Parsing IKEA Objects: Fine Pose Estimation

Joseph J. Lim Hamed Pirsiavash Antonio Torralba
Massachusetts Institute of Technology
Compter Science and Artificial Intelligence Laboratory
{lim,hpirsiav,torralba}@csail.mit.edu

Abstract

We address the problem of localizing and estimating the fine-pose of objects in the image with exact 3D models. Our main focus is to unify contributions from the 1970s with recent advances in object detection: use local keypoint detectors to find candidate poses and score global alignment of each candidate pose to the image. Moreover, we also provide a new dataset containing fine-aligned objects with their exactly matched 3D models, and a set of models for widely used objects. We also evaluate our algorithm both on object detection and fine pose estimation, and show that our method outperforms state-of-the art algorithms.

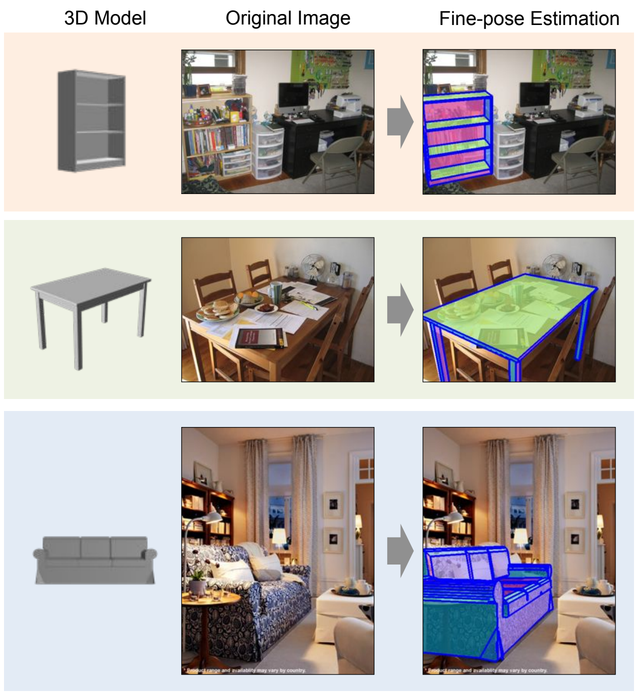


Figure 1. Our goal in this paper is to detect and estimate the fine-pose of an object in the image given an exact 3D model.

1. Introduction

Just as a thought experiment imagine that we want to detect and fit 3D models of IKEA furniture in the images as shown in Figure 1. We can find surprisingly accurate 3D models of IKEA furniture, such as *billy* bookcase and *ektorp* sofa, created by IKEA fans from Google 3D Warehouse and other publicly available databases. Therefore, detecting those models in images could seem to be a very similar task to an instance detection problem in which we have training images of the exact instance that we want to detect. But, it is not exactly like detecting instances. In the case of typical 3D models (including IKEA models from Google Warehouse), the exact appearance of each piece is not available, only the 3D shape is. Moreover, appearance in real images will vary significantly due to a number of factors. For instance, IKEA furniture might appear with different colors and textures, and with geometric deformations (as people building them might not do it perfectly) and occlusions (e.g., a chair might have a cushion on top of it).

The problem that we introduce in this paper is detecting and accurately fitting exact 3D models of objects to real images, as shown in Figure 1. Detecting 3D objects in images and estimating their pose was a popular topic in the early days of computer vision [14] and has gained a renewed interest in the last few years. The traditional approaches [12, 14] were dominated by using accurate geometric representations of 3D objects with an emphasis on viewpoint invariance. Objects would appear in almost any pose and orientation in the image. Most of the approaches were designed to work at instance level detection as it was expected that the 3D model would accurately fit the model in the image. Instance level detection regained interest with the introduction of new invariant local descriptors that dramatically improved the detection of interest points [17]. Those models assumed knowledge about geometry and appearance of the instance. Having access to accurate knowledge about those two aspects allowed precise detections and pose estimations of the object on images even in the presence of clutter and occlusions.

In the last few years, researchers interested in category

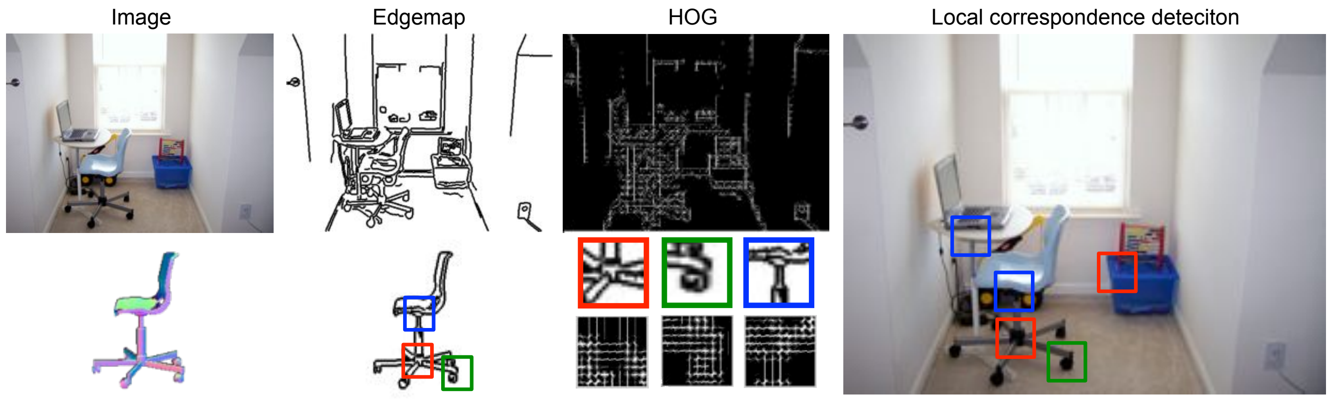


Figure 2. **Local correspondence:** for each 3D interest point $\mathbf{X}_i$ (red, green, and blue), we train an LDA patch detector on an edgemap and use its response as part of our cost function. We compute HOG on edgemaps to ensure a real image and our model share the modality.

level detection have extended 2D constellation models to include 3D information in the object representation. Many of these models [9, 20, 10, 6, 5, 19, 16, 7, 21] rely on gradient-based features [1, 13]. Category level detection requires the models to be generic and flexible, as they have to deal with all the variations in shape and appearance of the instances that belong to the same category. Therefore, the shape representations used for those models are coarse (e.g., modeled as the constellation of a few 3D parts or planes) and the expected output is at best an approximate fitting of the 3D model to the image.

In this paper we introduce a detection task that is in the intersection of these two settings; it is more generic than detecting instances, but we assume richer models than the ones typically used in category level detection. In particular, we assume that accurate CAD models of the objects are available. Hence, there is little variation on the shape of the instances that form one category. However, there might be large variation in the appearance, as the CAD models do not completely constrain the appearance of the objects placed in the real world. Although assuming that CAD models are available might seem to be a restrictive assumption, there are available 3D CAD models for most man-made artifacts, used for manufacturing or virtual reality. All those models could be used as training data for our system. Hence, we focus on detection and pose estimation of objects *in the wild* given their 3D CAD models. Our goal is to provide an accurate localization of the object, as in the instance level detection problem, but dealing with some of the variability that one finds in category level detection.

Our contributions are three folds: (1) Proposing a detection problem that has some of the challenges of category level detection while allowing for an accurate representation of the object pose in the image. This problem will motivate the development of better 3D object models and the algorithms needed to find them in images. (2) We propose a novel solution that unifies contributions from the 1970s with recent advances in object detection. (3) And we introduce a new dataset of 3D IKEA models obtained from Google Warehouse and real images containing instances of IKEA furniture and annotated with ground truth pose.

2. Methods

We now propose the framework that can detect objects and estimate their poses simultaneously by matching to one of the 3D models in our database.

2.1. Our Model

The core of our algorithm is to define the final score based on multiple sources of information such as local correspondence, geometric distance, and global alignment. Suppose we are given the image I containing an object for which we want to estimate the projection matrix P with 9 degrees of freedom [1]. We define our cost function S with three terms as follows:

$$S(P,c) = L(P,c) + w^D D(P,c) + w^G G(P) \tag{1}$$

where c refers to the set of correspondences, L measures error in local correspondences between the 3D model and 2D image, D measures geometric distance between correspondences in 3D, and G measures the global dissimilarity in 2D. Note that we designed our model to be linear in weight vectors of w^D and w^G. We will use this linearity later to learn our discriminative classifier.

2.2. Local correspondence error

The goal here is to measure the local correspondences. Given a projection P, we find the local shape-based matching score between the rendered image of the CAD model and the 2D image. Because our CAD model contains only

[1]We assume the image is captured by a regular pinhole camera and is not cropped, so the the principal point is in the middle of image.

shape information, a simple distance measure on standard local descriptors fails to match the two images. In order to overcome this modality difference, we compute HOG on the edgemap of both images since it is more robust to appearance change but still sensitive to the change in shape. Moreover, we see better performance by training an LDA classifier to discriminate each keypoint from the rest of the points. This is equivalent to a dot product between descriptors in the whitened space. One advantage of using LDA is its high training speed compared to other previous HOG-template based approaches. Figure 2 illustrates this step.

More formally, our correspondence error is measured by

$$L(P,c) = \sum_i H(\beta_i^T \Phi(\mathbf{x}_{c_i}) - \alpha) \quad (2)$$

$$\beta_i = \Sigma^{-1}(\Phi(P(\mathbf{X}_i)) - \mu) \quad (3)$$

where β_i is the weight learned using LDA based on the covariance Σ and mean μ obtained from a large external dataset [8], $\Phi(\cdot)$ is a HOG computed on a 20×20 pixel edgemap patch of a given point, $\mathbf{x}_{c_i}$ is the 2D corresponding point of 3D point $\mathbf{X}_i$, $P(\cdot)$ projects a 3D point to 2D coordinate given pose P, and lastly $H(x-\alpha)$ binarizes x to 0 if $x \geq \alpha$ or to ∞ otherwise.

2.3. Geometric distance between correspondences

Given a proposed set of correspondences c between the CAD model and the image, it is also necessary to measure if c is a geometrically acceptable pose based on the 3D model. For the error measure, we use euclidean distance between the projection of $\mathbf{x}_i$ and its corresponding 2D point $\mathbf{x}_{c_i}$; as well as the line distance defined in [12] between the 3D line l and its corresponding 2D line c_l.

$$D(P,c) = \sum_{i \in V} \|P(\mathbf{X}_i) - \mathbf{x}_{c_i}\|^2 + \quad (4)$$
$$\sum_{l \in L} \left\| \left[\cos\phi_{c_l}, \sin\phi_{c_l}, -\rho_{c_l} \right] \left[\begin{array}{cc} P(\mathbf{X}_{l_1}) & P(\mathbf{X}_{l_2}) \\ 1 & 1 \end{array} \right] \right\|^2$$

where $\mathbf{X}_{l_1}$ and $\mathbf{X}_{l_2}$ are end points of line l, and ϕ_{c_l} and ρ_{c_l} are polar coordinate parameters of line c_l.

The first term measures a pairwise distance between a 3D interest point $\mathbf{X}_i$ and its 2D corresponding point $\mathbf{x}_{c_i}$. The second term measures the alignment error between a 3D interest line l and one of its corresponding 2D lines c_l [12]. We use the Hough transform on edges to extract 2D lines from I. Note that this cost does not penalize displacement along the line. This is important because the end points of detected lines on I are not reliable due to occlusion and noise in the image.

2.4. Global dissimilarity

One key improvement of our work compared to traditional works on pose estimation using a 3D model is how we measure the global alignment. We use recently developed features to measure the global alignment between our proposal and the image:

$$G(P) = [f_{HOG}, f_{region}, f_{edge}, f_{corr}, f_{texture}], \quad (5)$$

The description of each feature follows:

HOG-based: While D and L from Eq 1 are designed to capture fine local alignments, the local points are sparse and hence it is yet missing an object-scale alignment. In order to capture edge alignment per orientation, we compute a fine HOG descriptor (2x2 per cell) of edgemaps of I and the rendered image of pose P. We use a similarity measure based on cosine similarity between vectors .

$$f_{HOG} = \left[\frac{\Phi(I)^T\Phi(P)}{\|\Phi(P)\|^2}, \frac{\Phi(I)^T\Phi(P)}{\|\Phi(I)M\|^2}, \right.$$
$$\left. \frac{\Phi(I)^T\Phi(P)}{\|\Phi(I)M\|\|\Phi(P)\|} \right], \quad (6)$$

where $\Phi(\cdot)$ is a HOG descriptor and M is a mask matrix for counting how many pixels of P fall into each cell. We multiply $\phi(I)$ by M to normalize only within the proposed area (by P) without being affected by other area.

Regions: We add another alignment feature based on super-pixels. If our proposal P is a reasonable candidate, super-pixels should not cross over the object boundary, except in heavily occluded areas. To measure this spill over, we first extract super-pixels, R_I, from I using [3] and compute a ratio between an area of a proposed pose, $|R_P|$, and an area of regions that has some overlap with R_P. This ratio will control the spillover effect as shown in Figure 3.

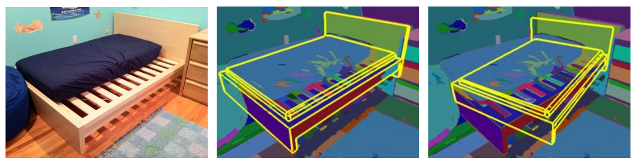

(a) Original image (b) Candidate 1 (c) Candidate 2

Figure 3. **Region feature:** One feature to measure a fine alignment is the ratio between areas of a proposed pose (yellow) and regions overlapped with the proposed pose. (a) is an original image I, and (b) shows an example to encourage, while (c) shows an example to penalize.

$$f_{region} = \left[\frac{\sum_{|R_P \cap R_I| > 0.1|R_P|} |R_P|}{|R_I|} \right] \quad (7)$$

Texture boundary: The goal of this feature is to capture appearance by measuring how well our proposed pose separates object boundary. In other words, we would like to measure the alignment between the boundary of our proposed pose P and the texture boundary of I. For this purpose, we use a well-known texture classification feature, Local Binary Pattern (LBP) [15].

We compute histograms of LBP on inner boundary and outer boundary of proposed pose P. We define an inner boundary region by extruding the proposed object mask followed by subtracting the mask, and an outer boundary by diluting the proposed object mask. Essentially, the histograms will encode the texture patterns of near-inner/outer pixels along the proposed pose P's boundary. Hence, a large change in these two histograms indicates a large texture pattern change, and it is ideal if the LBP histogram difference along the proposed pose's boundary is large. This feature will discourage the object boundary aligning with contours with small texture change (such as contours within an object or contours due to illumination).

Edges: We extract edges [11] from the image to measure their alignment with the edgemap of estimated pose. We used a modified Chamfer distance from Satkin, et. al. [18].

$$f_{edge} = \Bigg[\frac{1}{|R|}\sum_{a\in R}\min(\min_{b\in I}\|a-b\|,\tau),\\ \frac{1}{|I|}\sum_{b\in I}\min(\min_{a\in R}\|b-a\|,\tau)\Bigg] \quad (8)$$

where we use $\tau \in \{10, 25, 50, \infty\}$ to control the influence of outlier edges.

Number of correspondences: f_{corr} is a binary vector, where the i'th dimension indicates if there are more than i good correspondences between the 3D model and the 2D image under pose P. Good correspondences are the ones with local correspondence error (in Eq 2) below a threshold.

2.5. Optimization & Learning

Algorithm 1 Pose set $\{P\}$ search

```
For each initial seed pose,
while less than $n$ different candidates found do
    Choose a random 3D interest point and its 2D correspondence (this determines a displacement)
    for i = 1 to 5 do
        Choose a random local correspondence agreeing with correspondences selected (over $i-1$ iterations)
    end for
    Estimate parameters by solving least squares
    Find all correspondences agreeing with this solution
    Estimate parameters using all correspondences
end while
```

Our cost function $S(P, c)$ from Eq 1 with $G(P)$ is a non-convex function and is not easy to solve directly. Hence, we first simplify the optimization by quantizing the space of solutions. Because our $L(P, c)$ is ∞ if any local correspondence score is below the threshold, we first find all sets of correspondences for which all local correspondences are above the threshold. Then, we find the pose P that minimizes $L(P, c) + w^D(P, c)$. Finally, we optimize the cost function in the discrete space by evaluating all candidates.

We use RANSAC in populating a set of candidates by optimizing $L(P, c)$. Our RANSAC procedure is shown in Alg 1. We then minimize $L(P, c) + w^D D(P, c)$ by estimating pose P for each found correspondence c. Given a set of correspondences c, we estimate pose P using the Levenberg-Marquardt algorithm minimizing $D(P, c)$.

We again leverage the discretized space from Alg 1 in order to learn weights for Eq 1. For the subset of $\{(P, c)\}$ in the training set, we extract the geometric distance and global alignment features, $[D(P, c), G(P)]$, and the binary labels based on the distance from the ground truth (defined in Eq 9) . Then, we learn the weights using a linear SVM classifier.

3. Evaluation

3.1. Dataset

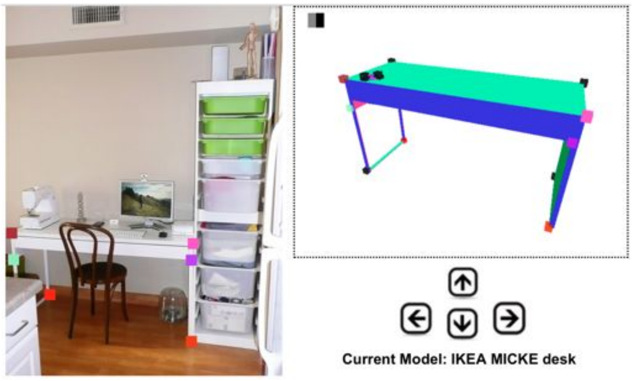
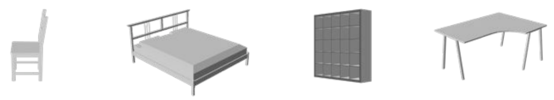


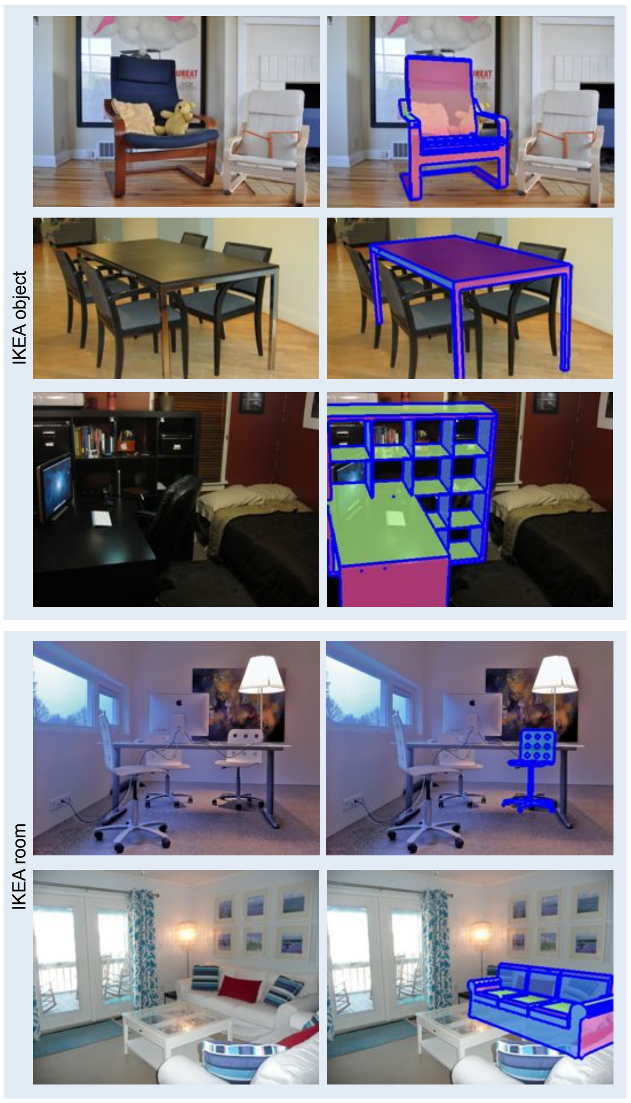
Figure 4. **Labeling tool:** our labeling tool enables users to browse through 3D models and label point correspondences to an image. The tool provides a feedback by rendering estimated pose on the image and an user can edit more correspondences.

In order to develop and evaluate fine pose estimation based on 3D models, we created a new dataset of images and 3D models representing typical indoor scenes. We explicitly collected IKEA 3D models from Google 3D Warehouse, and images from Flickr. The key difference of this dataset from previous works [18, 20] is that we align exact 3D models with each image, whereas others provided coarse pose information without using exact 3D models.

For our dataset, we provide 800 images and 225 3D models. All 800 images are fully annotated with 90 dif-



(a) 3D models



(b) Aligned Images

Figure 5. **Dataset:** (a) examples of 3D models we collected from Google Warehouse, and (b) ground truth images where objects are aligned with 3D models using our labeling tool. For clarity, we show only one object per image when there are multiple objects.

ferent models. Also, we separate the data into two different splits: **IKEAobject** and **IKEAroom**. IKEAobject is the split where 300 images are queried by individual object name (*e.g. ikea chair poang* and *ikea sofa ektorp*). Hence, it tends to contain only a few objects at relatively large scales. IKEAroom is the split where 500 images are queried by *ikea room* and *ikea home*; and contains more complex scene where multiple objects appear at a smaller scale. Figure 5ab show examples of our 3D models and annotated images.

For alignment, we created an online tool that allows a user to browse through models and label point correspondences (usually 5 are sufficient), and check the model's estimated pose as the user labels. Given these correspondences, we solve the least square problem of Eq 4 using Levenberg-Marquardt. Here, we obtain the full intrinsic/extrinsic parameters except the skewness and principal points. Figure 4 shows a screenshot of our tool.

3.2. Error Metric

We introduce a new error metric for fine-pose estimation. Intuitively, we use the average 3D distance between all points in the ground truth and the proposal. When the distance is small, this is close to the average error in viewing angle for all points. Formally, given an estimated pose P_e and a ground truth pose P_{gt} of image I, we obtain corresponding 3D points in the camera space. Then, we compute the average pair-wise distance between all corresponding points divided by their distance to the camera. We consider P is correct if this average value is less than a threshold.

$$score(P_e, P_{gt}) = \frac{\sum_{X_i} \|E_e X_i - E_{gt} X_i\|_2}{\sum_{X_i} \|E_{gt} X_i\|_2} \qquad (9)$$

4. Results

4.1. Correspondences

First of all, we evaluate our algorithm on finding good correspondences between a 3D model and an image. This is crucial for the rest of our system as each additional poor correspondence grows the search space of RANSAC exponentially.

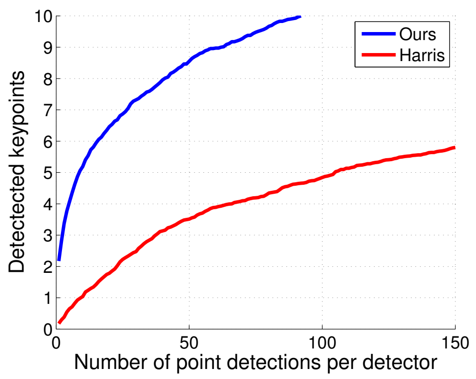


Figure 6. **Correspondence evaluation:** we are comparing correspondences between our interest point detector and Harris detector. The minimum number of interest points we need for reliable pose estimation is 5. Ours can recall 5 correct correspondences by considering only the top 10 detections per 3D interest point, whereas the Harris detector requires 100 per point. This results in effectively 10^5 times fewer search iterations in RANSAC.

Figure 6 shows our evaluation. We compare Harris corner detectors against our detector based on LDA classifiers. On average, to capture 5 correct correspondences with our method, each interest point has to consider only its top 10 matched candidates. In contrast, using the Harris corner detector, one needs to consider about 100 candidates. This helps finding good candidates in the RANSAC algorithm.

4.2. Initial Poses

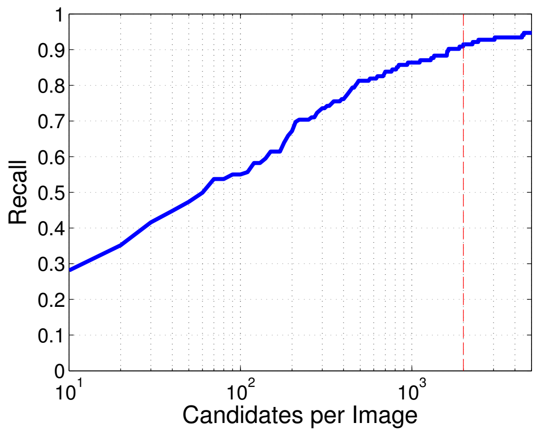


Figure 7. **RANSAC evaluation:** we evaluated how many pose candidates per image we need in order to obtain a certain recall. We need only about 2000 candidates in order to have 0.9 recall. This allows us to perform a computationally expensive feature extraction on only a small number of candidates.

In order for us to find the pose P minimizing $S(P, c)$, we need to ensure that our set of poses $\{(P, c)\}$ minimizing $L(P, c) + D(P, c)$ contains at least one correct pose. The recall of our full model is upper bonded by that of this set. Figure 7 shows a semi-log plot of recall vs minimum number of top candidates required per image. Considering the top 2000 candidate poses (shown with a red line) from RANSAC, we can obtain 0.9 recall. In other words, the later optimization, where feature extraction is computationally heavy, can run with only the top 2000 candidate poses and still have a recall 0.9.

4.3. Final Pose Estimation

Table 1 shows the evaluation of our method with various sets of features as well as two state-of-the-art object detectors: Deformable part models [4] and Exemplar LDA [8] in IKEAobject database.

For both [4] and [8], we trained detectors using the code provided by the authors. The training set contains rendered images (examples are shown in Figure 5b) in order to cover various poses (which are not possible to cover with real images). Note the low performances of [4] and [8]. We believe there are two major issues: (1) they are trained with a very large number of mixtures. If there are some mixtures with high false positive rates, then the whole system can break down, and (2) they are trained with rendered images due to their requirement of images for each different mixture. Having a different modality can result in a poor performance.

Our average precision (AP) is computed based on our error metric (Eq. 9). We score each pose estimation P as a true detection if its normalized 3D space distance to the ground truth is within the threshold; and for the rest, we precisely follow a standard bounding box criterion [2]. Here, we would like to emphasize that this is a much harder task than a typical bounding box detection task. Hence, low AP in the range of 0-0.02 should not be misleading.

$D + L$, based on RANSAC, has a low performance by itself. Though it is true that the top candidates generally contain a correct pose (as shown in Figure 7), it is clear that scores based only on local correspondences (D and L) are not robust to false positives, despite high recall.

We also evaluated two other features based on HOG and Region features. Adding these two features greatly boosts performance. HOG and Region features were added for capturing global shape similarity. Our full method takes about 12 minutes on a single core for each image and model.

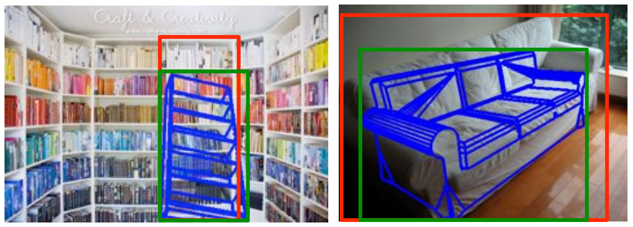


Figure 8. **Corner cases:** these two images illustrate cases which are incorrect under our pose estimation criteria, but are still correct under standard bounding box detection criterion.

Also, we compared our method against [4, 8] on a detection task as shown in Table 2. For our method, we ran the same full pipeline and extracted bounding boxes from the estimated poses. For the detection task, we trained [4, 8] with real images, because there exist enough real images to train a small number of mixtures. We train/test on 10 different random splits. Because our method is designed for capturing a fine alignment, we measured at two different thresholds on bounding box intersection over union. One is the standard threshold of 0.5 and the other one is a higher threshold of 0.8. As the table shows, our method does not fluctuate much with a threshold change, whereas both [8] and [4] suffer and drop performances significantly. Figure 8 shows several detection examples where pose estimation is incorrect, but still bounding box estimation is correct with a threshold of 0.5.

Lastly, Figure 9 shows our qualitative results. We first show our pose predictions by drawing blue outlines, and predicted normal directions. To show that our algorithm obtained an accurate pose alignment, we also render different

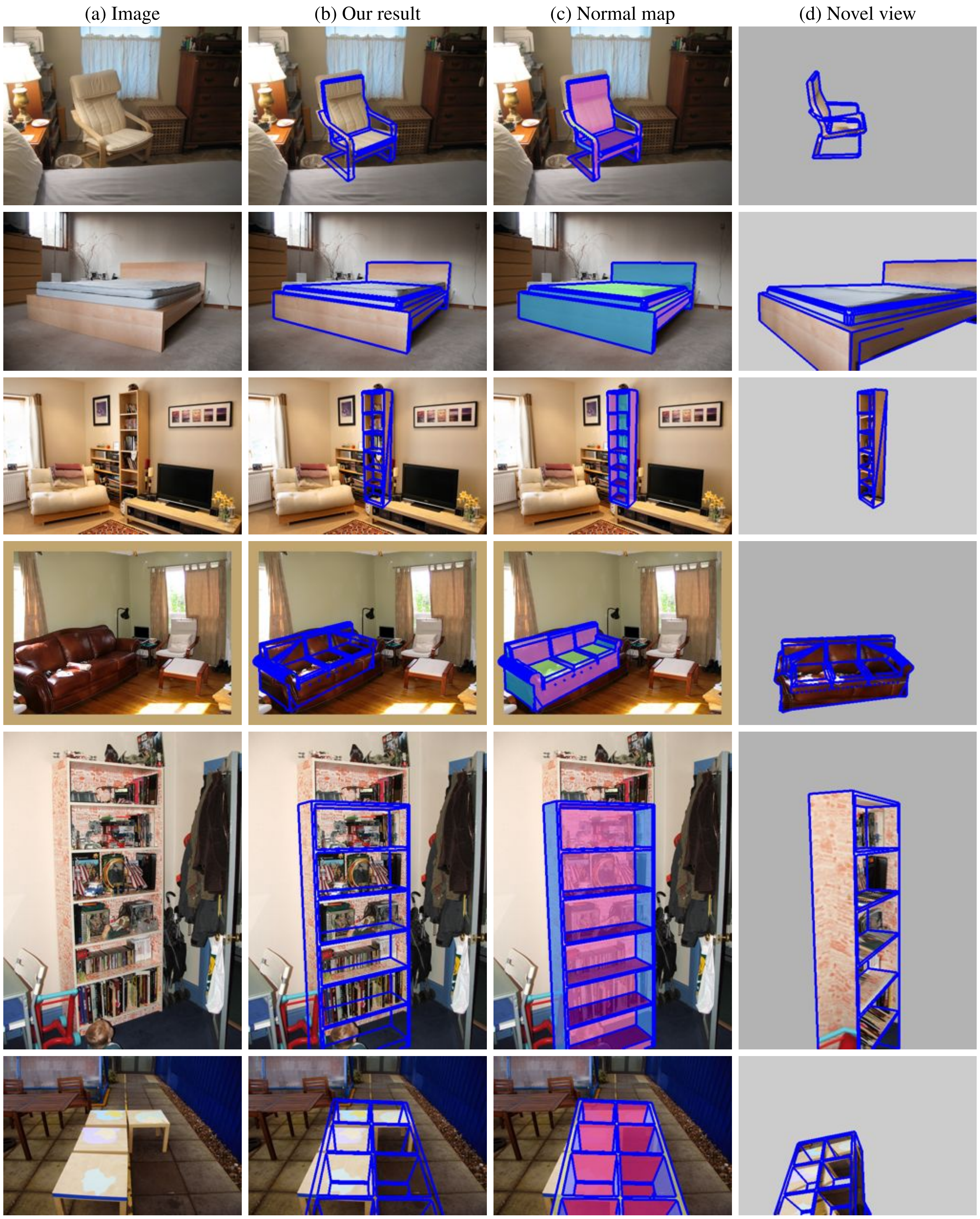


Figure 9. **Final results:** we show qualitative results of our method. (b) shows our estimated pose, (c) shows a normal direction, and (d) shows a synthesized novel view by rotating along y-axis. First 4 rows are correct estimations and last 2 rows are incorrect. Note that an error on the 5th row is misaligning a bookcase by 1 shelf. Near symmetry and repetition cause this type of top false positives.

	chair	bookcase	sofa	table	bed	bookcase	desk	bookcase	bed	bookcase	stool	
	poang	*billy1*	*ektorp*	*lack*	*malm1*	*billy2*	*expedit*	*billy3*	*malm2*	*billy4*	*poang*	mean
DPM	0.02	0.27	0.08	0.22	0.24	0.35	0.09	0.18	0.66	0.11	0.76	0.27
ELDA	0.29	0.24	0.35	0.14	0.06	0.77	0.03	0.20	0.60	0.41	0.13	0.29
D+L	0.045	0.014	0.071	0.011	0.007	0.069	0.008	0.587	0.038	0.264	0.003	0.10
D+L+HOG	4.48	2.91	0.17	5.56	0.64	9.70	0.12	5.05	**15.39**	7.72	0.79	4.78
D+L+H+Region	17.16	11.35	2.43	7.24	2.37	17.18	1.23	7.70	14.24	9.08	3.42	8.49
Full	**18.76**	**15.77**	**4.43**	**11.27**	**6.12**	**20.62**	**6.87**	**7.71**	14.56	**15.09**	**7.20**	**11.67**

Table 1. **AP Performances on Pose Estimation:** we evaluate our pose estimation performance at a fine scale in the IKEAobject database. As we introduce more features, the performance significantly improves. Note that DPM and ELDA are trained using rendered images.

	chair	bookcase	sofa	table	bed	bookcase	desk	bookcase	bed	bookcase	stool	
	poang	*billy1*	*ektorp*	*lack*	*malm1*	*billy2*	*expedit*	*billy3*	*malm2*	*billy4*	*poang*	mean
LDA @ 0.5	15.02	5.22	8.91	1.59	15.46	3.08	37.62	**34.52**	1.85	**46.92**	0.37	15.51
DPM @ 0.5	**27.46**	**24.28**	**12.14**	10.75	3.41	13.54	**46.13**	34.22	0.95	0.12	0.53	15.78
Ours @ 0.5	23.17	24.21	6.27	**13.93**	**27.12**	**26.33**	18.42	23.84	**22.34**	32.19	**8.16**	**20.54**
LDA @ 0.8	4.71	0.62	7.49	1.24	3.52	0.11	18.76	**21.73**	1.27	7.09	0.17	6.06
DPM @ 0.8	7.78	0.56	**10.13**	0.01	1.25	0.00	**35.03**	12.73	0.00	0.00	0.44	6.18
Ours @ 0.8	**21.74**	**18.55**	5.24	**11.93**	**11.42**	**25.87**	8.99	10.24	**18.14**	**17.92**	**7.38**	**14.31**

Table 2. **AP Performances on Detection:** we evaluate our method on detection against [4] and [8] at two different bounding box intersection over union thresholds (0.5 and 0.8) in the IKEAobject database. The gap between our method and [4] becomes significantly larger as we increase the threshold; which suggests that our method is better at fine detection.

novel views with a simple texture mapping.

5. Conclusion

We have introduced a novel problem and model of estimating fine-pose of objects in the image with exact 3D models, combining traditionally used and recently developed techniques. Moreover, we also provide a new dataset of images fine-aligned with exactly matched 3D models, as well as a set of 3D models for widely used objects. We believe our approach can extend further to more generic object classes, and enable the community to try more ambitious goals such as accurate 3D contextual modeling and full 3D room parsing.

Acknowledgements: This work is funded by ONR MURI N000141010933. We also thank Phillip Isola and Aditya Khosla for important suggestions and discussion.

References

[1] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *CVPR*, 2005. 2

[2] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes Challenge 2007 (VOC2007) Results. 6

[3] P. Felzenszwalb and D. Huttenlocher. Efficient graph-based image segmentation. *IJCV*, 59(2):167–181, 2004. 3

[4] P. F. Felzenszwalb, R. B. Girshick, and D. McAllester. Discriminatively trained deformable part models, release 4. 6, 8

[5] S. Fidler, S. J. Dickinson, and R. Urtasun. 3d object detection and viewpoint estimation with a deformable 3d cuboid model. In *NIPS*, 2012. 2

[6] M. Fisher and P. Hanrahan. Context-based search for 3d models. *ACM Trans. Graph.*, 29(6), Dec. 2010. 2

[7] S. Gupta, P. Arbelaez, and J. Malik. Perceptual organization and recognition of indoor scenes from RGB-D images. In *CVPR*, 2013. 2

[8] B. Hariharan, J. Malik, and D. Ramanan. Discriminative decorrelation for clustering and classification. In *ECCV*, 2012. 3, 6, 8

[9] M. Hejrati and D. Ramanan. Analyzing 3d objects in cluttered images. In *NIPS*, 2012. 2

[10] P. Henry, M. Krainin, E. Herbst, X. Ren, and D. Fox. Rgbd mapping: Using depth cameras for dense 3d modeling of indoor environments. In *RGB-D: Advanced Reasoning with Depth Cameras Workshop in conjunction with RSS*, 2010. 2

[11] J. J. Lim, C. L. Zitnick, and P. Dollar. Sketch tokens: A learned mid-level representation for contour and object detection. In *CVPR*, 2013. 4

[12] D. Lowe. Fitting parameterized three-dimensional models to images. *PAMI*, 1991. 1, 3

[13] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *IJCV*, 2004. 2

[14] J. L. Mundy. Object recognition in the geometric era: A retrospective. In *Toward CategoryLevel Object Recognition, volume 4170 of Lecture Notes in Computer Science*, pages 3–29. Springer, 2006. 1

[15] T. Ojala, M. Pietikinen, and T. Menp. Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. *PAMI*, 2002. 4

[16] B. Pepik, P. Gehler, M. Stark, and B. Schiele. 3d2pm - 3d deformable part models. In *ECCV*, 2012. 2

[17] F. Rothganger, S. Lazebnik, C. Schmid, and J. Ponce. 3d object modeling and recognition using local affine-invariant image descriptors and multi-view spatial constraints. *IJCV*, 66:2006, 2006. 1

[18] S. Satkin, J. Lin, and M. Hebert. Data-driven scene understanding from 3D models. In *BMVC*, 2012. 4

[19] Y. Xiang and S. Savarese. Estimating the aspect layout of object categories. In *CVPR*, 2012. 2

[20] J. Xiao, B. Russell, and A. Torralba. Localizing 3d cuboids in single-view images. In *NIPS*. 2012. 2, 4

[21] Y. Zhao and S.-C. Zhu. Image parsing via stochastic scene grammar. In *NIPS*, 2011. 2