Rent3D: Floor-Plan Priors for Monocular Layout Estimation

Chenxi Liu[1] * Alexander G. Schwing [2,*] Kaustav Kundu[2] Raquel Urtasun[2] Sanja Fidler[2]

[1]State Key Lab. on Intelligent Technology and Systems
Tsinghua Nat. Lab. for Inf. Science and Tech. (TNList)
Department of Automation, Tsinghua University

[2]Department of Computer Science
University of Toronto

chenxi_liu@live.cn, {aschwing, kkundu, urtasun, fidler}@cs.toronto.edu

Abstract

The goal of this paper is to enable a 3D "virtual-tour" of an apartment given a small set of monocular images of different rooms, as well as a 2D floor plan. We frame the problem as inference in a Markov Random Field which reasons about the layout of each room and its relative pose (3D rotation and translation) within the full apartment. This gives us accurate camera pose in the apartment for each image. What sets us apart from past work in layout estimation is the use of floor plans as a source of prior knowledge, as well as localization of each image within a bigger space (apartment). In particular, we exploit the floor plan to impose aspect ratio constraints across the layouts of different rooms, as well as to extract semantic information, e.g., the location of windows which are marked in floor plans. We show that this information can significantly help in resolving the challenging room-apartment alignment problem. We also derive an efficient exact inference algorithm which takes only a few ms per apartment. This is due to the fact that we exploit integral geometry as well as our new bounds on the aspect ratio of rooms which allow us to carve the space, significantly reducing the number of physically possible configurations. We demonstrate the effectiveness of our approach on a new dataset which contains over 200 apartments.

1. Introduction

How many times have you switched apartments/houses in the past few years? According to the U.S. Census Bureau, out of a population of 307,243,000 people, 35,918,000 Americans moved between 2012 and 2013. That is 11.69% of the US population. On an average day, Craigslist, a popular classified advertisement website, has 90,000 rental posts only for New Your City, with any bigger city exceeding 10,000 ads on any given day. Each ad typically has a few images showing different rooms in the apartment, and in many cases also a floor plan which gives the potential tenants an idea about how the space is organized.

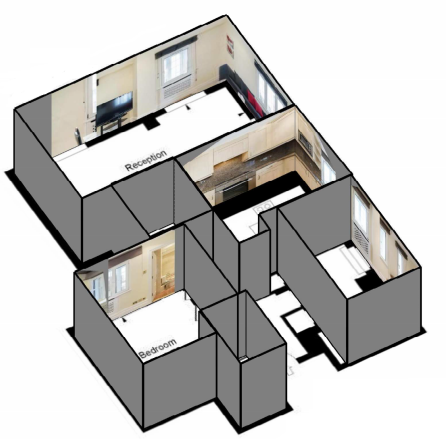

Figure 1. Our approach reconstructs rental apartments from a set of monocular images and a floor plan.

Given a rental ad that contains a few images and a floor plan of the apartment, our goal is to reconstruct the apartment in 3D. That is, given only monocular images of different rooms, we want to enable a "virtual-tour" allowing the potential tenant to virtually explore the apartment without having to physically visit the site. This adds another dimension to apartment rental sites, and is particularly convenient for cold cities like Toronto or Chicago where apartment search in the winter is particularly unpleasant.

Inferring 3D information from a single monocular image has been one of the holy grails of computer vision since its beginnings [15]. The problem itself is ill-posed, however, prior knowledge about the type of scene as well as scene semantics can help resolve some of the ambiguities [11]. By assuming that rooms conform to Manhattan world, impressive results have been achieved for the problem of room layout estimation [8, 22]. Estimating the layout means determining the 3D cuboid that defines the room, *i.e.*, inferring the accurate positions of the front wall towards which the camera is facing. This is a very challenging and exciting problem that received a lot of attention in the past years.

*Denotes equal contribution

In our work, we take this idea one step further. Our goal is to estimate a layout for each room from a monocular image and to accurately localize its camera within the full apartment. We phrase the problem as an energy minimization problem that exploits rich information contained in the apartment's floor plan as an additional source of information. We make use of the floor plan in order to get information about aspect ratios of the floor for each room. The room layouts across images are linked by the fact that all rooms share the same height in 3D, which imposes aspect ratio constraints in the layout estimation problem. Scene type classification helps us localize in which room the image was taken in. We also make use of semantics in the form of windows for which typically the floor plan contains additional ratio constraints. Windows are shown to be a very useful cue for camera localization within the apartment since they break the symmetry of parallel walls. We use structured prediction [26, 25] to train our model and a branch-and-bound inference algorithm that runs in real-time.

Since no suitable data exists for this task, we also collected our own dataset by crawling a London-based rental site. We collected posts for 215 apartments, each of which had at least one image and a floor plan, resulting in the total of 1259 images. We labeled our data with rich annotations including the ground-truth layout and pose of each room within the full apartment. Our dataset is available here: www.cs.utoronto.ca/∼fidler/projects/rent3D.html.

2. Related Work

Estimating the 3D room layout from monocular images has been popularized by [8]. It is a very challenging problem due to inherent ambiguities in 3D, large amount of clutter typically present in indoor scenes, and little contrast on the actual wall intersections. By assuming that the world is Manhattan, layout estimation was formalized as energy minimization in a Markov Random Field that tries to place a 3D room cuboid based on 3D-informed image features [6, 14, 27, 8, 18, 19, 7, 3]. While the optimization seemed intractable at first [8], it was recently shown that a globally optimal configuration can be found in real time using branch-and-bound [24]. Our work builds on these ideas. However, in contrast to prior work, we make use of floor plans in order to both, estimate the layout of each image and localize it within the apartment. Using floor plans for indoors is related to using cartographic maps for localization [1] or semantics [28, 17] in outdoor scenarios.

Numerous approaches have also been proposed addressing 3D detection or segmentation of indoor scenes [14, 4, 9, 10]. While this is an important line of work, particularly in scenes where objects dominate, our focus here is on layout and camera estimation. Note also that rental ads typically show cleaner, less cluttered rooms.

Using floor plans to facilitate 3D reconstruction is not a new idea. However, past work relied on 3D point clouds obtained via video scans of a site or RGB-D cameras [29, 5, 16, 2]. This is in contrast to our work which aims to perform reconstruction from monocular images. To the best of our knowledge, this has not been tackled in the literature before.

3. Rent3D Apartment Dataset

Since our goal here is to reconstruct apartments in 3D, we collected our dataset by crawling a rental website. We chose a London rental site that contained rental information in a unified format and the majority of posts also had floor plans. To get the data, we queried the site for "Central London". Out of many hits, we took the first 215 that had a floor plan of sufficient resolution and at least one photo.

Statistics: The 215 apartments have in total 1312 rooms, 6628 walls, 1923 doors, and 1268 windows. The number of photos per apartment ranges from 2 to 30, with the total number in our dataset being 1259, not counting the outdoor images. Note that in their rental ads, people sometimes also post pictures of the apartment building, or public facilities inside the building. We keep these photos as they represent the challenge of dealing with real world data. The number of all images are 1570.

Collecting Ground-truth: Since the goal of the paper is layout estimation and camera localization (computing the pose of the camera with respect to the full apartment), we annotated the data with three types of ground-truth. To annotate the floor plan, we asked in-house annotators to annotate the scale mapping the plan to the real world. This can easily be done since most of the floor plans contain information about the physical dimensions of the rooms. We further annotated the room outline, room type, as well as the position of doors and windows. Note that the outline of each room is not necessarily rectangular. Roughly 10% of rooms in our dataset have more complex polygonal shape.

Following the layout annotation setup [8], we collected annotations for the front, left, and right wall in each photo, as well as ceiling and floor. To get ground-truth for camera localization, the annotators were asked to first select the room in the floor plan in which the photo was taken, as well as to click on a wall in the photo and link it to the corresponding wall (line) in the floor plan. Since rooms share walls, clicking on only the wall is ambiguous, thus room information is also required.

Note that finding the alignment is a very challenging task even for a human. This is due to the minimal information typically contained in the floor plan. The annotators in most cases resolved the problem by checking the semantics: the type of scene, the presence and number of windows, and sometimes even objects, *e.g.*, some floor plans have information about ovens, toilets or bathtubs. There were however cases where even humans could not find the correct

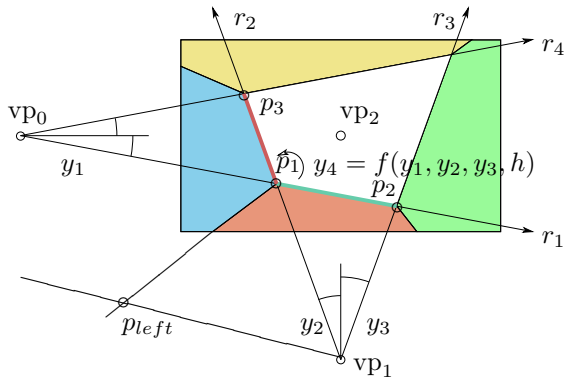

Figure 2. Parameterization of our layout task. Using projective geometry, y_4 can be computed from parameters y_1, y_2, y_3 and a known apartment height h. Further, given intersection points p_1 and p_2 we can compute the left bottom coordinate of the left wall using the aspect ratio from the floor plan. Our layout configuration is valid if the ray through vp_1 and p_{left} lies outside the image.

alignment. We allowed the annotators to choose "unknown" in the ambiguous examples. Out of the 1259 indoor photos, 71 of them do not have ground-truth room alignment, and 11 photos do not have ground-truth wall alignment.

Scenes: In the raw annotations, there were a total of 22 room types, which we merged into 5 scene labels: Reception, Bedroom, Kitchen, Bathroom, Outdoor. Of all 1570 photos in the dataset, the five labels have 485, 332, 213, 235, 305 photos, respectively.

4. Floor Plan Priors for Layout Estimation

We are interested in jointly solving two tasks: estimating the layout of the room for each given image, and finding the room's 3D pose relative to the full apartment. Here, our apartment is defined via the floor plan. Note that this is in principle a continuous problem, which we represent in a discrete manner. Instead of parametrizing the pose in 3D, we phrase the problem as that of choosing the room in the floor plan in which the photo was taken, and choosing which wall in that room the camera (image) is facing. This is much like our apartment dataset annotation was done. To jointly address this task we first formulate the problem in the context of energy minimization.

4.1. Energy Formulation

More formally, let $r \in \{1, \ldots, R\}$ be a discrete random variable representing the room, where R are the number of rooms in the floor plan. Let $c_r \in \{1, \ldots, C_r\}$ be a discrete variable representing within room r which wall the picture is facing. The number of walls within a room is denoted by C_r. We parameterize the location and orientation of the camera in 3D space in terms of the room layout. We assume the world is Manhattan [8], and thus our room is a cuboid. It can be represented via 4 corners of the front wall, *i.e.*, 4 rays

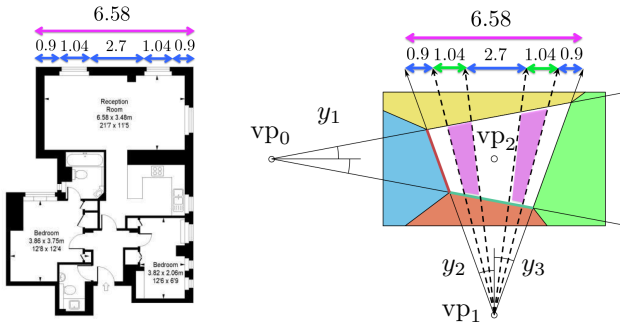

Figure 3. **(Left)** Example of a floor-plan with window-to-wall ratios, **(Right)** The window-to-wall ratios in the floor plan enable us to compute the rays (dashed lines) for the windows given y_2 and y_3 in our layout problem. Windows are denoted with green lines.

originating from a pair of orthogonal vanishing points [14]. Fig. 2 illustrates this parametrization. We compute the vanishing points using the implementation provided by [8].

Note that given the vanishing points, the camera intrinsics and rotation (relative to room) can be computed [21]. This enables us to reason in 3D, up to scale. We thus cannot exploit the actual physical dimensions contained in the floor plan, but we can exploit their ratios. We use the meta-data contained in our dataset about the room height as well as the dimensions of rooms available in floor plans in order to compute the aspect ratios of different walls in the apartment. We use (r, c_r) to look them up for a particular hypothesis.

The aspect ratio $a(r, c_r)$ allows us to parameterize the layout problem in terms of only three variables y_1, y_2, y_3 (and not the typical four [23]). We can compute the fourth ray by using $a(r, c_r)$ as explained in Sec. 5. Let $\mathbf{y} = (y_1, y_2, y_3)$ be the set of rays encoding the layout.

We formulate the localization problem within the apartment as inference in a Markov Random Field which estimates room layout ($\mathbf{y}$), the room in which the picture was taken as well as the assignment of the front wall to the floor plan. Our energy is a sum of energies encoding the agreement with geometric features (orientation maps (OM) [14] and geometric context (GC) [12, 8]), the presence of objects such as windows (which are a useful cue for disambiguating alignment), as well as the aspect ratio of the room, which links the floor plan and the room layout. We define:

$$E(r, c_r, \mathbf{y}) = E_{layout}(\mathbf{y}) + E_{win}(r, c_r, \mathbf{y}) + E_{as-ratio}(r, c_r, \mathbf{y}) + E_{scene}(r),$$

We next provide details regarding each energy term.

Layout: We follow [23], and employ features for each possible face $\alpha \in \mathcal{F} = \{\text{left}, \text{front}, \text{right}, \text{floor}, \text{ceiling}\}$ visible within the image. More specifically

$$E_{layout}(\mathbf{y}) = \sum_{\alpha \in \mathcal{F}} w_{lay,\alpha}^{\top} \phi_{lay,\alpha}(\mathbf{y}),$$

for all r, c **do**
  put pair $(\bar{f}(\mathcal{Y}), \mathcal{Y})$ into queue and set $\hat{\mathcal{Y}} = \mathcal{Y}$
  repeat
    split $\hat{\mathcal{Y}} = \hat{\mathcal{Y}}_1 \times \hat{\mathcal{Y}}_2$ with $\hat{\mathcal{Y}}_1 \cap \hat{\mathcal{Y}}_2 = \emptyset$
    put pair $(\bar{f}(\hat{\mathcal{Y}}_1), \hat{\mathcal{Y}}_1)$ into queue
    put pair $(\bar{f}(\hat{\mathcal{Y}}_2), \hat{\mathcal{Y}}_2)$ into queue
    retrieve $\hat{\mathcal{Y}}$ having highest score
  until $|\hat{\mathcal{Y}}| = 1$
end for

Figure 4. Inference uses exhaustive search and branch & bound.

with w a vector of weights and $\phi_{lay,\alpha}$ being 11-dimensional occurrence counts of orientation map (5 values) and geometric context (6 values) features within face α.

Windows: Given the room r as well as the wall c_r we can access the floor plan to determine whether, and if, at which location, a window should be present in the scene. We represent the window locations with rays originating from the vertical vanishing point. Note that the locations of the windows in the floor plan relative to the wall they belong to define a cross-ratio projective invariant which can be used to compute the vertical window rays in the image given the wall rays y_1 and y_2, illustrated in Fig. 3. We provide the details in the supplementary material.

On the image side, we predict windows by training a pixel-level classifier on the training set [20]. Let $\phi_{win,\alpha}(r, c_r, \mathbf{y})$ refer to the fraction of the predicted window pixels falling within the window rays for face $\alpha \in \mathcal{F}$. In addition we also augment this vector with information about the fraction of window pixels falling outside the window area, for a total of 6 features. We have:

$$E_{win}(r, c_r, \mathbf{y}) = \sum_{\alpha \in \mathcal{F}} w_{win,\alpha}^{\top} \phi_{win,\alpha}(r, c_r, \mathbf{y}).$$

Note that, just like for windows, the same information is also available for doors. However, our semantic classifiers for doors yielded poor results and we thus only opted to incorporate information about the windows in our model.

Aspect Ratios of Side Walls: We note that we have an additional constraint imposed by the floor plan: our camera relative to the estimated room layout in 3D needs to fall within the bounds of room r, where the bounds are defined by both, r and c_r (room's orientation). We impose this constraint by traveling on a ray from vp_2 through p_1 (defining the edge between the left wall and the floor). Via the distance between p_1 and p_2, and the given aspect ratio for the floor, it is possible to compute the second corner point p_{left} on the line vp_0 through p_1, defining the corner of the room where the left and the back wall meet. The details of this computation are in suppl. material. We say that y_1 and y_2 form an impossible configuration given (r, c_r), if the ray vp_1 through p_{left} intersects within the image. This would essentially imply that the camera is behind the back wall, and thus we will not be able to see the room. To incorporate such infeasible configurations we assign an ∞ penalty:

$$E_{as-ratio}(r, c_r, \mathbf{y}) = \begin{cases} 0 & (r, c_r \mathbf{y}) \text{ valid} \\ \infty & (r, c_r \mathbf{y}) \text{ invalid} \end{cases}.$$

We impose a hard constraint for the left and right wall.

Scene: In most cases floor plans also contain information about the location of the different rooms, *e.g.*, dining area, kitchen, bedroom *etc*. To leverage this information we add an additional scene energy term that favors room types depending on the score of a scene classifier, *i.e.*,

$$E_{scene}(r) = w_{scene} \phi_{scene}(r).$$

4.2. Efficient Inference via Branch and Bound

Given the above energy terms, inference amounts to solving the following program:

$$\min_{r, c_r \mathbf{y}} E(r, c_r, \mathbf{y}). \tag{1}$$

[24] shows how to employ branch and bound to optimize the layout of a room from a single monocular image. In our case, we have a different parameterization as well as new potentials which we need to design bounds for. This is not trivial, particularly for the aspect ratio constraints. Given (r, c_r) the layout problem is parametrized by only three variables $\mathbf{y}$ (rather than 4). This makes the branch and bound procedure for our case potentially even faster. Since the number of rooms r and the walls c_r is typically small we simply just exhaustively enumerate all possible combinations and perform a branch and bound inference to obtain the global optimum of the program given in Eq. (1).

The branch and bound approach w.r.t. $\mathbf{y}$ operates on interval product spaces, *i.e.*, we start with a set of all possible layouts $\mathcal{Y}$. This set is subsequently divided into smaller and smaller subsets, by choosing the lowest scoring layout set $\hat{\mathcal{Y}}$ from a priority queue as the next one to be split into disjunct subsets $\mathcal{Y}_1, \mathcal{Y}_2$. Before adding each of those sets into a priority queue we compute a lower bound $\bar{f}$ on the energy, *i.e.*, we make sure that the energy of every containing element is larger. The algorithm is provided in Fig. 4 and we detail the bounds in the following.

Layout bounds: We follow [24] to compute the bounds for the layout. We divide the energy into two parts depending on the sign of the weight vector. This is sufficient since our features ϕ are counts and hence necessarily positive. Lower bounds are then easily computable by adding the smallest possible face for the positive features to which we add the biggest possible negative contribution. We use integral geometry and divide all functions into sums of accumulators which depend on at most two variables. Both storage and computation are efficient.

Aspect ratio bounds: An aspect ratio bound $\bar{f}_{as-ratio}$ can also be obtained by setting its value to equal zero as long as there is at least a single feasible configuration within the set $\mathcal{Y}$ of the considered intervals. This essentially carves the space of possible layouts. Computing if there exists a feasible configuration can be done efficiently. We refer there reader to the suppl. material for details on the derivation.

Window bounds: Similar to the layout energy, the window energy consists of only weighted counts. Contrasting the layout energy, the window contributions indirectly depend on the parameterized variables $\mathbf{y}$ via the cross-ratios imposed by the floor plan. Careful consideration of maximally possible and minimally achievable regions yields computable lower bounds for the window energy E_{win}.

Scene energy: No bounding is required for the scene energy since it only depends on the room r. We exhaustively search for the minimum over this variable.

4.3. Learning

We learn the parameters of the MRF using a structural SVM [26]. In particular, we employ the parallel cutting plane implementation of [22] to learn the parameters. As task loss we use the layout pixel-wise loss.

5. Image and Floor-Plan Geometry

It is well known that the camera intrinsic matrix K and the rotation R can be recovered from three orthogonal vanishing points [21]. Given K and R we can compute a homography for each face of the room, *i.e.*, a projection which maps the face defined by two vanishing points vp_i and vp_j to fronto-parallel view:

$$H_{ij} = K \cdot R_{ij} \cdot K^{-1}.$$

Here R_{ij} denotes a column-wise permutation of R, placing the columns i and j to column 1 and 2.

Let a be a known aspect ratio defined as h/w, where h is the height and w the width of the wall in the physical world (we know these dimensions from the rental site). Then, given two corner points p_1 and p_2 of a wall, defined by the two intersections of rays y_1, y_2 and y_1, y_3, respectively, we can compute the corner point p_3 defined by the intersection of rays y_2 and y_4. Thus from p_3 we can get y_4. We now show how to compute p_3.

We want a point p_3 on a ray (vp_1, p_1) such that the aspect ratio $\frac{||\hat{p}_2 - \hat{p}_1||}{||\hat{p}_3 - \hat{p}_1||}$ equals a pre-specified value a. Here $\hat{p}$ denotes a point in the fronto-parallel view. Since p_1 and p_2 lie on a line through vp_0, let's express p_2 as follows: $p_2 = p_1 + \lambda(p_1 - \mathrm{vp}_0)$, where λ is easily computable from the two points. Since p_3 lies on a line through vp_1 and p_1, let: $p_3 = p_1 + \mu(p_1 - \mathrm{vp}_1)$. Via projective geometry we can then compute μ as

$$\mu = \frac{1}{\tilde{a} \cdot \left|1 + \frac{1}{\lambda}\right| - 1}, \quad \mu > 0. \tag{2}$$

Note that in cases when the denominator is negative, the rectangle formed by p_1, p_2 and p_3 is not valid (it crosses the infinity point). Here $\tilde{a} = a \cdot f_{01}$ and $f_{01} = \frac{||H\mathrm{vp}_0||}{||H\mathrm{vp}_1||}$. We refer the interested reader to the supplementary material for a more detailed derivation.

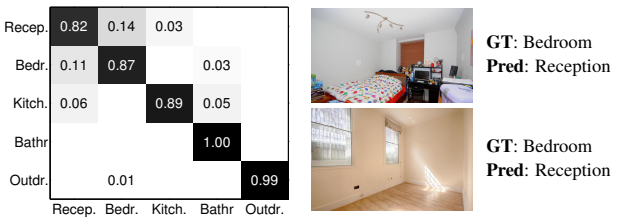

Figure 5. **Left**: Scene confusion matrix. **Right:** Examples of misclassification. Note that some cases are ambiguous since the room is empty and the annotator made use of the floor plan to label it.

	Train		Test	
	Top1	Top5	Top1	Top5
Ours (aspect only)	0.2429	0.3887	0.2563	0.4050
Ours (windowGT)	0.3320	0.7166	0.3249	0.7529
Ours (window)	0.3252	0.7139	0.3249	0.7414

Table 4. Accuracy for predicting the room correctly ($C = 1000$). Top 1 denotes how many of our top scoring predictions are correct. Top 5 evaluates accuracy if we are allowed to make 5 predictions.

6. Experimental Evaluation

We evaluate our approach on our Rent3D dataset. We use the first 100 apartments (751 photos) as our training set, the next 30 apartments (222 photos) as a validation set, and the last 85 apartments (597 photos) as test set. We evaluate different instantiations of the model, and propose different metrics to evaluate localization. We start our evaluation by first providing details on the features used in our model. In particular, we explain how we trained our scene and semantic segmentation classifiers.

6.1. Scene Classifier

In order to classify the scene into the five different types *reception*, *bedroom*, *kitchen*, *bathroom*, and *outdoor* we utilize Caffe [13] and the pre-trained model provided by [30]. We extract the fc7 feature (length 4096) for each image and train a multi-class support vector machine (SVM). Table 6 shows the confusion matrix showing good accuracy of 91.4% on our 5-class setting.

6.2. Window Segmentation

We train our pixel-level predictors for *door*, *window* and *other* using the approach by Ren *et al.* [20], where we add

	C	Train	Test	Train Eval	Train Time [s]	Test Eval	Test Time [s]
[24] (no floor plan), Transfer	1	16.70	17.00	20438.6	0.0190	22266.2	0.0208
[24] (no floor plan), Train on data	1	13.57	13.88	15643.3	0.0147	16012.4	0.0150
Ours (aspect only)	1	11.80	11.81	1164.6	0.0017	1269.5	0.0019
Ours (windowGT)	1	11.78	11.79	1144.3	0.0025	1250.0	0.0026
Ours (windowGT)	10	11.76	11.86	1158.0	0.0023	1263.4	0.0025
Ours (windowGT)	100	11.74	11.79	1199.4	0.0024	1322.8	0.0026
Ours (windowGT)	1000	11.74	11.79	1199.4	0.0024	1322.8	0.0026
Ours (window)	1	11.73	11.90	1149.3	0.0027	1258.7	0.0029
Ours (window)	10	11.78	11.83	1180.9	0.0035	1292.3	0.0035
Ours (window)	100	11.69	11.90	1164.1	0.0031	1271.5	0.0037
Ours (window)	1000	11.69	11.90	1164.1	0.0031	1271.5	0.0037

Table 1. Layout pixelwise prediction error for different methods. Average number of branch&bound evaluations and average inference time is also given. By using the aspect ratio constraint, our model runs an order of magnitude faster than [24] and results in better prediction.

	Accuracy Train			Accuracy Test		
	Window+Aspect	+Scene	+Room	Window+Aspect	+Scene	+Room
Random	0.0315	0.1108	0.1818	0.0328	0.1138	0.1954
Ours (aspect only)	0.0472	0.1538	0.2213	0.0686	0.1945	0.2654
Ours (windowGT)	0.2321	0.4683	0.5911	0.2128	0.4737	0.5995
Ours (window)	0.2051	0.4305	0.5587	0.1670	0.3982	0.5080

Table 2. Localization accuracy: Comparing different settings of our proposed approach, *i.e.*, "Window+Aspect", "Window+Aspect+Scene" and "Window+Aspect+Room". We also provide random guessing as a baseline. We use $C = 1000$ in S-SVM training.

Figure 6. Failure mode: Biggest apartment in dataset (16 rooms, 5 bedrooms, 88 walls). (**left**) our 3D recons., (**right**) GT recons.

an additional feature (Geometric Context). Note that the "other" label takes the majority of the photo, so the 3 labels have very unbalanced number of pixels. We compute the accuracy as intersection-over-union following the standard PASCAL evaluation. The result on test is 0.4% for door, 51.75% for window, and 95.6% for other. Since the door classifier performs very poorly, we decided to only incorporate windows in our model.

6.3. Layout Estimation

We next evaluate layout estimation using a pixelwise classification error as our metric. In this experiment, we do not reason about the localization in the apartment and only evaluate layout prediction as the isolated task. We use several baselines. We first run the state-of-the-art layout approach of [24], trained on the indoor dataset by Hedau *et al*. [8], on our Rent3D dataset. This first baseline is denoted as "Transfer" and its performance results on our training and test set are given in Table 1. With a test set error of 16.97% [24] obtains a decent performance, even though the predictor has not seen our data in training. As a second baseline we train [24] on our dataset. We refer to this method as "Train on data" which improves both training and test error by more than 3%.

We evaluate 3 different settings of our model, one where we only use wall aspect ratio information, one where we use GT windows and one with the trained window classifier. In this experiment we assume we know which wall the camera is facing, and thus our model imposes the correct aspect ratio on the front wall. The results are in Table 1. Performance improves around 2% irrespective of whether we employ GT window annotation or the classifier output. This highlights the robustness of our model. We show accuracy at different C values used in S-SVM training.

In Table 1 we also compare the number of evaluations

	Top5 Accuracy, **Train**			Top5 Accuracy, **Test**		
	Window+Aspect	+Scene	+Room	Window+Aspect	+Scene	+Room
Ours (aspect only)	0.2348	0.6343	0.8475	0.2769	0.6682	0.8970
Ours (windowGT)	0.5803	0.8003	0.9366	0.5858	0.8009	0.9519
Ours (window)	0.5398	0.7895	0.9312	0.5492	0.7895	0.9474

Table 3. Top 5 accuracy for predicting which wall in an apartment the camera is facing for different information used in the model, *i.e.*, "Window+Aspect", "Window+Aspect+Scene" and "Window+Aspect+Room". We use $C = 1000$ in S-SVM training.

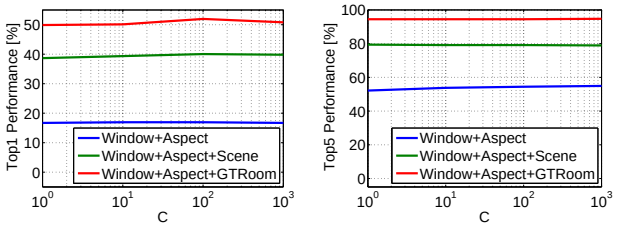

Figure 7. Top 1 (left) and Top 5 test set performance over a large range of values for the regularization parameter C when employing predicted window features.

required for the branch and bound method to attain the optimum. Additionally we provide the average time for inference. We observe a reduction of both the number of evaluations and the inference time by more than an order of magnitude on both the training and the test set over [24]. This is because our model has one less random variable (it's deterministic given the aspect ratio).

6.4. Localization

In this experiment we use our model in order to predict which room and wall the camera is facing. We evaluate the effect of using different features in our model. We employ a simple metric which counts the prediction as correct if the localization in terms of both the room and front wall matches the ground-truth alignment. We report the results in Table 2. Since there is no competing approach we provide random guessing as a baseline for both the training and the test set. On the training set we achieve an accuracy of about 20% compared to a test set performance of above 17%.

In columns we add features to our model. By +Scene we denote the case where the scene classifier is included in our potentials. Notice that the scene feature more than doubles the performance. We also evaluate +Room, in which case we give the model information in which room the photo was taken but not which wall the camera was facing. The random baseline in this case is 18%, which is due to the fact that some rooms are not cuboids and have more walls.

To better assess the performance of our approach we provide the top 5 wall prediction accuracy in Table 3. In this case we take top 5 predictions from our model and count the assignment as correct if any of the predictions is correct. In Table 4 we evaluate the performance of correctly predicting the room in which each photo was taken. We see that windows have a huge impact on performance.

In Fig. 7 we visualize the top 1 (we are only allowed one guess) prediction performance (left) and the top 5 prediction performance on the test set over a large range of values for the regularization parameter C. For this experiment we used the predicted window features. We observe that the performance of the different approaches is very robust w.r.t. the choice of regularization.

6.5. Visual Results

We also provide some qualitative results in Fig. 8. We show the apartment 3D reconstructions obtained from our model. We show results for three settings "window+aspect", "window+aspect+scene", and "window+aspect+room", as well as GT. Information about the number of images (which room the photo was taken in) and walls (which wall the camera was facing) which were predicted correctly for the full apartment is provided.

We show the layout and localization (alignment) results in Fig. 9. Solid faces are predictions from our model and dashed lines are GT. Color coding conveys different walls.

7. Conclusion

We have proposed an approach that enables a 3D "virtual-tour" of an apartment given a small number of monocular images of different rooms as well as a floor plan. Our approach exploits the floor plan to impose aspect ratio constraints across the layouts of different rooms, as well as to extract semantic information, *e.g.*, the location of windows and scene types. We showed that this information significantly helps in resolving the challenging localization problem. We derived an efficient exact inference algorithm which takes only a few milliseconds per apartment. We demonstrated the effectiveness of our approach in a new dataset which contains over 200 apartments. In the future we plan to exploit other objects such as ovens and toilets which are typically also labeled in floor-plans, and provide additional cues for solving this challenging problem.

Acknowledgments

This work was partially supported by ONR-N00014-14-1-0232 and the Undergraduate Overseas Research Training Program of Tsinghua Univ. We thank Qiurui He, Cong Ma, Long Qian, Jun Yang, Liang Ruan, Yansong Tang, Yifan Hou, Qianlin Tang, Kai Chen, Weishen Pan, Shenlong Wang, Liang-Chieh Chen, Yinan Zhao, and Yukun Zhu for helping us annotate the Rent3D dataset.

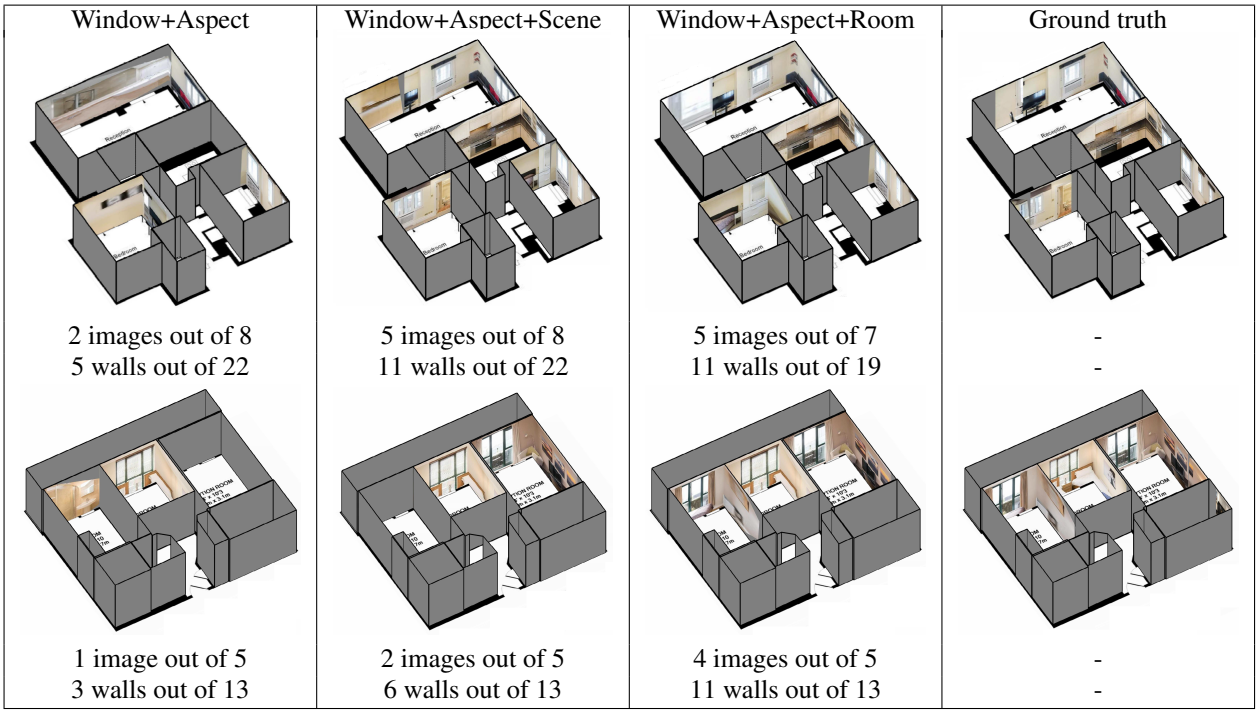

Figure 8. For different settings of our model, "window+aspect", "window+aspect+scene", "window+aspect+room", as well as for GT we illustrate the apartment reconstructions in 3D. Below each we provide the numbers of how many images and walls were matched correctly: "x images out of y" means that out of a total of y indoor photos for this apartment, x of them have correct front-wall alignment. "a walls out of b" means that out of a total of b visible walls, a of them are correctly aligned.

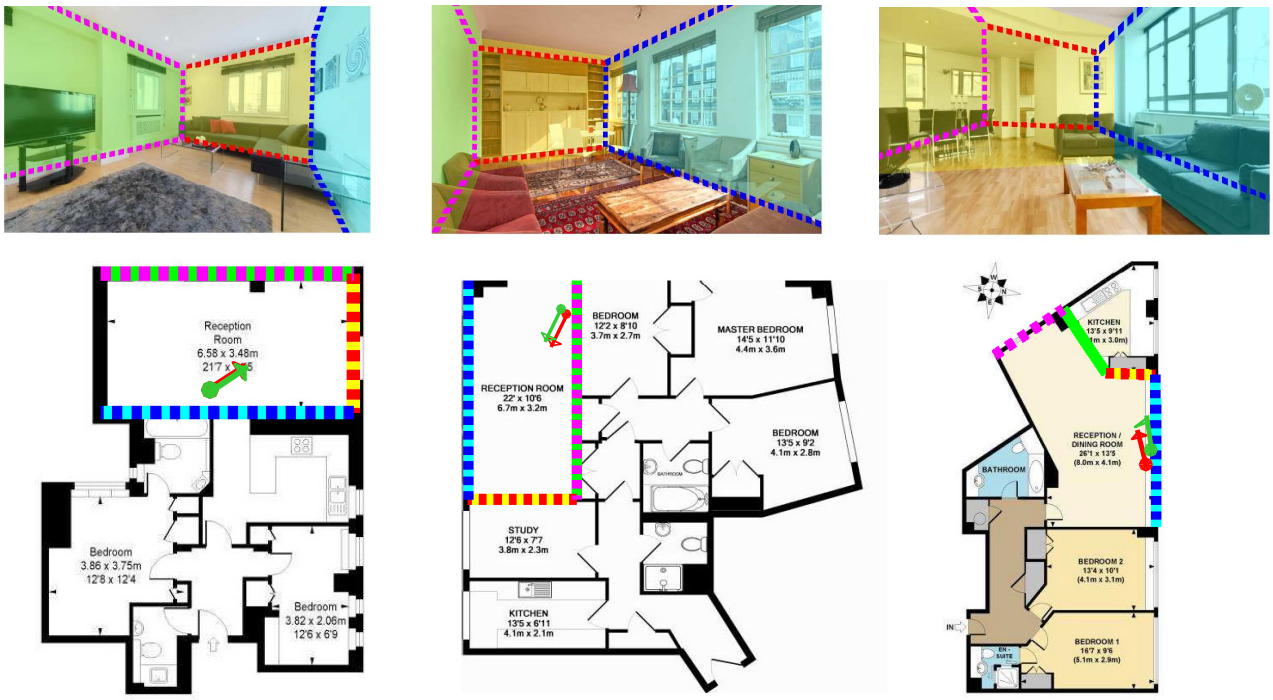

Figure 9. Room layout estimation and room-apartment alignment with our model. The top row of images color code the estimated vertical walls, and ground-truth layout is shown with dashed lines. In the bottom row, we show the floor plans with predicted and GT alignments. The solid colors show the alignment predicted by the model, where the yellow solid line denotes the alignment prediction for the front face. Similarly, GT alignment is shown with dashed lines. A red point shows the GT location of the camera, and the arrow is the viewing direction. Green point and line is our estimated localization. On the right we show a failed example due to a weird shape of the room.

References

[1] M. A. Brubaker, A. Geiger, and R. Urtasun. Lost! leveraging the crowd for probabilistic visual self-localization. In *CVPR*, 2013. 2

[2] R. Cabral and Y. Furukawa. Piecewise Planar and Compact Floorplan Reconstruction from Images. In *Proc. CVPR*, 2014. 2

[3] V. Delaitre, D. F. Fouhey, I. Laptev, J. Sivic, A. Gupta, and A. A. Efros. Scene semantics from long-term observation of people. In *Proc. ECCV*, 2012. 2

[4] S. Fidler, S. Dickinson, and R. Urtasun. 3D Object Detection and Viewpoint Estimation with a Deformable 3D Cuboid Model. In *Proc. NIPS*, 2012. 2

[5] Y. Furukawa, B. Curless, S. M. Seitz, and R. Szeliski. Reconstructing Building Interiors from Images. In *Proc. ICCV*, 2009. 2

[6] A. Gupta, A. A. Efros, and M. Hebert. Blocks World Revisited: Image Understanding Using Qualitative Geometry and Mechanics. In *Proc. ECCV*, 2010. 2

[7] A. Gupta, S. Satkin, A. A. Efros, and M. Hebert. From 3D Scene Geometry to Human Workspace. In *Proc. CVPR*, 2011. 2

[8] V. Hedau, D. Hoiem, and D. Forsyth. Recovering the Spatial Layout of Cluttered Rooms. In *Proc. ICCV*, 2009. 1, 2, 3, 6

[9] V. Hedau, D. Hoiem, and D. Forsyth. Thinking Inside the Box: Using Appearance Models and Context Based on Room Geometry. In *Proc. ECCV*, 2010. 2

[10] V. Hedau, D. Hoiem, and D. Forsyth. Recovering Free Space of Indoor Scenes from a Single Image. In *Proc. CVPR*, 2012. 2

[11] D. Hoiem, A. A. Efros, and M. Hebert. Putting objects in perspective. *IJCV*, 2008. 1

[12] D. Hoiem, A. A. Efros, and M. Hebert. Putting Objects in Perspective. *IJCV*, 2008. 3

[13] Y. Jia. Caffe: An Open Source Convolutional Architecture for Fast Feature Embedding. http://caffe.berkeleyvision.org/, 2013. 5

[14] D. C. Lee, A. Gupta, M. Hebert, and T. Kanade. Estimating Spatial Layout of Rooms using Volumetric Reasoning about Objects and Surfaces. In *Proc. NIPS*, 2010. 2, 3

[15] D. Marr and H. K. Nishihara. Representation and recognition of the spatial organization of three dimensional shapes. *Proc. of the Royal Soc. of London B*, 1978. 1

[16] R. Martin-Brualla, Y. He, B. C. Russell, and S. M. Seitz. The 3D Jigsaw Puzzle: Mapping Large Indoor Spaces. In *Proc. ECCV*, 2014. 2

[17] K. Matzen and N. Snavely. Nyc3dcars: A dataset of 3d vehicles in geographic context. In *ICCV*, 2013. 2

[18] L. Pero, J. Bowdish, D. Fried, B. Kermgard, E. Hartley, and K. Barnard. Bayesian geometric modeling of indoor scenes. In *Proc. CVPR*, 2012. 2

[19] L. Pero, J. Guan, E. Brau, J. Schlecht, and K. Barnard. Sampling Bedrooms. In *Proc. CVPR*, 2011. 2

[20] X. Ren, L. Bo, and D. Fox. Scene Labeling: Features and Algorithms. In *Proc. CVPR*, 2012. 4, 5

[21] C. Rother. A new approach for vanishing point detection in architectural environments. In *Proc. BMVC*, 2000. 3, 5

[22] A. G. Schwing, S. Fidler, M. Pollefeys, and R. Urtasun. Box In the Box: Joint 3D Layout and Object Reasoning from Single Images. In *Proc. ICCV*, 2013. 1, 5

[23] A. G. Schwing, T. Hazan, M. Pollefeys, and R. Urtasun. Efficient Structured Prediction for 3D Indoor Scene Understanding. In *Proc. CVPR*, 2012. 3

[24] A. G. Schwing and R. Urtasun. Efficient Exact Inference for 3D Indoor Scene Understanding. In *Proc. ECCV*, 2012. 2, 4, 6, 7

[25] B. Taskar, C. Guestrin, and D. Koller. Max-Margin Markov Networks. In *Proc. NIPS*, 2003. 2

[26] I. Tsochantaridis, T. Joachims, T. Hofmann, and Y. Altun. Large margin methods for structured and interdependent output variables. *JMLR*, 2005. 2, 5

[27] H. Wang, S. Gould, and D. Koller. Discriminative Learning with Latent Variables for Cluttered Indoor Scene Understanding. In *Proc. ECCV*, 2010. 2

[28] S. Wang, S. Fidler, and R. Urtasun. Holistic 3d scene understanding from a single geo-tagged image. In *CVPR*, 2015. 2

[29] J. Xiao and Y. Furukawa. Reconstructing the World's Museums. In *Proc. ECCV*, 2012. 2

[30] B. Zhou, A. Lapedriza, J. Xiao, A. Torralba, and A. Oliva. Learning Deep Features for Scene Recognition using Places Database. In *Proc. NIPS*, 2014. 5