From RGB-D Images to RGB Images: Single Labeling for Mining Visual Models

QUANSHI ZHANG, University of Tokyo; University of California, Los Angeles
XUAN SONG and XIAOWEI SHAO, University of Tokyo
HUIJING ZHAO, Peking University
RYOSUKE SHIBASAKI, University of Tokyo

Mining object-level knowledge, that is, building a comprehensive category model base, from a large set of cluttered scenes presents a considerable challenge to the field of artificial intelligence. How to initiate model learning with the least human supervision (i.e., manual labeling) and how to encode the structural knowledge are two elements of this challenge, as they largely determine the scalability and applicability of any solution. In this article, we propose a model-learning method that starts from a single-labeled object for each category, and mines further model knowledge from a number of informally captured, cluttered scenes. However, in these scenes, target objects are relatively small and have large variations in texture, scale, and rotation. Thus, to reduce the model bias normally associated with less supervised learning methods, we use the robust 3D shape in RGB-D images to guide our model learning, then apply the properly trained category models to both object detection and recognition in more conventional RGB images. In addition to model training for their own categories, the knowledge extracted from the RGB-D images can also be transferred to guide model learning for a new category, in which only RGB images without depth information in the new category are provided for training. Preliminary testing shows that the proposed method performs as well as fully supervised learning methods.

Categories and Subject Descriptors: I.4.8 [**Image Processing and Computer Vision**]: Scene Analysis; H.2.8 [**Database Applications**]: Data Mining; I.2.10 [**Artificial Intelligence**]: Vision and Scene Understanding

General Terms: Design, Algorithms, Performance, Theory

Additional Key Words and Phrases: Data mining, computer vision, big visual data, visual mining, transfer learning, visual knowledge base, RGB-D sensor

ACM Reference Format:
Quanshi Zhang, Xuan Song, Xiaowei Shao, Huijing Zhao, and Ryosuke Shibasaki. 2015. From RGB-D images to RGB images: Single labeling for mining visual models. ACM Trans. Intell. Syst. Technol. 6, 2, Article 16 (March 2015), 29 pages.
DOI: http://dx.doi.org/10.1145/2629701

16

This work was supported by Microsoft Research. It was also supported by a Grant-in-Aid for Young Scientists (23700192) of Japan's Ministry of Education, Culture, Sports, Science, and Technology (MEXT); and a grant from Japan's Ministry of Land, Infrastructure, Transport and Tourism (MLIT).
Authors' addresses: Q. Zhang is with the University of California, Los Angeles, USA; emails: {zhangqs@g.ucla.edu, zqs1022@csis.u-tokyo.ac.jp}; X. Song, X. Shao, and R. Shibasaki are with the Center for Spatial Information Science (CSIS), the University of Tokyo, Tokyo, Japan; emails: {songxuan, shaoxw, shiba}@csis.u-tokyo.ac.jp; H. Zhao is with Key Laboratory of Machine Perception (MoE), Peking University, Beijing, China; email: zhaohj@cis.pku.edu.cn.
This work was mainly done when Quanshi Zhang was a PhD student at the Center for Spatial Information Science (CSIS), University of Tokyo.
Permission to make digital or hard copies of part or all of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies show this notice on the first page or initial screen of a display along with the full citation. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, to republish, to post on servers, to redistribute to lists, or to use any component of this work in other works requires prior specific permission and/or a fee. Permissions may be requested from Publications Dept., ACM, Inc., 2 Penn Plaza, Suite 701, New York, NY 10121-0701 USA, fax +1 (212) 869-0481, or permissions@acm.org.
© 2015 ACM 2157-6904/2015/03-ART16 $15.00
DOI: http://dx.doi.org/10.1145/2629701

1. INTRODUCTION

Knowledge mining of big visual data presents a significant challenge to the field of artificial intelligence. Cognition-level symbolization of visual data is considerably more difficult than that of text data. A number of pioneering studies have addressed this challenge in recent years. Deep-learning methods [Le et al. 2012] have been widely used. Then, Zhang et al. [2014a] proposed a general platform for object-level visual mining, which is based on attributed graph mining. This platform has a wide range of extended applications, such as mining category-specific knowledge from ubiquitous images for single-view 3D reconstruction [Zhang et al. 2014b]. However, without sufficient manual labeling and training, the gap between feature-level image processing and object-level visual knowledge remains large. To bridge this gap, we propose use of RGB-D image data, instead of conventional RGB data, as a more productive basis for training.

We focus on the problem of constructing a category model base that can provide high-level guidance for many visual tasks, such as image understanding and object recognition. The construction of such a category base poses three main challenges, illustrated in Figure 1.

- *Single labeling:* Model base construction requires a minimum amount of manual labeling, but given a more idealized implementation of semi-supervised learning, can we learn a category model from just one labeled object and a large set of cluttered images? Such images are typical of what can be retrieved using most search engines. The target objects they contain are usually small, requiring significant hand-cropping and manual alignment for detailed learning, since automatic image segmentation usually cannot ensure object-level results.
- *Structural knowledge:* In many cases, it is structures, rather than textures, that determine functions and categories, especially for many daily-use commercial objects that have regular shapes but a variety of textures. For this reason, we hope to model structural knowledge of different object categories, unlike the conventional mining of "bag-of-words" knowledge.
- *Bias problem:* With the first two challenges, model learning in RGB images is caught in a dilemma, which makes the bias problem extremely serious. On one hand, training the structure-based model requires a large collection of small target objects from the image pool, as well as the extraction of part correspondences between these objects, to overcome intracategory variations. On the other hand, without sufficient labeling and training, object detection and matching based on a single labeling is hampered by large variations in texture and rotation, both great challenges even for state-of-the-art algorithms. Thus, in this case, bias in object collection in early learning steps will affect subsequent steps, and accumulate into significant model bias.

Thus, the goal of this article is to present an approach to learning a structure-based category model from a single-labeled object and a number of large, cluttered images with significant variations in texture and rotation.

To deal with these variations, we make use of the Kinect [Microsoft 2011] device to detect the 3D shape of the object. Kinect RGB-D images provide explicit spatial structures that are robust across variations in texture, 2D scale, and viewpoint. In many cases, the use of 3D structure can greatly improve the reliability of category detection. Meanwhile, ordinary RGB images are more widely used than RGB-D images.

Therefore, we attempt to build a bridge between the two formats, and employ a model learning strategy in which the model is trained with RGB-D images and is then applied to ordinary RGB images (Figure 1). We explore this problem at the following three levels.

First, the general idea of this strategy is simple. We use the more reliable 3D matching results to guide the learning of the less discriminative image-based models, as

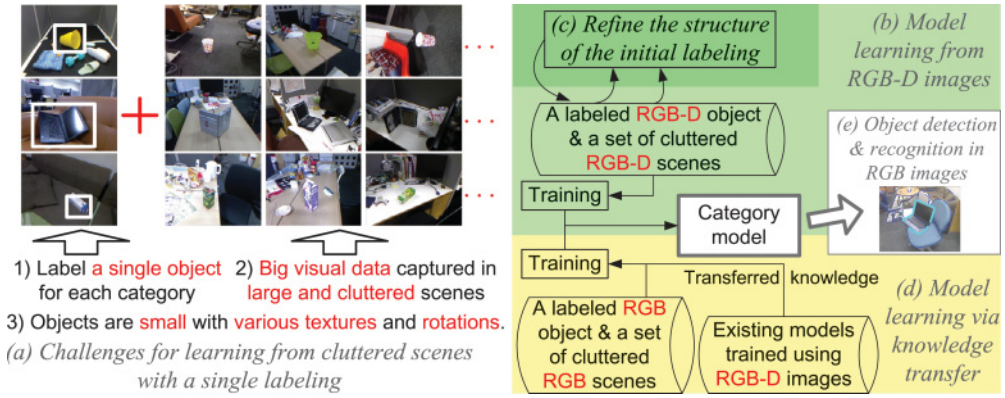

Fig. 1. How can a system learn a structure-based category model from a single-labeled object and a number of cluttered scenes? (a) We need to correctly detect and incrementally collect more object samples for training, but since our target objects in large scenes are usually captured informally, they will show large variations in texture and rotation. (b, *green part*) Thus, we can use 3D shapes in RGB-D images to guide model learning. (c) We propose a method to refine the initially labeled object using the cluttered RGB-D images in order to remove inaccuracy or subjective bias in manual labeling. (d, *yellow part*) When a large number of RGB-D images cannot be collected for a category, we can use knowledge transfer to learn the category model directly from RGB images. The knowledge is transferred from the models that are pretrained from RGB-D images to guide the training sample collection for a new category. (e, *white part*) The trained category model can be applied to ordinary RGB images.

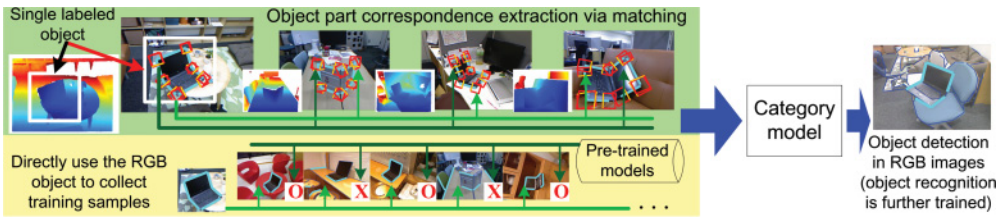

Fig. 2. Flowchart of the proposed method. First, for model learning from RGB-D images (*green part*), we use the 3D structure of the labeled object to match other objects in the RGB-D images. We then use the part correspondences to train a category model. Second, for model learning via knowledge transfer (*yellow part*), we label the RGB object to collect training objects directly from RGB images. The models that are trained for other categories provide knowledge to identify the correct and incorrect object matches during sample collection.

illustrated in Figure 2. We begin by using structure-based 3D matching to collect objects from RGB-D images, simultaneously obtaining part correspondences, even in the presence of significant texture variations. This yields a local codebook of visual words learned for each part of the object. The part correspondences in 3D space are then used to train the 2D structural knowledge in the category model. The category model can be applied to object matching in RGB images; we can also encode the knowledge for object recognition in the model by combining a set of negative[1] RGB images for training.

The second important issue is transfer learning. RGB-D images are used less widely than RGB images, and sometimes we can only collect RGB images for model training. In this case, we need to design a method for transferring the knowledge extracted from the existing category models to guide the training of new category models.

Finally, we focus on the problem of the inaccuracy of initial labeling. The training object collection is based on and sensitive to the labeling of the single initial object.

[1]In this case, negative RGB images are RGB images that do not contain objects in the target category, which can be collected directly by search engines.

Thus, inaccurate initial labeling may lead to a major bias in model learning. Therefore, we develop a method that uses the cluttered RGB-D images to refine the labeled object into a good category detector for further object collection. This method deletes the redundant parts of the object, while simultaneously modifying the local textures and structures of the other parts.

To implement these learning strategies, we design a novel graphical model that uses object edges as its basic and concise structural elements. Compared to texture features, object edges have a closer relationship to the overall object structure, especially where large texture variations exist. We also develop different sets of attributes to guide the 3D object collection from RGB-D images and the training of 2D category models.

In this study, we use graph-matching techniques to achieve both 3D object collection and model-based object matching. Given a template graph (the category model) and multiple target graphs, conventional methods for learning graph matching [Leordeanu and Hebert 2012; Cho et al. 2013; Caetano et al. 2009; Torresani et al. 2008] primarily train matching parameters (e.g., the weights of different graphical attributes). In contrast, our method estimates a prototype of the category model and eliminates the specificity of the labeled object in an unsupervised manner. Therefore, we extend the method proposed by Leordeanu and Hebert [2012] to learn model attributes. We have also combined our previous study [Zhang et al. 2013b] with this research to refine the structure of the initially labeled object.

The contributions of this article can be summarized as follows. First, we present a method that allows single labeling to start the learning of structural knowledge from informally captured scenes with significant clutter. Second, three model-learning strategies are proposed to avoid the bias problem caused by texture variations and various rigid transformations. If RGB-D images in the target category can be collected, we use depth information in the RGB-D images to assist the training of the category model for RGB images. Otherwise, we transfer the knowledge extracted from models of other categories to guide model learning. In particular, when the initial object labeling is inaccurate, we can refine the structure of the labeled object using the cluttered RGB-D images. Third, we design a new type of graphical model based on object edges as a concise representation of object structures in RGB and RGB-D images. Finally, we build and publish a large dataset of RGB-D images, which contain several challenging cases for graph matching.

Related work is discussed in the Section 2. Section 3 introduces the graphical models for RGB-D and RGB images. The basic algorithm for model learning and its technical extensions, as well as model-based recognition, are presented in Section 4. Section 5 presents the experiments, and the conclusions are summarized in Section 6.

A preliminary version of this paper appeared in Zhang et al. [2013a].

2. RELATED WORK

2.1. Visual Mining

Learning category models is a familiar problem in the field of computer vision, and many approaches have been proposed over the last few decades. In this section, we limit our discussion to those techniques related to the concept of data mining of big visual data. Generally speaking, these techniques should (1) have loose requirements for training data, and (2) limit the amount of human labeling, perhaps even to the point at which unsupervised learning is possible.

The effort to minimize manual labeling makes visual mining related to one-shot learning [Li et al. 2006]. For totally unsupervised approaches, object discovery (reviewed by Tuytelaars et al. [2010]) is a familiar goal in object-level knowledge mining. Most methods that have achieved this goal use bag-of-words models [Li et al. 2010]

for category representation. Others [Kang et al. 2011; Li et al. 2012; Faktor and Irani 2012; Zhu et al. 2012] have managed to detect repetitive objects based on similarities in appearance and visual context. Lee and Grauman [2011], Kang et al. [2011], and Liao et al. [2012] used unsupervised segmentation to generate object candidates, relying on foreground–background discrimination. Zhang et al. [2013c, 2014c] mined categories models from unlabeled 3D point clouds of large urban environment.

In lieu of conventional learning from a large sample pool, Li et al. [2010] and Vijayanarasimhan and Grauman [2011] proposed the collection of training images using image search engines via semi-supervised or active learning. These were more efficient ways of constructing a category model base. Approaches to co-segmentation [Kim et al. 2011; Chiu and Fritz 2013; Joulin et al. 2012; Kim and Xing 2012; Mukherjee et al. 2012] have provided a plausible way to detect and segment common objects from a big image set collected using a search engine.

Most methods for object discovery, one-shot learning, and co-segmentation have proven to be relatively poor for learning structural knowledge of objects. It is much easier to encode structural knowledge using link analysis (or graph mining) techniques in visual mining. When images are modeled as graphs, it is possible to extract the frequent subgraphs within these graphs as common objects [Jiang and Ngo 2003; Hong and Huang 2004; Kim et al. 2008; Tan and Ngo 2009; Yuan et al. 2012; Zhao and Yuan 2010; Xie et al. 2012; Liu and Yan 2010; Cho et al. 2010; Leordeanu et al. 2007].

However, most of these methods rely heavily on the similarity of object textures. Many object discovery and one-shot learning methods have directly used bag-of-words models, and co-segmentation approaches have leveraged texture distribution as the primary feature. Even most approaches based on graph link analysis have had to use the interimage consistency of local patches to generate potential patch correspondences between images, in order to prune the search space of frequent subgraphs. Thus, these approaches are not suitable for mining daily-use objects with large texture variations.

We also focus on the extraction of precise structural models from large cluttered scenes, rather than observing the probability of patch textures. We use the depth information in RGB-D images to guide the learning process, thus avoiding errors caused by variations in texture, scale, and rotation.

Actually, the RGB-D images have been widely used for object detection and classification. However, conventional methods [Lai et al. 2011a; Ren et al. 2012] mainly learned category models from well-labeled training samples. They required a great amount of human labeling to prepare training samples for each category, whereas, our method is close to the spirit of "model learning," which aims to directly mine the category models from cluttered scenes (a kind of big visual data) without much labeling of "what is where." Model mining involves almost all of the typical issues in computer vision, such as lack of manual alignments and intracategory variations in texture, rotation, scale, and illumination. Thus, it proposes continuous challenges for state-of-the-art algorithms. In recent years, people began to apply RGB-D images to some "mining" tasks. For example, Fouhey et al. [2013] and Zhang et al. [2014b] proposed learning reconstruction knowledge from RGB-D images for single-view 3D reconstruction on RGB images.

2.2. Category Modeling

In this section, we compare different types of category models, and analyze their applicability to informally captured scenes.

Bag-of-words models [Li et al. 2010] have been widely used to model categories without encoding structural information. The global structures of objects can be simply represented by the histograms of oriented gradients (HoG) templates [Dalal and Triggs 2005] or silhouette templates [Leordeanu et al. 2007; Wang et al. 2012; Xu et al. 2012]. Hough-style methods [Maji and Malik 2009; Razavi et al. 2012; Liu et al. 2012; Ferrari

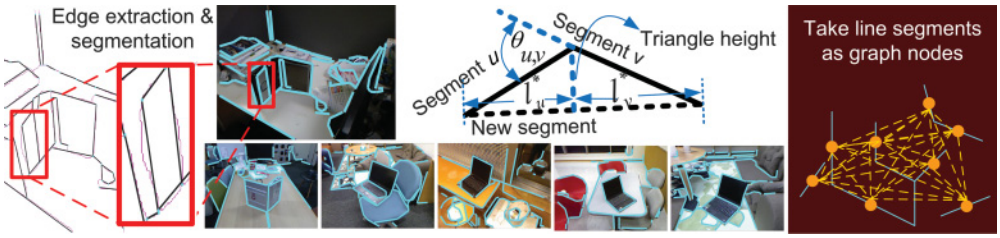

Fig. 3. Edge segmentation and illustration of variables: Edge extraction and segmentation (*left*); notation for edge segmentation (*middle*); line segments are taken as the graph nodes (*right*).

2010; Wang et al. 2013; Chen et al. 2013] have been developed more recently as a sophisticated, supervised means of modeling the spatial relationship between object parts. Lai and Fox [2010] and Hsiao et al. [2010] directly used a 3D model to detect objects in 2D images, whereas Hu [2010, 2012] and Pepik et al. [2012] used object multiview appearances to estimate the 3D structure. Recently, the appearance of RGB-D imaging technology has made object detection considerably easier [Aldoma et al. 2012; Lai et al. 2011b; Susanto et al. 2012], such that some low-level segmentation techniques [Collet et al. 2011; Ren et al. 2012; Silberman et al. 2012] can roughly achieve object-level results.

However, when we just use a single-labeled object to start the category modeling, it can only provide specific 2D structure and appearance from one viewpoint. Thus, the category model should (1) encode the structural information, (2) be able to detect objects with various scales and rotations (even roll rotations), and (3) be learned without further manual labeling.

Graph matching satisfies the first two requirements and has been widely used [Duchenne et al. 2011; Kim et al. 2012; Cho and Lee 2012; Hu et al. 2013; Zhou and De la Torre 2013]. The third requirement has been solved by Leordeanu and Hebert [2012], who first proposed an unsupervised[2] method for learning graph-matching-based models. Though we can use graphical models to represent object categories in both RGB and RGB-D images, 2D object structures in RGB images are not robust with respect to viewpoint changes. Thus, we use a 3D model based on RGB-D images to collect training samples for the learning of 2D models.

3. GRAPHICAL MODEL OF OBJECT EDGE SEGMENTS AND GRAPH MATCHING

We use a graphical model to encode the local and pairwise attributes of objects, that is, the part features and the spatial relationship between them. Model-based object detection is thus achieved via graph matching. This design ensures the robustness of viewpoint variation and roll rotations. In contrast to previous studies based on POI in images, voxels, or surfaces [Olsson and Boykov 2012; Liu and Yan 2012] in point clouds, our model uses object edges to represent object structures. We use the method of Arbelaez et al. [2011] to extract object edges in RGB images[3] and then discretize them into line segments. The line segments are used as graph nodes, as shown in Figure 3.

We connect each pair of edge segments in the labeled object to generate a complete undirected graph G as the initial category model, in which parameters will be trained

[2]In the context of learning graph matching, the term "unsupervised" [Leordeanu and Hebert 2012] refers to the ability to learn the model without manually specifying each individual matching assignment from the model to target graphs (cluttered scenes). In other words, under unsupervised learning, we do not have to manually label target objects in the cluttered scenes.

[3]Actually, object edges can be directly extracted in 4D space (i.e., RGB-D space) [Ren et al. 2012], but we simply extract edges just from RGB images. This is because we have to ensure that the edge extraction procedure for training is the same as that for testing in RGB images, in order to avoid potential differences between training and testing.

later. Given a target scene, let its corresponding target graph be G'. $\mathbf{f}_i$ and $\mathbf{f}_{ij}$ denote the local attributes of node i and pairwise attributes of edge ij in G, respectively. The matching assignments between G and G' can be defined using a matching matrix $\mathbf{y}$, where $y_{ii'} \in \{0, 1\}$. Further, $y_{ii'} = 1$ if node i in G maps to node i' in G', otherwise $y_{ii'} = 0$. We set $\sum_{i'} y_{ii'} = 1$ for all i. Therefore, the graph matching is formulated as

$$\hat{\mathbf{y}} = \underset{\mathbf{y}}{\operatorname{argmax}} \mathcal{C}, \qquad \mathcal{C} = \sum_{ii'} \rho_{ii'} y_{ii'} + \sum_{ii'jj'} \rho_{ii'jj'} y_{ii'} y_{jj'}, \tag{1}$$

where $\rho_{ii'}$ and $\rho_{ii'jj'}$ denote attribute compatibility for the unary assignment $i \to i'$ and the pairwise assignment $ij \to i'j'$, respectively. In general, they are functions of graph attributes:

$$\rho_{ii'} = \Phi_1\left(\mathbf{f}_i, \mathbf{f}_{i'}; \mathbf{w}^U\right), \qquad \rho_{ii'jj'} = \Phi_2\left(\mathbf{f}_{ij}, \mathbf{f}_{i'j'}; \mathbf{w}^P\right), \tag{2}$$

where $\mathbf{w}^U$ and $\mathbf{w}^P$ are parameter weightings for attributes.

In our study, we bring in an additional dummy matching choice—*none*—that is organized as a node in G'. This is necessary because some parts of the target objects in large cluttered scenes may be occluded; therefore some model nodes should not be matched to target scenes:

$$\rho_{i,none} = \kappa E(\rho_{ii'}), \qquad \rho_{i,none,jj'} = \rho_{ii'j,none} = \kappa E(\rho_{ii'jj'}), \tag{3}$$

where κ ($=1^4$, here) controls the matching priority of *none*.

Note that many-to-one matches may introduce errors into the learning of pairwise attributes. Considering that the compatibility in (2) is positive in our study, we simply modify unary compatibility as $\rho_{ii'jj'} = -1$ if and only if $i' = j'$ to avoid many-to-one matches.

We design two sets of local and pairwise attributes for the graphical model, so that the model can be applied to object collection (from RGB-D images) and object matching (from RGB images), respectively.

Edge segmentation: We use the local growth strategy to achieve edge segmentation. First, we initialize each pair of neighboring edge points as a tiny line segment, then gradually merge neighboring segments into longer and straighter lines. We finally map the edge segments in RGB images to the depth space to represent the 3D object structure.

Note that there is local nonsmoothness on the edges due to low image quality and texture variations. We, therefore, design a penalty metric to overcome such nonsmoothness in the segment-merging process. As illustrated in Figure 3, we merge neighboring segments u and v into a longer segment. Because angles between shorter segments are more sensitive to local perturbations, the penalty of their supplementary angle $\theta_{u,v}$ is calculated as $Pen_{u,v}^{\text{angle}} = \theta_{u,v}(1 - U_{u,v})$, where $U_{u,v} = e^{-\tau \min\{l_u^*, l_v^*\}}$ measures the noise level for segment pair of u and v, τ ($= 0.2$, here) controls the decrease speed, and l_u^* and l_v^* are the projected lengths of segments u and v on the new segment.

In addition, we propose another penalty metric that prevents a very short segment connecting to a long one, as $Pen_{u,v}^{\text{length}} = \frac{l_u^*}{l_u^*+l_v^*} \log \frac{l_u^*}{l_u^*+l_v^*} + \frac{l_v^*}{l_u^*+l_v^*} \log \frac{l_v^*}{l_u^*+l_v^*}$. This metric is created because the orientation measurement of long segments suffers less from local nonsmoothness than that of short segments. We want to avoid transferring the orientation unreliability from the short segment to the long segment during the merge process.

Therefore, the total penalty is calculated as follows:

$$Pen_{u,v} = Pen_{u,v}^{\text{angle}} + \eta Pen_{u,v}^{\text{length}}, \tag{4}$$

[4]In this article, most parameters are simply set to 1, and we only manually set a few parameters, such as τ, η, β, and k. This parameter setting is equally applied to all the categories.

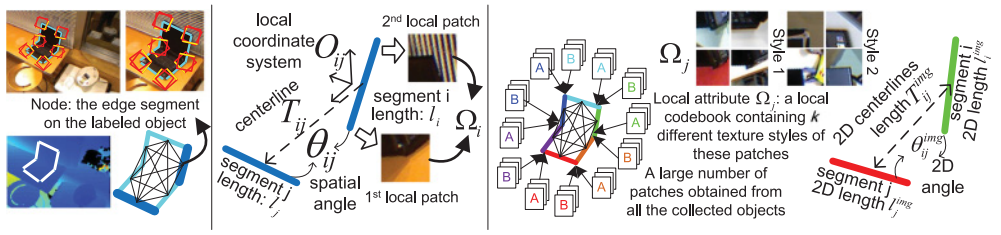

Fig. 4. Graphical models. Line segments in RGB-D/RGB images are taken as the nodes in a complete graph (*left*). The red squares indicate the image patches collected at terminals of the line segments. Model for object collection from RGB-D images (*middle*). Category model trained for ordinary RGB images (*right*).

where η ($=0.5$, here) is a weighting for the two penalty metrics. Segment pairs with lower penalty scores are merged earlier. We set the stopping criterion as follows. For each merge, the height of the triangle consisting of old and new segments should not be greater than six pixel units (Figure 3). Finally, we consider the line segments longer than 15 pixel units to be reliable ones, and select them as graph nodes.

3.1. Model for Object Collection from RGB-D Images

The graphical model proposed earlier is a paradigm. We design a set of attributes to adapt it for collecting objects in RGB-D scenes, while simultaneously extracting the correspondences of local patches between objects for further learning. Figure 4 presents the notation of this model.

Spatial length: We take the spatial length, denoted by l_i, as a local attribute. The length penalty for assignment $i \to i'$ is calculated as $|\log \frac{l_{i'}}{l_i}|$. We can thus obtain length attributes' matching compatibility as

$$P_{ii'}^{length} = e^{-|\log l_i - \log l_{i'}|/\beta}, \tag{5}$$

where β ($=2$, here) responds to the deformability level.

Patch features: We collect two local patches at the terminal points of each edge segment, then normalize them to their *right* orientations. We use their HoG features [Dalal and Triggs 2005] as another type of local attribute (details follow in Section 3.3). Let $\Omega_i = \{\varpi_i^A, \varpi_i^B\}$ denote the HoG features of the two patches of node i in G. We formulate the patch features' compatibility using a Gaussian distribution as

$$P_{ii'}^{patch} = \mathcal{G}\left(\left[dist\left(\varpi_i^A, \Omega_{i'}\right), dist\left(\varpi_i^B, \Omega_{i'}\right)\right]^T | \mu = 0, (\sigma^{patch})^2\mathbf{I}\right) \tag{6a}$$

$$dist(\varpi_i, \Omega_{i'}) = \min_{\varpi_{i'} \in \Omega_{i'}} \|\varpi_i - \varpi_{i'}\|_2, \tag{6b}$$

where $\mathcal{G}(\cdot)$ denotes a Gaussian function and $(\sigma^{patch})^2$ ($=1$, here) is the covariance. $dist(\cdot,\cdot)$ is a metric for distance measurement between patch features.

Spatial angle: The spatial angle between nodes i and j in G, θ_{ij}, is a widely used pairwise attribute. We assume its compatibility to follow a Gaussian distribution:

$$P_{ii'jj'}^{angle} = \mathcal{G}(\theta_{i'j'} | \mu = \theta_{ij}, (\sigma^{angle})^2), \tag{7}$$

where $(\sigma^{angle})^2$ ($=1$, here) is the variation.

Centerline: Another pairwise attribute describes the relative spatial translation between two nodes. We use the *centerline*—connecting the centers of two node segments—to measure the translation. We define a local 3D coordinate system based on the

segments in order to make the centerline's measurement independent of the global rotation of the object. Let $\mathbf{o_i}$ and $\mathbf{o_j}$ denote the unit 3D orientation of node segments i and j. The three orthogonal unit vectors of this coordinate system are calculated as $\mathbf{O}_{ij} = [\frac{\mathbf{o_i}+\mathbf{o_j}}{\|\mathbf{o_i}+\mathbf{o_j}\|_2}, \frac{\mathbf{o_i}-\mathbf{o_j}}{\|\mathbf{o_i}-\mathbf{o_j}\|_2}, \mathbf{o_i} \times \mathbf{o_j}]$. Thus, the 3D translation $\mathbf{T}_{ij}$ can be measured as $\mathbf{d}^{ij} = \mathbf{O}_{ij}^T \mathbf{T}_{ij}$. Note that we have two choices to define the orientation of node segment i, $\mathbf{o_i}$, and $-\mathbf{o_i}$; therefore, we instead use $\mathbf{c}_{ij} = [\min\{|d_1^{ij}|, |d_2^{ij}|\}, \max\{|d_1^{ij}|, |d_2^{ij}|\}, |d_3^{ij}|]^T$ as the centerline coordinates. The compatibility of centerline coordinates is also assumed to follow a Gaussian distribution:

$$P_{ii'jj'}^{center} = \mathcal{G}(\mathbf{c}_{i'j'} | \mu = \mathbf{c}_{ij}, \left(\sigma_{ij}^{\text{cen}}\right)^2 \mathbf{I}), \qquad \left(\sigma_{ij}^{\text{cen}}\right)^2 = (\alpha \|\mathbf{c}_{ij}\|_2)^2 + (\sigma^{noise})^2 \tag{8}$$

In fact, the variation is caused by both the structural deformability and noise; we use parameters $\alpha = 1$ and $\sigma^{noise} = 5$ to control the two factors, respectively.

Now, we summarize the model for 3D objects as follows. Its local and pairwise attributes are defined as $\mathbf{f}_i = [l_i, \Omega_i]$, $\mathbf{f}_{ij} = [\theta_{ij}, \mathbf{c}_{ij}]$, and the parameters are denoted by $\mathbf{w}^U = [\beta, \sigma^{patch}]$, $\mathbf{w}^P = [\sigma^{angle}, \sigma^{noise}, \alpha]$. We thus calculate the overall compatibility for unary and pairwise assignments as

$$\rho_{ii'} = \Phi_1(\mathbf{f}_i, \mathbf{f}_{i'}; \mathbf{w}^U) = P_{ii'}^{length} P_{ii'}^{patch}, \qquad \rho_{ii'jj'} = \Phi_2(\mathbf{f}_{ij}, \mathbf{f}_{i'j'}; \mathbf{w}^P) = P_{ii'jj'}^{angle} P_{ii'jj'}^{center}. \tag{9}$$

Because the local and pairwise attributes are designed based on the explicit 3D object structures, the problem of scale changes in RGB images can be overcomed. $\Phi_1(\cdot)$ and $\Phi_2(\cdot)$ are positive-bounded functions. We can, therefore, transform the compatibility maximization in (1) to an energy minimization problem and solve it by TRW-S [Kolmogorov 2006]. Finally, we use the matching rate Υ to ensure the overall matching quality: $\Upsilon = N^{detect}/(N^{detect} + N^{none})$, where N^{detect} and N^{none} denote the number of nodes that are matched to real segments in images and *none*, respectively. An incorrect match will produce a large N^{none} and thus a small Υ. Therefore, we only select those matches with $\Upsilon \geq 0.7$ as reliable results for further model learning. See Figure 6 for object-collection performances.

3.2. Category Model for Ordinary RGB Images

In ordinary RGB images, depth information can no longer be used. We thereby design new local and pairwise attributes to match objects in RGB images. See Figure 4 for the notations.

We learn a local codebook for each node i in G as the only local attribute, which consists of a set of patch features $\Omega_i = \{\varpi_i^k\}, (k = 1, 2, \ldots)$. The codebook contains different local texture styles to overcome texture variations. Details follow in Section 4.1.1.

We then define three types of pairwise attributes as follows. (1) θ_{ij}^{img} represents the angle between nodes i and j in G on the image plane. (2) The other pairwise attribute is $[\lambda_{ij}^A, \lambda_{ij}^B] = \frac{1}{T_{ij}^{img}}[l_i^{img}, l_j^{img}]$, where l_i^{img} denotes the segment length of node i, and T_{ij}^{img} denotes the length of the centerline between nodes i and j in G. Considering scale changes in RGB images, the length measurement is normalized by T_{ij}^{img}. (3) The third pairwise attribute describes the relative angles between the centerline and line segments of nodes i and j, denoted by $[\theta_{ij}^A, \theta_{ij}^B]$.

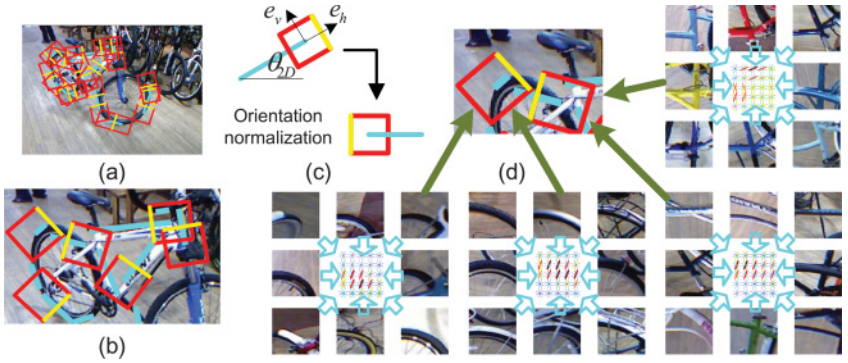

Fig. 5. Local codebook extraction. (a) The bicycle is detected by 3D matching. Patches (*red*) are extracted at terminals of the detected segments (*blue*). Yellow sides indicate patch orientations. (b) A detailed view. (c) Patch orientation normalization. (d) Given image patches corresponding to each of the two bicycle parts, we use k-means clustering ($k = 2$ for clarity) to obtain two texture styles as a sparse local codebook. We use the HoG template to represent the texture style of its surrounding image patches.

We absorb local compatibilities into the pairwise compatibilities, $\mathbf{f}_{ij} = \{\theta_{ij}^{img}, \lambda_{ij}^{A}, \lambda_{ij}^{B}, \theta_{ij}^{A}, \theta_{ij}^{B}, \Omega_i, \Omega_j\}$.

$$\rho_{ii'} = 0, \quad \rho_{ii'jj'} = \Phi_2(\mathbf{f}_{ij}, \mathbf{f}_{i'j'}; \mathbf{w}) = e^{-w_1|\theta_{ij}^{img} - \theta_{i'j'}^{img}|^2 - \sum_{k\in\{A,B\}}\{w_2|\lambda_{ij}^{k} - \lambda_{i'j'}^{k}|^2 + w_3|\theta_{ij}^{k} - \theta_{i'j'}^{k}|^2 + w_4[dist^2(\varpi_i^k, \Omega_i) + dist^2(\varpi_j^k, \Omega_j)]\}} \tag{10}$$

The distance between the local codebook $dist(\cdot, \cdot)$ is defined in Equation (6b). Similar to the model for RGB-D images, we use the TRW-S [Kolmogorov 2006] to solve the maximization problem.

3.3. HoG Feature Extraction

This section introduces the details of HoG feature extraction that is present in both the graphical model for RGB-D images and the one for RGB images. For each model node, we extract a set of patches from its matched node segments in the target scenes, as shown in Figure 6. We first extract these patches from the two terminals of the edge segment, then normalize them to their *right* orientations to eliminate rotation effects. As shown in Figure 5, the patches are collected using a square, which is rotated to the orientation of the edge segment.

HoG features [Dalal and Triggs 2005] are extracted using 5×5 cells, each of which covers half of its neighboring cells. We use four orientation bins (from 0° to 180°) to compute the gradient histogram in each cell. Because the patch is locally collected without significant illumination changes, we normalize all of the cells within a single block.

4. MODEL LEARNING ALGORITHMS

In this section, we focus on the model-learning algorithm and its several technical extensions. The framework of the basic algorithm, that is, model learning from RGB-D images, is proposed in Section 4.1. Then, in Section 4.2, we present the recognition method based on the trained models. In addition, we further extend the basic model-learning method to overcome two main challenges of visual mining by applying the strategy of knowledge transfer and initial labeling refinement, presented in Sections 4.3 and 4.4, respectively.

Fig. 6. Object collection. Given the single-labeled object (to the left of the red line), target objects are detected using the 3D structure (1^{st} and 4^{th} rows), and local patches are collected for each object part (2^{nd} and 5^{th} rows). The 3^{rd} and 6^{th} rows show detailed views of patch extraction. Patches (*red*) are extracted at terminals of the detected segments (*blue*). Yellow sides indicate patch orientations.

4.1. Basic Framework: Model Learning from RGB-D Images

In this section, we use relatively reliable 3D matches to guide the training of the category model for ordinary RGB images in order to avoid the bias problem. Based on the part correspondences estimated in 3D matching, we construct a local codebook for each model node that covers all possible local texture styles. The work of Leordeanu and Hebert [2012] is then extended for the training of both matching parameters and model attributes.

4.1.1. Local Codebook Extraction. During RGB-D object collection, we collect a set of image patches that correspond to each node in the category model. We then cluster the HoG features of each node i's patches via k-means clustering ($k = 5$). Consequently, the cluster centers represent a sparse set of visual words for this node, which compose the local codebook Ω_i. Figure 5(d) shows that we use a set of HoG patterns to describe an object part's different texture styles exhibited in different images. The local codebook therefore contains sufficient texture patterns to overcome texture variations during object detection.

4.1.2. Model-Learning Algorithm. We can rewrite the model-based graph matching defined by (1) and (10) as

$$\arg\max_{\mathbf{y}} \mathcal{C} = \arg\max_{\mathbf{y}} \mathbf{y}^T \mathbf{M} \mathbf{y}, \tag{11}$$

where $M_{(ii'),(jj')} = \rho_{ii'jj'}$. In this equation, $\mathbf{y}$ is transformed from a matching matrix to a vector.

According to Leordeanu and Hebert [2005], elements of the principal eigenvector $\mathbf{x}$ of $\mathbf{M}$, *e.g.* $x_{ii'}$, can be taken as the confidence value of the assignment $i \rightarrow i'$[5]. To reduce the

[5]Note that $\rho_{i,none,jj'}$ and $\rho_{ii'j,none}$ are not involved in M.

large computation, we apply the approximate principal eigenvector used in Leordeanu and Hebert [2012].

$$\mathbf{x} = \mathbf{M^n 1}/\sqrt{(\mathbf{M^n 1})^T(\mathbf{M^n 1})}. \tag{12}$$

The partial derivatives of $\mathbf{x}$ are thus computed as

$$\mathbf{x}' = [(\mathbf{M^n 1})'\|\mathbf{M^n 1}\| - ((\mathbf{M^n 1})^T(\mathbf{M^n 1})')\mathbf{M^n 1}/\|\mathbf{M^n 1}\|]/\|\mathbf{M^n 1}\|^2, \tag{13}$$

where $(\mathbf{M^n 1})' = \mathbf{M}'(\mathbf{M^{n-1}1}) + \mathbf{M}(\mathbf{M^{n-1}1})'$ and $n = 10$, as in Leordeanu and Hebert [2012].

Leordeanu and Hebert [2012] proposed training matching parameters $\mathbf{w}$ to increase confidence values $x_{ii'}$ of the correct assignments. At the same time, confidence values of incorrect assignments will decrease, as $\mathbf{x}$ is normalized. We extend this idea to learn both the parameters and the model attributes $\{\mathbf{w}, \mathbf{f}\}$ by maximizing the following function:

$$\mathcal{F}(\mathbf{w}, \mathbf{f}) = \sum_{i=1}^{N} \left(\mathbf{x}^{(i)}(\mathbf{w}, \mathbf{f})\right)^T \mathbf{t}^{(i)}, \tag{14}$$

where $i = 1, 2, \ldots, N$ indicates each target scene for training, and $\mathbf{t}^{(i)}$ denotes the predicted matching assignment in scene i.

We implement the whole model-training framework as follows. We first initialize $\{\mathbf{w}, \mathbf{f}\}$ using the labeled object, then iteratively modify $\{\mathbf{w}, \mathbf{f}\}$ to maximize $\mathcal{F}(\mathbf{w}, \mathbf{f})$. Intuitively, we can directly predict the correct matching assignments as the 3D matching results in the RGB-D image, $\mathbf{t}^{(i)} = \hat{\mathbf{y}}^{3D,(i)}$. However, some categories may have several potential assignment states owing to their symmetric 3D structures, for example, notebook PCs. These matching states are equivalent in terms of the 3D structure; however, they may show different matching compatibilities for ordinary 2D image matching. Thus, the matching assignments predicted by the category model (denoted by $\hat{\mathbf{y}}^{img,(i)}$) are not always the same as $\hat{\mathbf{y}}^{3D,(i)}$.

We therefore use $\hat{\mathbf{y}}^{img,(i)}$ to compute $\mathbf{t}^{(i)}$. Errors in $\hat{\mathbf{y}}^{img,(i)}$ are detected and eliminated by $\hat{\mathbf{y}}^{3D,(i)}$ to avoid the bias problem. We use the following criterion to identify correct matches from $\hat{\mathbf{y}}^{img,(i)}$. Correct 2D matches in $\hat{\mathbf{y}}^{img,(i)}$ should match the nodes in image i that are also matched by 3D matching $\hat{\mathbf{y}}^{3D,(i)}$. Thus, we get

$$\mathbf{t}^{(i)} = diag\{a^{(i)}_{jj'}\}\hat{\mathbf{y}}^{img,(i)}, \qquad a^{(i)}_{jj'} = \sum_{j} \hat{y}^{3D,(i)}_{jj'}. \tag{15}$$

In the k-th iteration, we use (15) to estimate matching assignment $\mathbf{t}^{(i),k}$ and modify each $w \in \mathbf{w}$ and $f \in \mathbf{f}$ via gradient ascent:

$$w^{k+1} \leftarrow w^k + \zeta \sum_{i=1}^{N} \left(\mathbf{t}^{(i),k}\right)^T \frac{\partial \mathbf{x}^{(i),k}(\mathbf{w}, \mathbf{f})}{\partial w}, \quad f^{k+1} \leftarrow f^k + \zeta \sum_{i=1}^{N} \left(\mathbf{t}^{(i),k}\right)^T \frac{\partial \mathbf{x}^{(i),k}(\mathbf{w}, \mathbf{f})}{\partial f}. \tag{16}$$

4.2. Object Recognition Based on Category Models

Originally, the category model is trained for object detection in RGB images. This section introduces three methods to further train object recognition, given the trained category models. Successful object recognition is defined as the correct determination of whether an image contains the target object of a model. Given a category model, we use a set of positive RGB images and the same number of negative RGB images

to train object recognition. The positive images are the RGB channels of the training RGB-D images used in Section 4.1. The negative images are the images that do not contain the target objects, which are selected randomly from the RGB image sets of other categories.

The most popular and simplest method for object recognition is based on matching compatibility[6] [Hong and Huang 2004; Yuan et al. 2012; Xie et al. 2012]. Given a testing RGB image, we compute the matching compatibility $\mathcal{C}$ in Equation (1) between this image and the model. If $\frac{\mathcal{C}-\mu_{\mathcal{C}}}{\sigma_{\mathcal{C}}}$ is greater than a threshold υ, the matching results are considered to represent the true detection of the target object, otherwise, a false detection. $\mu_{\mathcal{C}}$ and $\sigma_{\mathcal{C}}$ are the mean and standard deviation of $\mathcal{C}$ when we match the model to all of the positive and negative training images[7].

The second method trains a kernel-based support vector machine (SVM) for object recognition as

$$f(\mathbf{y}) = sgn\left(\sum_{i^+} \alpha_{i^+} K(\mathbf{y}_{i^+}, \mathbf{y}) - \sum_{i^-} \alpha_{i^-} K(\mathbf{y}_{i^-}, \mathbf{y}) + b\right), \tag{17}$$

where $\mathbf{y}$ denotes the matching results in the testing image computed in Equation (1); $\mathbf{y}_{i^+}$ and $\mathbf{y}_{i^-}$ are the matching results in each positive and negative image, respectively.

We modify the kernel for graph matching proposed by Wallraven and Caputo [2003] and Duchenne et al. [2011] to classify the matching results in positive and negative images:

$$\begin{aligned} K(\mathbf{y}, \mathbf{y}') = \sum_{ij} \exp\Bigg\{ & -w_1 \left|\theta^{img}_{y(i)y(j)} - \theta^{img}_{y'(i)y'(j)}\right|^2 - w_4 \left| \frac{1}{2} \sum_{k\in\{A,B\}} \left(\varpi^k_{y(i)} - \varpi^k_{y'(i)}\right)\right|^2 \\ & - \sum_{k\in\{A,B\}} \left[w_2 \left|\lambda^k_{y(i)y(j)} - \lambda^k_{y'(i)y'(j)}\right|^2 + w_3 \left|\theta^k_{y(i)y(j)} - \theta^k_{y'(i)y'(j)}\right|^2 \right] \Bigg\} \end{aligned}. \tag{18}$$

This is a Mercer kernel (see Wallraven and Captuo [2003] for the proof), which measures the similarity between the corresponding parts of $\mathbf{y}$ and $\mathbf{y}'$. The functions $y(i)$ and $y'(i)$ denote the matching assignments of model node i with regard to $\mathbf{y}$ and $\mathbf{y}'$. Let k be a node in an image. $y(i) = k$, if and only if $\mathbf{y}_{ik} = 1$. The other notation is presented in Equation (10). Note that we simply define $f_{y(i)y(j)} = -\mathbf{1}$, if $y(i)$ or $y(j)$ is equal to *none* for any $f_{y(i)y(j)} \in \{\theta^{img}_{y(i)y(j)}, \lambda^A_{y(i)y(j)}, \lambda^B_{y(i)y(j)}, \theta^A_{y(i)y(j)}, \theta^B_{y(i)y(j)}, \Omega_{y(i)}, \Omega_{y(j)}\}$.

The final approach for the recognition of graph matching results is the technique used by Zhang et al. [2013b]. This method designs a set of features to measure the matching quality of each graph-matching result, then trains a linear SVM for object recognition. The features of matching result $\mathbf{y}$ are defined as $\mathbf{f}' =< f'_i >$, where f'_i denotes the feature for node i of the category model, and N^{model} is the node number of

[6]This recognition approach can be used if the target function of graph matching is designed to maximize the matching compatibilities or to minimize the matching penalties.

[7]The value range of $\mathcal{C}$ varies among models because of variations in the value of $\mathbf{w}$, but the normalization of $\mathcal{C}$ aims to remove such effects.

the model. f_i' is computed as

$$f_i' = \text{sqrt}\left\{ w_4 \sum_{k\in\{A,B\}} dist^2\left(\varpi_{y(i)}^k, \Omega_i\right) + \frac{1}{N^{model}} \sum_j \left[w_1 \left|\theta_{ij}^{img} - \theta_{y(i)y(j)}^{img}\right|^2 + \sum_{k\in\{A,B\}} \left(w_2 \left|\lambda_{ij}^k - \lambda_{y(i)y(j)}^k\right|^2 + w_3 \left|\theta_{ij}^k - \theta_{y(i)y(j)}^k\right|^2 \right) \right] \right\}. \quad (19)$$

4.3. Technical Extension 1: Model Learning Via Knowledge Transfer

In this section, we focus on model learning from ordinary RGB images, because it is not realistic to obtain a large number of RGB-D training images for every category. The RGB images can be easily collected using search engines[8]. However, without the guidance of 3D structural information, the bias problem in model learning becomes severe due to incorrect object collection (graph matching) in large and cluttered scenes (demonstrated by the experimental results to follow). Fortunately, we can transfer some common knowledge from the existing models of some categories (trained using RGB-D images) to guide the model learning for a new category. We use this knowledge to identify incorrect matches, which reduces the bias problem to some extent.

Model attributes and parameters of the existing models contain information related only to their own categories, not the new category. However, they have similar rules for identifying correct and incorrect graph matching. In general, if graph matching is correct, the attribute difference between two matched graphs is small; it is large otherwise. Thus, for each type of attribute, we can learn the value ranges of the attribute differences for correct and incorrect graph matching. We define the following feature for the matching result $\mathbf{y}$ to measure the attribute differences:

$$F_{\mathbf{y}} = \left[\underset{ij}{mean}\left(F_{ij}^1\right), \underset{ij}{var}\left(F_{ij}^1\right), \underset{ij}{mean}\left(F_{ij}^2\right), \underset{ij}{var}\left(F_{ij}^2\right), \underset{ij}{mean}\left(F_{ij}^3\right), \underset{ij}{var}\left(F_{ij}^3\right), \underset{i}{mean}\left(F_i^4\right), \underset{i}{var}\left(F_i^4\right) \right]^T, \quad (20)$$

where the functions $mean(\cdot)$ and $var(\cdot)$ compute the mean and variation, and $y(i)$ denotes the matching assignments of model node i with regard to $\mathbf{y}$, which is similar to that in Equation (18). $F_{ij}^1 = |\theta_{y(i)y(j)}^{img} - \theta_{ij}^{img}|$, $F_{ij}^2 = \sum_{k\in\{A,B\}} |\lambda_{y(i)y(j)}^k - \lambda_{ij}^k|$, $F_{ij}^3 = \sum_{k\in\{A,B\}} |\theta_{y(i)y(j)}^k - \theta_{ij}^k|$, $F_i^4 = |\sum_{k\in\{A,B\}} (\varpi_{y(i)}^k - \varpi_i^k)|$.

Therefore, we can use feature $F_{\hat{\mathbf{y}}^{img,(i)}}$ to identify the correctness of the model's matching result in image i. We prepare a set of correct (positive) and incorrect (negative) matching results[9] for each existing model. Thus, we obtain a high number of matching results based on the existing models to train a nonlinear SVM for classifying the matching results.

$$\min_{\mathbf{w},b,\xi} \frac{1}{2}\mathbf{w}^T\mathbf{w} + C_1 \sum_{i^+} \xi_{i^+} + C_2 \sum_{i^-} \xi_{i^-}$$
$$\text{subject to } \mathbf{w}^T\phi(F_{i^+}) + b \geq 1 - \xi_{i^+};\ -(\mathbf{w}^T\phi(F_{i^-}) + b) \geq 1 - \xi_{i^-};\ \xi_{i^+}, \xi_{i^-} \geq 0, \quad (21)$$

where $K_{RBF}(\cdot,\cdot) = \phi(\cdot)^T\phi(\cdot)$ is a radial basis function (RBF) kernel. We set $C_1 = 1$ and $C_2 = 10$, which consider the unavoidable incorrect matching results produced by

[8]We use the RGB channels of the RGB-D images in the dataset [Zhang 2013] as the training images in experiments.

[9]The correct and incorrect matching results are the matching results in the positive and negative RGB images with regard to the category of the model, which are the same as those described in Section 4.2.

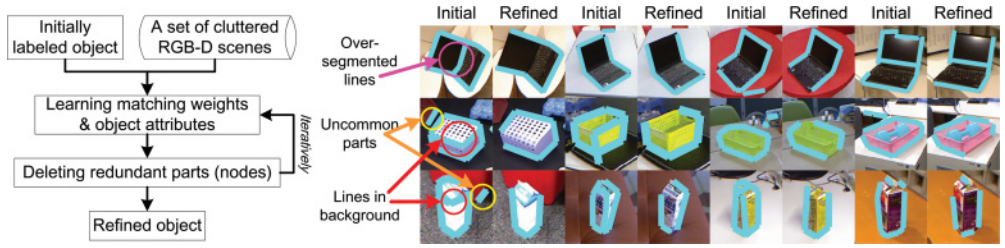

Fig. 7. Structure refinement of the initially labeled object. The flowchart is shown on the left, and the results are shown on the right.

the existing models. F_{i^+} and F_{i^-} denote the positive and negative training samples[10], respectively.

We modify the algorithm for "learning from RGB-D images" proposed in Section 4.1 to implement this knowledge-transfer-based learning. Two modifications are applied, as follows. First, we modify the learning of model parameters and attributes in the k-th iteration from Equation (16) to

$$w^{k+1} \leftarrow w^k + \zeta \sum_{i=1}^{N} \pi_i^k \left(\hat{\mathbf{y}}^{img,(i),k}\right)^T \frac{\partial \mathbf{x}^{(i),k}(\mathbf{w},\mathbf{f})}{\partial w}, \quad f^{k+1} \leftarrow f^k + \zeta \sum_{i=1}^{N} \pi_i^k \left(\hat{\mathbf{y}}^{img,(i),k}\right)^T \frac{\partial \mathbf{x}^{(i),k}(\mathbf{w},\mathbf{f})}{\partial f}, \tag{22}$$

where $\hat{\mathbf{y}}^{img,(i),k}$ denotes the matching assignments in image i determined by the category model in the k-th iteration, which is the same as the notation in Equation (15). The parameter π_i^k corresponds to the transferred knowledge. π_i^k is the reliability of the matching results $\hat{\mathbf{y}}^{img,(i),k}$, which is used to assign a low weight to a possibly biased matching result in the learning process. π_i^k is computed using the decision value of the SVM, as described by Lin and Weng [2004].

$$\pi_i^k = 1/\left\{1+\exp\left\{-A\left[\sum_{i^+}\alpha_{i^+}K_{RBF}\left(F_{i^+}, F_{\hat{\mathbf{y}}^{img,(i),k}}\right) - \sum_{i^-}\alpha_{i^-}K_{RBF}\left(F_{i^-}, F_{\hat{\mathbf{y}}^{img,(i),k}}\right) + b\right]\right\}\right\} \tag{23}$$

We set $A = 3$ in this case.

Second, we cannot pretrain a local-texture codebook for each node in the category model for preprocessing because the part correspondences between training images cannot be extracted reliably without depth information. Instead, we simply set the local codebook as the patch features $\Omega_i = \{\varpi_i^A, \varpi_i^B\}$ for each node i, and train Ω_i based on Equation (22), just as other parameters w_j.

4.4. Technical Extension 2: Initial Labeling Refinement

The basic model learning method introduced in Section 4.1 demands appropriate labeling of the initial RGB-D object to avoid bias during training object collection. People are capable of accurate labeling in most cases, but subjective labeling cannot be guaranteed to reflect the underlying structural features of a category in all cases. Therefore, as shown in Figure 7 we also utilize our previous method [Zhang et al. 2013b] to refine the initial labeling before model learning.

[10]The existing models are all well trained, which means that they usually produce smaller attribute differences than a target model that is still in training. Thus, during the preparation of each training sample (F_{i^+} or F_{i^-}), we randomly select another positive matching result $\mathbf{y}^+$, and use the matched attributes of $\mathbf{y}^+$ to replace the model attributes. In other words, Equation (20) is modified, for example, $F_{ij}^1 = |\theta_{y(i)y(j)}^{img} - \theta_{ij}^{img}| \rightarrow F_{ij}^1 = |\theta_{y(i)y(j)}^{img} - \theta_{y'(i)y'(j)}^{img}|$.

In addition to the set of positive RGB-D images (containing target objects), we collect a set of negative (background) RGB-D images for learning. We train the local and pairwise attributes of the labeled RGB-D object, which is similar to the method introduced in Section 4.1.2. Further, we modify the object structure at the same time. The labeled object should comprise the object parts (nodes) that perform most reliably during graph matching, facilitating appropriate guidance when training RGB object models.

Graph matching in RGB-D images: Similar to Equations (1) and (11), we rewrite the graph matching between the labeled object $G(V, E)$ and an RGB-D image $G'(V', E')$, as follows.

$$\begin{aligned} \hat{\mathbf{y}} &= \underset{\mathbf{y}}{\operatorname{argmax}}\, \mathcal{C}(\mathbf{y}|G, G'), \quad \mathcal{C}(\mathbf{y}|G, G') = \mathbf{y}^T\mathbf{M}\mathbf{y} \\ \text{s.t.} \quad & \forall i \in V, \sum_{i' \in V'} y_{ii'} \le 1, \quad \forall i' \in V', \sum_{i \in V} y_{ii'} \le 1 \end{aligned}. \tag{24}$$

In graph G, we use the local and pairwise attributes introduced in Section 3.1, including two local attributes ($n^U = 2$) and three pairwise attributes ($n^P = 3$). We clarify the notation as follows. The local attributes of node i include the HoG patch features $f_i^{(1)} = \Omega_i$ and segment length $f_i^{(2)} = \log l_i$. The three pairwise attributes comprise the segment angle $f_{ij}^{(1)} = \theta_{ij}$ and the centerline's attributes $f_{ij}^{(2)} = \|\mathbf{c}_{ij}\|$, $f_{ij}^{(3)} = \mathbf{c}_{ij}/\|\mathbf{c}_{ij}\|$. $F_V = \{f_i^{(k)}|k = 1, 2; i \in V\}$, $F_E = \{f_{ij}^{(k)}|k = 1, 2, 3; (i, j) \in E\}$. Thus, we apply the following matching compatibility matrix:

$$M_{ii',jj'} = \begin{cases} \exp\left(-(\mathbf{w}^U)^{\mathbf{T}}\mathbf{d}_{ii'}^2 - (\mathbf{w}^U)^{\mathbf{T}}\mathbf{d}_{jj'}^2 - (\mathbf{w}^P)^{\mathbf{T}}\mathbf{d}_{ii',jj'}^2\right), & (i, j) \in E, (i', j') \in E' \\ 0, & \text{Otherwise} \end{cases}, \tag{25}$$

where we define $\mathbf{d}_{ii'} = [d_{ii'}^{(1)}, d_{ii'}^{(2)}, \ldots, d_{ii'}^{(n^U)}]^T$ as the distances of the corresponding unary attributes during matching, $d_{ii'}^{(k)} = \|f_i^{(k)} - f_{i'}^{(k)}\|$[11]. In addition, $\mathbf{d}_{ii',jj'} = [d_{ii',jj'}^{(1)}, d_{ii',jj'}^{(2)}, \ldots, d_{ii',jj'}^{(n^P)}]^T$ denotes the distances of the pairwise attributes, $d_{ii',jj'}^{(l)} = \|f_{ij}^{(l)} - f_{i'j'}^{(l)}\|$. $\mathbf{w}^U = [w_1^U, w_2^U, \ldots, w_{n^U}^U]^T$ and $\mathbf{w}^P = [w_1^P, w_2^P, \ldots, w_{n^P}^P]^T$ denote the weights for each unary and pairwise attribute, respectively.

During the actual application of graph matching, we add a dummy node *none* and a penalty for many-to-one matches. Equation (24) is reformulated as

$$\hat{\mathbf{a}} = \underset{\mathbf{a}}{\operatorname{argmax}} \sum_{i,j \in V \cup \{none\}} \mathcal{C}_{ij}, \quad \mathcal{C}_{ij} = \begin{cases} M_{ia_i,ja_j}, & a_i \ne a_j \in V' \\ -\infty, & a_i = a_j \in V' \\ \frac{\lambda(\mathbf{1}^T\mathbf{M}\mathbf{1})}{n_v^2 n_{v'}^2}, & a_i \text{ or } a_j = none \end{cases}, \tag{26}$$

where a_i indicates the matching assignment of node $i \in V$ and $a_i = i' \in V'$, if and only if $y_{ii'} = 1$. λ (= 5) is the parameter weighting for the penalty of *none*. n_v and $n_{v'}$ denote the node number of G and G', respectively.

Learning matching weights and attributes: The method used for learning the matching weights and attributes is similar to that described in Section 4.1.2. Let $\mathbf{x}$ denote the principal eigenvector of $\mathbf{M}$, which is approximated by $\mathbf{x} = \mathbf{M^n 1}/\sqrt{(\mathbf{M^n 1})^T(\mathbf{M^n 1})}$. The learning process is performed iteratively. During each iteration, we use the current G to predict the matching assignments; then, we modify the matching weights

[11]Note that the distance of patch features $f_i^{(1)} = \Omega_i$ is defined in Equation (6b).

($w \in \mathbf{w}^U \cup \mathbf{w}^P$) and attributes ($f \in F_V \cup F_E$) by gradient ascent:

$$w^{k+1} \leftarrow w^k + \zeta \sum_{i^+} \left(\mathbf{t}^{(i^+),k}\right)^T \frac{\partial \mathbf{x}^{(i^+),k}}{\partial w}, \quad f^{k+1} \leftarrow f^k + \zeta \sum_{i^+} \left(\mathbf{t}^{(i^+),k}\right)^T \frac{\partial \mathbf{x}^{(i^+),k}}{\partial f}, \tag{27}$$

where similar to (16), $\mathbf{t}^{(i^+),k}$ denotes the predicted matching assignments for positive image i^+ computed in iteration k. $t_{jj'}^{(i^+),k}$ is set to 1, if $a_j = j'$, 0 otherwise. See Zhang et al. [2013b] for further details.

Structure modification: Structure modification is combined with the iterative framework for training weights and attributes. During each iteration, after parameter regression in Equation (27), we delete a redundant part (node) of G, until G has the predetermined number of nodes. We identify the redundant node based on the assumption that a redundant node usually contributes less than other nodes to the classification of positive and negative images.

First, we need to define the features of each node for classification. Let $\hat{\mathbf{A}} = \{\hat{a}_i^k | k = 1, 2 \ldots, N^+, i \in V\}$ and $\mathring{\mathbf{A}} = \{\mathring{a}_i^l | l = 1, 2 \ldots, N^-, i \in V\}$ denote the predicted matching assignments of G, where $\hat{a}_i^k$ and $\mathring{a}_i^l$ indicate the assignment mapping node i to positive graph k and that to negative graph l based on Equation (26).[12] Thus, according to Equation (25), $\mathbf{d}_{i\hat{a}_i^k}$ indicates the distance of the unary attributes for matching node $i \in V$ to node $\hat{a}_i^k$ in positive graph k. $\mathbf{d}_{i\mathring{a}_i^l}$ indicates such distance for matching to the negative graph l. Similarly, $\mathbf{d}_{i\hat{a}_i^k, j\hat{a}_j^k}$ and $\mathbf{d}_{i\mathring{a}_i^l, j\mathring{a}_j^l}$ denote the pairwise attribute distances in positive and negative graphs, respectively.

Therefore, we use $\hat{\mathbf{u}}_i^k$ and $\hat{\mathbf{p}}_i^k$ (n^U-dimensional and n^P-dimensional vectors) as the matching incompatibilities of the unary and pairwise attributes, respectively, for the match between node $i \in V$ and node $\hat{a}_j^k$:

$$\hat{\mathbf{u}}_i^k = \mathbf{d}_{i\hat{a}_i^k}^T, \quad \hat{\mathbf{p}}_i^k = \sum\nolimits_{j:j\neq i} \mathbf{d}_{i\hat{a}_i^k, j\hat{a}_j^k}^T \Big/ \sum\nolimits_{j:j\neq i} 1. \tag{28}$$

The features for recognizing matches with positive graph k and negative graph l are defined as

$$\hat{\mathcal{F}}^k = \left[\hat{\mathbf{u}}_1^k, \hat{\mathbf{p}}_1^k, \hat{\mathbf{u}}_2^k, \hat{\mathbf{p}}_2^k, \ldots, \hat{\mathbf{u}}_{n_v}^k, \hat{\mathbf{p}}_{n_v}^k\right]^T, \quad \mathring{\mathcal{F}}^l = \left[\mathring{\mathbf{u}}_1^l, \mathring{\mathbf{p}}_1^l, \mathring{\mathbf{u}}_2^l, \mathring{\mathbf{p}}_2^l, \ldots, \mathring{\mathbf{u}}_{n_v}^l, \mathring{\mathbf{p}}_{n_v}^l\right]^T. \tag{29}$$

Consequently, we use these features to train a linear-SVM classifier for match classification. We then use this classifier to find the redundant node in G.

$$\min_{\mathcal{W},\xi,b} \left\{ \frac{1}{2}\|\mathcal{W}\|^2 + C \sum_{k=1}^{N^+ + N^-} \xi_k \right\}, \quad \begin{array}{l} \text{s.t.} \quad \forall k = 1, 2, \ldots, N^+, \mathcal{W} \cdot \hat{\mathcal{F}}^k - b \geq 1 - \xi_k, \ \xi_k \geq 0; \\ \forall k = 1, 2, \ldots, N^-, -(\mathcal{W} \cdot \mathring{\mathcal{F}}^k - b) \geq 1 - \xi_{k+N^+}, \ \xi_{k+N^+} \geq 0 \end{array} \tag{30}$$

We represent the normal vector *with regard to* the hyperplane $\mathcal{W}$ as $[\boldsymbol{\mu}_1^T, \boldsymbol{\rho}_1^T, \boldsymbol{\mu}_1^T, \boldsymbol{\rho}_1^T, \ldots, \boldsymbol{\mu}_{n_v}^T, \boldsymbol{\rho}_{n_v}^T]^T$. $\boldsymbol{\mu}_i$ is an n^U-dimensional vector, which weights for the matching incompatibility of node i's unary attributes ($\hat{\mathbf{u}}_i^k$ or $\mathring{\mathbf{u}}_i^l$), whereas the n^P-dimensional $\boldsymbol{\rho}_i$ weights for the matching incompatibility of its pairwise attributes ($\hat{\mathbf{p}}_i^k$ or $\mathring{\mathbf{p}}_i^l$).

Clearly, a good node i in G should match better with positive graphs than negative graphs. In general, therefore, $\hat{\mathbf{u}}_i^k$ and $\hat{\mathbf{p}}_i^k$ should be less than $\mathring{\mathbf{u}}_i^l$ and $\mathring{\mathbf{p}}_i^l$, respectively. Thus, the weights of node i (i.e., $\boldsymbol{\mu}_i$ and $\boldsymbol{\rho}_i$) should be negative, according to Equation (30). Therefore, we use the following metric to evaluate the reliability of node $i \in V$:

$$R_i = -\sqrt{n_v}\left(\mathbf{1}^T \boldsymbol{\mu}_i^{(j)} + \mathbf{1}^T \boldsymbol{\rho}_i^{(j)}\right) / \|\mathcal{W}\|. \tag{31}$$

[12] In this case, we set $\lambda = -\infty$ to avoid matching with *none*.

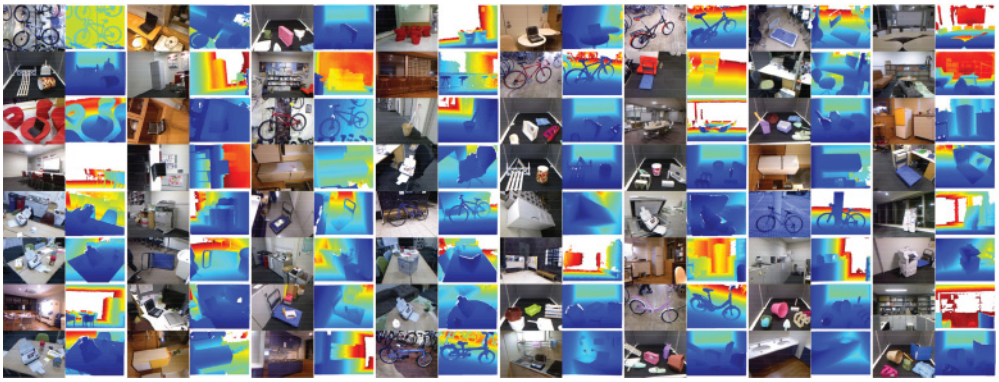

Fig. 8. Some samples in our category dataset of Kinect RGB-D images, published at http://sites.google.com/site/quanshizhang.

Iterative structural modification is performed as follows. During each iteration, we eliminate the node with the lowest reliability from G, that is, $i^* = \text{argmin}_{i \in V} R_i$, thereby updating the structure of G as the induced subgraph. See Zhang et al. [2013b] for further details.

5. EXPERIMENTS

We perform two experiments to evaluate the object matching and object recognition performance of the trained category models. We test the models trained using the strategies proposed in Sections 4.1, 4.3, and 4.4. Six competing methods are designed to train the category models, including semi-supervised and supervised approaches.

5.1. Data

A number of RGB-D datasets have been built in recent years. Lai et al. [2011a] built a large RGB-D object dataset containing 300 objects in 50 categories, Lai et al. [2011a] as well as an RGB-D scene dataset, primarily for use in object recognition. Koppula et al. [2011], Silberman et al. [2011], and Browatzki et al. [2011] constructed RGB-D image sets of indoor environments. A dataset for the perception challenge [Bradski and Hong 2011] consists of 35 objects for training; the UBC robot vision survey dataset [Helmer et al. 2010] contains four categories of objects; and the 3D table-top dataset [Sun et al. 2010] covers three categories without significant variation in viewpoint. Janoch [2012] and Wohlkinger et al. [2012] also built large datasets.

However, in our scenario of learning from large and cluttered RGB-D scenes, we have two requirements for the dataset. First, the target objects in the dataset should not be hand-cropped or aligned, and they should have different scales, textures, and rotations. Second, each category should include a large number of samples for training (in most existing datasets, each category contains only a few objects). A relatively large number of training images can ensure stability of model-learning processes of both the proposed method and the competing methods introduced in Section 5.2, although theoretically, all of these methods can be successfully applied, given just two cluttered scenes for each category as training images. Therefore, we have constructed our own RGB-D image dataset [Zhang 2013], published as a standard Kinect RGB-D object dataset oriented to graph matching[13], as shown in Figure 8. We use five categories in the dataset that are large enough for training and testing: *notebook PC*, *drink box*,

[13]This is one of the largest RGB-D object datasets, containing about 900 objects in complex environments. It is available at http://sites.google.com/site/quanshizhang, produced mainly for the testing of graph matching.

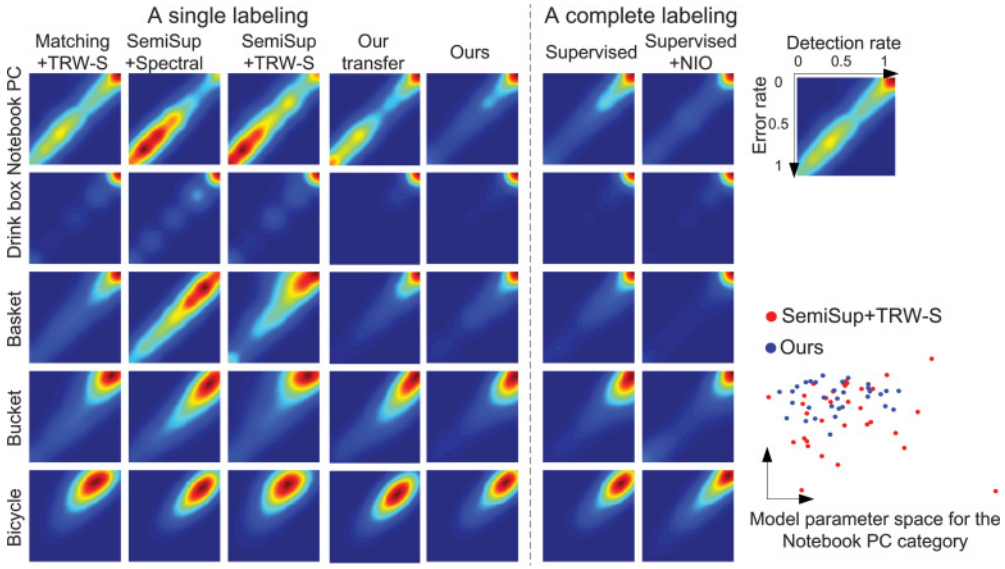

Fig. 9. Bias problems for different approaches. (Main part) Distribution of the detection (matching) rate and error rate for the learned models. In cross validation, a set of models is learned for each category by setting different initial labeling, and the object matching of each target RGB image based on any of these models produces a pair of detection and error rates (Please see Section 5.3.1 for details). The sub-figure shows the distribution of the detection and error rates. The existence of low detection rates and high error rates is mainly caused by the biased models. Note that, despite being based on a single labeling, *Ours* outperforms other semi-supervised methods, even approaches the performance of supervised methods with complete labeling. Except for the learning of *notebook PC* models, *Our transfer* does not show an obvious bias in model learning. (Bottom right) Model parameters ($\mathbf{w}$) of the *notebook PC* category projected onto a 2D space. Different points indicate the values for $\mathbf{w}$ learned from a different initial labeling. Note that the values for $\mathbf{w}$ learned by *Ours* are more convergent, whereas the outliers provided by *SemiSup+TRW-S* indicate biased models.

basket, *bucket*, and *bicycle*. These categories contain 33, 36, 36, 67, and 92 scenes, respectively, which are enough for training and testing.

5.2. Competing Methods

As discussed in Section 2, the scenario of learning from large and cluttered RGB-D or RGB scenes proposes several high-level requirements for competing methods. Generally speaking, only four styles of methods (object discovery,[14] one-shot learning, multi-image co-segmentation, and learning graph matching) can be used to learn from large and cluttered images where the target objects are not manually aligned. However, except for learning graph matching, these styles are usually hampered by large texture variations (e.g., texture variations in the *drink box* category) in the informally captured training images, because they rely heavily on the texture consistency between objects, and some related studies have even directly used "bag-of-words" models. We, therefore, limited our comparison to approaches of learning graph matching that make good use of structural information. Both supervised and semisupervised methods for learning graph matching are used as competing methods.

We use *Ours*, *Our transfer*, and *Ours+GraphRefine* to denote the model-learning method based on RGB-D images (proposed in Section 4.1), the method based on

[14]We regard semisupervised learning [Li et al. 2010] and active learning [Vijayanarasimhan and Grauman 2011] from results of image search engines as an extension of the object discovery, here.

knowledge transfer (proposed in Section 4.3), and the model learning combined with initial labeling refinement (proposed in Section 4.4), respectively. Then, we introduce five competing methods. Pure graph matching based on TRW-S [Kolmogorov 2006] (without learning) is used as our benchmark method, denoted by *Matching+TRW-S*. We then use four typical methods of semi-supervised and supervised learning of graph matching as the competing methods. The two methods based on Leordeanu and Hebert [2012] learn graph matching in an unsupervised manner, using spectral techniques [Leordeanu and Hebert 2005] and TRW-S [Kolmogorov 2006], respectively, to solve graph matching. We call these methods "semi-supervised" approaches, because they require a preprovided template graph, as in our methods. Thus, we refer to them as *SemiSup+Spectral* and *SemiSup+TRW-S*. We also apply our previous work for semi-supervised learning of graph structures from RGB images [Zhang et al. 2013b], denoted by *GraphRefine*, as a competing method for *Ours+GraphRefine*. The remaining two methods perform supervised learning of the proposed category model. *Supervised* is an extension of Leordeanu and Hebert [Leordeanu and Hebert 2012] that uses the ground truth, instead of 3D matching assignments, to guide model learning. *Supervised+NIO* chooses nonlinear inverse optimization (NIO) to learn models, as introduced by Torresani et al. [2008] and Liu et al. [2005]. We simply set $\mathbf{w} = \mathbf{1}$ for *Matching+TRW-S*, because *Matching+TRW-S* does not learn the matching weights. We transform our semi-supervised learning into supervised learning, *Supervised*. In *Supervised*, $a_{jj'}$ in Equation (15) is redefined as 1 or 0 depending on whether the matched node j' in the image is a true detection of the target object part. Finally, in *Supervised+NIO*, model parameters and attributes are trained using the NIO [Liu et al. 2005]. The NIO minimizes the compatibility gap between the true assignments and predicted assignments, as given by

$$\arg\min_{\mathbf{f}_{ij},\mathbf{w}} \sum_{k=1}^{N} \left\{ \max_{\mathbf{y}} \mathcal{C}\left(\mathbf{f}_{ij}, \mathbf{w}, \mathbf{y} | G'^{(k)}\right) - \mathcal{C}\left(\mathbf{f}_{ij}, \mathbf{w}, \mathbf{y}^{(k)}_{truth} | G'^{(k)}\right) \right\}, \tag{32}$$

where $\mathcal{C}(\cdot)$ denotes the matching compatibility in Equation (1), $G'^{(k)}$ denotes the k-th target graph for matching, and $\mathbf{y}^{(k)}_{truth}$ represents the matching ground truth of $G'^{(k)}$.

5.3. Experiment 1: Learning from a Single-Labeled Object

This experiment tests the performance of "model learning from a single-labeled object" using different methods, including *Ours*, *Our transfer*, and the other five competing methods. Note that method performance is sensitive to the initial labels; therefore, we uniformly apply the object labels used by Zhang et al. [2013a] to all the competing methods to ensure a fair comparison. All the object labeling is published in Zhang [2013].

5.3.1. Evaluation 1: Object Matching. This experiment tests object matching between a trained model and RGB images known to contain the target objects. The object matching performance is evaluated via cross-validation. For the method *Ours*, we use each RGB-D image to start a single-model learning process, thus obtaining a set of models for each category for cross-validation. The training process for each model is performed as follows. Given an RGB-D image, we use the labeled object in this image as the template graph. We then randomly select 2/3 and 1/3 of the remaining RGB-D images in this category for training and testing, respectively. Note that only the RGB channels of the RGB-D images are used for testing.

The cross-validation of the other competing methods is similar to that for *Ours*. The only difference is that we use only RGB channels in the RGB-D images for training. In particular, for the method *Our transfer*, we transfer the knowledge extracted from the

existing models in any four of the five categories to guide the model learning of the fifth category. The existing models are trained using the method *Ours*, and we use the RGB channels of the RGB-D images in the fifth category as the training images for model learning.

According to convention, we calculate the average detection (matching) rate (ADR) as a measurement of matching performance[15] [Leordeanu and Hebert 2012, 2008; Zhang et al. 2013b, 2013a]. Each detection rate is defined as $DR = N^T/\min\{N^{model}, N^{target}\}$, indicating the proportion of model nodes that are correctly matched to the target object. N^T denotes the number of nodes in the model that are matched to the target object; N^{model} and N^{target} indicate the total number of segments in the model and the target object. The ADR represents the average of individual detection rates across all matching results produced by the trained models of a category.

Given the inclusion of *none* as a matching choice, we are also concerned with the average error rate (AER). The error rate of an individual matching result is $ER = \frac{N^B}{N^{model}}$, where N^B is the number of nodes matched to the background. Note that $N^T + N^B \leq N^{model}$, as some model nodes may be matched to *none*. Similarly, AER is the average of ER.

Figure 10 and Figure 11 illustrate object matching results using the category models learned by *Ours*. Error matching cases are shown in Figure 12. Table I shows the quantitative results, and demonstrates that the performance of 3D matching from RGB-D images is good enough to guide the learning of category models. As shown in Figure 9, conventional semisupervised methods suffer greatly from accumulated bias, whereas the strategy of learning from RGB-D images and that of learning based on knowledge transfer make *Ours* and *Our transfer* outperform the other semisupervised methods. *Ours* approaches the performance of supervised methods with complete labeling, and except for the *notebook PC* category, *Our transfer* does not show an obvious bias in model learning. Note that for some categories, *Ours* exhibits a better performance than *Supervised*. This is because for *Supervised*, the labeling of ground truth only determines a subset of line segments in target scenes as target objects, whereas *Ours* uses 3D matching to provide the exact matching assignments that more fit the target model. Moreover, in Leordeanu and Hebert [2012], the regression of the prototype model is not sensitive to outliers in training samples; therefore, *Ours* performs even better than *3D matching* for the *drink-box* category.

5.3.2. Evaluation 2: Object Recognition. We combine each competing method with the three object recognition approaches proposed in Section 4.2 to compare their object recognition performance. To test each model trained with each competing method, we use the testing RGB images from Experiment 1 as the positive images. We then randomly select the same number of negative images from the image sets in the other four categories.

In the same manner as object matching, object recognition is evaluated by cross-validation. We generate a curve of the recall and error rate by setting different values of threshold υ to assess the matching-compatibility-based object recognition. All of the models trained for a category[16] are used in the cross-validation. Given a specific value of υ, each model produces a set of object-recognition results; we can obtain a high number of recognition results from all the models. We compute the curve based on all of these results to represent the overall recognition performance. Next, to assess the Wallraven and Caputo [2003], Duchenne et al. [2011], -based and Zhang et al. [2013b]

[15]Note that the matching rate in Zhang et al. [2013b] is defined as $MR = N^T/N^{model}$, which has a slight difference to the detection rate used in Zhang et al. [2013a] and this work.

[16]See Experiment 1 for the training of these models in the same category.

Fig. 10. Object matching results using the category models learned by *Ours*.

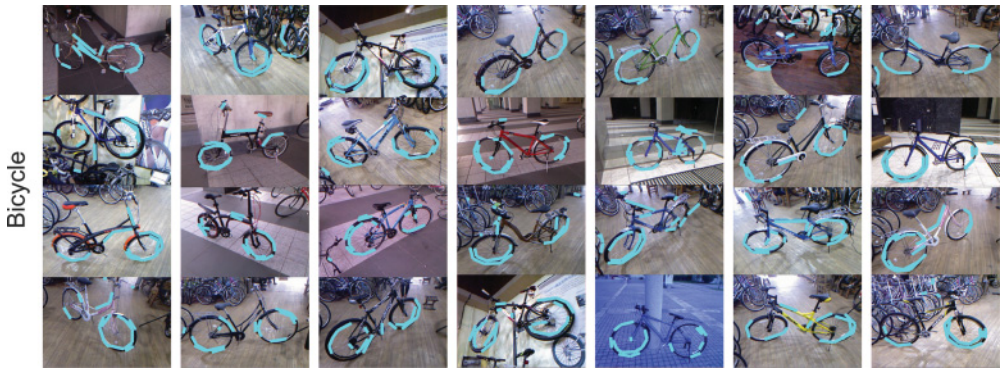

Fig. 11. Object matching results using the category models learned by *Ours*.

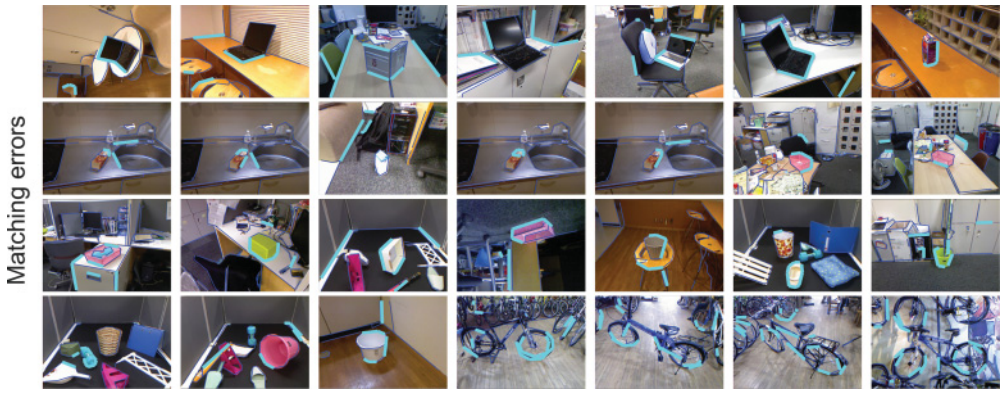

Fig. 12. Error results of object matching. Detection results are shown in cyan, and other edge segments in images are shown in dark blue.

Table I. Object-Matching Performance in Experiment 1: "Learning from a Single-Labeled Object"

Detection↑/Error↓ rate	Notebook PC	Drink Box	Basket	Bucket	Bicycle
Matching+TRW-S	56.17 / 42.82	84.84 / 14.93	74.12 / 24.67	73.43 / 22.76	67.62 / **18.31**
SemiSup+Spectral	41.89 / 58.16	78.01 / 21.99	61.69 / 39.11	74.60 / 30.17	76.28 / 23.72
SemiSup+TRW-S	43.57 / 54.43	77.95 / 20.89	62.87 / 30.83	69.47 / 22.41	61.37 / 20.40
Our transfer	51.25 / 48.80	97.97 / 2.03	87.92 / 13.45	80.38 / 24.52	72.45 / 27.55
Ours	**74.24 / 25.98**	**98.03 / 1.97**	**88.04 / 13.22**	**87.99 / 17.77**	**81.56** / 18.44
Supervised	73.13 / 27.08	**98.61 / 1.39**	87.21 / 14.15	**87.69 / 18.04**	80.98 / 19.02
Supervised+NIO	**78.11 / 22.13**	95.54 / 4.46	**92.05 / 9.42**	79.08 / 25.55	**82.68 / 17.31**
Object Matching in RGB-D Images					
3D matching	93.68 / 6.49	90.57 / 9.43	90.35 / 11.00	96.12 / 10.57	93.87 / 4.58

Note: This table lists detection and error rates. The models are trained using initial labeling of Zhang et al. [2013a]. *Ours* starts with minimal labeling, but performs nearly as well as supervised methods that require manual labeling of all training samples. In general, *Our transfer* outperforms the other semi-supervised methods.

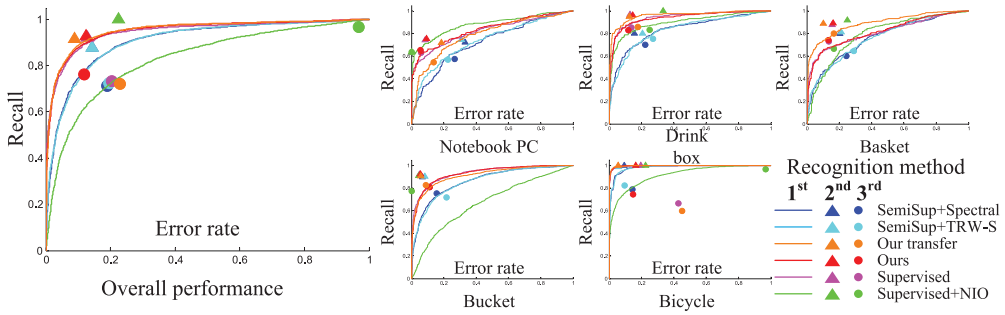

Fig. 13. Comparison of object-recognition performance. We use the curve, triangle, and circle to represent the first, second, and third methods for object recognition. These methods are used to evaluate the trained category models. The graph on the left shows the overall recognition performance based on the three recognition methods proposed in Section 4.2. The other graphs show recognition performance for different categories. Generally speaking, *Ours* and *Our transfer* approach the performance of *Supervised,* which requires complete labeling, and exhibits superior performance to other competing methods.

object-recognition methods based on Wallraven and Caputo [2003] and Zhang et al. [2013b], we compute the average recall and error rates for all of the trained models.

Object-recognition performances of competing methods are compared in Figure 13. Figure 14 shows the object-recognition performance of *Ours* for each category. Except for the *bicycle* category, *Ours* and *Our transfer* outperform the other semi-supervised methods, and even approach the performance of the *Supervised* method. For the learning of the *bicycle* models, *SemiSup+Spectral* and *SemiSup+TRW-S* are usually biased to some specific parts of the bicycle, but these parts are often more distinguishing than the entire shape of the bicycle. Therefore, *SemiSup+Spectral* and *SemiSup+TRW-S* exhibit better performance for the *bicycle* category. Based on the NIO, *Supervised+NIO* is prone to forcing "bad" parts of objects to be well matched, rather than achieving a high value of total matching compatibility. Thus, *Supervised+NIO* has poor performance for complex objects (with many edge segments), that is, *bucket* and *bicycle*.

5.4. Experiment 2: Learning from Refined Object Labeling

In this experiment, we also test the performance of "model learning from inaccurately labeled objects." We compare the proposed *Ours+GraphRefine* with the original method *Ours* and other competing methods.

We apply the "inaccurate object labeling" used by Zhang et al. [2013b], which is also published in Zhang [2013], in the experiment. We evaluate the object-matching performance of models trained using different methods. The object-matching performance is evaluated by cross-validation. The design of the cross-validation is the same as the one described in Section 5.3.1. We select each image to start a single model-learning process, and use the same distribution of the training and testing images as before. The only difference is that the initial template is not accurately labeled in the selected image as it was in Section 5.3.1.

In addition to ADR and AER, we also use the average matching rate (AMR) described by Zhang et al. [2013b] as an evaluation metric. Each matching rate is defined as $MR = N^T/N^{model}$, where N^T and N^{model} denote the model node number that matches the target object and the total model node number, respectively. In the same manner as ADR and AER, the AMR is the average of MR.

We set the proposed method *Ours+GraphRefine* to refine the initial labeled objects in the *notebook PC*, *drink box*, *basket*, and *bucket* categories to yield five, four, five, and

Fig. 14. Recognition performance for each category. Different colors indicate different categories: *notebook PC* (*cyan*), *drink box* (*blue*), *basket* (*red*), *bucket* (*yellow*), and *bicycle* (*green*). The last two rows show some error results.

Table II. Object-Matching Performance in Experiment 2: "Learning from Refined Object Labeling"

	Average (detection↑ / error↓ / matching↑) rate			
	Notebook PC	Drink box	Basket	Bucket
Matching+TRW-S	57.6 / 44.0 / 56.1	69.1 / 43.9 / 56.2	67.3 / 44.7 / 55.3	72.7 / 42.0 / 58.0
SemiSup+Spectral	47.8 / 53.6 / 46.4	61.1 / 50.5 / 49.5	61.2 / 49.8 / 50.3	69.5 / 45.3 / 54.7
SemiSup+TRW-S	50.2 / 51.2 / 48.8	63.3 / 48.8 / 51.2	62.1 / 49.0 / 51.0	71.5 / 43.6 / 56.4
GraphRefine	54.5 / 43.8 / 54.5	83.4 / 16.6 / 83.4	81.8 / 18.0 / 81.8	77.4 / 28.1 / 70.4
Ours	57.9 / 43.8 / 56.1	71.8 / 42.0 / 57.8	75.8 / 38.3 / 61.7	**78.9** / 37.4 / 62.5
Ours+GraphRefine	**68.6 / 31.2 / 68.6**	**89.0 / 11.0 / 89.0**	**86.3 / 13.7 / 86.3**	78.6 / **28.4 / 71.6**
Supervised	54.5 / 47.1 / 52.9	65.0 / 47.3 / 52.7	62.9 / 48.3 / 51.7	72.0 / 43.2 / 56.8

Note: This table lists detection, error, and matching rates. Category models are trained using initial labeling of Zhang et al. [2013b].

nine parts (nodes), respectively. These settings are also applied to *GraphRefine*, which learns the graph structure from RGB images, ensuring a fair comparison.

Table II provides the quantitative comparisons, demonstrating the superior performance of the proposed *Ours+GraphRefine*. Generally speaking, compared to results in Table I, the results in Table II show higher error rates and lower detection rates, due to

the inaccurate manual labeling. Note that in Table II, the *bucket* category shows lower detection rates for *Matching+TRW-S* and *SemiSup+TRW-S*, they also exhibit significantly lower error rates. Only for the *Notebook PC* category do we see *Matching+TRW-S*, *SemiSup+Spectral*, and *SemiSup+TRW-S* exhibiting better performance in Table II than in Table I. However, because *Matching+TRW-S* directly matches initially labeled templates without training, its superior performance for the *Notebook PC* category in Table II demonstrates that compared to subjectively well-labeled notebook PCs in Zhang et al. [2013a], subjectively inaccurate labeling of PCs providedZhan by Zhang et al. [2013b] can actually produce a better fit to the target objects in cluttered images in some cases.

Then, we limit our comparison to the results given in Table II. Compared to *Ours*, *Ours+GraphRefine* clearly benefits from the Zhang et al. [2013b] technique, exhibiting more robustness under inaccurate manual labeling.

6. CONCLUSIONS

In this study, we developed a method for category model learning that proceeds from a single-labeled object to a number of informally captured RGB-D images. We used the RGB-D information to guide the training of the category model, which was applied to ordinary RGB images. The trained category models were also used to provide knowledge to guide the model training for new categories, if users could only obtain RGB images in these categories. All of these techniques were designed to facilitate efficient model learning. The minimization of labeling saves considerable human labor during the construction of the model base. Both the depth information obtained from RGB-D images and the knowledge transferred from other category models can guide the learning framework to overcome the problem of bias. The effectiveness of the proposed method has been demonstrated in various experiments.

In this study, we used the trained model only to match a single object in an image for testing. However, graph matching can easily be extended to matching multiple objects in the same image in real applications. We can use the model directly to match a new object if we remove all the parts (nodes) of the previously matched object from the target image (graph). Recognizing that most objects in daily use have standard and recognizable shapes, but may have a variety of tones and textures, we designed our category model to focus on structural information, namely, object edge segments. This design makes the model robust across texture variations. However, it is difficult for our category model to describe largely occluded objects or objects with highly deformable or irregular shapes, such as natural scenes and animals. Therefore, we need to develop more kinds of graphical models with new local and pairwise attributes to represent these irregular and deformable objects in future work.

REFERENCES

A. Aldoma, F. Tombari, L. D. Stefano, and M. Vincze. 2012. A global hypotheses verification method for 3D object recognition. In *12th European Conference on Computer Vision (ECCV)*. 511–524.

P. Arbelaez, M. Maire, C. Fowlkes, and J. Malik. 2011. Contour detection and hierarchical image segmentation. In *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)* 33, 5, 898–916.

G. Bradski and T. Hong. 2011. NIST and willow garage: Solution in perception challenge. Retrieved from http://www.willowgarage.com/blog/2011/02/28/nist-and-willow-garage-solutions-perception-challenge.

B. Browatzki, J. Fischer, G. Birgit, H. Bulthoff, and C.Wallraven. 2011. Going into depth: Evaluating 2d and 3D cues for object classification on a new, large-scale object dataset. In *IEEE International Conference on Computer Vision Workshop (ICCV Workshops)*. 1189–1195.

C. K. Liu, A. Hertzmann, and Z. Popović. 2005. Learning physics-based motion style with nonlinear inverse optimization. In *ACM Transactions on Graphics (TOG)—Proceedings of ACM SIGGRAPH* 24, 3 (2005), 1071–1081.

T. S. Caetano, J. J. McAuley, L. Cheng, Q. V. Le, and A. J. Smola. 2009. Learning graph matching. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)* 31, 6, 1048–1058.

H.-Y. Chen, Y.-Y. Lin, and B.-Y. Chen. 2013. Robust feature matching with alternate Hough and inverted Hough transforms. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2762–2769.

W.-C. Chiu and M. Fritz. 2013. Multi-class video co-segmentation with a generative multi-video model. In *2013 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 321–328.

M. Cho, K. Alahari, and J. Ponce. 2013. Learning graphs to match. In *International Conference on Computer Vision (ICCV)*.

M. Cho and K. M. Lee. 2012. Progressive graph matching: Making a move of graphs via probabilistic voting. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 398–405.

M. Cho, Y. M. Shin, and K. M. Lee. 2010. Unsupervised detection and segmentation of identical objects. In *International Conference on Computer Vision and Pattern Recognition (CVPR)*. 1617–1624.

A. Collet, S. S. Srinivasay, and M. Hebert. 2011. Structure discovery in multi-modal data: A region-based approach. *In ICRA*.

N. Dalal and B. Triggs. 2005. Histograms of oriented gradients for human detection. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 886–893.

O. Duchenne, A. Joulin, and J. Ponce. 2011. A graph-matching kernel for object categorization. In *IEEE International Conference on Computer Vision (ICCV)*. 1792–1799.

A. Faktor and M. Irani. 2012. "Clustering by Composition"–Unsupervised discovery of image categories. In *12th European Conference on Computer Vision (ECCV)*. 474–487.

V. Ferrari, F. Jurie, C. Schmid. 2010. From images to shape models for object detection. In *International Journal on Computer Vision (IJCV),* 87, 3, 284–303.

D. F. Fouhey, A. Gupta, and M. Hebert. 2013. Data-driven 3D primitives for single image understanding. In *International Conference on Computer Vision (ICCV)*.

S. Helmer, D. Meger, M. Muja, J. J. Little, and D. G. Lowe. 2010. UBC robot vision survey. Retrieved February 13, 2015 from http://www.cs.ubc.ca/labs/lci/vrs/index.html.

P. Hong and T. S. Huang. 2004. Spatial pattern discovery by learning a probabilistic parametric model from multiple attributed relational graphs. *Discrete Applied Mathematics* 139, 113–135.

E. Hsiao, A. Collet, and M. Hebert. 2010. Making specific features less discriminative to improve point-based 3D object recognition. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2653–2660.

N. Hu, R. M. Rustamov, and L. Guibas. 2013. Graph matching with anchor nodes: A learning approach. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2906–2913.

W. Hu. 2012. Learning 3D object templates by hierarchical quantization of geometry and appearance spaces. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2336–2343.

A. Janoch. 2012. The Berkeley 3D Object Dataset. Retrieved February 13, 2015 from http://www.eecs.berkeley.edu/Pubs/TechRpts/2012/EECS-2012-85.html. Master's thesis. EECS Department, University of California, Berkeley.

H. Jiang and C.-W. Ngo. 2003. Image mining using inexact maximal common subgraph of multiple ARGs. In *International Conference on Visual Information System*. 63–76.

A. Joulin, F. Bach, and J. Ponce. 2012. Multi-class cosegmentation. In *International Conference on Computer Vision and Pattern Recognition (CVPR)*. 542–549.

H. Kang, M. Hebert, and T. Kanade. 2011. Discovering object instances from scenes of daily living. In *13th International Conference on Computer Vision (ICCV)*. 762–769.

G. Kim, C. Faloutsos, and M. Hebert. 2008. Unsupervised modeling of object categories using link analysis techniques. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 1–8.

G. Kim and E. P. Xing. 2012. On multiple foreground cosegmentation. In *International Conference on Computer Vision and Pattern Recognition (CVPR)*. 837–844.

G. Kim, E. P. Xing, L. Fei-Fei, and T. Kanade. 2011. Distributed cosegmentation via submodular optimization on anisotropic diffusion. In *IEEE International Conference on Computer Vision (ICCV)*. 169–176.

K. I. Kim, J. Tompkin, M. Theobald, J. Kautz, and C. Theobalt. 2012. Match graph construction for large image databases. In *12th European Conference on Computer Vision (ECCV)*. 272–285.

V. Kolmogorov. 2006. Convergent tree-reweighted message passing for energy minimization. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)* 28, 10, 1568–1583.

H. S. Koppula, A. Anand, T. Joachims, and A. Saxena. 2011. Semantic labeling of 3D point clouds for indoor scenes. In *Neural Information Processing Systems (NIPS)*. 244–252.

K. Lai, L. Bo, X. Ren, and D. Fox. 2011a. A large-scale hierarchical multi-view RGB-D object dataset. In *IEEE International Conference on Robotics and Automation (ICRA)*. 1817–1824.

K. Lai, L. Bo, X. Ren, and D. Fox. 2011b. Sparse distance learning for object recognition combining RGB and depth information. In *IEEE International Conference on Robotics and Automation (ICRA)*. 4007–4013.

K. Lai and D. Fox. 2010. Object recognition in 3D point clouds using web data and domain adaptation. *Proceedings of International Journal of Robotic Research* 29, 8, 1019–1037.

Q. Le, M. Ranzato, R. Monga, M. Devin, K. Chen, G. Corrado, J. Dean, and A. Ng. 2012. Building high-level features using large scale unsupervised learning. In *ICML*.

Y. J. Lee and K. Grauman. 2011. Learning the easy things first: Self-paced visual category discovery. In *International Conference on Computer Vision and Pattern Recognition (CVPR)*. 1721–1728.

M. Leordeanu and M. Hebert. 2005. A spectral technique for correspondence problems using pairwise constraints. In *10th International Conference on Computer Vision (ICCV)*. 1482–1489.

M. Leordeanu and M. Hebert. 2008. Smoothing-based optimization. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 1–8.

M. Leordeanu and M. Hebert. 2012. Unsupervised learning for graph matching. *International Journal of Computer Vision* 96, 1, 28–45.

M. Leordeanu, M. Hebert, and R. Sukthankar. 2007. Beyond local appearance: Category recognition from pairwise interactions of simple features. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 1–8.

C. Li, D. Parikh, and T. Chen. 2012. Automatic discovery of groups of objects for scene understanding. *International Conference on Computer Vision and Pattern Recognition (CVPR)*. 2735–2742.

F.-F. Li, R. Fergus, and P. Perona. 2006. One-shot learning of object categories. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)* 28, 4, 594–611.

L.-J. Li, G. Wang, and F.-F. Li. 2010. OPTIMOL: Automatic online picture collection via incremental model learning. *International Journal on Computer Vision (IJCV)*, 88, 2, 147–154.

Z. Liao, A. Farhadi, Y. Wang, I. Endres, and D. Forsyth. 2012. Building a dictionary of image fragments. In *International Conference on Computer Vision and Pattern Recognition (CVPR)*. 3442–3449.

C.-J. Lin and R. C. Weng. 2004. Simple probabilistic predictions for support vector regression. In *Technical report Department of Computer Science*, National Taiwan University, Taiwan.

H. Liu and S. Yan. 2010. Common visual pattern discovery via spatially coherent correspondences. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 1609–1616.

H. Liu and S. Yan. 2012. Efficient structure detection via random consensus graph. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 574–581.

K. Liu, Q. Wang, W. Driever, and O. Ronneberger. 2012. 2D/3D rotation-invariant detection using equivariant filters and kernelweighted mapping. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 917–924.

S. Maji and J. Malik. 2009. Object detection using a max-margin Hough transform. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 1038–1045.

Microsoft. 2011. Introducing Kinect for Xbox 360.

L. Mukherjee, V. Singh, J. Xu, and M.D. Collins. 2012. Analyzing the subspace structure of related images: Concurrent segmentation of image sets. In *12th European Conference on Computer Vision (ECCV)*. 128–142.

C. Olsson and Y. Boykov. 2012. Curvature-based regularization for surface approximation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 1576–1583.

B. Pepik, P. Gehler, M. Stark, and B. Schiele. 2012. 3D2PM—3D deformable part models. In *12th European Conference on Computer Vision (ECCV)*. 356–370.

N. Razavi, J. Gall, P. Kohli, and L. v. Gool. 2012. Latent Hough transform for object detection. In *12th European Conference on Computer Vision (ECCV)*. 312–325.

X. Ren, L. Bo, and D. Fox. 2012. RGB-(D) scene labeling: Features and algorithms. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2759–2766.

N. Silberman and R. Fergus. 2011. Indoor scene segmentation using a structured light sensor. In *IEEE International Conference on Computer Vision Workshop (ICCV Workshops)*. 601–608.

N. Silberman, D. Hoiem, P. Kohli, and R. Fergus. 2012. Indoor segmentation and support inference from RGBD images. In *12th European Conference on Computer Vision (ECCV)*. 746–760.

M. Sun, G. Bradski, B.-X. Xu, and S. Savarese. 2010. Depth-encoded Hough voting for joint object detection and shape recovery. In *11th European Conference on Computer Vision (ECCV)*. 658–671.

W. Susanto, M. Rohrbach, and B. Schiele. 2012. 3D object detection with multiple kinects. In *12th European Conference on Computer Vision (ECCV)*. 93–102.

H.-K. Tan and C.-W. Ngo. 2009. Localized matching using Earth Movers Distance towards discovery of common patterns from small image samples. *Image and Vision Computing* 27, 1470–1483.

L. Torresani, V. Kolmogorov, and C. Rother. 2008. Feature correspondence via graph matching: Models and global optimization. In *10th European Conference on Computer Vision (ECCV).* 596–609.

T. Tuytelaars, C. H. Lampert, M. B. Blaschko, and W. Buntine. 2010. Unsupervised object discovery: A comparison. *International Journal on Computer Vision* 88, 2, 284–302.

S. Vijayanarasimhan and K. Grauman. 2011. Large-scale live active learning: Training object detectors with crawled data and crowds. In *International Conference on Computer Vision and Pattern Recognition (CVPR).* 1449–1456.

C. Wallraven and B. Caputo. 2003. Recognition with local features: The kernel recipe. In *9th International Conference on Computer Vision (ICCV).* 257–264.

T. Wang, X. He, and N. Barnes. 2013. Learning structured Hough voting for joint object detection and occlusion reasoning. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR).* 1790–1797.

X. Wang, X. Bai, T. Ma, W. Liu, and L. J. Latecki. 2012. Fan shape model for object detection. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR).* 151–158.

W. Wohlkinger, A. Aldoma, R. B. Rusu, and M. Vincze. 2012. Large-scale object class recognition from CAD models. In *IEEE International Conference on Robotics and Automation (ICRA).* 5384–5391.

H. Xie, K. Gao, Y. Zhang, J. Li, and H. Ren. 2012. Common visual pattern discovery via graph matching. *ACM Multimedia* 1385–1388.

Y. Xu, Y. Quan, Z. Zhang, and H. Ji. 2012. Contour-based recognition. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR).* 3402–3409.

J. Yuan, G. Zhao, Y. Fu, Z. Li, A. K. Katsaggelos, and Y. Wu. 2012. Discovering thematic objects in image collections and videos. In *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)* 21, 4, 2207–2219.

Q. Zhang. 2013. Category Dataset of Kinect RGBD Images.

Q. Zhang, X. Song, X. Shao, R. Shibasaki, and H. Zhao. 2013a. Category modeling from just a single labeling: Use depth information to guide the learning of 2D models. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR).* 193–200.

Q. Zhang, X. Song, X. Shao, H. Zhao, and R. Shibasaki. 2013b. Learning graph matching for category modeling from cluttered scenes. In *IEEE International Conference on Computer Vision (ICCV).*

Q. Zhang, X. Song, X. Shao, H. Zhao, and R. Shibasaki. 2013c. Unsupervised 3D category discovery and point labeling from a large urban environment. In *Proceeding of the IEEE International Conference on Robotics and Automation (ICRA).*

Q. Zhang, X. Song, X. Shao, H. Zhao, and R. Shibasaki. 2014a. Attributed graph mining and matching: An attempt to define and extract soft attributed patterns. In *International Conference on Computer Vision and Pattern Recognition (CVPR).*

Q. Zhang, X. Song, X. Shao, H. Zhao, and R. Shibasaki. 2014b. When 3D reconstruction meets ubiquitous RGB-D images. In *International Conference on Computer Vision and Pattern Recognition (CVPR).*

Q. Zhang, X. Song, X. Shao, H. Zhao, and R. Shibasaki. 2014c. Start from minimum labeling: Learning of 3D object models and point labeling from a large and complex environment. In *Proceeding of the IEEE International Conference on Robotics and Automation (ICRA).*

G. Zhao and J. Yuan. 2010. Mining and cropping common objects from images. *ACM Multimedia* 975–978.

F. Zhou and F. de la Torre. 2013. Deformable graph matching. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR),* 2922–2929.

J.-Y. Zhu, J. Wu, Y. Wei, E. Chang, and Z. Tu. 2012. Unsupervised object class discovery via saliency-guided multiple class learning. In *International Conference on Computer Vision and Pattern Recognition (CVPR).* 3218–3225.

Received February 2014; revised April 2014; accepted May 2014