates the horizontal 3D lines in two different groups based on their orientations in the scene. To separate the horizontal 3D lines based on their orientations, we adopt a similar graph construction procedure as is used in the vertical 3D line association. For computing the edges in the graph $G = (V, E)$ that encode adjacency information between the lines, we compute the angular similarity instead of 3D distance between the lines. We compute a form of adjacency matrix in this graph which we refer to as $H^{angular}$. Once $H^{angular}$ is computed, we find the connected components from this matrix using a Depth First Search (DFS). Each component identifies the set of horizontal 3D lines that are oriented towards the same direction.

For all the horizontal 3D lines in a single connected component, we associate them using a 3D distance-based data association as used in our vertical data association. Assume there are M connected components. We construct a set of adjacency matrices $\{H^{3d}_1, H^{3d}_2, \ldots, H^{3d}_M\}$, where each H^{3d}_k represents the adjacency matrix for the 3D distance-based horizontal 3D line association. For each H^{3d}_k, we again compute the connected component, where each component represents all the annotated horizontal 3D lines that come from different

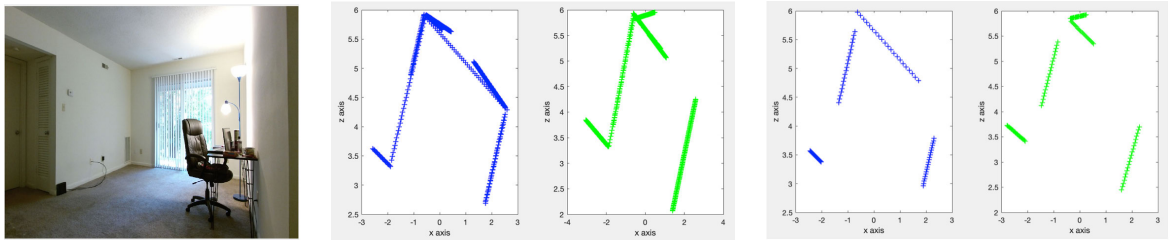


FIGURE 8. The process of the horizontal 3D lines association in a given frame (best viewed in color). The horizontal lines are shown in a bird-eye view in two different XZ planes: i) ceiling XZ plane (in blue color) and ii) floor XZ plane (in green color). (a) shows the original RGB image of a frame, (b) shows all the horizontal lines before they are associated together. In the left panel, blue represents the horizontal lines on the ceiling XZ plane, and in the right panel, green denotes the horizontal line on the floor XZ plane, and (c) a set of lines each fitted with RANSAC.

keyframes. Like the vertical 3D line data association method, we accumulate all the 3D points associated with each horizontal 3D line in a component computed from H_k^{3d}. We estimate a single 3D line from these accumulated 3D points using a RANSAC-based line fitting algorithm. We execute our horizontal 3D line association in the two XZ planes: a) ceiling XZ plane and b) floor XZ plane separately. We determine this partitioning of horizontal 3D lines based on distance offset. Figure 8(a) shows a sample frame in question in which we want to propagate the layout. Figure 8(b) shows the horizontal 3D lines projected on the ceiling XZ plane (left) and on the floor XZ plane (right). Finally, Figure 8(c) shows the set of RANSAC-fitted lines after the data association.

These fitted horizontal lines – in conjunction with the fitted vertical lines – are used to estimate the final scene layout for the current frame. One advantage of partitioning the horizontal lines using orientation first is that it permits our algorithm to estimate the layout for a scene with arbitrary shapes. In other words, our algorithm is not restricted to environments that follow the "Manhattan-World" assumption that the planes in a scene are oriented towards one of the three orthogonal vanishing directions: our algorithm can find the layout for a more general rooms layouts including pentagonal, hexagonal, etc.

3) COMBINE THE 3D LINES FOR FINAL LAYOUT ESTIMATION

Once we find the set of RANSAC-fitted horizontal and vertical 3D lines in the current frame's camera coordinate space, we combine them to estimate our final scene layout for the frame. We trace extensions of the layout in the ceiling XZ plane and the floor XZ plane separately. The steps of finding boundaries of the layout in the ceiling XZ plane are as follows. First, we project the 3D horizontal lines pertaining to the ceiling on the XZ plane as shown in Figure 9(a). Then, we project all the vertical 3D lines in the XZ plane. We find the mean of these projected vertical lines (shown by the red dots in Figure 9(a)). We refer to these points as vertical junctures.

We also compute all pairs of intersections among the projected lines in the XZ plane (shown by the blue dots in Figure 9(a)). For each intersection of a pair of horizontal lines, we find the closest vertical junctures. These vertical junctures act as boundaries for horizontal lines to limit their extensions. We extend each horizontal line until its boundary limits if both ends of the line have associated vertical junctures. If any side of a horizontal line is not limited by a vertical juncture, then we extend that horizontal line until it reaches the image boundary.

Similarly, we project the 3D horizontal lines pertaining to the floor on the XZ plane as shown in Figure 9(c), and then trace their boundaries as shown in Figure 9(d). Once we have the extensions of the horizontal lines in both floor and ceiling XZ planes, we render their projections in the 2D image space to define the final layout. The vertical lines are extended in both directions until they reach the ceiling and the floor. Figure 9 illustrates these steps of the final layout estimation.

V. EXPERIMENTS

A. SEMANTIC LABEL GENERATION

We experimented on the eight RGB-D video sequences from SUN3D [34] to validate our automatic annotation approach. Table 2 shows statistics for the eight video sequences. Each video consists of thousands of frames captured across various indoor locations in university campuses and dormitories. Only a fraction of these frames have been manually annotated. We validate the automatically generated annotations from our approach on these frames.

We used 10 categories, including both fine-grained (*Bed, Chair, Table, TV, Floor, Ceiling*) and generic categories (*Props, Furniture, Structure*). We conform to this selection based on the labeling criteria laid out by the popular indoor scene understanding dataset NYUD-V2 [45].

We used an open-source implementation of Mask-RCNN [46] pretrained on MS COCO [47] as our object detector. MS COCO consists of 80 categories commonly found in both indoor and outdoor scenes; we selected only the indoor object categories. We mapped categories of MS COCO to categories used in our experiments, as shown in Table 5.

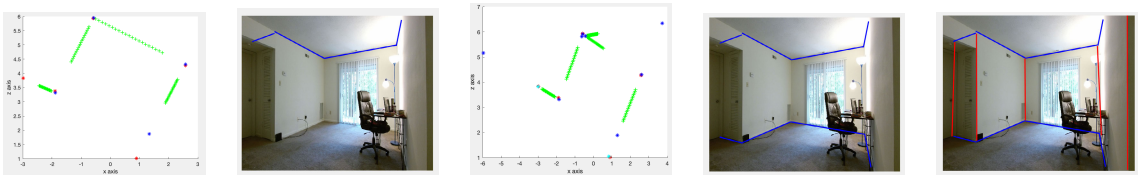


FIGURE 9. An illustrative example of the final layout estimation (best viewed in color). (a) Projection of the horizontal lines in the ceiling XZ plane. (b) Projected 2D points of the estimated ceiling boundaries of the layout. (c) Projection of the horizontal lines in the floor XZ plane. (d) Projected 2D points of the estimated layout floor boundaries also augmented in the image space. (e) Final layout including the vertical lines that are extended until they reach both XZ planes (ceiling and floor).

TABLE 2. Statistics of 8 video sequences in SUN3D [34]

Scene	# Frames	# of Human-annotated Frames
hotel-umd	1869	82
hv-c5	2063	24
studyroom	3322	49
mit-32	5444	109
hv-c6	961	36
hv-c8	1003	23
dorm	2675	58
mit-lab	1906	14

For 3D scene layout estimation, we used the implementation by the author of [39]. The estimated layout provides a single mask for *floor* and *ceiling* categories, and the remaining layout is represented as series of other masks such as *Wall, Office-partition, Door,* etc. We map these categories to a generic *Structure* category as in NYUD-V2 [45].

To measure the performance of our automatic annotation, we used two metrics: *per-class accuracy:* for each class, find the proportion of correctly-labeled pixels, and *per-class IoU:* for each class, compute the ratio of the size of the intersection of ground truth label and estimated label regions, and the size of the union between the ground truth and estimated label.

1) AUTOMATIC ANNOTATION RESULTS

We validated the annotations generated automatically by our method against the ground truth labels manually prepared by a human in each video sequence. As the manual annotation is laborious and expensive, each video sequence has only a small fraction of the frames manually labeled (as shown in Table 2). This is exactly the motivation for our work: we can generate automatic annotations for all the frames in a video sequence, allowing a larger quantity of annotations with minimal human effort.

The results of our evaluation using the two metrics defined above are shown in Table 1. The table evaluates for each individual category as well as the average across categories (last column). To evaluate the category specific performance across all the videos, we also report an aggregated mean in the last row. Each entry in the table lists two numbers: *per-class accuracy* and *per-class IoU*. A missing entry signifies that the object is not present in that video (e.g., *TV* is present only in *hotel-umd*).

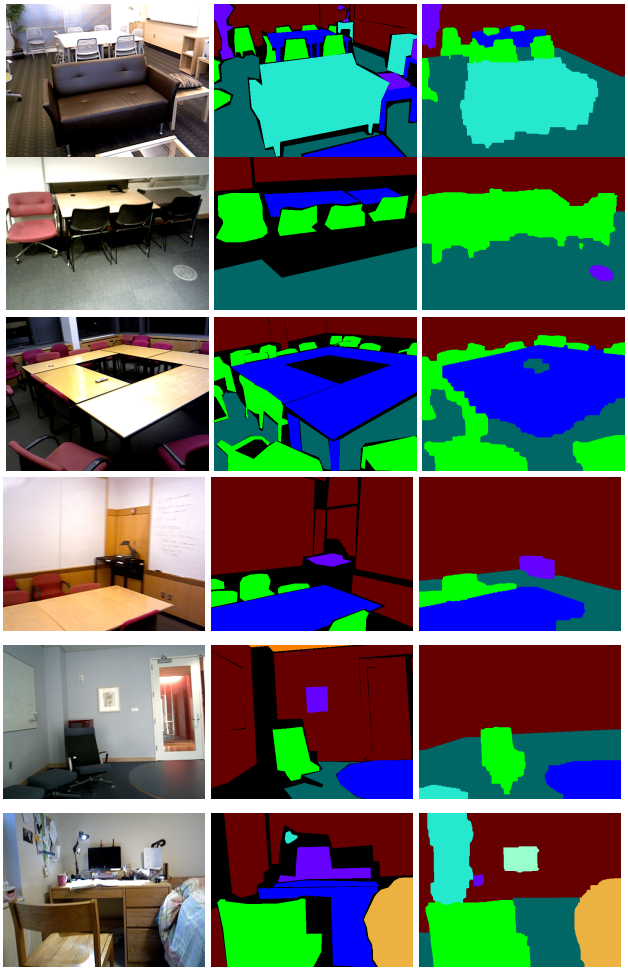


FIGURE 10. Qualitative results for automatic annotation experiment on different video sequences from SUN3D [34]. From left to right we show the RGB image, the ground truth, and the automatic annotations from our method.

Our automatic annotation method performs well on object categories such as *Chair, Table,* and *Bed*, presumably because Mask-RCNN trained on MS COCO [47] has modeled these categories well. Some qualitative visualizations are shown in Figure 10. As we notice, our method can reliably annotate *chair, table, bed* categories in most cases. Our method had weaker performance on the generic object categories such as *Props* and *Furniture*. Our method solely relies on the

TABLE 3. Semantic segmentation performance comparison. First and second items in each table entry denote metrics *average per-class* and *average IoU* respectively. Last column shows the aggregated performance across all 8 classes

Train Set	Bed	Ceiling	Chair	Floor	Furniture	Props	Structure	Table	Average value
GT	39.3 / 33.6	68.5 / 54.7	73.3 / 40.2	66.3 / 39.4	16.0 / 10.5	15.4 / 12.0	84.2 / 71.8	88.3 / 74.2	56.4/42.0
Auto	1.2 / 1.0	11.3 / 9.5	53.1 / 21.8	72.0 / 15.7	0.9 / 0.7	4.0 / 2.9	70.8 / 53.9	67.6 / 55.3	35.1/20.1
GT + Auto-sample	15.1 / 13.4	33.8 / 30.7	**81.0** / 38.1	**73.3** / 22.9	10.5 / 7.0	6.4 / 5.4	78.3 / 61.9	**89.6** / 60.2	48.5/30.0

TABLE 4. Videos used in our experiments from Active Vision Dataset [33]

Video name	# Total frames	# hand-annotated keyframes
Home 001	1536	11
Home 002	1224	10
Home 003	1500	21
Home 004	1488	14
Home 005	744	12
Home 006	2412	15
Home 007	1728	13
Home 008	840	14
Home 009	1320	15
Home 010	1548	17
Home 011	1644	15
Home 013	696	14
Home 015	972	16
Home 016	1128	13

signals from our object detectors to capture the annotation information; when a detector consistently fails to detect an object across a video sequence, our method fails to annotate that object. Our method also captures the annotations for *Floor* and Structure categories since our layout estimation can retrieve the structure of almost all of the scenes from the RGB-D images.

2) SEMANTIC SEGMENTATION WITH AUTOMATIC ANNOTATIONS

Since our goal is to generate automatic annotations that would be useful for training deep semantic segmentation models, we evaluated our technique as a means of generating ground truth labels for FCN [2] for the 10 object categories mentioned above. We partitioned the 8 videos of SUN3D into 4 for training and 4 for testing. The training video sequences include *hotel-umd, hv-c5, studyroom, mit-32* which have a total of 264 human-annotated keyframes. Our test partition, *hv-c6, hv-c8, dorm, mit-lab*, has 131 human-annotated frames in total. We use all the 264 training frames along with their ground truth labels to train a FCN model, which we refer to as *GT*. We then automatically generated annotations for these frames using our method, and used them to train another FCN model, which we call *Auto*. Both models were trained for 60,000 iterations with learning rate $1e^{-}5$ and cross-entropy loss.

Quantitative results are shown in Table 3 (excluding *TV* which is absent in the test partition), with the first value in each entry indicating per-class accuracy and the second indicating IoU accuracy. Qualitative results are shown in Figure 11. The average per-class accuracy for *GT* is 56.4% and average IoU accuracy is 42.0%. The average per-class accuracy of the model *Auto* is 35.1% and average IoU accuracy is 20.1%. Of course, this is to be expected: *GT* was trained on laboriously hand-labeled training data, whereas *Auto* required no human annotation whatsoever. *Auto* performs well on categories such as *chair, floor, structure* and *table*, although not as well as the *GT* model. Additionally, we observe that both models do not perform well on categories such as *furniture* and *props*, as SUN3D has very few instances of these categories, making it difficult for the segmentation network to learn a reasonable model even with perfect ground truth annotations.

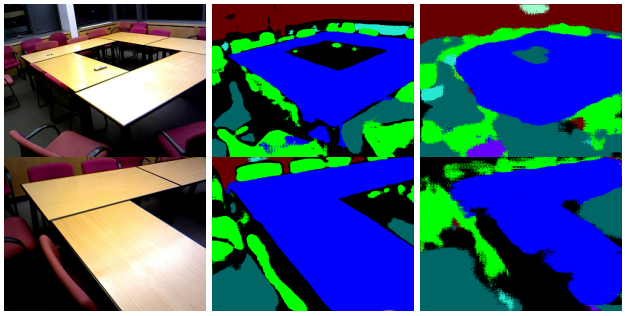


FIGURE 11. Qualitative comparison for semantic segmentation on the images on test set (left) when trained on human annotated (middle) vs automatic annotated (right).

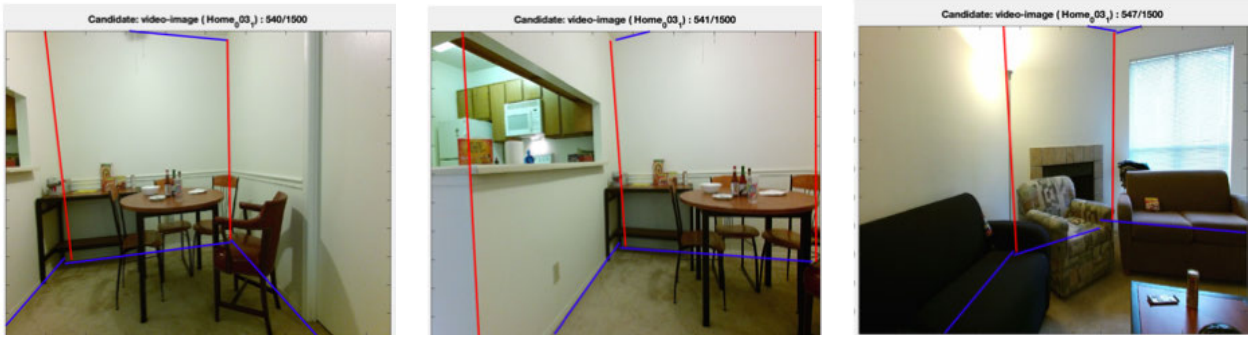
To further understand the effectiveness of our automatically-generated annotation, we trained another model, *GT + Auto-sample* (last row in Table 3), by adding more samples of automatically-annotated frames to the existing 264 human-annotated training frames. More specifically, *Auto-sample* was prepared by sampling the automatic annotation of every 15-*th* image in each training video, resulting in a total of 838 automatic annotated frames. Although the overall performance of *GT + Auto-sample* is inferior compared to *GT* (average per-class and IoU are 48.5% and 30.0% respectively), we observe performance improvements for some categories such as *Chair, Table*, and *Floor*. These three belong to the classes for which our automatic annotation method performed well (as reported in Table 2 and also discussed in Section V-A.1).

B. VIDEO GEOMETRIC LABEL GENERATION RESULTS

We evaluated our semi-automatic geometric annotation algorithm primarily on the Active Vision Dataset (AVD) [33], which contains diverse indoor environment types across

TABLE 5. Mapping of MS COCO [47] categories to indoor scene categories for our automatic annotation approach

MS COCO	bed	dining table	chair	tv	oven	refrigerator	toilet	couch	bench	microwave	sink	book	remote	clock	bowl	wine glass	hair drier	fork	toothbrush	spoon	knife	tie	suitcase	laptop	mouse	keyboard	toaster	vase
Our Approach	bed	table	chair	tv	structure	furniture	furniture	furniture	furniture	prop	prop	prop	prop	prop	prop	prop	prop	prop	prop	prop	prop	prop	prop	prop	prop	prop	prop	prop
MS COCO	vase	cell phone	cup	handbag	bottle	cake	potted plant	pizza	person	scissors	sports ball	frisbee	umbrella	banana	apple	teddy bear	donut	skis	snowboard	kite	baseball bat	baseball glove	skateboard	tennis racket	sandwich	orange	broccoli	hot dog	
Our Approach	prop	prop	prop	prop	prop	prop	prop	prop	prop	prop	prop	prop	prop	prop	prop	prop	prop	prop	prop	prop	prop	prop	prop	prop	prop	prop	prop	prop	



(a) (b) (c)

FIGURE 12. Final estimated layout in three unannotated frames. The blue lines are vertical 3D lines projected in 2D image space, while the red lines are similarly found vertical lines.

multiple geographic locations in the United States. Each video was captured by a camera mounted on a robot that was directed to roam through the rooms inside the apartment. We experimented with 14 videos from AVD where each video contains between approximately 700 and 2500 frames, for a total of 18780 frames. Table 4 reports the detailed statistics of the videos that are used in our experiments. We performed the manual point-level annotation for all 14 videos, and so far have applied the annotation algorithm on ''Home 001'',''Home 003'', and ''Home 011''. We show sample final layouts in some frames in Figure 12.

Our algorithm is flexible and applicable to generate annotations from other indoor data sources such as the SUN3D dataset [34] and the GMU Kitchen dataset [48]. There are 8 videos in SUN3D dataset containing 19243 frames in total, and it is a suitable a dataset for our semi-automatic annotation algorithm. GMU Kitchen contains 6735 frames from 9 videos. These videos from SUN3D and GMU kitchen datasets were captured in different types of indoor environments ranging from apartment, classroom, and office, and thus could be additional sources of training data generation using our algorithm.

VI. CONCLUSION

In this work, we presented a method for generating annotations for creating training data for two indoor scene understanding tasks, semantic object segmentation and 3d room layout estimation, using minimal human intervention. For semantic object segmentation, our method is fully-automatic and relies on two complementary sources of evidence: pre-trained object detectors and rough scene layout estimators. For 3d room layout, we proposed a semi-automated technique that requires a human operator to provide just a few key annotations for a handful of keyframes of an RGBD video, and then the dense room layout is automatically estimated and propagated across time to the unlabeled frames. These methods offer an alternative technique for generating a large quantity of dense pixel-level annotations for training data-hungry deep neural network models. In the future, we plan to augment the method to generate annotations for a large number of fine-grained indoor object categories. We also plan to explore the feasibility of our approach in the outdoor setting.

ACKNOWLEDGMENT

Kai Chen participated in this project while visiting Indiana University through the IU Global Talent Attraction Program (GTAP) program. All work was conducted while the authors were at Indiana University.

REFERENCES

[1] A. Anzai and G. C. DeAngelis, ''Neural computations underlying depth perception,'' *Current Opinion Neurobiol.*, vol. 20, no. 3, pp. 367–375, Jun. 2010.

[2] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 3431–3440.
[3] V. Badrinarayanan, A. Kendall, and R. Cipolla, "SegNet: A deep convolutional encoder-decoder architecture for image segmentation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 12, pp. 2481–2495, Dec. 2017.
[4] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, "DeepLab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected CRFs," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 40, no. 4, pp. 834–848, Apr. 2018.
[5] C. Liu, K. Kim, J. Gu, Y. Furukawa, and J. Kautz, "PlaneRCNN: 3D plane detection and reconstruction from a single image," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 4450–4459.
[6] X. Sun, J. Wu, X. Zhang, Z. Zhang, C. Zhang, T. Xue, J. B. Tenenbaum, and W. T. Freeman, "Pix3D: Dataset and methods for single-image 3D shape modeling," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 2974–2983.
[7] N. Wang, Y. Zhang, Z. Li, Y. Fu, W. Liu, and Y.-G. Jiang, "Pixel2mesh: Generating 3d mesh models from single rgb images," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2018, pp. 52–67.
[8] A. Mallya and S. Lazebnik, "Learning informative edge maps for indoor scene layout prediction," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Dec. 2015, pp. 936–944.
[9] H. Izadinia, Q. Shan, and S. M. Seitz, "IM2CAD," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 5134–5143.
[10] M. A. Reza, H. Zheng, G. Georgakis, and J. Kosecka, "Label propagation in RGB-D video," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Sep. 2017, pp. 4917–4922.
[11] V. Badrinarayanan, F. Galasso, and R. Cipolla, "Label propagation in video sequences," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, Jun. 2010, pp. 3265–3272.
[12] S. Mustikovela, M. Yang, and C. Rother, "Can ground truth label propagation from video help semantic segmentation?" in *Proc. ECCV Workshop Video Segmentation*, 2016, pp. 804–820.
[13] K. He, G. Gkioxari, P. Dollar, and R. Girshick, "Mask R-CNN," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 2961–2969.
[14] M. A. Reza, A. U. Naik, K. Chen, and D. J. Crandall, "Automatic annotation for semantic segmentation in indoor scenes," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Nov. 2019, pp. 4970–4976.
[15] S. Gupta, P. Arbelaez, and J. Malik, "Perceptual organization and recognition of indoor scenes from RGB-D images," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2013, pp. 564–574.
[16] N. Dong and E. Xing, "Few-shot semantic segmentation with prototype learning," in *Proc. Brit. Mach. Vis. Conf.*, 2018, pp. 1–5.
[17] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C. Fu, and A. Berg, "SSD: Single shot multibox detector," in *Proc. Eur. Conf. Comput. Vis.*, 2016, pp. 21–37.
[18] J. Redmon and A. Farhadi, "YOLOv3: An incremental improvement," 2018, *arXiv:1804.02767*. [Online]. Available: http://arxiv.org/abs/1804.02767
[19] L. Castrejon, K. Kundu, R. Urtasun, and S. Fidler, "Annotating object instances with a polygon-RNN," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 5230–5238.
[20] M. Suchi, T. Patten, D. Fischinger, and M. Vincze, "EasyLabel: A semi-automatic pixel-wise object annotation tool for creating robotic RGB-D datasets," in *Proc. Int. Conf. Robot. Autom. (ICRA)*, May 2019, pp. 6678–6684.
[21] P. Marion, P. R. Florence, L. Manuelli, and R. Tedrake, "Label fusion: A pipeline for generating ground truth labels for real RGBD data of cluttered scenes," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2018, pp. 1–8.
[22] J.-H. Lee, M. Heo, K.-R. Kim, and C.-S. Kim, "Single-image depth estimation based on Fourier domain analysis," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 330–339.
[23] R. Zhao, Y. Wang, and A. M. Martinez, "A simple, fast and highly-accurate algorithm to recover 3D shape from 2D landmarks on a single image," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 40, no. 12, pp. 3059–3066, Dec. 2018.
[24] I. Laina, C. Rupprecht, V. Belagiannis, F. Tombari, and N. Navab, "Deeper depth prediction with fully convolutional residual networks," in *Proc. 4th Int. Conf. 3D Vis. (3DV)*, Oct. 2016, pp. 239–248.
[25] L. Roberts, "Machine perception of three-dimensional solids," Ph.D. dissertation, Dept. Elect. Eng., Massachusetts Inst. Technol., Cambridge, MA, USA, 1961.
[26] C. Zou, A. Colburn, Q. Shan, and D. Hoiem, "LayoutNet: Reconstructing the 3D room layout from a single RGB image," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 2051–2059.
[27] C.-Y. Lee, V. Badrinarayanan, T. Malisiewicz, and A. Rabinovich, "RoomNet: End-to-End room layout estimation," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 4875–4884.
[28] J. Huang, Y. Zhou, T. Funkhouser, and L. Guibas, "FrameNet: Learning local canonical frames of 3D surfaces from a single RGB image," 2019, *arXiv:1903.12305*. [Online]. Available: http://arxiv.org/abs/1903.12305
[29] S. Dasgupta, K. Fang, K. Chen, and S. Savarese, "DeLay: Robust spatial layout estimation for cluttered indoor scenes," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 616–624.
[30] C. Liu, J. Yang, D. Ceylan, E. Yumer, and Y. Furukawa, "PlaneNet: Piecewise planar reconstruction from a single RGB image," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 2579–2588.
[31] L. Yi *et al.*, "Large-scale 3D shape reconstruction and segmentation from ShapeNet Core55," 2017, *arXiv:1710.06104*. [Online]. Available: http://arxiv.org/abs/1710.06104
[32] A. X. Chang, T. Funkhouser, L. Guibas, P. Hanrahan, Q. Huang, Z. Li, S. Savarese, M. Savva, S. Song, H. Su, J. Xiao, L. Yi, and F. Yu, "ShapeNet: An information-rich 3D model repository," 2015, *arXiv:1512.03012*. [Online]. Available: http://arxiv.org/abs/1512.03012
[33] P. Ammirato, P. Poirson, E. Park, J. Kosecka, and A. C. Berg, "A dataset for developing and benchmarking active vision," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2017, pp. 1378–1385.
[34] J. Xiao, A. Owens, and A. Torralba, "SUN3D: A database of big spaces reconstructed using SfM and object labels," in *Proc. IEEE Int. Conf. Comput. Vis.*, Dec. 2013, pp. 1625–1632.
[35] O. Miksik, D. Munoz, J. A. Bagnell, and M. Hebert, "Efficient temporal consistency for streaming video scene analysis," in *Proc. IEEE Int. Conf. Robot. Autom.*, May 2013, pp. 133–139.
[36] A. Handa, V. Patraucean, V. Badrinarayanan, S. Stent, and R. Cipolla, "Understanding RealWorld indoor scenes with synthetic data," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 4077–4085.
[37] S. Richter, V. Vineet, S. Roth, and V. Koltun, "Playing for data: Ground truth from computer games," in *Proc. Eur. Conf. Comput. Vis.*, 2016, pp. 108–112.
[38] S. Tsutsui, Y. Fu, and D. Crandall, "Meta-reinforced synthetic data for one-shot fine-grained visual recognition," in *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, 2019, pp. 3057–3066.
[39] C. Taylor and A. Cowley, "Parsing indoor scenes using RGB-D imagery," in *Proc. Robot. Sci. Syst. (RSS)*, 2012, pp. 401–408.
[40] P. F. Felzenszwalb and D. P. Huttenlocher, "Efficient graph-based image segmentation," *Int. J. Comput. Vis.*, vol. 59, no. 2, pp. 167–181, Sep. 2004.
[41] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and S. Sässtrunk, "SLIC superpixels compared to state-of-the-art superpixel methods," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 34, no. 11, pp. 2274–2282, Nov. 2012.
[42] P. Arbelaez, J. Pont-Tuset, J. Barron, F. Marques, and J. Malik, "Multiscale combinatorial grouping," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2014, pp. 328–335.
[43] Y. Boykov, O. Veksler, and R. Zabih, "Fast approximate energy minimization via graph cuts," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 23, no. 11, pp. 1222–1239, Sep. 2001.
[44] J. L. Schonberger and J.-M. Frahm, "Structure-from-motion revisited," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 4104–4113.
[45] N. Silberman, D. Hoiem, P. Kohli, and R. Fergus, "Indoor segmentation and support inference from RGBD images," in *Proc. Eur. Conf. Comput. Vis.*, 2012, pp. 746–760.
[46] W. Abdulla, "Mask R-CNN for object detection and instance segmentation on Keras and TensorFlow," GitHub Repository, 2017. [Online]. Available: https://github.com/matterport/Mask_RCNN
[47] T.-Y. Lin, M. Maire, S. Belongie, L. Bourdev, R. Girshick, J. Hays, P. Perona, D. Ramanan, C. Lawrence Zitnick, and P. Dollár, "Microsoft COCO: Common objects in context," 2014, *arXiv:1405.0312*. [Online]. Available: http://arxiv.org/abs/1405.0312
[48] G. Georgakis, M. A. Reza, A. Mousavian, P.-H. Le, and J. Kosecka, "Multiview RGB-D dataset for object instance detection," in *Proc. 4th Int. Conf. 3D Vis. (3DV)*, Oct. 2016, pp. 426–434.

MD ALIMOOR REZA received the M.S. degree in computer science from Drexel University, in 2011, and the Ph.D. degree in computer science from George Mason University, in 2018. He is a Postdoctoral Associate with the School of Informatics, Computing, and Engineering, Indiana University. His research interests include the intersection of computer vision, machine learning, and robotics, focusing on the development of novel algorithms for perception tasks, such as semantic segmentation and object detection exploiting the 3D information. His work appears in premier conferences in computer vision, robotics, and artificial intelligence, including 3DV and IROS. He worked at the U.S. Army Research Laboratory (ARL), in 2017, where he was a recipient of the Summer Journeyman Fellowship. He also worked as a Research Intern at 3M Company, in summer 2015.

KAI CHEN received the bachelor's degree from the School of Computer Science, Fudan University. He is currently pursuing the Ph.D. degree with the Hong Kong University of Science and Technology. He visited the Indiana University Computer Vision Laboratory, in 2019. His research is mainly focused on few-shot learning, semisupervised learning, and other computer vision-related problems.

AKSHAY NAIK received the bachelor's degree in computer engineering from Mumbai University and the master's degree in data science from the Luddy School of Informatics, Computing, and Engineering, Indiana University. He has been a Data Science Intern with the OSRAM Sylvania Research Center and has worked as a Trainee Decision Scientist at Mu Sigma. He is currently a Data Scientist with the Expedia Group. His research interests include the field of computer vision and recently gained interest in learning theory and optimization.

DAVID J. CRANDALL received the B.S. and M.S. degrees in computer science and engineering from Pennsylvania State University, in 2001, and the M.S. and Ph.D. degrees in computer science from Cornell University, in 2007 and 2008, respectively. He was a Senior Research Scientist with Eastman Kodak Company, from 2001 to 2003. He is currently an Associate Professor with the Luddy School of Informatics, Computing, and Engineering, Indiana University. His research on computer vision and data mining has been funded by NSF, IARPA, U.S. Navy, NASA, DTRA, ONR, ETRI, AFOSR, IN3, Facebook, Google, Yahoo, Kodak, Grant Thornton, and Nvidia. He has received the Best Paper Awards or Nominations at CVPR, WWW, CHI, ICCV, and ICDL. He has been an Area Chair for CVPR, ICCV, ECCV, WACV, AAAI, ICML, and IJCAI. He was a recipient of the NSF CAREER Award, two Google Faculty Research Awards, the IU Trustees Teaching Award, and the Grant Thornton Fellowship. He is an Associate Editor of the IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE and the IEEE TRANSACTIONS ON MULTIMEDIA.

SOON-HEUNG JUNG received the B.S. degree in electronics from Pusan National University, South Korea, in 2001, and the M.S. and Ph.D. degrees in electrical engineering from the Korea Advanced Institute of Science and Technology (KAIST), South Korea, in 2003 and 2016, respectively. From March 2003 to March 2005, he was with LG Electronics. Since April 2005, he has been a Principal Researcher with the Electronics and Telecommunications Research Institute (ETRI), Daejeon, South Korea. His research interests include immersive media, computer vision, video coding, and realistic broadcasting systems.