3D Semantic Parsing of Large-Scale Indoor Spaces Supplementary Material

Iro Armeni[1] Ozan Sener[1,2] Amir R. Zamir[1] Helen Jiang[1]
Ioannis Brilakis[3] Martin Fischer[1] Silvio Savarese[1]

[1] Stanford University [2] Cornell University [3]University of Cambridge

http://buildingparser.stanford.edu/

Abstract

The supplementary material contains: (I) most importantly, a video that shows detailed and comprehensive results, as well as a PDF which includes (II) more experimental results in terms of PR curves, sample RGB-D detection results, and comparison to the baselines, (III) a few example real-world applications of large-scale semantic parsing, (IV) implementation details of CRF to enforce contextual consistency, (V) more details on detecting space dividers for disjoint space parsing, and finally (VI) implementation details of locating the entrance to disjoint spaces for proper normalization in the canonical coordinate system.

1. Additional Results

In this section, we present additional results on the semantic element parsing. We first provide precision-recall curves for each semantic class. We also include more qualitative results on the comparison to RGB-D based methods, as explained in Sec. 5.4 in the main paper.

1.1. Precision-Recall Curves for Element Detection

Fig. 1 shows the precision-recall curves for each semantic class. The curves suggest the importance of the global features in all semantic classes. The precision-recall curves, as well as the class-specific average precision values presented in the main paper, suggest that our method performs very well in the case of structural elements, however the performance for furniture is limited. We attribute these results to the generalization of structural elements among different buildings, which does not apply to the same extent on furniture.

1.2. Additional Results on comparison with RGB-D based methods

In this section, we present additional results on our RGB-D experiments. In Fig 2, we qualitatively show the RGB-D segmentation results of [1], as well as the projected results of our algorithm on the same images. We also provide the input images as reference. As the resulting images suggest, our algorithm significantly outperforms the baseline. However, the results might be due to the train-test mismatch. It should be noted that our 3D detectors are trained on our training set whereas the RGB-D segmentation algorithm is trained by using NYU-Depth-V2.

2. Applications

Understanding the semantics of large-scale indoor space point clouds can give rise to numerous applications without the immediate need for 3D reconstruction. We will present here three examples that showcase the power of point cloud semantics: automatic extraction of space statistics, automatic coarse calculation of relative natural light score and automatic space manipulation. For detailed results, please see the provided video.

2.1. Space Statistics

Knowing space statistics of an existing space is the starting point for refurbishment, energy performance analysis, interior design, etc. Acquiring such information is currently a time consuming task since it requires manually extracting it from the point cloud. It is however possible to automate this process and automatically generate all necessary measurements. Such an application can benefit not only the construction industry but also the typical end-user.

We use our point-level results and compute a variety of space statistics, such as volume, area, size and width of semantic elements, per floorplan or disjoint space. To see an example of the computed statistics for a sample building area of our dataset, please see the provided video.

2.2. Estimation of Natural Illumination Model

Availability of detailed semantics of an indoor space can empower several applications. One such example is the ap-

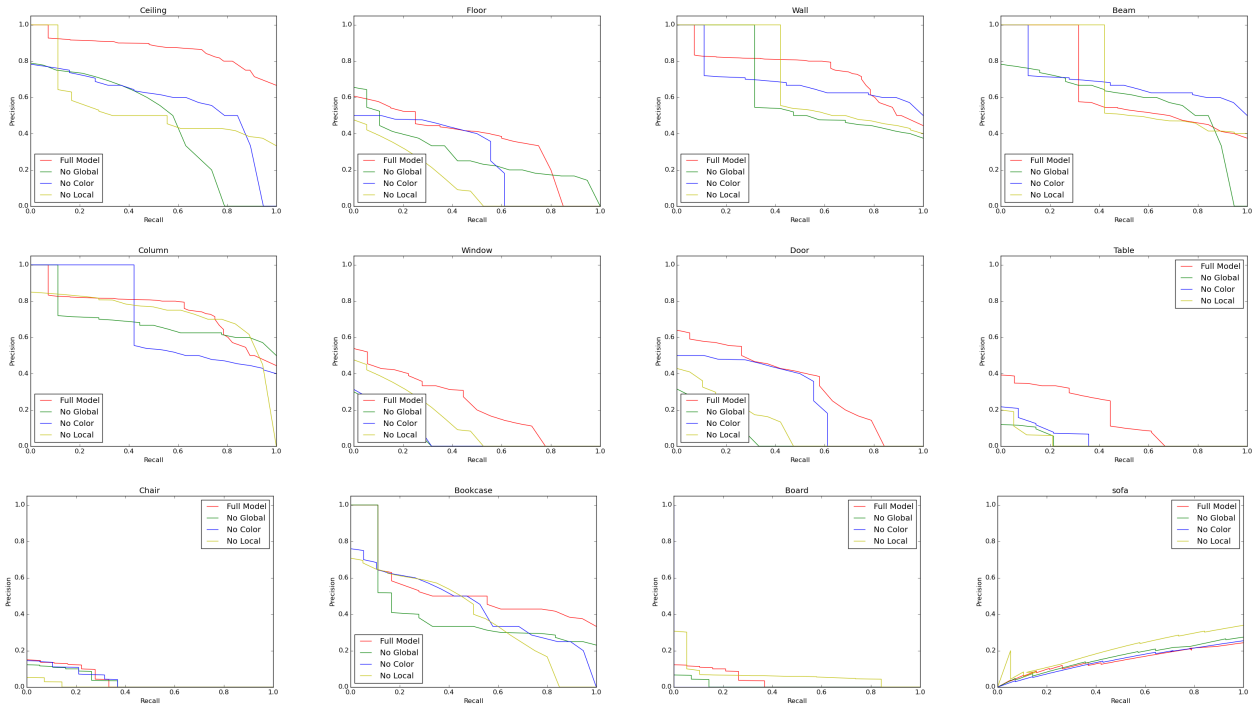

Figure 1. Additional Results for 3D element detection. Class specific PR curves for each semantic class.

proximate calculation of the relative gradient of natural light per space based on the proximity to a window (Fig. 3). To this end we compute the distance of each point from the closest window, taking into account occlusions due to walls and doors. This results to a coarse natural light illumination model.

2.3. Space Manipulation

One more possible application is space manipulation. By knowing the semantics, one can visualize how two or more adjacent spaces would look like if they were connected by removing the in-between walls (Fig. 4 b), or further how they would be perceived if they were empty, by removing the furniture (Fig. 4 c,d). This finds numerous applications in interior design, graphics, etc. For an example, please see the provided video.

3. Additional Details

3.1. Parsing Point Cloud into Disjoint Spaces

In this section we will provide details on detecting space dividers and finding the location of the entrance to a disjoint space.

3.1.1 Detecting space dividers

Bank of filters: We detect space dividers by convolving the histogram of density of points for each axis with a custom designed filter of a peak-gap-peak shape. However we have no prior knowledge of the width of space dividers in the building, neither do we know if their size remains constant throughout the floorplan. As a result we cannot define a single filter with set dimensions (the size of the gap between the two peaks). We thus create a bank of filters to accommodate all possible space divider widths. In specific, we form a bank of filters with $\delta\{2, 6, 8, ..., 80\}$, $c\{2, 4, 6, ..., 10\}$ and $w\{10, 15, 20, 80\}$, resulting to 754 differently shaped filters (see Fig. 2 right (b) in main paper).

Convolution: We convolve the histogram of density of points with all 754 filters. The output of this operation is a set of continuous signals that form peaks in the location of the gap between two closely located peaks in the histogram. The filter that fits the peak-gap-peak formation in the histogram the closest will output the maximum possible value among all filters. Since we have a number of space dividers per signal each of which might be best represented by different filters, we perform max pooling over $\delta, c : C(a) = max(C(a))$ to consolidate the results of all convolutions, as the right δ, c scale parameters are expected to give rise to the strongest peaks (see Fig. 2 right (c) in main paper).

Space Dividers: We detect the final space divider locations by applying non-maximum suppression on C, followed by bimodal clustering. Filters with different parameters close to the one that gives the maximum peak fire as well but in

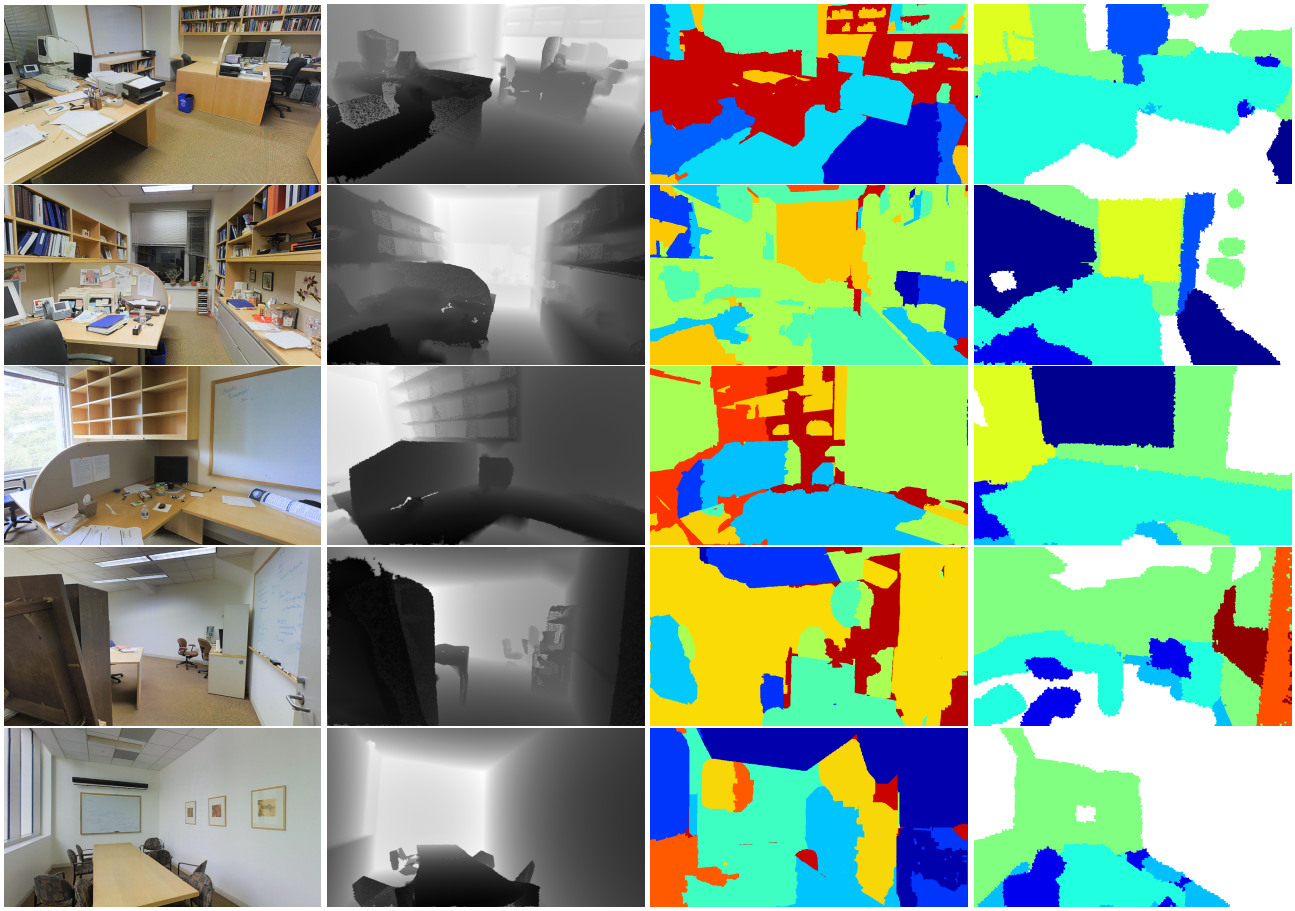

Figure 2. Additional Results. First column is the RGB image, second column is the depth image, 3rd column is the segmentation results of [1] and the last column is our projected results. It should be noted that [1] and our method are using different set of classes; hence, we show the segmentation results for visualization purposes only. In our quantitative evaluation in the main paper, we only use the intersecting classes.

slightly different locations, next to that of the strongest response. To select the maximum one, we use non-maximum suppression. The output of this step provides us with candidate locations for space dividers. However some of these candidates are a by-product of the moving nature of convolution and hence do not correspond to space dividers. Such peaks are due to the distribution of clutter or to a non-building element that might get represented in the histogram in a similar to a wall way but with smaller intensity. To filter them out, we use bimodal clustering. Due to the considerably lower magnitude of such peaks, they can be easily distinguished from correct peaks by maximizing the inter-class variance between the two groups of peaks without the need to manually set a threshold (we used Otsu's method [3] for this purpose). The peaks surviving this step are our final space divider locations (see Fig. 2 right (d) in main paper).

After over-segmenting our point cloud, we perform a series of merging operations to connect over-segments that belong to the same space. As a brief overview, we iteratively examine a pair of neighboring over-segments for the existence of a space divider in a localized area around their common side.

Merging Over-segments: We generate the histogram of density of points in this area along the axis that is orthogonal to the common side. As in the previous step, we convolve the histogram with the bank of 754 filters and perform the same operations to identify the existence or not of a candidate space divider in this area. If it results to a peak, then this pair of neighboring over-segments is not connected, i.e. the over-segments belong to separate spaces. Otherwise, they belong to the same space and no space divider exists between them.

Locating Doors: In case that the two segments belong to separate spaces, it is possible that a connection element exists between them (door) that allows mobility from one space to the other. If we chose to normalize the detected spaces based on their entrance location, we need to coarsely

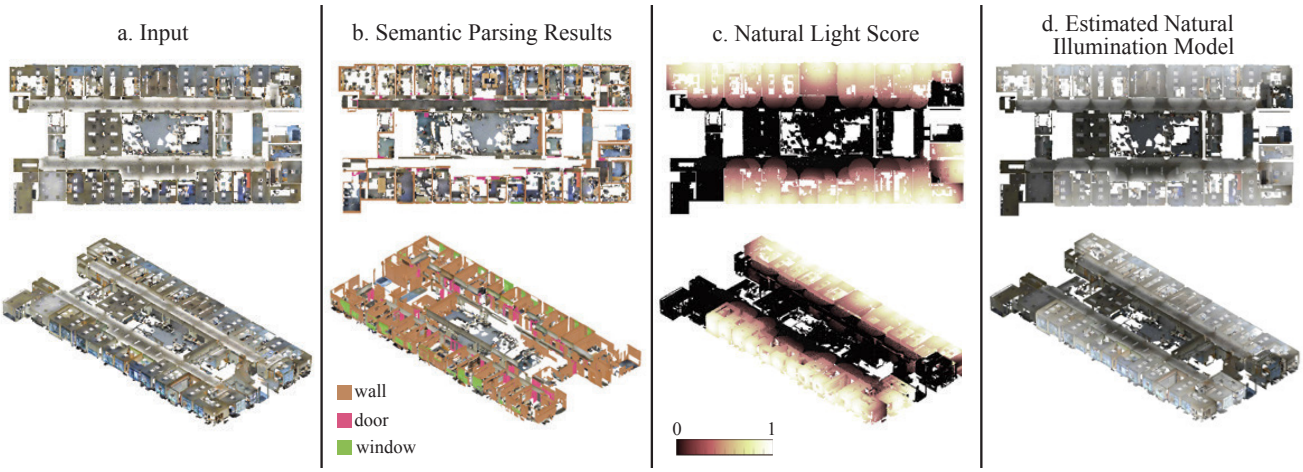

Figure 3. An application of large-scale semantic parsing: automatically estimating the natural illumination model of spaces based on the proximity to windows and placement of walls and doors for reflection reasoning.

identify the potential location of the door of each space prior to the next step of the framework (element parsing). We identify it during the merging operations using a simple detector, by once again utilizing the generated histogram of density of points in the connecting area of the over-segments. In the core of this detector lies the fact that a door is depicted in the point cloud as a concentrated group of points in between two wall surfaces. This means that if a door exists between two disjoint over-segments, there will be a number of points in between the two wall surfaces. We take advantage of this property and compute the mode of the density of points in a small area near the fired peak (space divider). Because we are only using a very small area around the peak and we are only interested in the extends of the rooms' sides that are common, we expect the mode to be zero if no connecting element exists, otherwise we identify a connection.

3.2. Parsing Disjoint Spaces into Elements

In this section, we provide more details on how do we define and solve the graphical model as well as give additional results on element parsing.

3.2.1 Details on Graphical Models

As we explained in equation (1) of the main paper (see Sec. 4.1), our model follows the log-linear model [2] and we predict the final elements as a maximization problem of the energy function;

$$\arg\max_{y} \sum_{v\in\mathcal{V}} w_0 l_v y_v + \sum_{(u,v)\in\mathcal{E}} y_v y_u \left(w_{e_u,e_v}\cdot\Phi_{u,v}\right), \quad (1)$$

which can be written as an integer program by introducing auxiliary variables $y_{uv} = y_u y_v \quad \forall u,v \in \mathcal{V}$ as;

$$\begin{aligned} \arg\max_{y} \quad & \sum_{v\in\mathcal{V}} w_0 l_v y_v + \sum_{(u,v)\in\mathcal{E}} y_{vu}\left(w_{e_u,e_v}\cdot\Phi_{u,v}\right) \\ & s.t. y_{uv} \le y_u \quad \forall u\in\mathcal{V}, \forall v\in\mathcal{N}(u) \\ & s.t. y_u + y_v \le y_{uv} + 1 \quad \forall u,v\in\mathcal{E}. \end{aligned} \quad (2)$$

This maximization is performed using an off-the-shelf LP/MIP solver and the weight vectors w are learned using Structured SVM [4]. In order to solve the optimization problem, we are using the GNU Linear Programming Toolkit.

In order to learn the weight vectors w, we are using the Structured-SVM (S-SVM) algorithm, which is based on cutting plane method. In a nutshell, it minimizes the loss function $\sum \Delta(y,\hat{y})$ such that Δ is a structured loss between the ground truth labelling $\hat{y}$ and the estimation y. We are using the Hamming loss as $\sum_i \mathbb{1}[\hat{y}_i = y_i]$. S-SVM optimizes this loss by merely requiring;

$$\begin{aligned} \arg\max_{y} \quad & \sum_{v\in\mathcal{V}} w_0 l_v y_v + \sum_{(u,v)\in\mathcal{E}} y_{vu}\left(w_{e_u,e_v}\cdot\Phi_{u,v}\right) + \Delta(y,\hat{y}) \\ & s.t. y_{uv} \le y_u \quad \forall u\in\mathcal{V}, \forall v\in\mathcal{N}(u) \\ & s.t. y_u + y_v \le y_{uv} + 1 \quad \forall u,v\in\mathcal{E}. \end{aligned} \quad (3)$$

Thanks to the specific structure of Hamming loss, this is equivalent to;

$$\begin{aligned} \arg\max_{y} \quad & \sum_{v\in\mathcal{V}} w_0 l_v y_v + \mathbb{1}[\hat{y_v} = y_v] + \\ & \sum_{(u,v)\in\mathcal{E}} y_{vu}\left(w_{e_u,e_v}\cdot\Phi_{u,v}\right) + \mathbb{1}[\hat{y_{vu}} = y_{vu}] \\ & s.t. y_{uv} \le y_u \quad \forall u\in\mathcal{V}, \forall v\in\mathcal{N}(u) \\ & s.t. y_u + y_v \le y_{uv} + 1 \quad \forall u,v\in\mathcal{E}. \end{aligned} \quad (4)$$

This loss augmented optimization problem can also be solved by using the same LP/MIP solver.

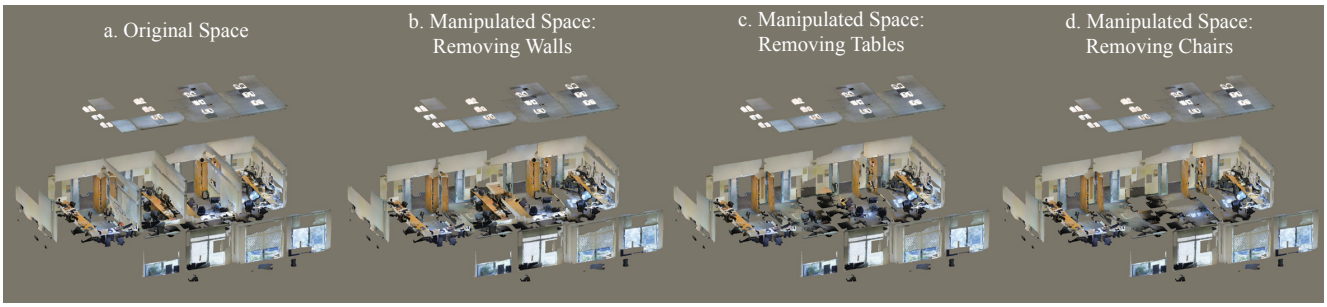

Figure 4. Space Manipulation

References

[1] L. Bo, K. Lai, X. Ren, and D. Fox. Object recognition with hierarchical kernel descriptors. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pages 1729–1736. IEEE, 2011.

[2] D. Koller and N. Friedman. *Probabilistic graphical models: principles and techniques*. MIT press, 2009.

[3] N. Otsu. A threshold selection method from gray-level histograms. *Automatica*, 11(285-296):23–27, 1975.

[4] I. Tsochantaridis, T. Hofmann, T. Joachims, and Y. Altun. Support vector machine learning for interdependent and structured output spaces. In *Proceedings of the twenty-first international conference on Machine learning*, page 104. ACM, 2004.