Int J Comput Vis (2015) 111:69–97
DOI 10.1007/s11263-014-0734-4

3DNN: 3D Nearest Neighbor

Data-Driven Geometric Scene Understanding Using 3D Models

Scott Satkin · Maheen Rashid · Jason Lin · Martial Hebert

Received: 11 November 2013 / Accepted: 23 May 2014 / Published online: 22 July 2014

Abstract In this paper, we describe a data-driven approach to leverage repositories of 3D models for scene understanding. Our ability to relate what we see in an image to a large collection of 3D models allows us to transfer information from these models, creating a rich understanding of the scene. We develop a framework for auto-calibrating a camera, rendering 3D models from the viewpoint an image was taken, and computing a similarity measure between each 3D model and an input image. We demonstrate this data-driven approach in the context of geometry estimation and show the ability to find the identities, poses and styles of objects in a scene. The true benefit of 3DNN compared to a traditional 2D nearest-neighbor approach is that by generalizing across viewpoints, we free ourselves from the need to have training examples captured from all possible viewpoints. Thus, we are able to achieve comparable results using orders of magnitude less data, and recognize objects from never-before-seen viewpoints. In this work, we describe the 3DNN algorithm and rigorously evaluate its performance for the tasks of geometry estimation and object detection/segmentation, as well as two novel applications: affordance estimation and photorealistic object insertion.

Keywords Computer vision · Machine learning · Scene understanding · Geometry estimation · 3D data

Communicated by Cordelia Schmid.

S. Satkin (✉)
Google Inc., Mountain View, CA, USA
e-mail: satkin@google.com

M. Rashid · M. Hebert
Carnegie Mellon University, Pittsburgh, PA, USA
e-mail: maheenr@andrew.cmu.edu

M. Hebert
e-mail: hebert@ri.cmu.edu

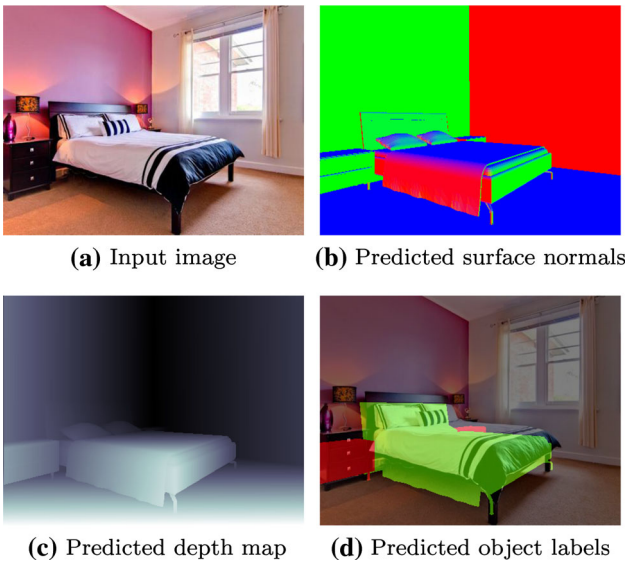
J. Lin
Microsoft Corp., Redmond, WA, USA
e-mail: jasonlin@alumni.cmu.edu

1 Introduction

This work explores the intersection of geometric reasoning and machine learning for scene understanding. Our objective is to produce a rich representation of the world from a single image by relating what we see in the image with vast repositories of 3D models, as shown in Fig. 1. By matching and aligning an image with 3D data, we can produce detailed reconstructions of scenes and transfer rich information from the models to answer a wide variety of queries. Our work builds upon recent advances in data-driven scene matching and single-view geometry estimation, which we now summarize.

1.1 Data-Driven Approaches in Computer Vision

Over the past decade, researchers have demonstrated the effectiveness of data-driven approaches for complex computer vision tasks. Large datasets such as Torralba et al. (2008)'s 80 Million Tiny Images and Deng et al. (2009)'s ImageNet have proven to be invaluable sources of information for tasks like scene recognition and object classification. Simple nearest-neighbor approaches for matching an input image (or patches of an image) with a large corpus of annotated images enables the "transfer" of information from one image to another. These non-parametric approaches have been shown to achieve amazing performance for a wide variety of complex computer vision and graphics tasks ranging



(a) Input image **(b)** Predicted surface normals

(c) Predicted depth map **(d)** Predicted object labels

Fig. 1 From a single image, we estimate detailed scene geometry and object labels

from semantic labelling (Sing and Košecká 2013; Tighe and Lazebnik 2010) and scene categorization (Oliva and Torralba 2001), to motion synthesis (Liu et al. 2008) and even image localization (Hays and Efros 2008).

Recently, large online repositories of 3D data such as Trimble 3D Warehouse (Trimble Inc. 2012) (formerly Google 3D Warehouse) have emerged. These resources, as well as the advent of low-cost depth cameras such as the Kinect (Microsoft Corporation 2010), have sparked interest in geometric data-driven algorithms. At the same time, researchers have (re-)started investigating the feasibility of recovering geometric information, for example, the layout of a scene (Bao et al. 2010; Hoiem et al. 2007; Saxena et al. 2009). The success of data-driven techniques for tasks based on appearance features, for example, interpreting an input image by retrieving similar scenes (Hays and Efros 2007; Torralba et al. 2008; Liu et al. 2008), suggests that similar techniques based on *geometric* data could be equally effective for 3D scene interpretation tasks. In fact, the motivation for data-driven techniques is the same for 3D models as for images: Real-world environments are not random; the sizes, shapes, orientations, locations and co-location of objects are constrained in complicated ways that can be represented given enough data. In principle, estimating 3D scene structure from data would help constrain bottom-up vision processes. For example, in Fig. 1, one nightstand is fully visible; however, the second nightstand is almost fully occluded. Although a bottom-up detector would likely fail to identify the second nightstand since only a few pixels are visible, our method of finding the best matching 3D model is able to detect these types of occluded objects. This is not a trivial extension of the image-based techniques. Generalizing data-driven ideas raises new fundamental technical questions never addressed before in this context: What features should be used to compare input images and 3D models? Given these features, what mechanism should be used to rank the most similar 3D models to the input scene? Even assuming that this ranking is correct, how can we transfer information from the 3D models to the input image?

To address these questions, we develop a set of features that can be used to compare an input image with a 3D model and design a mechanism for finding the best matching 3D scene using support vector ranking. We show the feasibility of these techniques for transferring the geometry of objects in indoor scenes from 3D models to an input image.

The graphics community has begun harvesting data from online repositories such as 3D Warehouse in an effort to better understand and model how objects are typically arranged in homes (Fisher and Hanrahan 2010; Fisher et al. 2011). Additionally, the vision community has begun utilizing this data to learn about the sizes, shapes and affordances of objects (Grabner et al. 2011; Zhao and Zhu 2013). There has also been work using this data for 3D to 3D matching with laser scans to aid in classification (Lai and Fox 2009). However, our work is one of the first to combine this geometric prior with image features in a framework capable of producing detailed 3D models from an image. Of course, this is not entirely new, the idea of relating 3D models to 2D projections was a foundation of earlier vision approaches (Brooks 1981; Lowe 1987; Grimson et al. 1990). In the area of outdoor scene understanding, prior work (Baboud et al. 2011; Baatz et al. 2012; Ramalingam et al. 2010) investigated matching images with models of terrain or cities. This body of work aims to localize images, relying mostly on matching features such as skylines which are specific to these scenarios, and do not transfer to other environments. The major difference here is our use of vast repositories of 3D data, which require novel vision and learning approaches.

This work is an important first step towards 3D data-driven techniques, which will contribute to addressing two major problems in image understanding. First, as most geometric scene understanding systems rely implicitly on sifting through a collection of hypotheses (iterative refinement (Hoiem et al. 2008), sampling (Pero et al. 2011), explicit search (Gupta et al. 2010), structured prediction (Hedau et al. 2009; Lee et al. 2010), matching and ranking mechanisms such as the ones we propose provide a data-driven way to *generate multiple hypotheses* which can be used as seeds for further processing. Second, 3D data offers potentially richer information for transfer. In this paper, we show that using 3D information for scene understanding enables us predict not only object type and location, but also viewpoint and even occlusions from other objects.

1.2 Single-View Geometry Estimation

For decades, vision researchers have strived to create high-quality 3D models of indoor scenes. Traditional approaches rely on having images taken from multiple viewpoints in order to recover the depth of each pixel using triangulation (Longuet-Higgins 1981). However, in recent years, the vision community has begun to focus on recovering the geometry of a scene from a single image (Hoiem et al. 2007; Lee et al. 2009; Saxena et al. 2009).

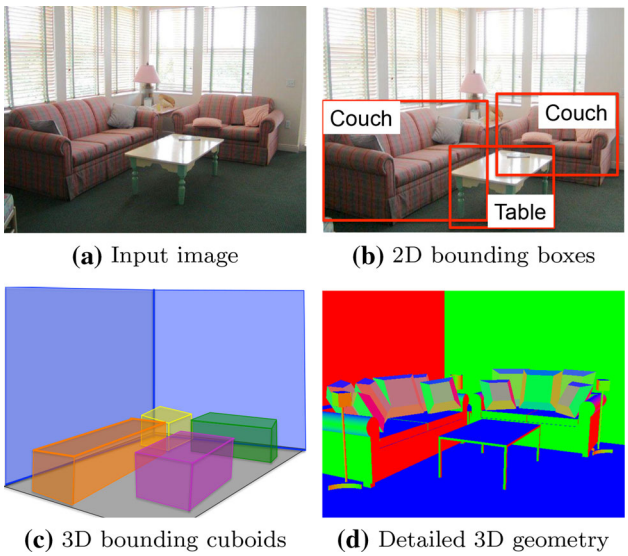
This is an inherently ill-posed problem—there exists an infinite number of 3D models which project to the same image. Despite the inherent mathematical ambiguity, humans excel at this task. When shown an image of an environment, we are not overwhelmed with an infinite number possible 3D models. On the contrary, we can quickly associate objects in images with objects we have seen before, to reason about the structure of the scene.

We are capable of this type of reasoning because the environments we live in are not completely random. The sizes, shapes, orientations, locations and co-location of objects are all dictated by the activities which the environment was designed to afford. For example, there are manufacturing standards for sizes of beds, couches, tables, etc. Moreover, when we place these objects in our homes, we tend to place them in specific locations relative to each other (e.g., nightstands adjacent to beds, coffee tables ∼2ft in front of couches). When confronted with the ill-posed problem of recovering the geometry of scene from a single image, we must exploit this statistical prior and only consider 3D models which contain reasonable objects, in reasonable arrangements.

Recently, tremendous progress has been made towards the task of estimating the geometry of a scene from a single image. The groundbreaking work of Hoiem et al. (2007) and Saxena et al. (2009), showed that machine learning can be be used to tackle this tremendously challenging task. The authors of these papers demonstrate that classifiers can be trained to predict the orientation and identity of image patches from outdoor scenes, which can be used to infer the 3D structure of the environment.

For indoor imagery, Yu et al. (2008) and Lee et al. (2009) showed that imposing a Manhattan-world constraint enables the robust detection of vanishing points allowing cameras to be autocalibrated from a single image. The authors use their model to detect planes and infer depth ordering to estimate the locations of the walls, floors and ceilings.

A fundamental problem when estimating the locations of walls in an indoor environment is clutter. Quite frequently, furniture or other objects will occlude the boundary between walls and where the walls meet the floor. Hedau et al. (2009) train a classifier to predict which pixels are the result of clutter, and which pixels correspond to the walls and floor of a room. Wang et al. (2010) also predict which pixels correspond to occluding objects; however, their technique uses latent variables and avoids the need for labeled training data. Both of these approaches show that by identifying the locations of clutter in a scene, the room layout can be more accurately estimated.



(a) Input image **(b)** 2D bounding boxes

(c) 3D bounding cuboids **(d)** Detailed 3D geometry

Fig. 2 Comparison of scene representations. In order of increasing geometric detail: traditional 2D bounding boxes (**b**), 3D bounding cuboids (**c**), our detailed 3D scene geometry (**d**)

More recently, researchers have begun analyzing the detected clutter in a scene and trying to model it with 3D bounding cuboids. This representation, depicted in Fig. 2c offers more information than traditional 2D bounding boxes (Fig. 2b), which only localize objects in the image plane. Cuboids can be used to reason about a scene in ways which cannot be inferred from 2D bounding boxes, such as estimating the freespace of an environment or analyzing where the supporting surfaces of objects are. Lee et al. (2010) combine the geometric context used in Hedau et al. (2009) with an orientation map to fit a parametric model to objects in a room with the goal of improving the estimates of wall locations. Pero et al. (2011) use 3D bounding cuboids fit to objects as part of a Markov Chain Monte Carlo framework to optimize over both the locations of the walls as well as camera pitch, roll and focal length. Both of these works show that modeling the clutter in a room improves the accuracy of room layout estimation; however, the authors do not evaluate how well their cuboids match the geometry of the objects in the scene.

In Hedau et al. (2010), the authors build upon their previous work by incorporating a cuboid detector capable of accurately detecting beds. Their algorithm searches for gradients in images which have been rectified to estimated Manhattan-world axes, and tries to align cuboids of fixed sizes (corresponding to common beds). Unlike previous work which aims only to recover the locations of walls and floors in

images, this work strives to detect and align objects in scenes and evaluates their results using typical object detection metrics.

Rather than representing objects and freespace with coarsely voxelized occupancy grids or bounding cuboids, we aim to produce high-quality detailed polygonal meshes of objects, as shown in Fig. 2d. We build upon and use the room layout estimates of Hedau et al. (2009), and mine through a database of 3D models to discover the identity, locations and orientations of objects from a single image. New work such as Choi et al. (2013), Hedau et al. (2012), Pero et al. (2012), Zhao and Zhu (2013) also aims to recover freespace by localizing cuboids representing object categories and sizes using parametric models as their prior. In contrast, we recover more detailed object geometries, similar to the recent approach presented in Lim et al. (2013), and we use non-parametric priors that can capture complex interactions between objects.

For each object in a scene, we aim to not only recover its exact location, orientation and dimensions in 3D (which can be modeled with cuboids), but also a detailed polygonal model of the object. In addition, we aim to recover the intrinsic (focal length and principal point) and extrinsic (position and rotation relative to corner of the room) parameters of the camera which captured the image. In this work, we show that recovering the detailed geometry of a scene and the corresponding camera parameters offers a complete representation of the world, which can be used to answer a wide variety of questions. For example, using the estimated camera parameteres, we can project the polygons of each object onto the image plane to produce a segmentation mask (Fig. 1d). We can compute the distance from the camera to each object to produce depth maps and reason about occlusions and depth ordering (Fig. 1c).

Although there exist many sensors which are designed to capture 2.5D representations of the world, such as laser scanners and RGBD cameras, these modalities capture only what is visible from a single viewpoint. On the contrary, because we have a *full 3D* representation of the scene, we can reason about portions of the environment which are not visible to the camera. For example, in the bedroom scene in Fig. 1, only a small portion of the nightstand in the corner of the room is visible, and the strip of floor between the bed and the wall is fully occluded. A 2.5D sensor will have no knowledge of these portions of the environment; however, our full 3D representation of the scene includes the geometry of these regions. This information cannot be measured directly, and must be inferred using prior knowledge of the world.

This brief summary of previous work shows how vibrant this research area is and how much progress has been made in a short time. The work presented here builds upon the authors' prior work (Satkin and Hebert 2013; Satkin et al. 2012) with the addition and analysis of new features for matching images to 3D models, and a geometry refinement stage which swaps hypothesized 3D objects with others from a library to produce instance-level matches. We also demonstrate new applications of 3DNN: object recognition, affordance estimation and geometry-aware object insertion. The data-driven techniques that we propose here should not be viewed as a substitute to any of the above approaches. Perhaps the most exciting aspect of our approach is that it can be used to augment *any* of these scene interpretation approaches: upstream, by providing a data-driven way to generate hypotheses; and downstream by providing richer mechanisms for information transfer. We show this by building upon the work of Hedau et al. (2009) and by demonstrating how prior 3D models can be integrated with this existing approach for room layout estimation to help discover the identity, locations and orientations of objects from a single image.

2 Scene Understanding via 3D Model Matching

We now describe our framework for comparing 3D models to monocular images. Naturally, we cannot compare 3D models directly to a 2D image. Thus, we first estimate the intrinsic and extrinsic parameters of the camera and use this information to render each of the 3D models from the same view as the image was taken from. We then compute similarity features between the models and the input image. Lastly, each of the 3D models is ranked based on how similar its rendering is to the input image using a learned feature weighting. See Fig. 3 for an overview of this process.

2.1 Autocalibration and Room Layout Estimation

Our algorithm for recovering the geometry of a scene is an *analysis via synthesis* approach. We render 3D models from the viewpoint from which an image was captured and compare these renderings to the input image. This requires we first recover the parameters of the camera used to capture the image via auto-calibration and room layout estimation.

We begin auto-calibrating the camera by estimating vanishing points using the approach of Lee et al. (2009). The vanishing points are also used to estimate the orientation of the camera with respect to the three Manhattan-world axes in our scene. Next, we run the pre-trained room layout estimation algorithm of Hedau et al. (2009) to determine the locations of the walls and the floor in the image, and use priors on camera height and room size to solve for the position of the camera. To resolve the scale ambiguity, we first fix the height of the camera to 5ft, and solve for the distance from the camera to the visible corners of the room. If the computed room size is too large or small (e.g., height $\geq$12ft or height $\leq$8ft), we raise or lower the height of the camera in 0.5ft increments and re-solve for the room size until the scale is

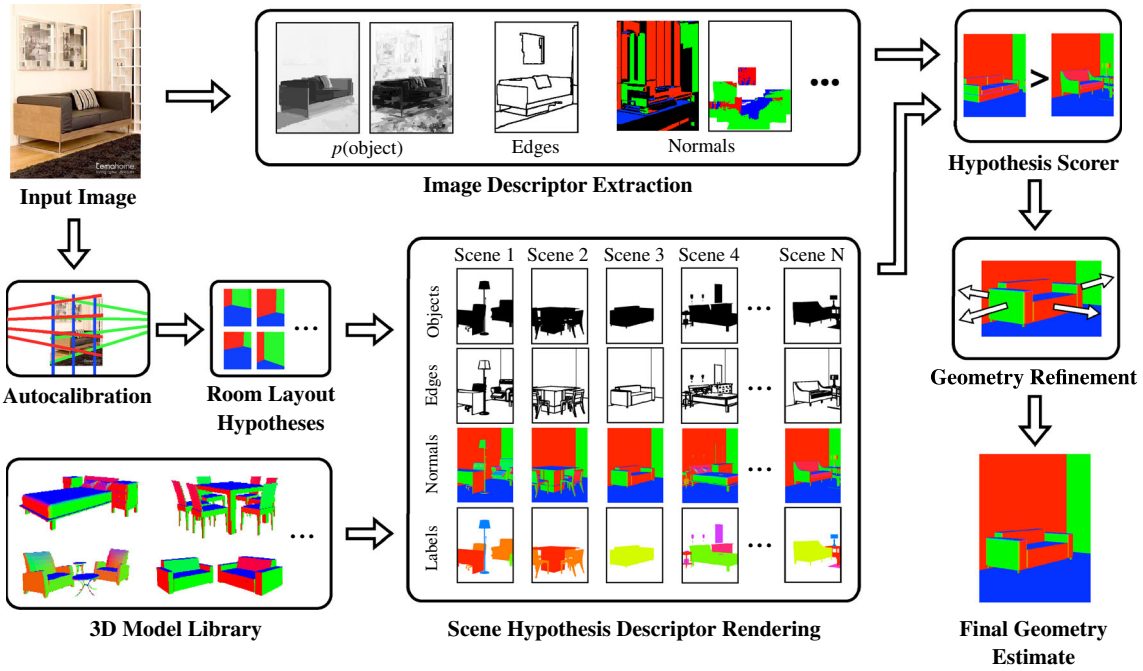


Fig. 3 Overview of our approach for matching a 3D model with a monocular image

within range. When rendering 3D models, we align the walls of the models with the estimated wall locations relative to the camera, and incorporate our calibrated camera parameters (focal length and principal point) with a viewing frustum to create renderings which align with our input image.

This setup allows us to project objects from our 3D model library into the image plane in a manner which is consistent with the estimated camera parameters. We use this renderer as a fundamental tool in computing similarity features from each 3D model. The following section details this process.

2.2 Similarity Features

An important question we address in this paper is, "What features are useful for matching 3D scenes with monocular images?" This issue is fundamentally complicated by the fact that we need to compare two objects of a completely different nature: an array of intensity/color pixels on the one hand, and a set of surfaces with no appearance information on the other hand.

To overcome this challenge, we introduce the concept of *similarity features*. Unlike traditional features which are extracted from a single image, similarity features involve comparing an image with a 3D model to describe how similar the model is to the input image. Our goal here is to rank each 3D model j, with respect to image i using similarity features denoted by x_j^i. x_j^i is a vector in which each entry corresponds to a different measure of similarity between the image i and the 3D model j. We use our renderer to produce synthetic image descriptors for each 3D model, and compare these to traditional image-based descriptors to compute each similarity feature. Figure 4 includes example image descriptors and their rendered counterparts used to compute each similarity feature. Note that these are not photorealistic renderings, we are simply rendering descriptors of each 3D model. This section introduces our preliminary set of similarity features for relating 2D images with 3D models.

***p*(object) masks:** For each 3D model, we render a simple object mask (i.e., each polygon in the model is rendered black on a white background) as shown in Fig. 4a. This rendered object mask is compared with p(object) image descriptors which predict the locations of objects in an image.

We use the pre-trained Indoor Geometric Context model of Hedau et al. (2009) and Hoiem et al. (2007) to estimate p(object) masks (see Fig. 4b). We also train a probabilistic classifier using the algorithm of Munoz et al. (2010) as a second predictor of object locations in an image.[1] This classifier was trained with more data than the Geometric Context model and is more robust to the diversity of object colors, textures, and illumination conditions seen in the SUN database (Xiao et al. 2010).

After scaling each of these masks to be in the range $[-1, 1]$, the dot product between the predicted object locations and the rendered object masks indicate how well the model matches our image. We treat our two p(object)

[1] Training was performed using 10-fold cross validation on a subset of the SUN Database (Xiao et al. 2010), for which there exist LabelMe annotations (Torralba et al. 2010).

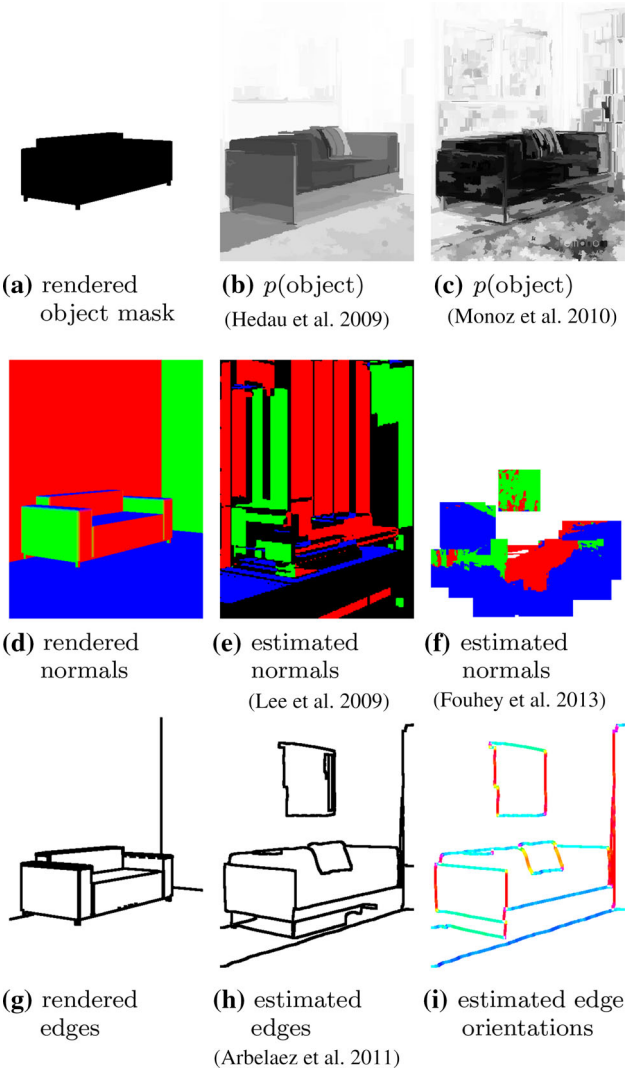


Fig. 4 Descriptors extracted from an input image (*right columns*), and their corresponding rendered descriptors from the top-ranked 3D model (**a**, **d**, **g**)

descriptors separately as two independant similarity features. The first compares the output of Hedau et al. (2009)'s descriptor to rendered object masks, and the second compares Munoz et al. (2010)'s output to rendered object masks. These similarity measures are the first features we use to compare 3D models to an input image.

Surface normals: We use the plane-sweeping algorithm of Lee et al. (2009) and the new 3D primitive algorithm of Fouhey et al. (2013) to predict the surface normals of each pixel in an input image. For each 3D model, we render a surface normal image, by simply setting the red, green and blue color components of each polygon to the x, y and z components of the polygon's surface normal. See Fig. 4d–f, for examples of predicted surface normals, and rendered surface normals. The normalized dot product between the rendered descriptor and each surface normal quantifies their similarity. We use these values as separate features when scoring each 3D model. Lee et al. (2009)'s plane-sweeping algorithm produces a single surface normal estimate for each image, resulting in a single similarity feature; on the contrary, Fouhey et al. (2013)'s data-driven 3D primitive algorithm produces outputs at varying levels of sparsity and confidence. We use the dense interpretations from Sect. 4 of Fouhey et al. (2013)'s algorithm, as well as the sparse surface normal estimates at eight different levels of sparsity. Each of these nine surface normal estimates are compared to 3D model surface normal renderings to produce a nine-dimensional similarity feature.

Edges: We extract edges from an input image using the globalPb algorithm (Arbelaez et al. 2011) (thresholded at $p(\text{boundary}) > 0.5$). These edges are compared to Canny edges which are extracted from rendered surface-normal images of each scene hypothesis (Fig. 4g, h). Pairs of edge images (extracted from the input image and each rendering) are compared using a modified symmetric Chamfer distance ($a \in A$ indicates a is an edge pixel in image A):

$$\Delta_{edge}(A, B) = \frac{1}{|a|}\sum_{a\in A}\min\left(\min_{b\in B}\|a-b\|, \tau\right) + \frac{1}{|b|}\sum_{b\in B}\min\left(\min_{a\in A}\|b-a\|, \tau\right). \quad (1)$$

To avoid the effect of outlier edges which do not match well, we truncate individual edge distance penalties at different thresholds ($\tau \in \{10, 25, 50, \infty\}$). Intuitively, distances computed with smaller values of τ encourage fine-grain matching of edges, while distances computed with larger thresholds aim to penalize large errors. Each of the distances computed with a different value of τ is treated as a separate feature, for a total of four features.

In addition, we compute a second edge similarity feature which takes into account not only the location of edges, but also their orientation. We use an oriented chamfer distance, which matches only edges which are within 30° of each other. This reduces the effects of spurious edges which are spatially close, but not properly oriented in the image. To efficiently compute the oriented chamfer distance, we discretize edges into 12 overlapping bins of 30° covering the half-circle. This is similar to the directional chamfer matching approach introduced by Liu et al. (2010), with the addition of overlapping bins to alleviate the effects of quantization artifacts at the boundaries between buckets. We use the same edge penalty truncation approach described above to reduce the influence of outlier edges, resulting in another four-dimensional similarity feature (corresponding to different thresholds).

2.3 Hypothesis Ranking

For a given input image, we render all of the 3D models in our scene library and compute similarity features from the renderings, as described above. We concatenate the object mask (2), surface normal (10), and edge (8) features into a 20-dimensional feature vector. A linear weighting of these similarity features is computed to determine a matching score indicating how similar each 3D model is to the given image.

We learn this weight vector using a max-margin learning framework. Using annotated training data, we can rank how well each 3D model in our library matches each training image. We compute a similarity score for each pair of images and 3D models by comparing both the surface normals and object locations of the rendered 3D models to renderings of hand-annotated scene geometries which are treated as ground-truth. For this ranking, we multiply the object mask agreement score with the surface normal agreement score for each pixel (both scaled to be in the range [0, 1]). This combined score aims to count how many pixels in the image satisfy two constraints: Firstly, objects in the rendered 3D models should appear only where they are in the ground-truth. Secondly, the surface normals of the 3D models at these locations should also agree with the ground-truth surface normals. Using this metric, we score how similar each 3D model is to each training image.

Our goal is to find a weight vector w which can correctly rank pairs of 3D scenes (i.e.: $w^\top x_j^i > w^\top x_k^i$ if scene j matches image i better than scene k). We use the difference in masked surface normal scores as the hinge loss margin δ_{jk}^i. This optimization takes the form of support vector ranking (Herbrich et al. 1999):

$$\begin{aligned} &\min_{w,\xi} \frac{\lambda}{2}\|w\|^2 + \sum \xi_{jk}^i \\ &\text{s.t.: } w^\top x_j^i \geq w^\top x_k^i + \delta_{jk}^i - \xi_{jk}^i, \;\; \xi_{jk}^i \geq 0. \end{aligned} \tag{2}$$

We optimize Eq. 2 using a stochastic subgradient method. In each iteration, we select a training image i and a pair of 3D models (j, k). If the current weight vector causes the pair of 3D models to be incorrectly ranked, or if their difference in scores is less than the margin δ_{jk}^i, we compute a subgradient and update the weight vector. This convex optimization is repeated until convergence.

Prior to learning weights for each feature, we first perform feature selection by incorporating an ℓ_1 penalty term, enforcing sparseness:

$$\begin{aligned} &\min_{w,\xi} \frac{\lambda}{2}\|w\|_1 + \sum \xi_{jk}^i \\ &\text{s.t.: } w^\top x_j^i \geq w^\top x_k^i + \delta_{jk}^i - \xi_{jk}^i, \;\; \xi_{jk}^i \geq 0. \end{aligned} \tag{3}$$

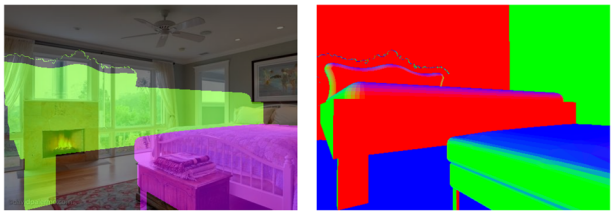
Features with negligible weights (less than 1%) relative to the average weight are discarded, and the selected features are re-weighted using the ℓ_2 regularized SVM ranking described in Eq. 2.

3 Viewpoint Selection

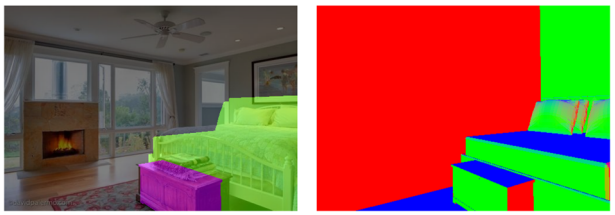
The problem of viewpoint estimation is very challenging. Estimating the layout of a room, especially in situations where objects such as furniture occlude the boundaries between the walls and the floor remains unsolved. Recently, researchers such as Hedau et al. (2010), Lee et al. (2010), Pero et al. (2012) proposed mechanisms for adjusting the estimated locations of walls and floors to ensure that objects (represented by cuboids) are fully contained within the boundaries of the scene. Inspired by these approaches, we aim to intelligently search over viewpoint hypotheses. Intuitively, if we can fit an object configuration using a particular viewpoint hypothesis with high confidence, then that room layout is likely correct (i.e., it allows for objects to be properly matched). By searching over possible viewpoints, we aim to alleviate the brittleness of our baseline scene matching approach (Satkin et al. 2012) which relied on hard decisions for the estimated viewpoint of an image. It should be noted that our geometry estimation algorithm is one of many recent works which rely on accurate viewpoint estimation Fouhey et al. (2012); Gupta et al. (2011). These types of geometry estimation algorithms are unable to recover when the room layout estimation process fails.

Thus, we present a framework which does not assume any individual viewpoint hypothesis is correct. Rather, we use our learned cost function to re-rank a set of room layout hypotheses by jointly selecting a combination of furniture and camera parameters, which together best match the image. We search over the top N room layout hypotheses, returned by the algorithm of Hedau et al. (2009). For each individual room layout, we use the estimated camera parameters corresponding to that room layout to render every 3D model. This approach scales linearly with the number of viewpoints and geometry hypotheses explored, and is trivially parallelizable. Moreover, we can leverage the self-similarity of environments to efficiently explore the search space in sublinear time as described in (Satkin 2013, Sect. 3.6). In all our experiments, we consider the top 20 results from Hedau et al. (2009)'s room layout algorithm. However, our approach is agnostic to the source of these viewpoint hypotheses, and additional hypotheses from Lee et al. (2009), Pero et al. (2011), Schwing and Urtasun (2012) or any other algorithm could easily be incorporated to improve robustness.

Figure 5 illustrates the benefit of searching over various camera parameters. The top row shows the result of 3DNN using only the top-ranking room layout from Hedau et al.



(a) Result using only the top-ranking camera parameters from (Hedau et al. 2009)



(b) Result after re-ranking the top-20 hypotheses from (Hedau et al. 2009)

Fig. 5 Example results highlighting the benefit of searching over viewpoint hypotheses. The *top row* shows the best matching scene geometry using the top-ranking room layout hypothesis of Hedau et al. (2009). The *bottom row* shows the best matching scene geometry after intelligently selecting the best room layout

(2009). Note that the failure to accurately estimate the height of the camera causes inserted objects to be incorrectly scaled. However, by not limiting ourselves to a single camera parameter hypothesis, we can automatically select a better room layout estimate, enabling a higher-scoring geometry match to be found. Figure 5b uses the 10th-ranking hypothesis from Hedau et al. (2009), and has the highest matching score using our learned cost function.

Figures 6 and 7 show the full set of hypotheses considered for a scene during our viewpoint selection process. Note that the beds and nightstands almost fully occlude the wall/floor boundaries in the image, resulting in an inaccurate room layout estimate (see Fig. 6a). As shown in Fig. 7a, our baseline scene matching algorithm does its best to find a 3D model which when rendered from this inaccurate viewpoint aligns with the input image; however, the result is incorrect. Looking through the best matching scene geometries for each viewpoint hypothesis, we see that the more accurate a viewpoint estimate is, the more precisely we can find a matching scene geometry which aligns with the input image. By independently scoring and ranking each of these hypotheses, we correctly identify Hypothesis 13 (Figs. 6m, 7m) as the best viewpoint for this scene.

Selecting the room layout hypothesis which affords for the best 3D model matching improves the accuracy of room box estimation (14.0 % per-pixel error with viewpoint selection versus 16.4 % error without viewpoint selection). See Sect. 5 for further analysis of the benefits of viewpoint selection for the task of 3D reconstruction.

4 Geometry Refinement

In order to accurately segment objects in an image, and reason about their occlusions, we must precisely estimate their positions. However, a fundamental limitation of nearest-neighbor approaches is that their outputs are restricted to the space of object configurations seen in training data. This is a problem which has affected both 2D and 3D non-parametric methods. Recently, algorithms such as SIFT flow (Liu et al. 2008) have been developed to address this issue. The SIFT flow algorithm perturbs a matched image by warping the pixels to better align with the input image. However, because this approach warps pixels in the image plane, there is no longer a coherent 3D interpretation of the result. Thus, we propose a two-stage geometry refinement algorithm which is inherently 3D. Our method first searches for the best location of each object in 3D, such that the projection of these objects best align in the image plane, producing a more precise result. Next, we search through a library of 3D models and try to replace each object in the scene with objects that more precisely match the size and style of the objects in the image. We now describe each of these refinement techniques and demonstrate their effectiveness (both qualitatively and quantitatively).

It should be noted, that our algorithm is not the only work to address 3D geometry refinement. In Hedau et al. (2012), the authors present a refinement approach which locally adjusts the position of cuboids to better match an image. Similarly, Pero et al. (2011) perturb the locations of cuboids as diffusion moves of a Markov Chain Monte Carlo optimization, which also includes the addition and removal of cuboids to the scene. The authors have recently extended their algorithm to refine hand-crafted parametric models of objects Pero et al. (2013). Our approach differs from these works in our use of vast repositories of non-parametric models. This data allows us to not only detect the positions and sizes of objects, but also their precise styles. Moreover, by not allowing objects to be arbitrarily resized, we ensure that they maintain real-world dimensions. For example, although beds have many possible styles, they cannot be arbitrarily resized; they come in discrete sizes (queen, king, etc.). Resizing household objects may violate real-world distributions, and result in scene geometries which no longer afford for the human actions they were designed to enable. Thus, we sample over discrete object hypotheses from 3D Warehouse to refine scenes while maintaining object functionality.

4.1 Object Location Refinement

We first search for local refinements of object locations which improve the overall geometric scene matching score using a stochastic geometry refinement algorithm. In each itera-

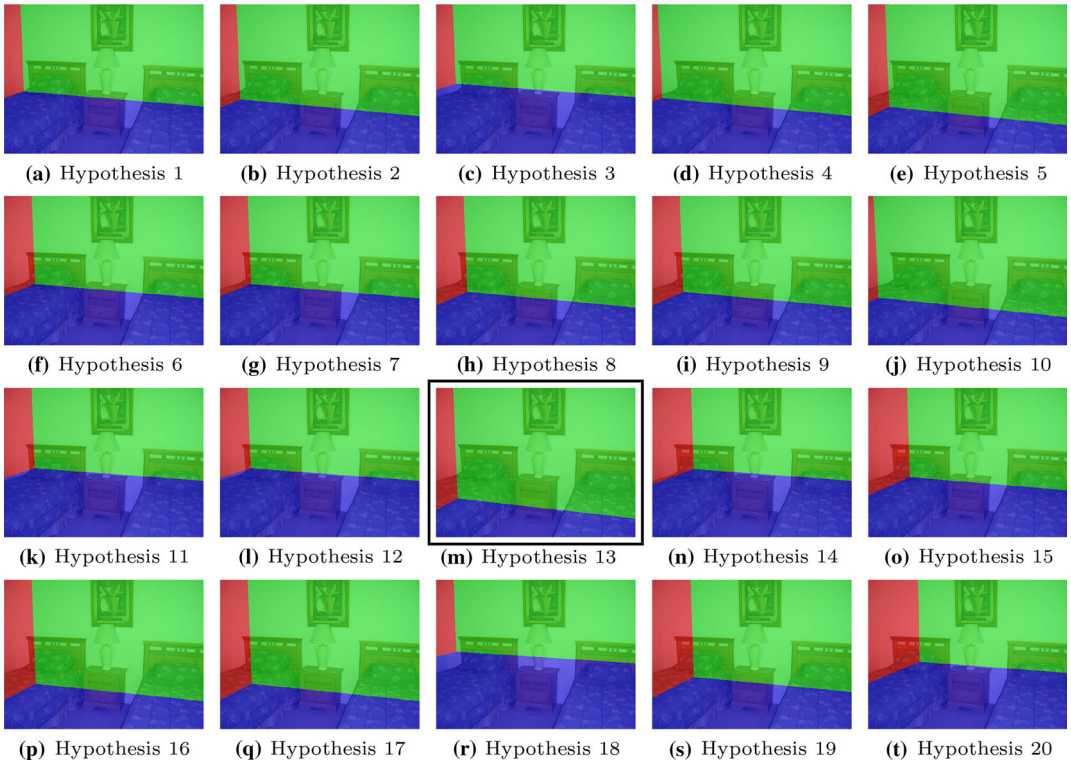


Fig. 6 Top 20 viewpoint hypotheses from Hedau et al. (2009) for an input image

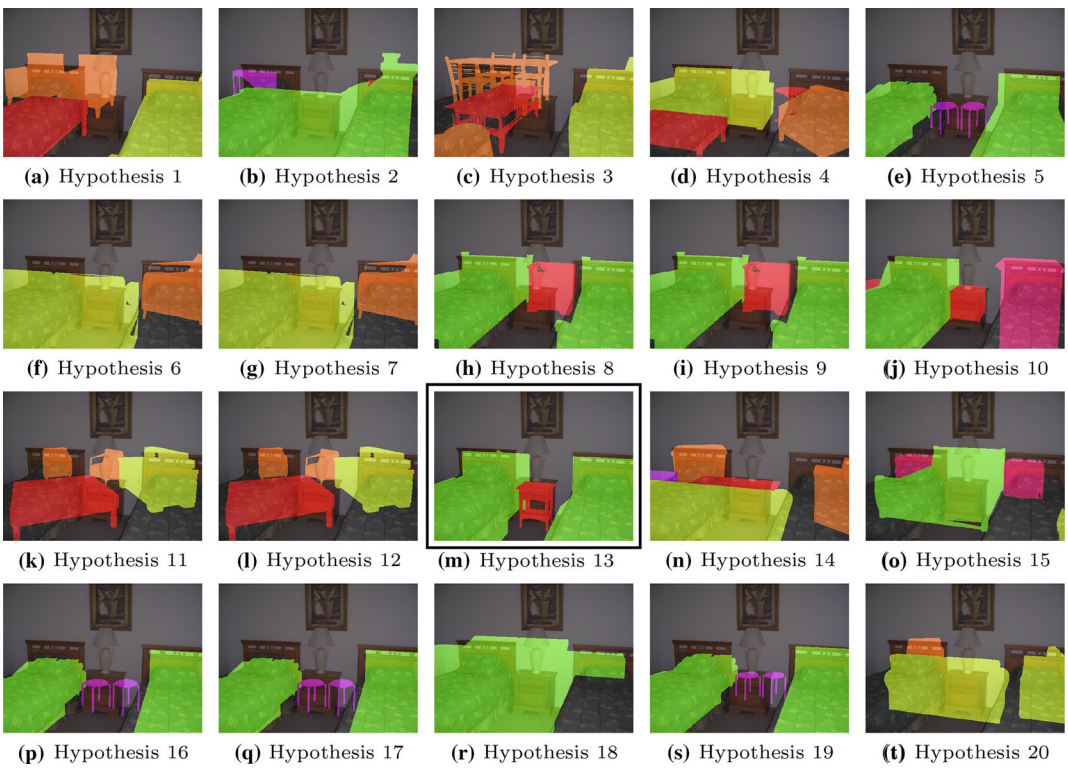
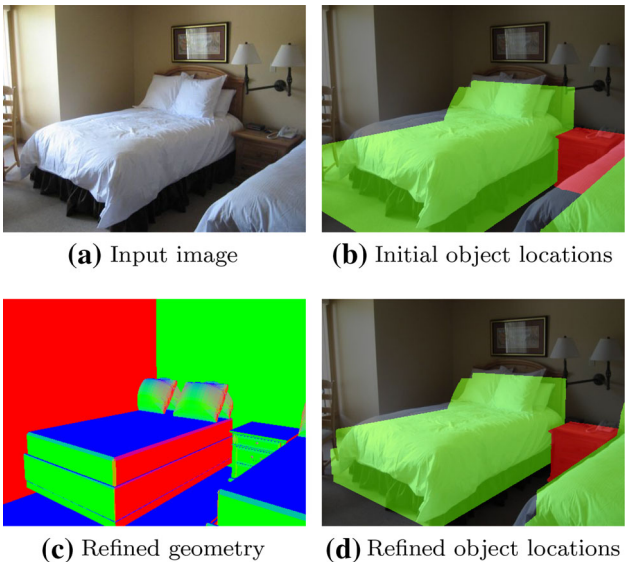


Fig. 7 Object overlays for the best matching scene geometries given each viewpoint hypothesis



(a) Input image **(b)** Initial object locations

(c) Refined geometry **(d)** Refined object locations

Fig. 8 Effects of the geometry refinement process. Note that object boundaries are well-delineated after refinement

tion of the refinement, the locations of objects on the x–y plane are perturbed (height off the ground remains fixed), by adding Gaussian noise (σ=1in) to the current objects' locations. If the adjusted objects' locations match the image better than the previous locations, the new locations are saved. This process repeats until convergence. In practice, a few hundred iterations are required to reach a final refined scene geometry.

Figure 8 highlights the effects of our geometry refinement process. Note the initial object locations in 8b, when projected into the image plane do not align with the actual object boundaries. However, after refinement, in 8d the objects very precisely align with the image boundaries. The projected objects produce an excellent segmentation mask, and because the scene interpretation is inherently 3D, we can properly reason about occlusions and depth ordering.

Figure 9 shows an interesting visualization of the object location refinement process for a scene with a bad initial geometry estimate. Note the poor p(object) masks in Fig. 9b, c, resulting in the incorrect geometry match shown in Fig. 9d. The location refinement algorithm slides these objects around until the couch on the left nicely aligns with the bed in the input image, and the end table is positioned where the nightstand appears in the image. This example demonstrates the capability of our refinement algorithm to adjust the locations of objects in 3D, such that their projections best align with the input image.

4.2 Object Swapping

The sizes, shapes and styles of objects found in real-world environments is quite diverse. Just as our initial scene matching approach is limited by the space of object configurations seen in training data, after location refinement, the accuracy of our results is still limited by the set of object geometries found in each matched 3D model. For example, Fig. 10b shows an initial scene geometry match for which the identities and locations of objects have been correctly predicted; however, the style of these objects are not accurately matched. The image contains a canopy-style bed with a tall metal frame, while our matched 3D model contains a more traditional bed with a rounded headboard and footboard.

The high level of diversity in object styles found in real-world scenes cannot be represented using parametric models. Thus, we leverage the vast collection of models in 3D Warehouse to search for objects which more precisely match the input image. Here, we aim to go from coarse-level object matches to instance-level matches. Our object swapping algorithm is simple and intuitive. For each object in a matched 3D model, we remove it from the scene and replace it with a new object from the same category. When replacing each object, we rotate and position the new models such that they best align with the original objects' positions. We use the same similarity features and learned cost function from Sect. 2.3 to score each object swapping hypothesis, and select the instance which maximizes this score. Hundreds of swapping hypotheses are considered for each object in the scene, and scored independently.

Figure 10d shows the resulting scene geometry after object swapping. Note that after object swapping, the unusual bed style is correctly matched. Figure 10f shows the diversity of hypotheses considered during the swapping process. By searching through hundreds of 3D models, we automatically select the instance which most precisely aligns with the input image. See Fig. 11 for additional examples of object swapping. Note that after swapping, the style and size of each object is more precise than in the initial geometry estimate.

5 Evaluation

We now evaluate the performance of our 3DNN algorithm. The goals of these experiments are two-fold. First, we analyze the added benefit of each component of the 3DNN system: improved similarity features, geometry refinement and viewpoint selection. In the next section, we compare 3DNN with state-of-the-art 2D appearance-based scene matching aproaches, to demonstrate the benefits of viewpoint invariant scene matching.

5.1 Evaluation Metrics

For each of these experiments, we report performance using the two "Pixelwise Surface Normal Accuracy" metrics from Satkin et al. (2012), one measuring how accurately the surface normals of all pixels are predicted, the second eval-

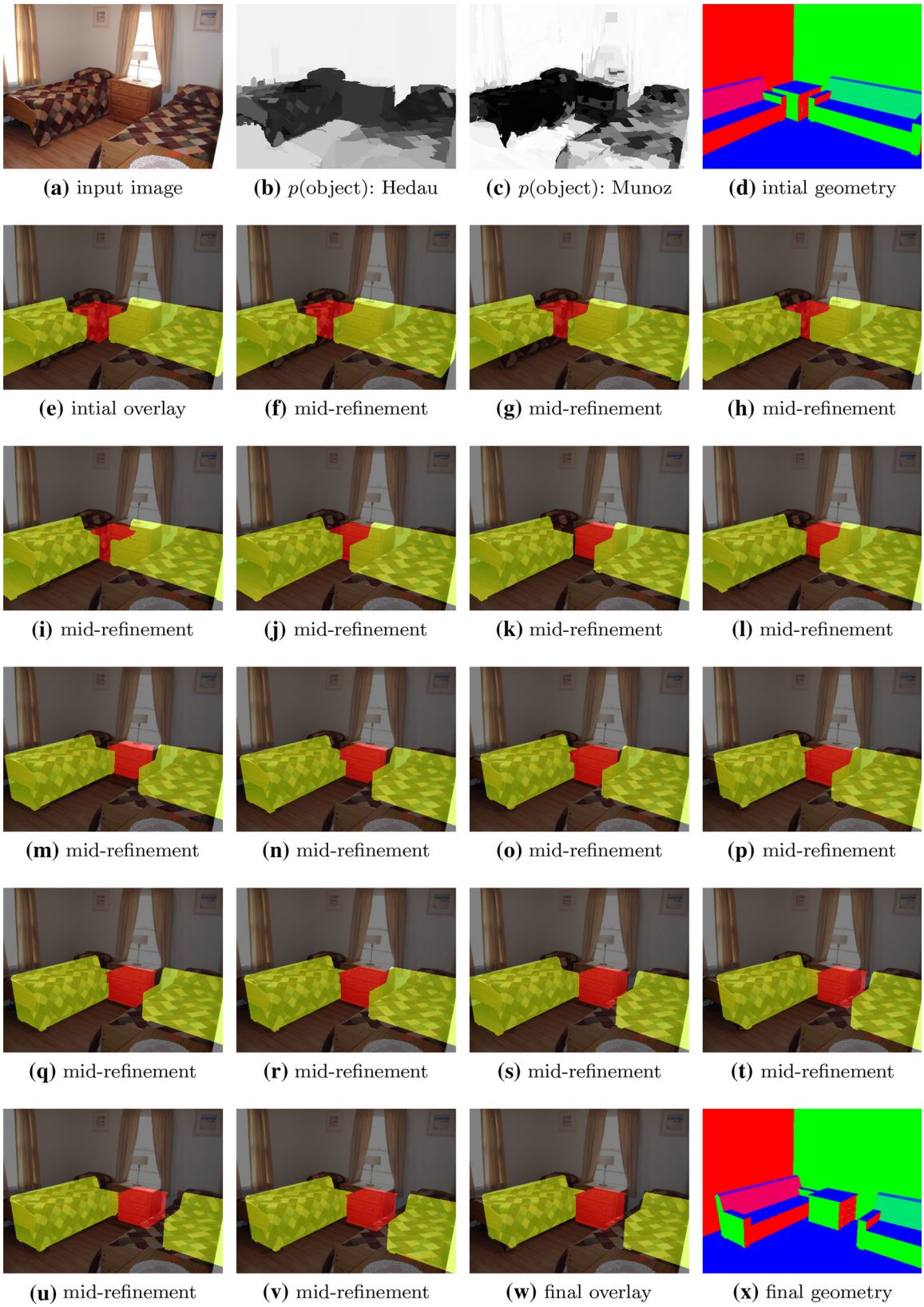


Fig. 9 Visualization of the geometry refinement process for a scene with a poor initial geometry match. Note the incorrect p(object) masks in (**b**) and (**c**), resulting in the poor initial geometry match (**d**). Given this geometry, the location refinement algorithm slides these objects around until the couch on the left nicely aligns with the bed in the input image, and the end table is positioned where the nightstand appears in the input image

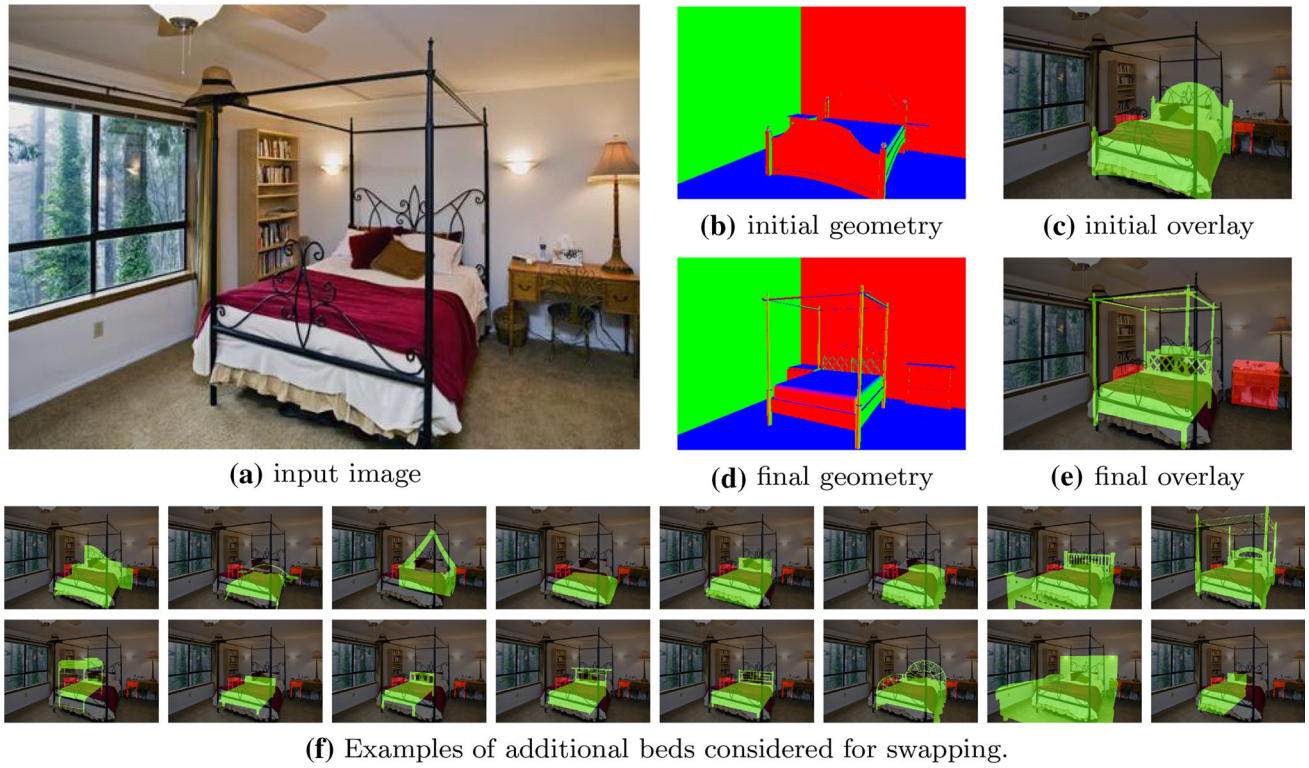


Fig. 10 Example effects of object swapping. Note that after swapping, the canopy style of the bed is correctly matched. Shown below are additional examples of the hundreds of beds considered during the swapping phase of the geometry refinement process

uating only those pixels which correspond to objects in the ground-truth annotations. Although these metrics are informative for the task of surface normal prediction, they are unable to capture how accurately objects in an image are localized. For example, a horizontal surface corresponding to a bed in an image may be scored as "correct" even if the predicted scene contains no objects. This is because the horizontal floor has the same orientation as the bed's surface. Thus, we present results computed using a new metric, "Matched Objects Surface Normal Accuracy." This is a strict metric which requires two criteria to be met: For each pixel corresponding to objects in the ground-truth annotation, we must first correctly predict that there is an object at that location. We compute the dot product between ground-truth and predicted surface normals only at those pixels for which we "match" an object. Unmatched object pixels receive a score of zero. This metric is more sensitive to correctly predicting the exact locations and geometries of objects in a scene.

We also evaluate how accurately our algorithm can estimate the freespace of a room from a single image. Figure 12 shows our ability to recover an architectural floorplan of a room. Note that we are able to identify which regions in space are occupied, estimate the distances between objects, and even make predictions about regions that are not visible in the input images due to occlusions. In Hedau et al. (2012) and Satkin et al. (2012), the authors present various metrics for how accurately their algorithms can predict the 3D freespace of a scene. These metrics require rectifying the predicted scene geometry, and are ill-posed when the estimated viewpoint deviates substantially from the ground-truth camera parameters. For example, if the estimated horizon computed from vanishing points is incorrect and intersects the ground-truth floor polygon, the rectification homography (computed using the ground-truth camera parameters) will produce incoherent results with points at infinity being projected to finite locations when applied to the estimated scene geometry.

Thus, we develop another new metric to measure freespace prediction in the image plane: "Floorplan Overlap Score." For each object in the scene, we render its "footprint" by setting the height of each polygon to 0. A simple pixel-wise overlap score (intersection/union) of the object footprints can now be used to compare the ground-truth floorplan of a scene with our estimated scene geometry.

5.2 Experimental Dataset and 3D Model Library

Since the problem of monocular geometry estimation is relatively new, there does not yet exist an established dataset of images with detailed ground-truth object geometry and surface normals. Thus, we have created a new dataset with anno-

Fig. 11 Example object swapping results. Besides each input image are the results before (*top*) and after (*bottom*) object swapping

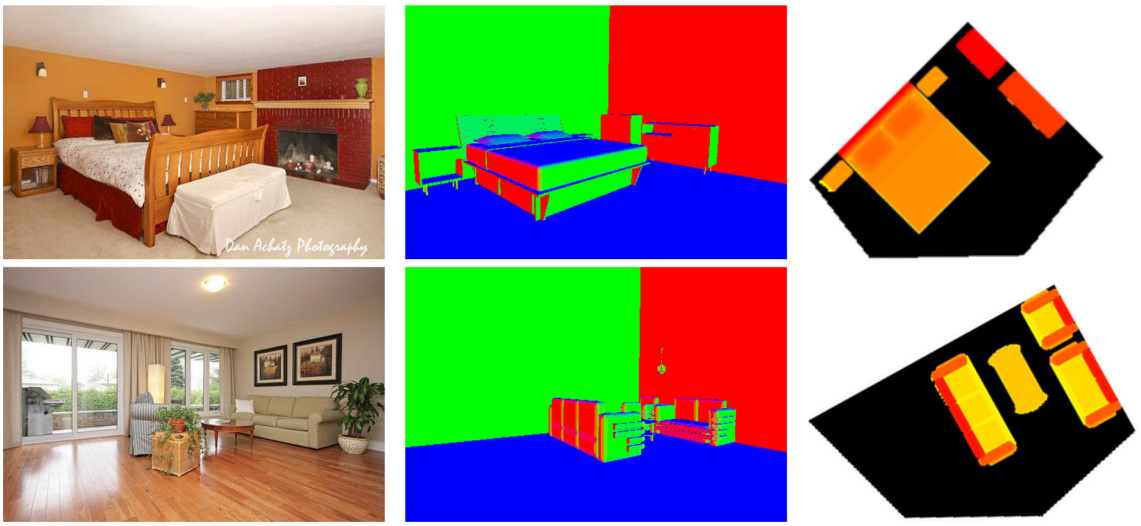


Fig. 12 Input images, automatically-selected 3D models, and overhead views (color indicates height: *yellow* low, *red* high). Results shown use annotated camera parameters

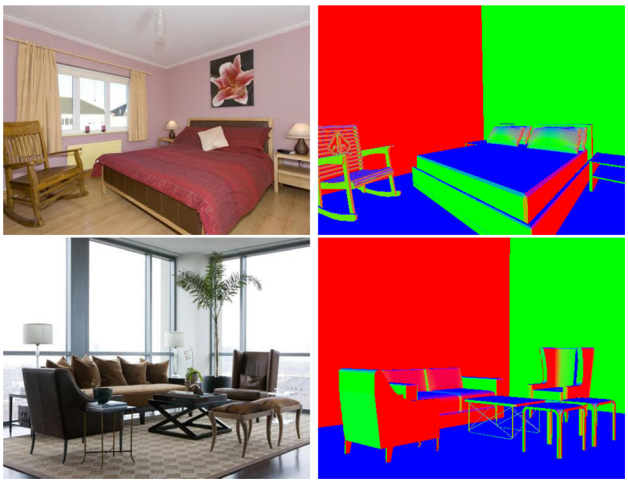


Fig. 13 Example images and hand-crafted 3D models from our dataset

tated scene geometry building upon the SUN database (Xiao et al. 2010). Our dataset consists of over 500 images from the categories "bedroom" and "living room." For each image, a detailed 3D model was constructed using SketchUp (Google Inc. 2000). This software allows users to label vanishing points for camera auto-calibration and insert existing 3D models of objects from the Internet to generate detailed models from an image. Figure 13 includes example 3D models from our dataset. This dataset has been made publically available to researchers as the CMU 3D-Annotated Scene Database Satkin et al. (2012).

We use these hand-crafted 3D models as ground-truth for training our scene ranker, as well as for evaluating the performance of our geometry estimation algorithm. Using images with associated 3D models enables us to relate pairs of images via their underlying geometry. Moreover, because each 3D model has an associated image (unlike data from 3D Warehouse), we are able to transfer any metadata from the source image when parsing an input image.

This type of 3D content with associated imagery is quickly emerging, in large part due to the availability of low-cost RGBD cameras, which has been a catalyst for the rapid increase in 2.5D data. Researchers are now working on automated methods for inferring the full 3D geometry of a scene given a 2.5D projection (Shao et al. 2012; Silberman et al. 2012). As these approaches become more effective, there will be massive amounts of images with associated 3D models, allowing for the first time the exciting possibilities afforded by using the full power of geometric information in conjunction with conventional appearance-based techniques. Our work shows how these emerging new sources of data can be used by quantifying their effectiveness in terms of matching efficiency (dataset size), generalization to unseen viewpoints, geometry estimation, and object segmentation.

5.3 Quantitative Results

We perform all experiments using the CMU 3D-Annotated Scene Database Satkin et al. (2012), containing 526 images of bedrooms and living rooms. All training was performed using 5-fold cross-validation to evaluate performance on all images in the dataset. Figure 14 reports the performance of our scene matching algorithm with individual components turned on/off. As a baseline, we measure the performance of Choi et al.'s 3D Geometric Phrases algorithm (Choi et al. 2013) on the dataset, which is shown in blue. Note that 3DNN offers dramatic performance improvements over 3DGP by producing detailed 3D models as opposed to coarse cuboids as shown in Fig. 15. Our approach goes beyond the bounding

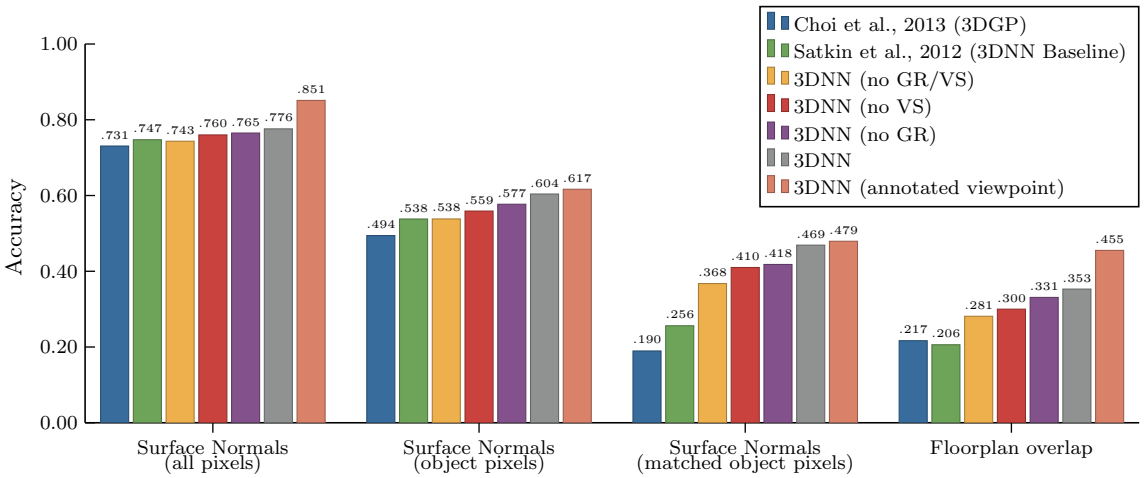
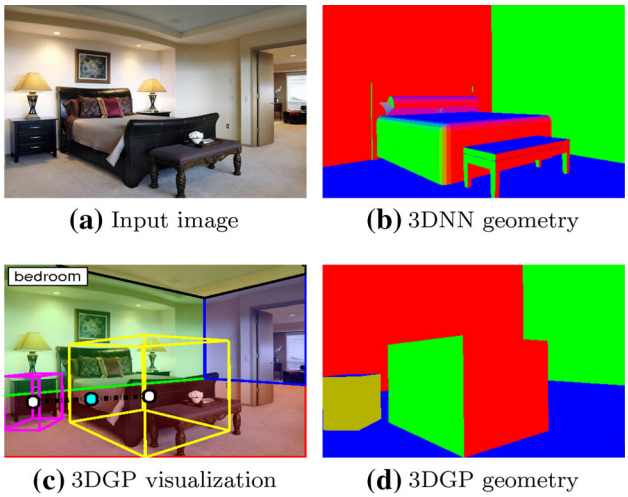


Fig. 14 Analysis of the benefit of each component in our improved scene matching algorithm

boxes to generate detailed 3D models with accurate surface normals and freespace estimates.

The green bars indicate the baseline performance using our preliminary 3DNN scene matching algorithm described in Satkin et al. (2012). The next set of bars (yellow) indicate the added benefit of using the expanded set of similarity features presented in Sect. 2.2. Note that the additional similarity features account for a large boost in performance. This effect is especially pronounced in the matched object surface normals and floorplan overlap scores, for which accurately predicting the locations of objects is crucial. The red bars show the performance by running our geometry refinement process (with the new feature set). The purple bars show the performace gains achieved via viewpoint selection (with the new feature set). The gray bars indicate the performance achieved by running the full 3DNN algorithm with new features, viewpoint selection and geometry refinement. Lastly, for comparison, we report in pink the performance of our scene matching approach if we were to use annotated viewpoints (similar to the analysis in Satkin et al. (2012)). The gap in performance between the gray and pink bars represents the remaining error due to incorrect viewpoint estimation. Comparing the red, gray, and pink bars shows that our viewpoint selection mechanism accounts for a substantial performance gain; however, there still remains considerable room for improvement. This is most pronounced in the all pixel surface normal and floorplan overlap scores, which are sensitive to correctly estimating the locations of the walls and floors in a scene.

These extensions to our baseline scene matching algorithm have resulted in considerable improvements to both the robustness and precision of our results. Figures 16 and 17 include example results of 3DNN on a wide variety of bedroom and living room scenes. Note that we are able to produce accurate 3D models shown in the surface normal renderings beside each input image. In addition, each object's



(a) Input image **(b)** 3DNN geometry

(c) 3DGP visualization **(d)** 3DGP geometry

Fig. 15 Comparison of the detailed geometry inferred by 3DNN with cuboids generated by 3DGP Choi et al. (2013)

boundaries are well-delineated due to our geometry refinement stage, as indicated in the overlaid object segmentation masks.

We quantify the benefits of viewpoint selection by comparing the accuracy of our results with and without viewpoint selection. Figure 18 shows the distribution of performance gains seen across all images in the CMU 3D-Annotated Scene Database as a result of the viewpoint selection process. The y-axis indicates how much the matched object surface normal score was affected via viewpoint selection. Note that for approximately two-thirds of the images, the viewpoint selection process results in an improved scene geometry (indicated in green). Not only does viewpoint selection result in more accurate object geometries, it also improves the accuracy of room box estimation by re-ranking viewpoint hypotheses based on which room layout affords for the best 3D model matching (14.0 % per-pixel error with viewpoint selection versus 16.4 % error without viewpoint selection).

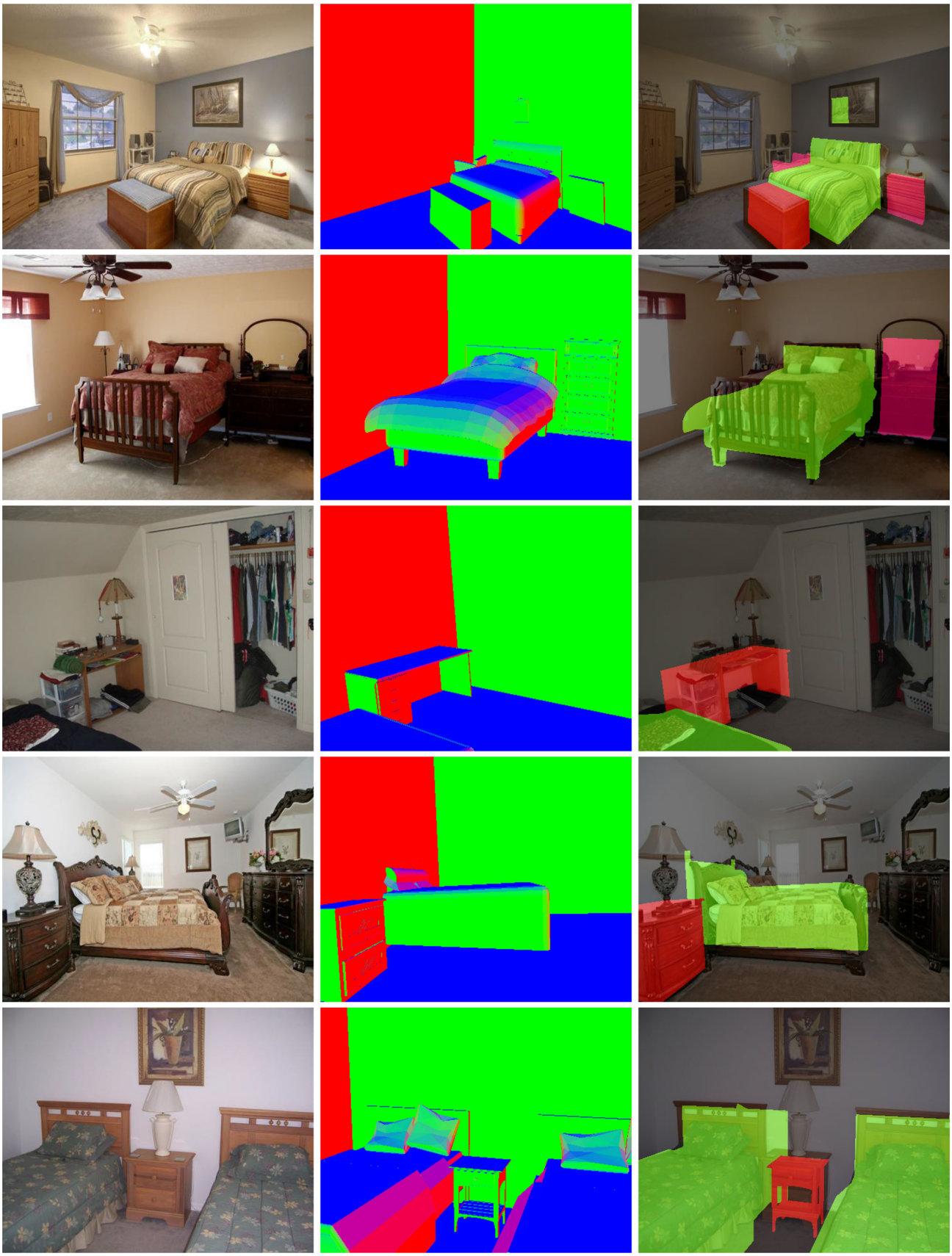


Fig. 16 Qualitative results of bedroom scenes. From *left* to *right* input images, surface normal renderings and overlaid object segmentation masks

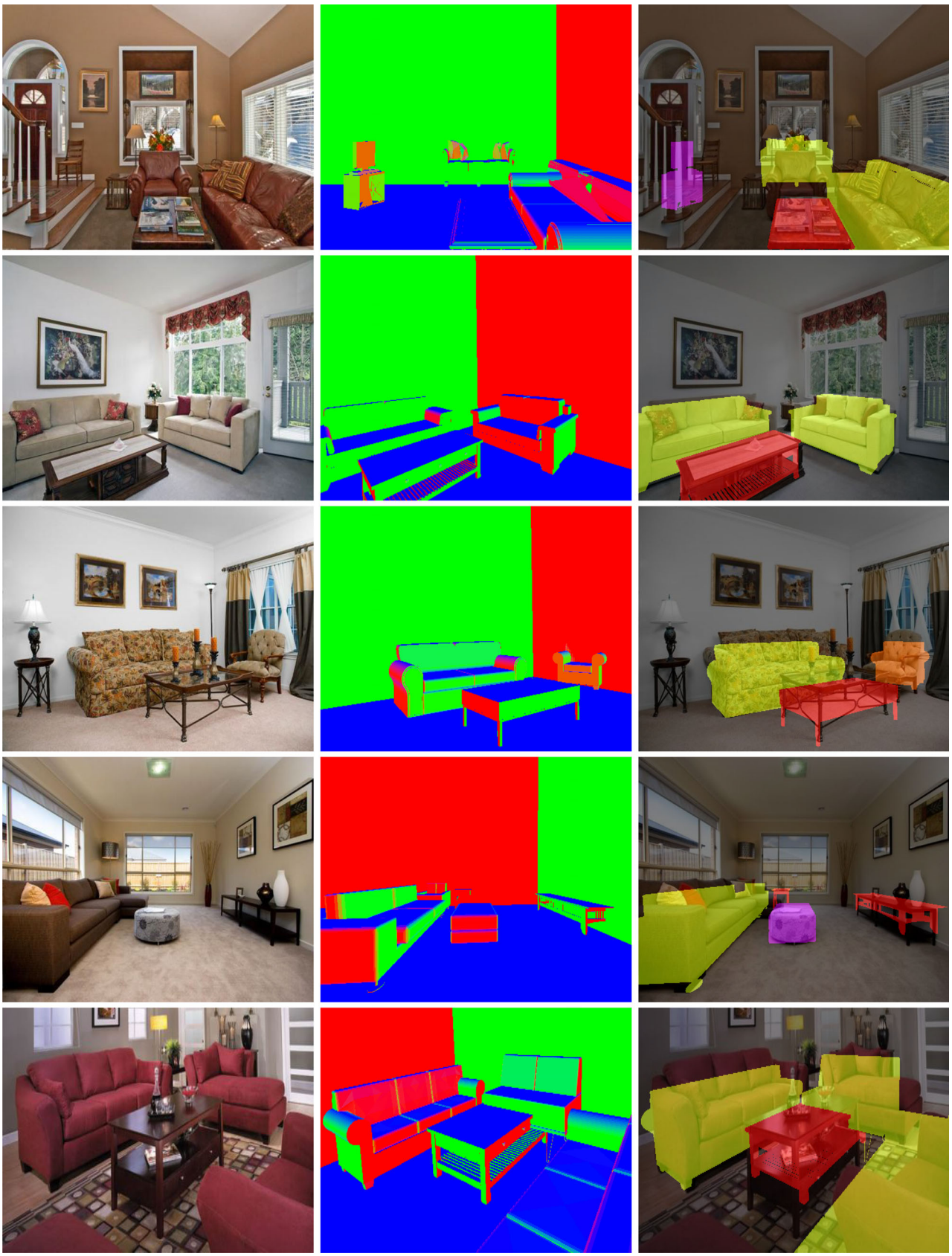


Fig. 17 Qualitative results of living room scenes. From *left* to *right* input images, surface normal renderings and overlaid object segmentation masks

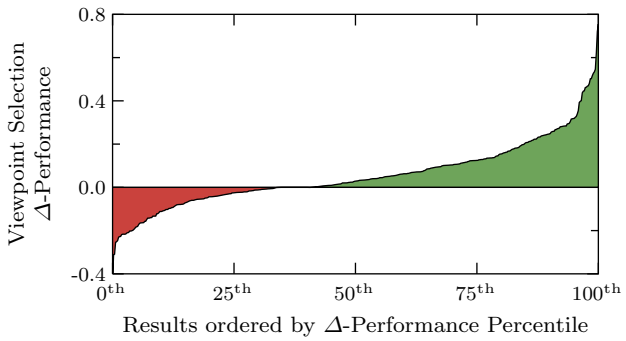


Fig. 18 Distribution of improvements resulting from the viewpoint selection process. Performance is measured using the matched object surface normal scores. *Green* indicates a performance increase and examples in *red* resulted in a marginal performance decrease

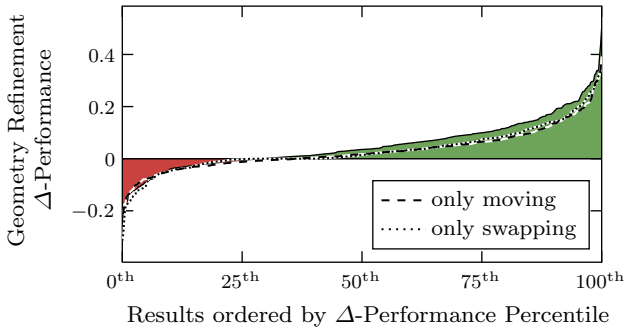


Fig. 19 Distribution of improvements resulting from the geometry refinement processes. Performance is measured using the matched object surface normal scores. *Green* indicates a performance increase and examples in *red* resulted in a marginal performance decrease. The *dashed* and *dotted lines* indicate the performance of only moving or only swapping objects, respectively

In Sect. 4.1, we presented an algorithm to refine the locations of objects in 3D, and in Sect. 4.2, we presented an algorithm to replace individual objects from a 3D model with objects which more closely match the input image. We now evaluate how each of these refinement methods affects the overall performance of our scene matching algorithm. Figure 19 shows the distribution of performance gains seen across all images in the CMU 3D-Annotated Scene Database as a result of the geometry refinement process. The dashed line shows the effects of only moving objects and the dotted line shows the effects of only swapping objects. Quantitatively, these two refinement mechanisms perform similarly, resulting in performance gains on approximately two-thirds of the images. However, by combining the two refinement processes, further performance gains can be achieved, as indicated in the green region of the paired error plot.

Quantitatively, the geometry refinement stages of 3DNN result in modest improvements. This is expected, as the refinement process is inherently local and designed to make small modifications, not major changes which would result in dramatic effects on performance (if a 3D model required substantial refinement, it is not a good scene match, and a different model should have been selected). However, qualitatively our results after refinement are markedly better, with objects being well-localized and their styles properly modeled. These effects are most pronounced in the matched object surface normal and floorplan overlap scores, for which precisely localizing and matching the style of objects is emphasized.

5.4 Feature Analysis

We perform two experiments to analyze the importance of each feature for 3D model matching. First, we run a standard ablative analysis to see how much each feature contributes to the overall performance of our system. Next, we run our scene matching pipeline using only a single feature (or type of feature), and compare the performance of each feature independently with the performance of the full feature set.

The results of both experiments emphasized the importance of our p(object) features. Figure 20 shows an example of a relatively simple scene for which Hedau et al. (2009)'s Indoor Geometric Context is unable to accurately estimate the locations of objects; however, our new approach succeeds. Note that the failure to correctly identify which pixels belong to objects (Fig. 20b) results in a poor 3D model match (Fig. 20c), which is consistent with the p(object) prediction. However the classifier of Munoz et al. (2010) correctly identifies the object locations, enabling better scene matching (Fig. 20e).

These p(object) features are vital to 3DNN's performance; they are most heavily weighted by our support vector ranker, and their ablation results in the largest drop in performance. Interestingly, the removal of either p(object) feature has only a small effect on system performance; howevever, removing both p(object) features results in a dramatic drop in performance, as shown in Fig. 21. This implies that the two features encode much of the same information, and can compensate when the other is removed.

Our inability to robustly rank geometric hypotheses is a result of the challenges inherent in computing the similarity between an image and 3D models. The similarity features used for this are at the core of the 3DNN algorithm; they are required not only to select the 3D model which is most similar to an image, but also to rank viewpoint hypotheses and to determine the location and style of objects during geometry refinement. Thus, continued research into the development of robust new similarity features is likely to result in further performance gains.

6 2D Versus 3D Nearest Neighbor

A natural question to explore is, "What are the benefits of 3DNN compared to a traditional 2D nearest-neighbor approach?" Although 2D nearest-neighbor approaches are

Fig. 20 Effects of incorrect object location estimation on scene matching. Note in (**b**) that only a small region near the headboard of the bed is predicted to belong to an object using our preliminary p(object) feature, resulting in a poor scene match (**c**). However, as seen in (**d**), our new p(object) feature can more accurately predict the locations of objects, enabling a good scene match (**e**)

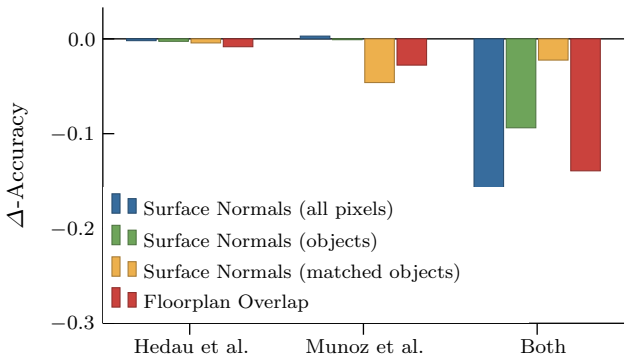


Fig. 21 Change in performance resulting from the removal of each p(object) similarity feature

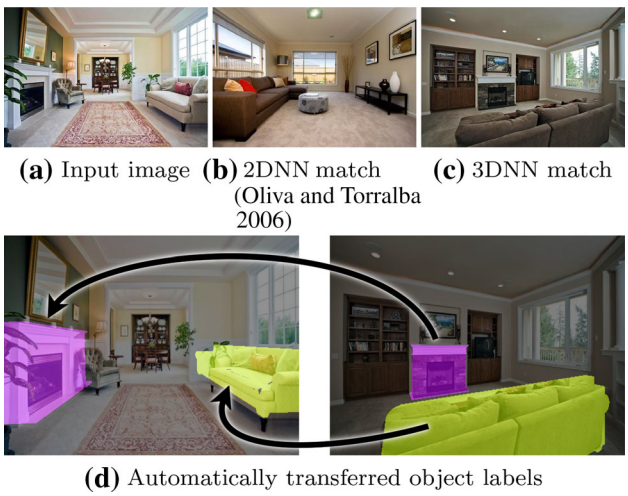


Fig. 22 Extreme viewpoint differences. Traditional appearance based image matching approaches fail to generalize across such extreme viewpoint differences; however, our approach is able to match the geometry of these two scenes, and transfer object labels

powerful, a fundamental limitation of these techniques is the need for vast amounts of data. For a traditional image matching approach to succeed, there must be an image in the recall corpus which is very similar to the input image (i.e., captured from a similar viewpoint, lighting conditions, etc.). This has propelled the growth of datasets, which now measure in the millions of images (Hays and Efros 2007; Torralba et al. 2008). Moreover, despite these massive datasets, 2D nearest-neighbor approaches cannot generalize to never-before-scene viewpoints.

Consider the pair of scenes in Fig. 22. Note the images were captured from drastically different viewpoints. A traditional appearance-based image matching approach such as Liu et al. (2008) and Oliva and Torralba (2006) fails to generalize across such extreme viewpoint differences. Although the scenes appear quite different from the viewpoints they were captured, they have a lot in common: both scenes contain a couch facing a fireplace at approximately the same distance from each other. In this section, we show that we are able to automatically match these images by comparing the appearance of one image with the geometry of another. By decoupling the viewpoint and the geometry of an image, we develop a scene matching approach which is truly 100 % viewpoint invariant.

6.1 Geometry Estimation

We now quantify the performance of 3DNN with a variety of baseline scene matching approaches, including state-of-the-art 2D nearest-neighbor approaches. We compare 3DNN with our baseline scene matching approach from Satkin et al. (2012) as well as two popular 2D nearest-neighbor approaches: GIST Oliva and Torralba (2006) and HoG Dalal

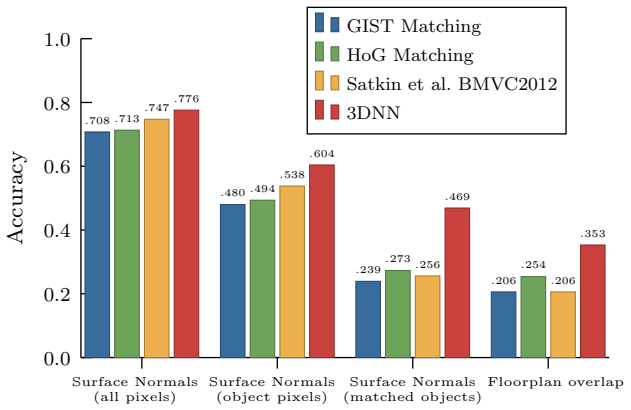


Fig. 23 Comparison of 3DNN with state-of-the-art 2D nearest-neighbor approaches and the geometry matching algorithm of Satkin et al. (2012)

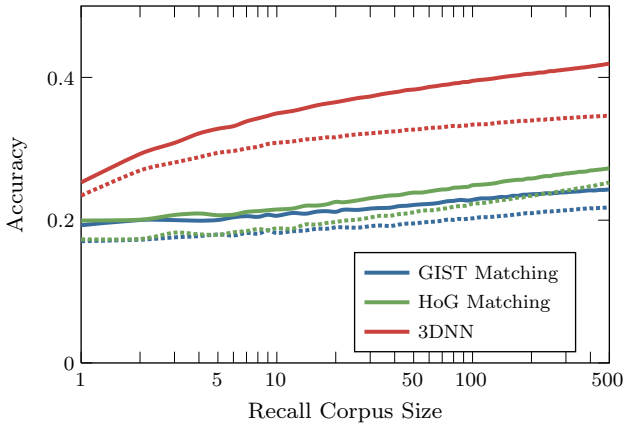


Fig. 24 Accuracy as a function of dataset size. *Solid lines* indicate "matched objects surface normal score," *dotted lines* indicate "floorplan overlap score." Note the logarithmic x-axis

and Triggs (2005) matching.[2] Figure 23, reports the results for 3DNN compared to each baseline, for the task of geometry estimation. Note that our baseline 3D scene matching algorithm (indicated in yellow) does not offer substantial improvement over the 2D nearest-neighbor approaches on the more challenging metrics (matched object surface normals and floorplan overlap score); however, 3DNN exhibits dramatic improvement on each of these metrics.

6.2 Dataset Size

It is well known that for appearance-based image matching to be effective, there must be a large recall corpus of images to match with Hays and Efros (2007) and Torralba et al. (2008). This is because the data set needs to include recall images captured from a similar viewpoint as the query image. On the contrary for 3DNN, the viewpoint and the geometry of the recall images are decoupled. Thus, each scene provides an exemplar which can be matched to images from any viewpoint.

We evaluate this by experimenting with the size of the recall corpus. Figure 24 shows how the performance of 3DNN increases as a function of dataset size, compared to GIST and HoG matching. We report results using two of the more challenging metrics: "matched object surface normal scores" (solid lines) and "floorplan overlap scores" (dashed lines). In these experiments, we consider recall dataset sizes between 1 and 500 images. For each dataset size, we select random subsets of images from the full recall set, and report the performance of each algorithm on the smaller datasets. Due to the high variance in performance using small recall sets, we average performance across 1,000 random subsets of each size. For fair comparison, we do not use our sequence optimization or hypothesis score prediction during these experiments.

There are two important properties of 3DNN we can identify from this graph. Firstly, note that the red plots for 3DNN start out with a higher accuracy (even for a dataset size of one image). This is because our algorithm starts by estimating the room layout of each image, identifying the locations of floors and walls. On the contrary, GIST and HoG matching do not incorporate this knowledge directly, and must infer the viewpoint of the scene by finding a similar image from the recall corpus.

Secondly, note that the curves for 3DNN are steeper than for the appearance-based approaches. This is because on average, each additional training image provides more information in its geometric form, than the raw pixels used in GIST or HoG matching. This indicates that performance is increasing more quickly as a function of the dataset size, and that fewer training examples are required to achieve the same level of performance using 3DNN compared to a traditional appearance-based 2D nearest-neighbor scene matching approach. Remarkably, 3DNN is able to achieve a noticeable performance boost using a recall set size of only 10 images or fewer, due to the algorithm's ability to generalize across never-before-seen viewpoints.

7 Application: Object Recognition

We now explore how our approach can be applied to one of the most common computer vision tasks—object recognition. We can transfer the identities of objects by projecting them onto the image plane to produce per-pixel object labels. Additionally, we can integrate the output of other object detectors to create more robust results.

[2] GIST: 4 × 4 blocks, 8 orientations (code from Oliva and Torralba (2006)). HoG: 20 × 20 blocks, 8 orientations (code from Vondrick et al. (2013)).

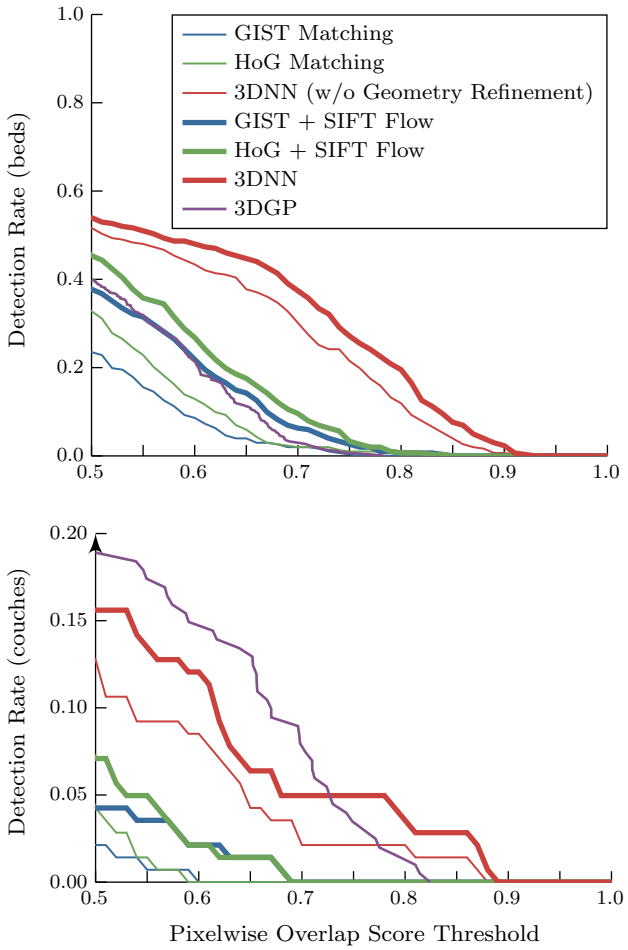


Fig. 25 Object detection rate as a function of overlap score strictness for the "bed" and "couch" categories

7.1 Object Detection and Segmentation

Our mechanism for inferring the structure of a scene in 3D provides us with rich information about the depth ordering and the occlusions of objects when projected onto the image plane. Thus, we should be able to not only detect the locations of objects, but also segment their spatial support in the image by precisely identifying their boundaries. To verify that using 3D cues is an attractive alternative for pixel-based object segmentation, we evaluate the per-pixel overlap score of the ground-truth and the object labels estimated by 3DNN.

Figure 25 analyzes the detection rate of 3DNN, compared to various appearance-based image matching baselines. We measure performance for the "bed" and "couch" categories, two of the most prominent objects in the CMU 3D-Annotated Scene Database. We vary the pixelwise overlap score threshold, and compute what percentage of objects are detected at each threshold. Note that at a stricter threshold of overlap score $\geq .75$, the baseline appearance-based approaches detect very few objects; however, 3DNN still performs well.

Naturally, 3DNN's ability to precisely segment objects is due in part to the geometry refinement stage. To analyze the benefits of this process, we measure the performance of 3DNN with and without the refinement stage. As anticipated, by refining the predicted locations of objects, we achieve a significant (on the order of 5 %) boost in detection rate. For fair comparison, we run the SIFT flow algorithm (the state-of-the-art 2D refinement process) as a baseline. The SIFT flow algorithm of Liu et al. (2008) has been shown to be a robust technique for aligning matched images. By warping each matched scene, SIFT flow refines the location of objects in the image plane, akin to our geometry refinement process. We apply the SIFT flow algorithm using code provided by Liu et al. (2008); this process takes the top-10 scene matches (using either GIST or HoG), warps each matched image, and computes the energy of each warping. We then re-rank the top-10 scene matches according to their SIFT flow energy, and score the top-ranking warped recall image. Although the SIFT flow process yields a significant boost in performance, the algorithm is still not as effective in accurately identifying and segmenting objects compared to 3DNN. Moreover, a key distinction between our geometry refinement process and the SIFT flow algorithm, is that our approach is inherently 3D and produces physically meaningful results. On the contrary, because SIFT flow warps pixels in the image plane, the result no longer has a coherent 3D interpretation.

As an additional baseline, we measure the performance of 3DGP Choi et al. (2013) for the task object detection and segmentation. 3DGP's performance, indicated by the purple plots in Fig. 25, is on-par with 2DNN methods for the bed category; however, their approach performs better on the couch category by leveraging the power of Felzenszwalb et al. (2010)'s discriminatively trained object detector and by training with an order of magnitude more imagery compared to 3DNN.

7.2 Integrating Discriminative Object Detections

Bottom-up appearance-based object detectors such as the discriminative deformable parts model of Felzenszwalb et al. (2010) have been at the forefront of the object recognition field. As a baseline for comparison, we train Felzenszwalb et al. (2010)'s part-based detector using code provided by Girshick et al. (2012). It is important to note that object detectors are in a fundamentally different class of algorithms than our approach. Most discriminative object detectors have a tunable parameter which can be adjusted to achieve arbitrarily high recall by predicting the object at all locations and scales in the scene (at the cost of reducing precision). However, our approach produces a single 3D model for each image, and uses physical constraints which do not allow objects to be hypothesized at arbitrary locations. Thus, our output represents a single point on the precision recall curves.

Many researchers have successfully demonstrated the ability to integrate the output of multiple object recognition systems to create a more robust detector. For example, in Hedau et al. (2010), the authors combine the scores from their "boxy object detector" with Felzenszwalb et al. (2010)'s discriminatively trained deformable part model to produce a state-of-the-art bed detector. Hedau et al. showed that while each of the two approaches perform well on their own, the integration of the two algorithms results in a significant performance boost. Their approach assumes the outputs of each algorithm are statistically independent and multiplies the scores from their boxy object detector with the scores from the deformable parts model.

Following the paradigm of Hedau et al. (2010), we integrate our object detections with the bounding-box detections from Girshick et al. (2012), as a post-processing step. For each object detection, we boost the DPM detector's confidence if our result is in agreement. Specifically, we compare each of the bounding boxes from our results with those from the DPM. If their overlap score is greater than a set threshold, we increase the DPM's confidence by a fixed amount. To determine the best values for this threshold and the amount by which to boost the DPM's confidence, we perform a grid search after partitioning the dataset into 5folds (80 % train, 20 % test).

Figure 26 shows the performance of Felzenszwalb et al. (2010)'s deformable parts model (DPM) with pairs of precision recall curves for the bed and couch categories, respectively. The plots on the left use the traditional 50 % overlap score measure. On the right, we measure performance using a 75 % overlap score threshold; this is a stricter metric, which requires objects to be more precisely localized. The purple plots report the performance of the DPM detector and the red plots show the performance of integrating the DPM's detections with our object predictions. Note that the combined detection results provide a modest improvement (∼10 % relative increase in average precision) over the baseline DPM accuracy. This indicates that our approaches to object classification and geometry estimation provide complementary information. Experimentally, we achieved similar results to Hedau et al. (2010). In isolation, the authors' cuboid detector does not perform as well as Felzenszwalb et al. (2010)'s DPM; however, by intelligently integrating the results of their cuboid detector and the DPM's results, they achieve state-of-the-art performance.

Quantitatively, the DPM out-performs our approach when measured using a liberal 50 % overlap score detection threshold. However, as the overlap score threshold is increased, our approach shows a modest improvement over DPM. This result is not surprising—the features and model used in Felzenszwalb et al. (2010)'s detector are discriminative in nature and explicitly aim to classify each object in the scene. On the contrary, our approach uses features which intentionally ambiguate between object categories. For example, the p(object) feature simply predicts whether or not an object is present, not the identity of the object. This suggests a new family of features which could be integrated into the 3DNN algorithm. Our similarity feature framework enables us to compare any information that can be estimated from an image and rendered from 3D models. Thus, we could incorporate the outputs of an object detector into a similarity feature to leverage the performance of these successful discriminative approaches in a manner similar to Li et al. (2010)'s Object Bank features.

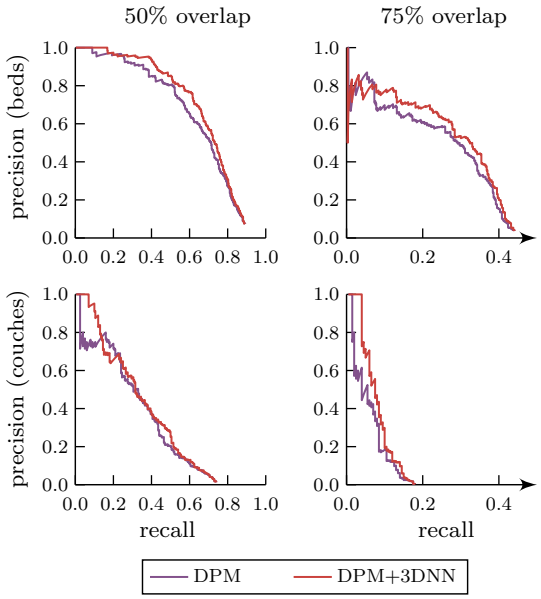


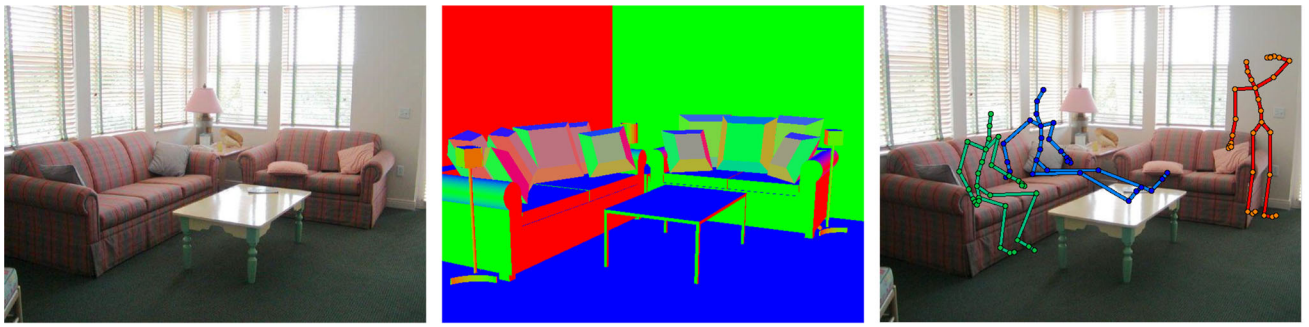
Fig. 26 Object detection precision recall curves for the "couch" category. Our approach provides complementary information to DPM; thus, integrating both results improves performance

8 Application: Affordance Estimation

Given a detailed geometric representation of a scene, there are many possible higher-level interpretations that we can generate. In this section, we summarize our work in affordance estimation (Gupta et al. 2011). This work leverages the availability of motion capture data and derives a novel human-scene interaction model capable of predicting the locations of possible human poses from a single image. Our affordance estimation process utilizes an intermediate geometric representation of a scene. We show the importance of high-quality geometry estimates for this problem, and incorporate our data-driven geometry estimation algorithm to improve upon baseline approaches.

8.1 Algorithmic Overview

This work is an attempt to marry 3D scene understanding with human action modeling. We present a novel qualitative



(a) Input image **(b)** Geometry estimation **(c)** Our human-centric representation

Fig. 27 **a** An input image, **b** geometry estimated using 3DNN, **c** our human-centric representation

scene representation that combines 3D scene geometry with a set of possible human actions, to create a joint space of human-scene interactions. A key insight is to note that there are only two constraints on a 3D human pose that are relevant for embedding it within a given geometry: (1) the 3D space (volume) the pose occupies, and (2) the surfaces it is in contact with. We divide the space around the human actor into blocks and associate each block with a 0 or 1 based on whether the block is occupied or not. In addition, each block may require an external support in a particular direction. For example, in the sitting pose (with back support), we need a horizontal surface below the pelvic joint to support the body and a vertical surface to rest the spine. In a similar manner, for the "reaching" action a horizontal support is required at the feet and a vertical surface of interaction is required to represent the point of contact of the hands (Fig. 27).

By discretizing the scene geometries and human poses into an occupancy matrix, we can efficiently search for locations and poses which satisfy the freespace and support constraints. We slide the discretized human blocks around the scene occupancy matrix using a binary correlation operation. Intuitively, we are searching for locations for which the human pose does not intersect any objects. Additionally, the locations must have the appropriate supporting surfaces to afford each pose. Both of these constraints can be satisfied using simple correlation operations. To account for the deformation of furniture and the human body, we perform an erosion and dilation process on a scenes's occupancy matrix before performing the correlation operations. See Gupta et al. (2011) for a detailed description of this algorithm.

8.2 Results and Evaluation

We now evaluate the utility of our 3D geometry estimation algorithm for determining the affordances of a scene. Figure 28 shows the results of this approach. To visualize the whole range of possible poses, we overlay colored masks indicating the locations of pertinent joints for a given pose. For example, we show in blue the locations where the pelvic joint makes contact with a valid surface of support for the "sitting reclined" task. We also indicate in cyan the locations where the back makes contact with a vertical support. Example human stick figures (extracted from the mocap data) show representative valid poses in each scene. As is evident from the stick figures, our approach predicts affordances that cannot be represented by basic object categories. For example, on the "sitting reclined" pose, our approach combines the vertical surface of the bed with the horizontal surface of the ground to predict human poses.

To quantify the performance of our affordance estimation algorithm, we measure how accurately we can predict the possible locations of human poses in each scene. We evaluate our affordance estimation algorithm using the data-driven scene matching approach for geometry estimation presented in this paper, as well as the geometry estimation approach from Gupta et al. (2011). The approach of Gupta et al. (2011) is a precursor to our geometry estimation algorithm, which uses a simplified geometry estimation process to find a set of cuboids which model the arrangement of furniture in a scene. This approach begins with the same autocalibration technique discussed in Sect. 2.1; however, it considers a limited number of geometry hypotheses, commits to a single viewpoint estimate, and uses only the original p(object) similarity feature for scoring each hypothesis.

We also compare our algorithm with a standard appearance-based baseline; training a separate classifier for each task. These methods have shown good performance for different pixel labeling tasks, such as object categorization and qualitative geometry estimation. Each pose classifier uses appearance features computed from an image to label the pixels where a relevant body joint can appear for that human pose. For example, the "sitting upright" classifier predicts where a person can sit by indicating where the pelvic joint could rest in an image when the person is sitting. Specifically, we use the image features and multiple segmentations classifier of Hoiem et al. (2007), and 50 training images for each clas-

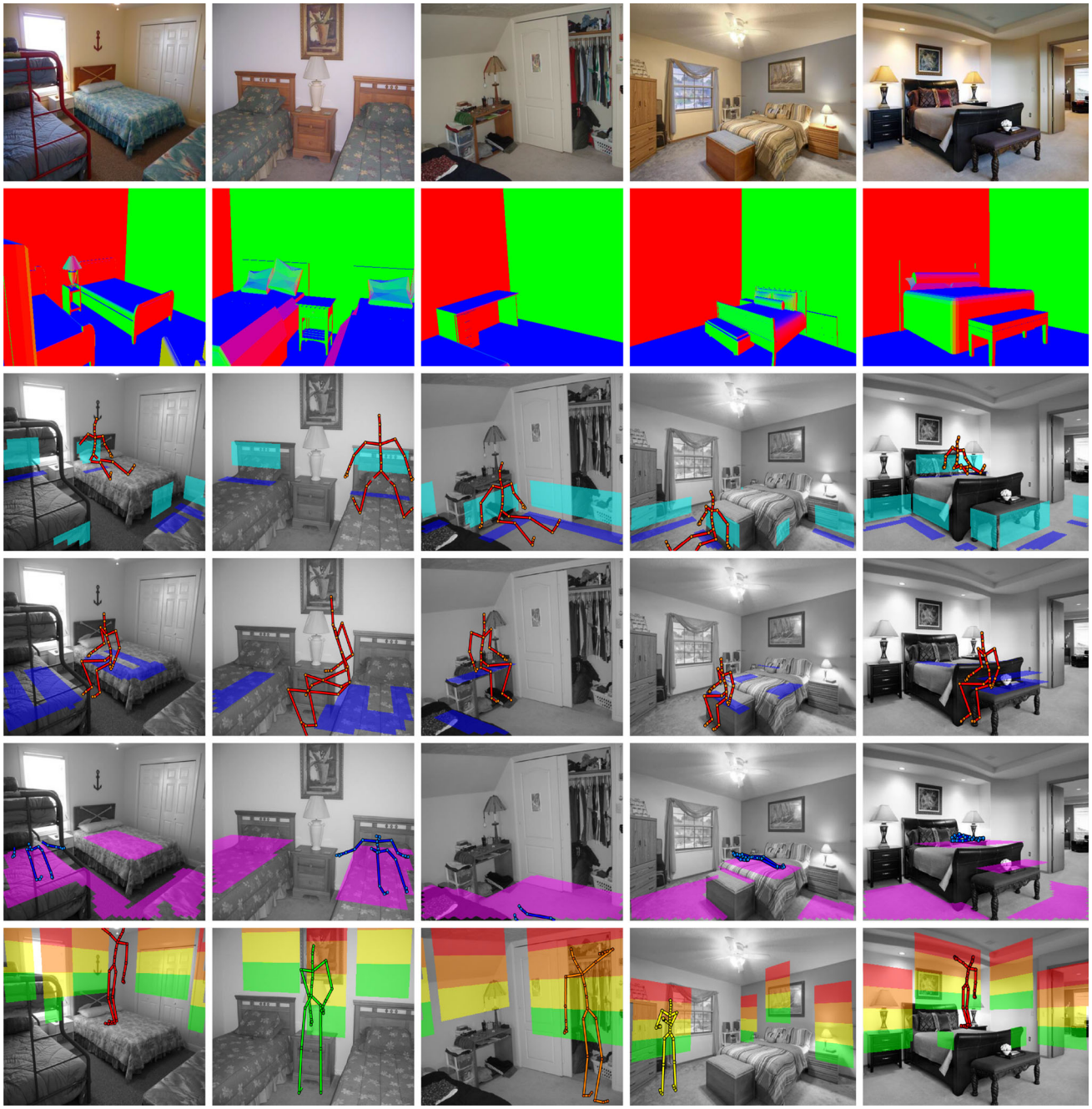


Fig. 28 Qualitative performance of our affordance estimation approach. The images in the *first row* are the input to our algorithm. The *second row* shows our estimated 3D scene geometry. The *third row* shows the possible pelvic joint and back support locations in *blue* and *cyan* respectively for the "sitting reclined" pose. The *fourth row* shows the possible pelvic joint locations in blue for the "sitting upright" pose. The *fifth row* shows the locations where a human's back can rest when "laying down." The *last row* shows the vertical surfaces a person's hand can touch from a standing position for the "reaching" pose, *color coded* to indicate the corresponding pose. Each scene also includes a representative stick figure for each pose

sifier. We manually labeled locations in 50 test images for four poses where:

(a) a pelvic joint can rest while sitting upright,
(b) a pelvic joint can rest while sitting reclined,
(c) a human's back can rest when laying down,
(d) a hand can reach on a vertical surface.

Using these annotations, we compute the pixelwise overlap score between each algorithm's predicted pose locations

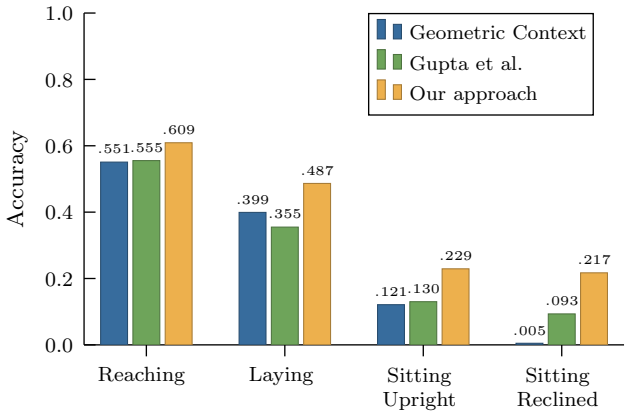


Fig. 29 Quantitative comparison of our affordance estimation algorithm with two baselines

and the ground-truth. Figure 29 shows the performance of our approach compared to the baseline appearance classifier using geometric context, and affordances computed using the coarse geometry estimates from Gupta et al. (2011) on each of the four classes.

Note that our approach outperforms the two baselines for all poses. While the baseline approaches do a decent job in prediction valid locations for "reaching" and "laying down," their performance is markedly lower for the "sitting upright" and "sitting reclined" poses[3]. Intuitively, this is because detecting locations for reaching and laying simply requires identifying the locations of walls and floors in an image, which spatial priors can easily encode. This is akin to easier categories such as ground and sky in pixel classification literature. However, accurately detecting possible sitting poses requires precise estimates of scene geometry and cannot be captured via appearance-based approaches.

9 Application: Geometry-Aware Object Insertion

Many graphics applications involve the insertion of new objects into existing images or videos (e.g., Colburn et al. 2013; Karsch et al. 2011; Lalonde et al. 2007). In order to achieve photorealistic results, the geometry and lighting of a scene must be known (or estimated). Without accurate models of a scene it is impossible to properly reason about how objects should be positioned and oriented, which portions of these objects are visible or occluded, and how complex lighting interactions such as reflections and shadows should be properly rendered. In this section, we explore how 3DNN's ability to automatically estimate the geometry of a scene enables photorealistic object insertion.

Karsch et al. (2011) present a framework which allows a user to quickly annotate the geometry of a scene, and roughly specify the locations of directed light sources. Their system then automatically generates a physical model of the scene including the position, shape and intensity of light sources. Given an estimated scene geometry (via user annotation), the authors present an algorithm to recover the material properties and illumination conditions in the scene. Their method uses intrinsic image decomposition to determine the albedo, direct illumination and indirect illumination of a scene by building upon Grosse et al. (2009)'s Color Retinex algorithm and Guo et al. (2011)'s shadow detection and removal algorithm. After estimating the geometry, lighting and material

(a) Input image.

(b) Synthetically inserted objects.

Fig. 30 Examples of synthetically inserted objects. Note the accurate occlusion and depth ordering of the dresser, and the realistic reflections, **a** input image, **b** synthetically inserted objects

[3] *Note* In Gupta et al. (2011), their approach outperformed the appearance baseline. This is due to the selection bias in creating the dataset for that paper, such that only images with accurate autocalibration estimates from Hedau et al. (2009) were used. However, for these experiments an unbiased sample of images were selected.

(a) input image **(b)** IKEA Grans

(c) IKEA Hemnes **(d)** IKEA Liatorp

(e) input image **(f)** IKEA Lack

(g) IKEA Klubbo **(h)** IKEA Liatorp

Fig. 31 Photorealistic synthetic furniture insertions from the IKEA catalog

properties of the scene, Karsch et al. (2011) allow users to position new 3D objects into the scene which are photorealistically rendered using the additive differential rendering method of Debevec (1998) and the spectral matting algorithm of Levin et al. (2008). The rendered results of their approach are high-fidelity. In fact, the results are often photorealistic enough that humans often cannot differentiate between objects which were originally in the image and objects which were inserted.

This approach to object insertion relies on reconstructing the geometry of the scene. This is done in Karsch et al. (2011) by requiring users to manually annotate each scene's geometry, and recently in Karsch et al. (2014), by matching the input image to RGBD images. Integrating our geometry estimation algorithm provides a way to take advantage of prior, detailed 3D models in estimating the 3D geometry of the scene. Specifically, we can incorporate our automatically estimated camera parameters, surface normals, occlusion masks and depth orderings rather than using human annotations.

Figure 30 shows an example scene with three synthetically inserted objects: a dresser behind the bed, a lamp on one nightstand, and a small kinetic sculpture on the other nightstand. By using our estimated camera parameters, the added objects are automatically scaled and oriented correctly, and the perspective effects of these objects are in correspondence with the scene's vanishing points.[4] The estimated scene geometry enables objects to be automatically positioned to lay upon horizontal surfaces. Moreover, our precise scene geometry enables automatic occlusion reasoning (as seen with the dresser behind the bed). Not only is the scene geometry useful for positioning each object, the estimated surface normals and depths are critical for ensuring proper lighting effects. For example, the reflection of the bed is visible in the inserted mirror, and the light emitted from the lamp is realistically scattered off the edge of the bed.

Our estimated object categories and orientations can also be used to automatically position new objects in a scene at realistic locations using co-occurrence priors. For example, if we know the position and orientation of a couch in a scene, we can infer the most likely location of a coffee table relative to the couch's position.

We now demonstrate a proof-of-concept application which incorporates 3DNN with Karsch et al.'s synthetic rendering algorithm to create an augmented reality product catalog. The goal of this application is to allow users to see how furniture would look in their home. We run our geometry estimation pipeline on images and identify the positions and categories of each object present in the scene. Then, using our known co-occurrence priors, we recommend objects which have a high likelihood of co-occurring with objects currently in the scene. Not only can we recommend what object could be added to a scene, we automatically position and orient the objects relative to other objects in the scene. For example, a coffee table should be centered in front of a couch with approximately 1.5ft in between the objects.

Figure 31 show example scenes for which we automatically insert coffee tables from the IKEA catalog (IKEA 2013), using texture mapped 3D models from Polantis (2010). Note that the coffee tables are automatically sized and positioned at realistic locations in the scenes. In addition, the high-fidelity rendering approach of Karsch et al. (2011) produces photorealistic results with accurate lighting effects (reflections, scattering, shadows, etc.). This application would not be feasible without precise and detailed geometric representations, which traditionally require manual annotation.

10 Closing Thoughts

Recovering the 3D structure of a scene from a single 2D projection is an inherently ill-posed problem, and remains a tremendous challenge for vision researchers. However, we believe it is an important problem to address, as determining the geometry of an environment provides a complete representation which can be used to answer virtually any question about our world, ranging from object categorization and localization to freespace and affordance estimation.

To tackle this problem, we leveraged the recent growth and availability of 3D data to integrate geometric reasoning and machine learning. The methods presented in this paper should not be thought of as a final algorithm, rather they represent a general framework which can be extended and adapted for different tasks. There are four key parts of this framework:

1. A mechanism for aligning 3D data with images via autocalibration
2. A data-driven means for generating geometric hypotheses
3. A means for comparing each hypothesis to an input image via rendering
4. An algorithm for ranking each hypothesis using similarity features

Different environments, such as outdoor settings, would require an alternative mechanism for alignment and similarity features which are tuned for the task; however, the overall 3DNN architecture can be directly adapted.

Acknowledgments This work was supported in part by ONR MURI N000141010934.

[4] *Note* The object insertion renderings included in this section were created by Kevin Karsch at the University of Illinois at Urbana Champaign during a collaboration with the author.

References

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

Baatz, G., Saurer, O., Köser, K., & Pollefeys, M. (2012). Large scale visual geo-localization of images in mountainous terrain. *Proceedings of the European Conference on Computer Vision (ECCV)*.

Baboud, L., Cadik, M., Eisemann, E., & Seidel, H.-P. (2011). Automatic photo-to-terrain alignment for the annotation of mountain pictures. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Bao, S. Y., Sun, M., & Savarese, S. (2010). Toward coherent object detection and scene layout understanding. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Brooks, R. (1981). Symbolic reasoning among 3D models and 2D images. *Artificial Intelligence*, Vol. 17.

Choi, W., Pantofaru, C., & Savarese, S. (2013). Understanding indoor scenes using 3D geometric phrases. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Colburn, A., Agarwala, A., Hertzmann, A., Curless, B., & Cohen, M. F. (2013). Image-based remodeling. *IEEE Transactions on Visualization and Computer Graphics*, *19*(1), 56–66.

Dalal, N., & Triggs, B. (2005). Histograms of oriented gradients for human detection. *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*.

Debevec, P. (1998). Rendering synthetic objects into real scenes: Bridging traditional and image-based graphics with global illumination and high dynamic range photography. *Proceedings of the 25th annual conference on Computer graphics and interactive techniques, SIGGRAPH '98*.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., & Fei-Fei, L. (2009). ImageNet: A Large-Scale Hierarchical Image Database. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Felzenszwalb, P. F., Girshick, R. B., McAllester, D., & Ramanan, D. (2010). Object detection with discriminatively trained part based models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *32*(9), 1627–1645.

Fisher, M., & Hanrahan, P. (2010). Context-based search for 3D models. *ACM Transactions on Graphics*, *29*, 182:1–182:10.

Fisher, M., Savva, M., & Hanrahan, P. (2011). Characterizing structural relationships in scenes using graph Kernels. *ACM Transactions on Graphics*, *30*, 34:1–34:12.

Fouhey, D., Gupta, A., & Hebert, M. (2013). Data-driven 3d primatives. *Submission to the Proceedings of the IEEE International Conference on Computer Vision (ICCV)*.

Fouhey, D.F., Delaitre, V., Gupta, A., Efros, A. A., Laptev, I., & Sivic, J. (2012). People watching: Human actions as a cue for single-view geometry. *Proceedings of the European Conference on Computer Vision (ECCV)*.

Girshick, R.B., Felzenszwalb, P.F., & McAllester, D. (2012). Discriminatively trained deformable part models, release 5. http://people.cs.uchicago.edu/rbg/latent-release5/.

Google Inc. (2000). Google SketchUp. http://sketchup.google.com/.

Grabner, H., Gall, J., & Gool, L.J.V. (2011). What makes a chair a chair? *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1529–1536.

Grimson, W., Lozano-Pérez, T., & Huttenlocher, D. (1990). *Object recognition by computer*. Cambridge: MIT Press.

Grosse, R., Johnson, M. K., Adelson, E. H., & Freeman, W. T. (2009). Ground truth dataset and baseline evaluations for intrinsic image algorithms. *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pp. 2335–2342.

Guo, R., Dai, Q., & Hoiem, D. (2011). Single-image shadow detection and removal using paired regions. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Gupta, A., Efros, A. A., & Hebert, M. (2010). Blocks world revisited: Image understanding using qualitative geometry and mechanics. *Proceedings of the European Conference on Computer Vision (ECCV)*.

Gupta, A., Satkin, S., Efros, A. A., & Hebert, M. (2011). From 3D scene geometry to human workspace. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Hays, J., & Efros, A. A. (2007). Scene completion using millions of photographs. *ACM Transactions on Graphics (SIGGRAPH 2007)*. doi:10.1145/1275808.1276382.

Hays, J., & Efros, A. A. (2008). im2gps: estimating geographic information from a single image. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Hedau, V., Hoiem, D., & Forsyth, D. (2009). Recovering the spatial layout of cluttered rooms. *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*.

Hedau, V., Hoiem, D., & Forsyth, D. (2010). Thinking inside the box: Using appearance models and context based on room geometry. *Proceedings of the European Conference on Computer Vision (ECCV)*.

Hedau, V., Hoiem, D., & Forsyth, D. (2012). Recovering free space of indoor scenes from a single image. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Herbrich, R., Graepel, T., & Obermayer, K. (1999). Support vector learning for ordinal regression. *Proceedings of the International Conference on Artificial Neural Networks (ANN)*, *1*, 97–102.

Hoiem, D., Efros, A. A., & Hebert, M. (2007). Recovering surface layout from an image. *International Journal of Computer Vision (IJCV)*, *75*(1), 151–172.

Hoiem, D., Efros, A. A., & Hebert, M. (2008). Closing the loop on scene interpretation. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

IKEA (2013). IKEA Catalog.

Karsch, K., Hedau, V., Forsyth, D., & Hoiem, D. (2011). Rendering synthetic objects into legacy photographs. *Proceedings of the 2011 SIGGRAPH Asia Conference. ACM*.

Karsch, K., Sunkavalli, K., Hadap, S., Carr, N., Jin, H., Fonte, R., et al. (2014). Automatic scene inference for 3D object compositing. *ACM Transactions on Graphics* (to appear).

Lai, K., & Fox, D. (2009). 3D laser scan classification using web data and domain adaptation. *Proceedings of Robotics: Science and Systems, Seattle, USA*.

Lalonde, J.-F., Hoiem, D., Efros, A. A., Rother, C., Winn, J., & Criminisi, A. (2007). Photo clip art. *ACM Transactions on Graphics (SIGGRAPH 2007)*, *26*(3), 3.

Lee, D., Gupta, A., Hebert, M., & Kanade, T. (2010). Estimating spatial layout of rooms using volumetric reasoning about objects and surfaces. *Proceedings of the Conference on Neural Information Processing Systems (NIPS)*.

Lee, D., Hebert, M., & Kanade, T. (2009). Geometric reasoning for single image structure recovery. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Levin, A., Rav-Acha, A., & Lischinski, D. (2008). Spectral matting. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, *30*(10), 1699–1712.

Li, L.-J., Su, H., Xing, E. P., & Fei-Fei, L. (2010). Object bank: A high-level image representation for scene classification & semantic feature sparsification. *Neural Information Processing Systems (NIPS)*, *2*(3), 5.

Lim, J. J., Pirsiavash, H., & Torralba, A. (2013). Parsing ikea objects: Fine pose estimation. *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*.

Liu, C., Yuen, J., Torralba, A., Sivic, J., & Freeman, W. T. (2008). SIFT flow: Dense correspondence across different scenes. *Proceedings of the European Conference on Computer Vision (ECCV)*.

Liu, M.-Y., Tuzel, O., Veeraraghavan, A., & Chellappa, R. (2010). Fast directional chamfer matching. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Longuet-Higgins, H. C. (1981). A computer algorithm for reconstructing a scene from two projections. *Nature, 293*, 133–135.

Lowe, D. (1987). Three-dimensional object recognition from single two-dimensional images. *Artificial Intelligence*, *31*(3), 355–395.

Microsoft Corporation (2010). Kinect for Xbox 360.

Munoz, D., Bagnell, J. A., & Hebert, M. (2010). Stacked hierarchical labeling. *Proceedings of the European Conference on Computer Vision (ECCV)*.

Oliva, A., & Torralba, A. (2001). Modeling the shape of the scene: A holistic representation of the spatial envelope. *International Journal of Computer Vision (IJCV)*, *42*, 145–175.

Oliva, A., & Torralba, A. (2006). Building the gist of a scene: the role of global image features in recognition. *Progress in Brain Research*, *42*, 145–175.

Pero, L. D., Bowdish, J., Fried, D., Kermgard, B., Hartley, E. & Barnard, K. (2012). Bayesian geometric modeling of indoor scenes. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Pero, L. D., Bowdish, J., Hartley, E., Kermgard, B., & Barnard, K. (2013). Understanding bayesian rooms using composite 3D object models. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Pero, L. D., Guan, J., Brau, E., Schlecht, J., & Barnard, K. (2011). Sampling bedrooms. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Polantis (2010). Polantis 3D Catalog. http://www.polantis.com/ikea.

Ramalingam, S., Bouaziz, S., Sturm, P., & Brand, M. (2010). SKYLINE2GPS: Localization in Urban Canyons using Omni-Skylines. *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*.

Satkin, S. (2013). Data-Driven Geometric Scene Understanding. PhD thesis, Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, August.

Satkin, S., & Hebert, M. (2013). 3DNN: Viewpoint invariant 3D geometry matching for scene understanding. *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*.

Satkin, S., Lin, J., & Hebert, M. (2012). CMU 3D-Annotated Scene Database. http://cmu.satkin.com/bmvc2012/.

Satkin, S., Lin, J., & Hebert, M. (2012). Data-driven scene understanding from 3D models. *Proceedings of the 23rd British Machine Vision Conference (BMVC)*.

Saxena, A., Sun, M., & Ng, A. Y. (2009). Make3D: Learning 3D scene structure from a single still image. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, *31*(5), 824–840.

Schwing, A. G., & Urtasun, R. (2012). Efficient Exact Inference for 3D Indoor Scene Understanding. *Proceedings of the European Conference on Computer Vision (ECCV)*.

Shao, T., Xu, W., Zhou, K., Wang, J., Li, D., & Guo, B. (2012). An interactive approach to semantic modeling of indoor scenes with an rgbd camera. *ACM Transactions on Graphics*, *31*(6), 136:1–136:11.

Silberman, N., Hoiem, D., Kohli, P., & Fergus, R. (2012). Indoor segmentation and support inference from rgbd images. *Proceedings of the European Conference on Computer Vision (ECCV)*.

Sing, G., & Košecká, J. (2013). Nonparametric scene parsing with adaptive feature relevance and semantic context. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Tighe, J., & Lazebnik, S. (2010). Superparsing: Scalable nonparametric image parsing with superpixels. *Proceedings of the European Conference on Computer Vision (ECCV)*.

Torralba, A., Fergus, R., & Freeman, W. T. (Nov. 2008). 80 million tiny images: A large data set for nonparametric object and scene recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, *30*(11), 1958–1970.

Torralba, A., Russell, B. C., & Yuen, J. (2010). Labelme: Online image annotation and applications. *Proceedings of the IEEE*, *98*(8), 1467–1484.

Trimble Inc. (2012). Trimble 3D Warehouse. http://sketchup.google.com/3dwarehouse.

Vondrick, C., Khosla, A., & Malisiewicz, T., & Torralba, A. (2013). Inverting and visualizing features for object detection. MIT Technical Report.

Wang, H., Gould, S., & Koller, D. (2010). Discriminative learning with latent variables for cluttered indoor scene understanding. *Proceedings of the European Conference on Computer Vision (ECCV)*.

Xiao, J., Hays, J., Ehinger, K., Oliva, A., & Torralba, A. (2010). Sun database: Large-scale scene recognition from abbey to zoo. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Yu, S., Zhang, H., & Malik, J. (2008). Inferring spatial layout from a single image via depth-ordered grouping. *Proceedings of the IEEE Workshop on Perceptual Organization in Computer Vision (CVPR-POCV)*.

Zhao, Y., & Zhu, S.-C. (2013). Scene parsing by integrating function, geometry and appearance models. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.