View-Invariant Object Detection by Matching 3D Contours

Tianyang Ma, Meng Yi, and Longin Jan Latecki

Dept. of Computer and Information Sciences,Temple University, Philadelphia, USA
`ma.tianyang@gmail.com, {Mengyi,latecki}@temple.edu`

Abstract. We propose an approach for view-invariant object detection directly in 3D with following properties: (i) The detection is based on matching of 3D contours to 3D object models. (ii) The matching is constrained with qualitative spatial relations such as above/below, left/right, and front/back. (iii) In order to ensure that any matching solution satisfies these constraints, we formulate the matching problem as finding maximum weight subgraphs with hard constraints, and utilize a novel inference framework to solve this problem. Given a single view of an RGB-D camera, we obtain 3D contours by "back projecting" 2D contours extracted in the depth map. As our experimental results demonstrate, the proposed approach significantly outperforms the state-of-the-art 2D approaches, in particular, latent SVM object detector, as well as recently proposed approaches for object detection in RGB-D data.

1 Introduction

Since the beginning of computer vision, the researchers have realized that 3D information makes object detection and recognition simpler and more robust than using 2D image information only. In particular, contours of 3D objects have been utilized in object recognition many decades ago, e.g., [1,2], since they offer a view invariant representation of 3D objects. Moreover, in contrast to 3D surfaces, 3D contours offer a simpler 1D like representation of complex shapes in 3D like chairs or other man-made objects. However, extraction of 3D contours from single 2D images or stereo image pairs turned out to be a challenging problem. Only due to recent progress of RGB-D sensors, robust extraction of 3D contours became possible. However, we still face the problem of matching of 3D contours. The main challenges are intra class object variance, e.g., everyday objects like chairs come in different sizes and shapes, and occlusion.

Contour is an important cue for human to recognize objects, and has been widely used in 2D single-view object detection in [3,4,5]. While contour has certain advantages, such as its low computation cost and its invariance to color and texture changes, it varies significantly under different viewpoints. This challenges most of current state-of-the-art shape-based detection approaches on a multi-view object detection task. As early computer vision approaches, we address this challenge by directly working with contours of 3D objects instead of their 2D projections. In our approach, we still utilize the fact that contours of 3D objects project to 2D contours. It allows us for efficient recovery of 3D contours from 2D contours extracted from depth maps. This is possible thanks to Kinect, which is the most popular RGB-D camera. Since depth information can be obtained from a single view of a given scene, it is possible to recover 3D point cloud representing object surfaces. Depth map certainly provides more information that a single RGB image, and has proved to boost the performance of object recognition methods [6].

Object detection in 3D point clouds is an active research topic in the robotic community. There objects are recognized by directly matching 3D point clouds or by fitting surfaces to 3D point clouds. While surfaces are appropriate models for certain object classes, e.g., a ball, it is very hard if impossible to model object classes like chairs with surfaces alone. Contours appear to be a very suitable representation for RGB-D images. We observe that contours of 3D objects project to contours in 2D images. This in particular means that we can obtain 3D contours by lifting back contours from 2D images to 3D.

The processing flow of the proposed approach is illustrated in Fig. 1. After obtaining an RGB and depth images of a single view of a scene with Kinect, we first run Canny edge detector on the depth map. By linking the edge pixels, we obtain 2D edge fragments shown overlaid on the depth map in Fig. 1(b) with different colors. Since for each pixel in the depth map we can recover the 3D point that projects to it (with exception of out of range readings), we can "back project" each edge fragment to a set of 3D points, which we call 3D contour fragment. In Fig. 1(c) we see the 3D points recovered form the depth map in (b); for clarity of visualization the floor points are not shown. In Fig. 1(d) we show the 3D contour fragments in different colors. Each 3D contour fragment is represented with a set of 3D line segments fitted to the 3D points "back projected" from the corresponding 2D edge fragment. While one can recognize there the 3D contours of the two chairs and the stand, there are also many other contours present. They represent edges of walls and the background clutter.

After this preprocessing phase, we are ready for the proposed object detection. The 3D contours that belong to two detected chairs are shown in red and green in Fig. 1(e). All other 3D contours are shown in cyan. The detection is obtained by matching the model chair shown in Fig. 1(f) to all 3D contours in (e). In our system we used only one extremely simplistic model chair, as shown in (f), in order to demonstrate the power of matching 3D contours. The main challenges addressed by the proposed approach are intra class variability of 3D contours and occlusion. Occlusion and self-occlusion results in missing parts of 3D contours, which makes their matching challenging. To address these challenges we utilize the fact that geometric relations between 3D contours have more expressive power, and consequently, are less ambiguous compared to 2D.

We propose to solve the object detection by 3D matching problem by finding maximal weight subgraphs (MWSs) that satisfy mutex constraints. An example result is shown in Fig. 2. There for each of the three detected chairs, we mark with the same color their 3D segments and the corresponding model segments. We observe that the three chairs vary in shape and size, and all are substantially

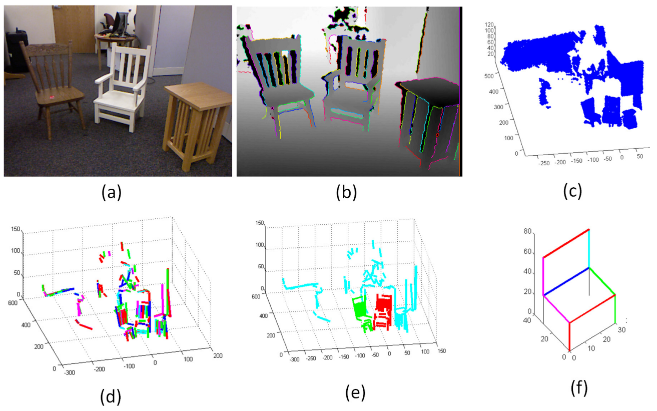

Fig. 1. An RGB image in (a) and the corresponding depth map in (b). The 3D points recovered from (a) are shown in (c). We recover 3D contour fragments, shown in different colors in (d) from edge fragments in (b). The line segments of two detected chairs in (d) are shown in green and red in (e). They are detected by matching segments of a single model shown in (f) to the segments in (d).

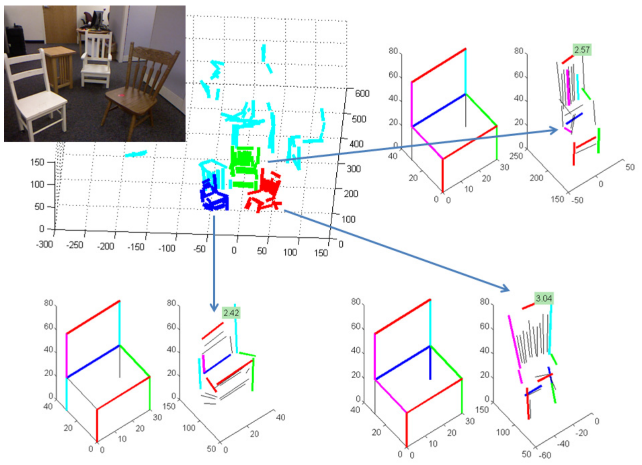

Fig. 2. A recovered 3D scene from a single RGB-D image. Contours of 3D objects are represented with 3D line segments. Object detection is performed by finding MWSs in the correspondence graph composed of pairs (model segment, 3D scene segment). We mark with the same colors the corresponding segments for three detected chairs shown in red, green, and blue in the 3D scene.

different form our single model chair. Moreover, due to self-occlusion, and since some edge fragments are not detected in the 2D depth images, all three chairs have some missing parts. The proposed matching approach is able to robustly deal with these challenges. This is possible due to our inference framework for finding MWSs that allows us to enforce hard, mutual exclusion (mutex) constrains. The mutex constraint, which express qualitative spatial relations such as above/below as well as prohibit grouping 3D contours that are too far from each other, eliminate the majority of impossible matching configurations. This allows us to obtain correct detections with weak shape similarity relations, which in turn allow us to tolerate a significant shape and size variance of 3D contours representing objects in the same shape class. In particular, we use only one chair exemplar in our experiments on chair detection.

We compute the MWSs on the correspondence graph composed of all pairs (model segment, 3D scene segment). As shown in Fig. 1(f), our exemplar chair is composed of 11 line segments. If we have 200 segments in a given 3D scene, for example, then the correspondence graph has 2200 nodes. In order to detect MWSs in this graph, we initialize with one correspondence, and compute a MWS that contains this correspondence, i.e., we have 2200 initializations. Then we sort the MWSs according to their weights. The three detected chairs in Fig. 2 represent MWSs with three highest weights. As can be seen the subgraphs have 8 to 10 nodes. Thus, our inference framework is capable of finding very small MWSs in graphs with a few thousand nodes.

In Sec. 2, we review related works. In Sec. 3, we introduce our shape representation and matching, also how to formulate the object localization problem as finding maximal weight subgraph with mutex constraints. In Sec. 4, a formal definition of maximal weight subgraph with mutex constraints will be given and an algorithm we used to solve it is described. Experiment results are shown in Sec. 5.

2 Related Work

There are some recent works utilizing 3D contour information to perform object detections in range images. Stiene et al. [7] proposed a detection method in range images based on silhouettes. Drost et al. [8] use a local hough-like voting scheme that uses pairs of points as features to detect rigid 3D objects in 3D point clouds. Hinterstoisser et al. [9] proposed a multimodal template matching approach based on RGB-D data that is able to detect objects in highly cluttered scenes.

In a very early work, Ponce et al. [10] established a 3D object recognition framework, where objects are collections of small (planar) patches, their invariants, and a description of their 3D spatial relationship. Ferrari et al. [11] proposed a method to compute feature tracks densely connecting multiple model views of a single object. In [12], Implicit Shape Model [13] and [11] are combined, and activation links for transferring votes across views are used to address the object detection from arbitrary viewpoints. Savarese and Fei-Fei [14] propose a compact model of an object by linking together diagnostic parts of the objects from different viewpoints. Instead of recovering a full 3D geometry, parts are mutually connected by homographic transformation in this approach. More recently, a probabilistic approach to learning affine constraints between object parts is introduced in [15]. In [16], discriminative part-based 2D detectors and generative 3D representation of the object class geometry which can be learned from a few synthetic 3D models are combined. Yan et al. [17] collect patches from viewpoint-annotated 2D training images and map them onto an existing 3D CAD model. In [18], a 3D implicit shape model is obtained via sparsely annotated 2D feature positions. Payet and Todorovic [19] proposed a shape-based 3D object recognition method, in which a few view-dependent shape templates are jointly used for detecting object occurrences and estimating their 3D poses.

A recent work by Janoch et al. [20] explores different options on how to utilize the depth information from RGB-D cameras to improve the detection accuracy of objects seen from different viewpoints. They call Deformable Part Model (DPM) [21] applied to depth images Depth HOG, and conclude that Depth HOG is never better than HOG on the original 2D image. The best performing system on their dataset is a linear combination of DPM running on the original image with the size distribution of a given object class, which is modeled with a single Gaussian. We call this system DPM-SIZE.

View-invariant object detection can also be addressed by directly using single 2D images, i.e., no 3D contour or surface reconstruction is attempted prior to the detection. Recent approaches of this type include [12,16,22]. While 2D single-view object detection methods can be used to addressed the task by combining the outputs of classifiers trained for different object views, such approaches are argued to be only effective when there are sufficient single-view detectors to cover all possible viewpoints [12]. However, this strategy requires a lot of training samples, and many independent detectors may lead to a substantial increase in the number of false-positives. In order to obtain a better multi-view object detector, many methods made an effort to learn a generative model by combining 2D appearance and geometric viewpoint information [15,23,16]. While promising results are obtained by such methods, they suffer from ambiguous 2D local features and lack of direct modeling of 3D viewpoint geometry.

In general graph matching frameworks [24], while local features' similarity (unary potential) and geometric relations between them (binary potential) are usually considered, very coarse qualitative geometric constraints such as above/below, or left/right do not draw much attention. We demonstrate in our work that using mutex constraints to enforce these qualitative geometric constraints makes our method more robust to the noise, and therefore, able to generate higher quality solutions.

3 Object Detection by Matching 3D Contours

In order to obtain contours of 3D objects form a given RGB-D image, we first find edge fragments in the depth map. They are obtained by linking edge pixels obtained by the Canny edge detector to 2D curves. Then we lift each 2D edge fragment back to a 3D curve. Let C be a single edge fragment. We first dilate it with a dilation radius of 2 pixels. Then we find the set of 3D points Z that project to pixels in dilated C. Finally we iteratively fit 3D line segments to points in Z. We run RANSAC to fit a line and identify the inlier points and outlier points. Then we repeat this process for the outlier points until the number of outlier points is lower than a threshold. Hence we represent each 3D curve Z as a set of 3D line segments, and consequently, we represent 3D contours obtained from a given RGB-D image as set of line segments in 3D. An example is shown in Fig. 1(d).

Object detection in the proposed approach is formulated as finding configurations of line segments recovered from a given RGB-D image that are similar to the line segment configuration of the exemplar modeling a given shape class. Thus, we need to identify a subset of 3D line segments that best matches the exemplar. This computation is formulated here as finding maximum weight subgraphs (MWS) in a weighted correspondence graph. We begin with definitions of pairwise similarities of line segments.

3.1 Similarity of 3D Vectors

We use a set of straight line segments $\mathcal{S} = \{B_1E_1, \cdots, B_nE_n\}$ to approximate object contours in 3D, where B_i is the beginning point and E_i is the endpoint of segment B_iE_i. An example is shown in Fig 1 (b). Since the line segments are oriented, they are vectors in 3D, and from now on we treat them as vectors. For the model contour each line segment is represented with just one vector. In contrast, each contour line segment in 3D image is represented by two vectors that differ by their orientation.

Although we know the exact size of objects in 3D, the size of objects in the shape shape class may still vary significantly. To obtain a size-invariant vector representation, we characterize each B_iE_i by its angle with a reference vector r defined as

$$\angle(B_iE_i, r) = \arccos(\frac{B_iE_i \cdot r}{||B_iE_i||\ ||r||}) \in [0, \pi] \tag{1}$$

We take vector $r = [0, 0, 1]$ representing the z-axis as the reference vector. Since 3D objects are supported by the floor, which is represented as xy-plane, the representation in (1) is invariant to the rotation around the z-axis. This means it is invariant to object location on the floor, under the assumption that the object is standing on the floor. To simplify the notation, we omit the direction r below when possible, and use $\angle B_iE_i$ to represent the angle of vector B_iE_i with z-axis.

Given the above angle-based segment representation, we treat two vectors as similar if they have similar angles with the z-axis. We compute this similarity value as

$$\psi(B_iE_i, B_jE_j) = \exp(-\frac{(\angle B_iE_i - \angle B_jE_j)^2}{\sigma^2}) \tag{2}$$

where σ represents the tolerance of angle differences (it is set to $\frac{\pi}{3}$ in all our experiments).

3.2 Similarity of Vector Configurations

Let $\mathcal{E} = \{B_1^eE_1^e, \cdots, B_m^eE_m^e\}$ be 3D vectors that represent an exemplar (model) of a given shape class, and let $\mathcal{S} = \{B_1^sE_1^s, \cdots, B_n^sE_n^s\}$ be 3D vectors representing the vectors of the recovered 3D scene.

We construct a weighted association graph $G = (V, A)$ with $V = \mathcal{E} \times \mathcal{S}$. Hence each node represents a correspondence $u = (B_i^e E_i^e, B_j^s E_j^s)$ between a model vector i and an image vector j. Consequently, there are $N = m \times n$ nodes in the graph.

We define now the entries of the adjacency matrix A. If $u = v = (B_i^e E_i^e, B_j^s E_j^s)$, then $A(u, u) = \psi(B_i^e E_i^e, B_j^s E_j^s)$, which simply the similarity of the angle with z-axis of both vectors. Given a pair of different correspondences $u \neq v$, where $u = (B_i^e E_i^e, B_j^s E_j^s)$ and $v = (B_k^e E_k^e, B_l^s E_l^s)$, the weight $A(u, v)$ between nodes u and v represents the consistency of the their assignments. We measure it by computing the similarity of the spatial configuration of exemplar vectors $B_i^e E_i^e, B_k^e E_k^e$ to the configuration of the 3D scene vectors $B_j^s E_j^s, B_l^s E_l^s$. For this we consider new vectors that join their start points. For example, in Fig. 3 vectors $B_i^e E_i^e, B_k^e E_k^e$ are the cyan lines in the model, and the new vector $B_i^e B_k^e$ is marked with the black dashed line while the new vector $E_i^e E_k^e$ is marked with the red dashed line. The same colors are used for the corresponding vectors in the 3D scene. The similarity of this configuration is determined by the similarity of the angles between the corresponding dashed vectors:

$$A(u, v) = \psi(B_i^e B_k^e, B_j^s B_l^s) \cdot \psi(E_i^e E_k^e, E_j^s E_l^s). \tag{3}$$

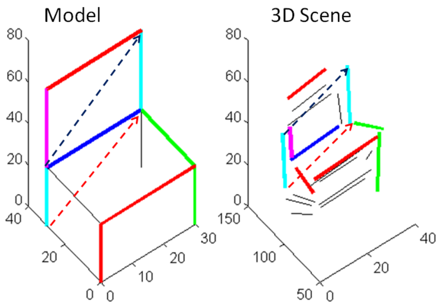

Fig. 3. Similarity of the two configurations of cyan lines is defined as similarity of the angles between two black dashed vectors and between two red dashed vectors

3.3 Mutex Constraints between Contour Vectors

Compared to other graph matching frameworks, the key and unique property of our formulation is usage of qualitative spatial constraints, such as above/below or left/right or front/back. For example, if for a given pair $u = (B_i^e E_i^e, B_j^s E_j^s)$ and $v = (B_k^e E_k^e, B_l^s E_l^s)$, the model vector $B_k^e E_k^e$ is above vector $B_i^e E_i^e$, then we require the same for the corresponding vectors in the 3D scene, i.e., $B_k^s E_k^s$ should be above $B_i^s E_i^s$. By enforcing the qualitative geometric relations in the correspondence computation, we can significantly improve the solution quality. In particular, the matching becomes robust to significant variance in shape and size of objects form a given class.

We define a symmetric mutex relation $M \subseteq V \times V$ between vertices of the graph defined in Section 3.2. It is represented with a binary matrix $M \in \{0,1\}^{N \times N}$. If $M(u,v) = 1$ then the two vertices u, v cannot belong to the same maximum clique. In other words, mutex represents incompatible vertices that cannot be selected together. Since a vertex cannot exclude itself, we set $M(u,u) = 0$ for all vertices $u \in V$.

Given a pair of two vertices representing the correspondences $u = (B_i^e E_i^e, B_j^s E_j^s)$ and $v = (B_k^e E_k^e, B_l^s E_l^s)$, where $u \neq v$, $M(u,v)$ represents the compatibility of the the spatial relations between vectors $B_i^e E_i^e$ and $B_k^e E_k^e$ in the model, and $B_j^s E_j^s$ and $B_l^s E_l^s$ in the 3D scene. For example, if $B_i^e E_i^e$ is above $B_k^e E_k^e$ in the model and $B_j^s E_j^s$ is below $B_l^s E_l^s$ in the scene, then $M(u,v) = 1$. One the other hand, if $B_j^s E_j^s$ is also above $B_l^s E_l^s$, then $M(u,v) = 0$. Similarly, $M(u,v) = 1$ if front/back or left/right spatial relations are violated.

In order to define M without checking different cases, we project the 4 points $B_i^e, E_i^e, B_k^e, E_k^e$ to vectors $B_i^e E_i^e$ and $B_k^e E_k^e$ in the model and the 4 points B_j^s, E_j^s, B_l^s, E_l^s to vectors $B_j^s E_j^s$ and $B_l^s E_l^s$ in the scene. Then we check whether the two 1D orders on the projection lines are compatible. If yes, we set $M(u,v) = 0$, and if not, we set $M(u,v) = 1$. We skip the technical details, since they only require elementary 3D geometry and the limited space.

4 Maximum Weight Subgraphs with Mutex Constraints

Given the weighted correspondence graph G, we formulate the problem of localizing objects in images as finding constrained maximum weight subgraphs. Each node in our graph is a matching between an image vector and a model vector. Therefore, a configuration of nodes, i.e., subgraph, corresponds to a set of selected scene vectors matched to the model shape. Hence each subgraph represents a configuration of 3D scene vectors and the corresponding configuration of model vectors. The unary and binary potentials in Section 3.2 are defined so that the more similar are both configurations the larger is the weight of their subgraph, which is just the sum of unary and binary potentials. Therefore, maximum weight subgraphs identify the instances of the model exemplar present in a given 3D scene recovered from a single RGB-D image.

Formally, the input is a weighted graph $G = (V, A)$, where $V = \{v_1, \ldots, v_N\}$ is the set of nodes representing the matches between model segments and image segments, N is the number of nodes, and A is a symmetric $N \times N$ affinity matrix, defined in Section 3.2, with all nonnegative entries, i.e., $A_{ij} \geq 0$ for all $i, j = 1, \ldots, N$. The selected matches are identified with and indicator vector $\mathbf{x} = (x_1, \ldots, x_N) \in \{0,1\}^N$, where a given match v_i is selected if and only if $x_i = 1$.

We are also given a symmetric mutex relation $M \subseteq V \times V$ between vertices of the graph defined in Section 3.3. The mutex relation M imposes constraints on the indicator vector $\mathbf{x} \in \{0,1\}^N$: if $M(i,j) = 1$, then $x_i + x_j \leq 1$. This formulation is equivalent to the requirement $\mathbf{x}^{\mathbf{T}} M \mathbf{x} = 0$.

We find the contours belong to the target object by solving the following maximization problem

$$\text{maximize} \;\; f(\mathbf{x}) = \mathbf{x}^T A\mathbf{x} \;\; \text{s.t.} \;\; \mathbf{x} \in \{0,1\}^n \;\; \text{and} \;\; \mathbf{x}^{\mathbf{T}} M\mathbf{x} = 0. \tag{4}$$

The goal of (4) is to select a subset of vertices of graph G such that f is maximized and the mutex constraints are satisfied. Since f is the sum of pairwise affinities of the elements of the selected subset, the larger is the subset, the larger is the value of f. However, the size of the subset is limited by mutex constraints. The problem (4) is a combinatorial optimization problem and is NP-hard [25].

By setting $W = A - \gamma M$, with a large positive γ we reformulate problem (4) into the following form:

$$\text{maximize} \;\; \mathbf{x}^{\mathbf{T}} W\mathbf{x} = \mathbf{x}^{\mathbf{T}} A\mathbf{x} - \gamma\mathbf{x}^{\mathbf{T}} M\mathbf{x} \;\; \text{s.t.} \;\; \mathbf{x} \in \{0,1\}^n. \tag{5}$$

Finally, we relax (5) to

$$\text{maximize} \;\; \mathbf{x}^{\mathbf{T}} W\mathbf{x} = \mathbf{x}^{\mathbf{T}} A\mathbf{x} - \gamma\mathbf{x}^{\mathbf{T}} M\mathbf{x} \;\; \text{s.t.} \;\; \mathbf{x} \in [0,1]^n. \tag{6}$$

We utilize the algorithm described in [26] to solve problem (6). Its key property is that if $\gamma > \max_i \sum_j A_{ij}$ and if the solution $\mathbf{x}^*$ is discrete, then $\mathbf{x}^*$ is guaranteed to satisfy all mutex constraints, i.e., $(\mathbf{x}^*)^{\mathbf{T}} M\mathbf{x}^* = 0$.

Although this algorithm solves the relaxed problem (6) the obtained solutions were discrete in all of our experiments. Hence the solutions satisfy all mutex constraints.

Since the algorithm in [26] converges to a local optimum, multiple initializations are required to increase the change of getting a globally optimal solution. In our implementation, we initialize from every node in the graph. More precisely, for every $u \in V$ we set $(\mathbf{x}_{(0)})_u = 1$ and $(\mathbf{x}_{(0)})_i = 0$ for all $i \neq u$, where $\mathbf{x}_{(0)}$ denotes the initial vector $\mathbf{x}$. Starting from the $\mathbf{x}_{(0)}$, we obtain a maximal subgraph indicated by a binary vector $\mathbf{x}^*$. $\mathbf{x}^*$ is a local maximizer of $\mathbf{x}^T A\mathbf{x}$ while satisfying $\mathbf{x}^{*T} M\mathbf{x}^* = 0$.

Therefore, we obtain N maximal subgraphs in total. Since there may be duplicated subgraphs among these N maximal subgraphs, we perform a non-maximum suppression over these subgraphs according to their $\mathbf{x}^T A\mathbf{x}$ values. Finally, we take the remaining subgraphs as object detections.

We are not only able to find out which contours in image belong to a detected object, but more importantly, we are also able to establish a correspondence of these contours. This is very important in some applications, such as robot manipulation.

To obtain the location of the object, i.e., its bounding box, in RGB image. We project the 3D segments back to RGB images, and compute the bounding box.

5 Experiments

Chair is an icon object class that has gained much attention form the beginning of AI. Although humans have no problem in identifying chairs, until today no artificial system is able to cope with chair detection. Chair detection is a challenging problem for most computer vision, detection algorithms [27], considering that the chair shape in 2D images varies significantly due to different viewpoints and due to resulting perspective distortion. Moreover, chairs come in different shapes and sizes. Therefore, we focus our performance evaluation on chair detection. We selected a stand as the second object class, since it is visually very similar to the chair in that it usually has 4 legs supporting a flat rectangular surface on top. The main difference is that the stand does not have any back support and its legs are longer, e.g., see the left image in Fig. 4.

We collected a dataset containing 109 RGB-D images captured with the Kinect sensor. It contains a total of 213 chairs shown from many different view points and 40 stands. Our dataset also contains other objects that may be confused with chairs and stands like tables and trash cans as can be seen in Fig. 4. Moreover, may objects are occluded and are shown in many different views.

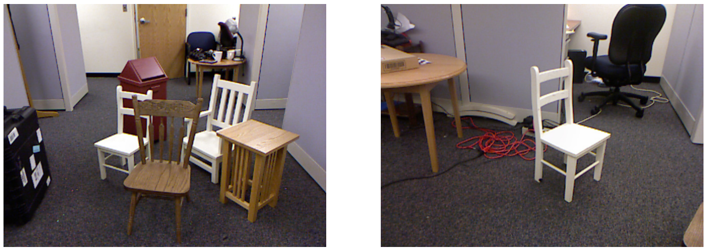
Fig. 4. Example images in our chair-stand dataset

In order to demonstrate that our dataset is very challenging and in order to compare to state-of-the-art object detectors, we compare the performance of our approach to DPM by Felzenszwalb et al. [21] and to DPM-SIZE recently proposed in Janoch et al. in [20]. DPM-SIZE augments DPM with depth information. It utilizes the expected object sizes in 3D scenes to boost DPM performance. We also compare to the popular contour based detection method PAS by Ferrari et al. [3]. For a quantitative evaluation, we use recall-precision curves and average precision (AP) computed as described in [28].

The detection results of chairs are summarized in Fig. 5. The proposed approach achieves a significantly better AP value compared to DPM and to DPM-SIZE. Our AP is nearly 30% higher than the second best performing method DPM-SIZE [20]. Moreover, the fact that DPM-SIZE, DPM, and PAS have all very low recall clearly demonstrates that these methods cannot cope with significant view changes and perspective distortions. This comes at no surprise for DPM and PAS, since both methods are based on 2D image analysis. In contrast, the direct matching of 3D contours in 3D allows us to overcome the challenges of view changes and of perspective distortion. We stress that our approach does not require any training, as opposed to the other three approaches, and we only have one extremely simplistic chair model. Moreover, our chair model is not extracted from the test dataset.

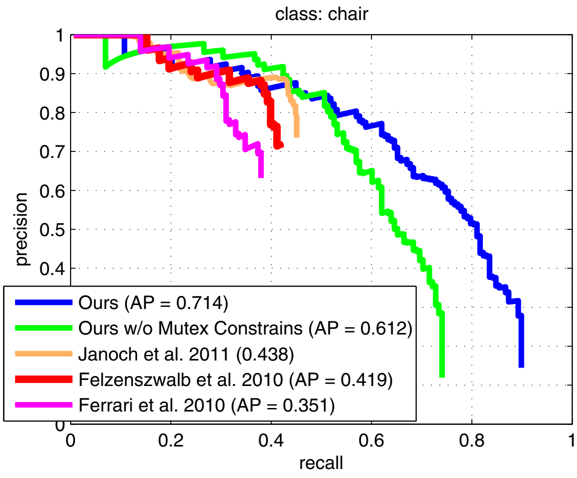

Fig. 5. Recall-Precision and AP comparison for the class chair

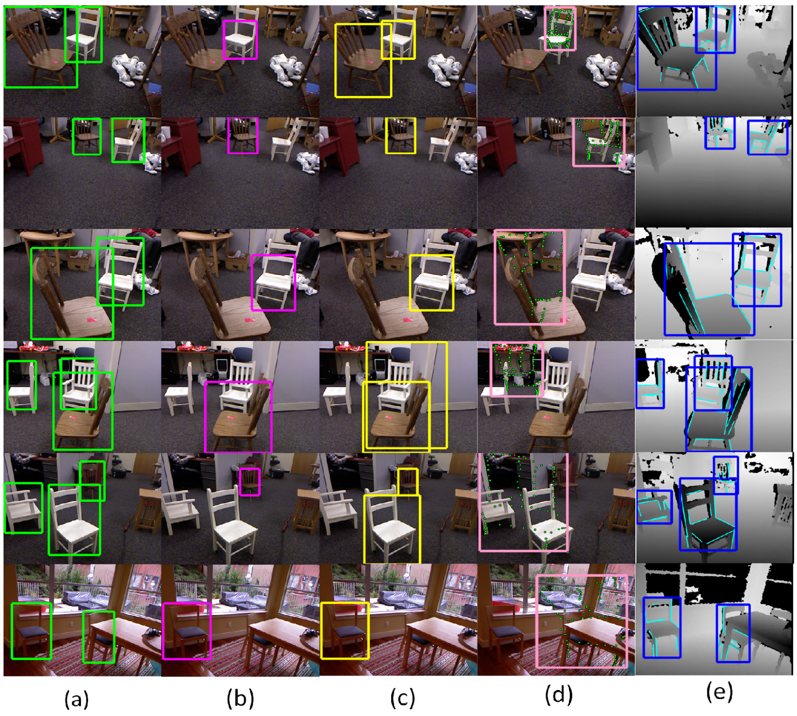

Fig. 6. Some chair detection results. (a) ground truth, (b) DPM [21], (c) DPM-SIZE [20]. (d) PAS [3] with transformed model shown with dots, and (e) The proposed method with results shown on depth map to stress that they are obtained in 3D.

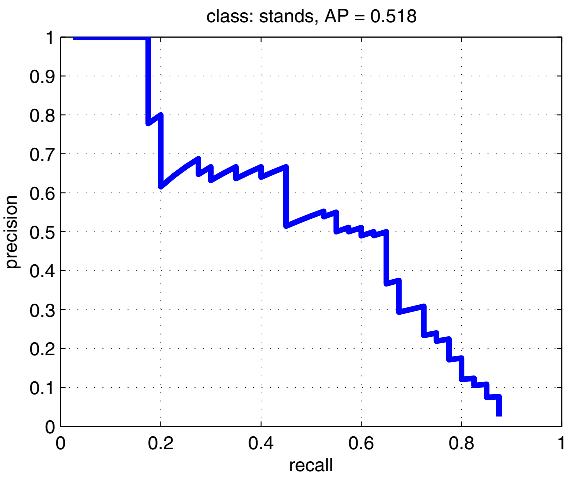

Fig. 7. Recall-Precision and AP of our detector with mutex constraints on class stand

The significance of the qualitative mutex constraints is demonstrated by the fact that the performance of our method drops by 10% when these constraints are not used. This in turn illustrates the importance of the utilized inference framework.

In Fig. 6, we show some detection results. As seen in Fig. 6(b), DPM [21], DPM-SIZE [20], PAS [3] missed many chairs. Adding 3D information about expected object sizes in the 3D scenes (DPM-SIZE [20]) is able to improve the performance of DPM, but still some chairs are missed. The main reason is that the initial detection is still performed in the 2D images (using sliding window processing of DPM).

We use the already trained version of DPM, which is publicly available on the authors' webpage. DPM [21] attempts to solve the object detection problem by using a multiple components object model, and each component is aimed to capture the object appearance under certain view-point. The 2D chair appearance model of DPM is trained using images from [28] with thousands of chairs. We also tried to train DPM detector on half of our dataset and test on the other half as opposed to using the trained detector from images in [28]. This process yields a much worse AP of 0.01. However, the DPM detector is able to get 0.96 AP on training images. This again demonstrates how challenging is significant view point variance, and perspective distortion to state-of-the-art 2D object detectors. The expected size of the chair for DPM-SIZE was learned as described in [20]. We trained it on a random half of our dataset and test on the other half. This process was repeated 10 times. We also used the software of the authors of PAS [3] to perform experiments on our chair dataset. A shape is learned automatically using this software, following the same procedure as for size training of DPM-SIZE.

Since there does not exist any trained version of DPM for the class stand and our dataset exhibits too large view variance for training DPM, we only report the result of our detector with mutex constraints on the class stand in Fig. 7.

6 Discussion and Future Work

We only used one simplistic chair model, which differs in both size and shape from the various chairs captured in our dataset. This allows us to demonstrate the robustness of the proposed 3D matching framework. Our matching framework is also robust to occlusion, and of course, it is not influenced by view point changes. Similarly we only used one simplistic stand model.

However, more 3D contour models are needed to capture the intra class variability. In particular, some chairs may only have one leg like the office chair shown in the right image in Fig. 4. Such models can be easily learned by clustering training objects using the proposed similarity measure.

One of the biggest challenges of our 3D contour-based object detection are objects without clear 3D contours like humans or sofas. For such objects it is still possible to extract occluding contours from the RGB-D data, and those contours exhibit significantly lower variation than contours extracted form 2D RGB images. Also the contour detection problem in RGB-D images is significantly simpler. However, the 3D occluding contours exhibit larger variation than intrinsic 3D contours of objects like chair or stand. Our future work will focus on matching the occluding 3D contours.

Acknowledgement. This work has been supported by AFOSR FA9550-09-1-0207 and by NSF grants BCS-0924164, IIS-0812118, and OIA-1027897.

References

1. Barrow, H., Tenenbaum, J.: Interpreting line drawings as three-dimensional surfaces. Artificial Intelligence 17, 75–116 (1981)
2. Lowe, D.G.: Three-dimensional object recognition from single two-dimensional images. Artificial Intelligence 31(3), 355–395 (1987)
3. Ferrari, V., Jurie, F., Schmid, C.: From images to shape models for object detection. International Journal of Computer Vision 87, 284–303 (2010)
4. Shotton, J., Blake, A., Cipolla, R.: Multiscale categorical object recognition using contour fragments. IEEE Trans. Pattern Anal. Mach. Intell. 30, 1270–1281 (2008)
5. Opelt, A., Pinz, A., Zisserman, A.: Learning an alphabet of shape and appearance for multi-class object detection. International Journal of Computer Vision 80, 16–44 (2008)
6. Bo, L., Lai, K., Ren, X., Fox, D.: Object recognition with hierarchical kernel descriptors. In: CVPR, pp. 1729–1736 (2011)
7. Stiene, S., Lingemann, K., Nuchter, A., Hertzberg, J.: Contour-based object detection in range image. In: Third International Symposium on 3D Data Processing, Visualization and Transmission (2006)
8. Drost, B., Ulrich, M., Navab, N., Ilic, S.: Model globally, match locally: Efficient and robust 3d object recognition. In: IEEE Computer Society Conference on Computer Vision and Pattern Recognition, pp. 998–1005 (2010)
9. Hinterstoisser, S., Holzer, S., Cagniart, C., Ilic, S., Konolige, K., Navab, N., Lepetit, V.: Multimodal templates for real-time detection of texture-less objects in heavily cluttered scenes. In: IEEE International Conference on Computer Vision, pp. 858–865 (2011)

10. Ponce, J., Lazebnik, S., Rothganger, F., Schmid, C.: Toward true 3d object recognition. In: Congres de Reconnaissance des Formes et Intelligence Artificielle (2004)
11. Ferrari, V., Tuytelaars, T., Van Gool, L.J.: Integrating multiple model views for object recognition. In: CVPR, pp. 105–112 (2004)
12. Thomas, A., Ferrari, V., Leibe, B., Tuytelaars, T., Schiele, B., Van Gool, L.J.: Towards multi-view object class detection. In: CVPR, pp. 1589–1596 (2006)
13. Leibe, B., Leonardis, A., Schiele, B.: An Implicit Shape Model for Combined Object Categorization and Segmentation. In: Ponce, J., Hebert, M., Schmid, C., Zisserman, A. (eds.) Toward Category-Level Object Recognition. LNCS, vol. 4170, pp. 508–524. Springer, Heidelberg (2006)
14. Savarese, S., Li, F.F.: 3d generic object categorization, localization and pose estimation. In: ICCV, pp. 1–8 (2007)
15. Sun, M., Su, H., Savarese, S., Li, F.F.: A multi-view probabilistic model for 3d object classes. In: CVPR, pp. 1247–1254 (2009)
16. Liebelt, J., Schmid, C.: Multi-view object class detection with a 3d geometric model. In: CVPR, pp. 1688–1695 (2010)
17. Yan, P., Khan, S.M., Shah, M.: 3d model based object class detection in an arbitrary view. In: ICCV, pp. 1–6 (2007)
18. Arie-Nachimson, M., Basri, R.: Constructing implicit 3d shape models for pose estimation. In: ICCV, pp. 1341–1348 (2009)
19. Payet, N., Todorovic, S.: From contours to 3d object detection and pose estimation. In: ICCV, pp. 983–990 (2011)
20. Janoch, A., Karayev, S., Jia, Y., Barron, J.T., Fritz, M., Saenko, K., Darrell, T.: A category-level 3-d object dataset: Putting the kinect to work. In: ICCV Workshops, pp. 1168–1174 (2011)
21. Felzenszwalb, P.F., Girshick, R.B., McAllester, D., Ramanan, D.: Object detection with discriminatively trained part-based models. IEEE Transactions on Pattern Analysis and Machine Intelligence 32, 1627–1645 (2010)
22. Savarese, S., Tuytelaars, T., Van Gool, L.J.: Special issue on 3d representation for object and scene recognition. Computer Vision and Image Understanding 113, 1181–1182 (2009)
23. Liebelt, J., Schmid, C., Schertler, K.: Viewpoint-independent object class detection using 3d feature maps. In: CVPR (2008)
24. Berg, A.C., Berg, T.L., Malik, J.: Shape matching and object recognition using low distortion correspondences. In: CVPR, pp. 26–33 (2005)
25. Asahiro, Y., Hassin, R., Iwama, K.: Complexity of finding dense subgraphs. Discrete Applied Mathematics (2002)
26. Ma, T., Latecki, L.J.: Maximum weight cliques with mutex constraints for video object segmentation. In: CVPR (2012)
27. Grabner, H., Gall, J., Van Gool, L.J.: What makes a chair a chair? In: CVPR, pp. 1529–1536 (2011)
28. Everingham, M., Van Gool, L.J., Williams, C.K.I., Winn, J., Zisserman, A.: The pascal visual object classes (voc) challenge. International Journal of Computer Vision 88, 303–338 (2010)