Interactive Images: Cuboid Proxies for Smart Image Manipulation

Youyi Zheng[1] Xiang Chen[1,2] Ming-Ming Cheng[3] Kun Zhou[2] Shi-Min Hu[3] Niloy J. Mitra[4,1]

[1]KAUST [2]Zhejiang Univ. [3]TNList, Tsinghua Univ. [4]University College London

Figure 1: *Starting from single images with segmented interest regions, we generate cuboid proxies for partial scene modeling enabling a range of smart image manipulations. Here, we replace furniture in one image using candidates from other images, while automatically conforming to the non-local relations extracted from the original scene, e.g., the sofas have the same heights, table is correctly placed, etc.*

Abstract

Images are static and lack important depth information about the underlying 3D scenes. We introduce *interactive images* in the context of man-made environments wherein objects are simple and regular, share various non-local relations (e.g., coplanarity, parallelism, etc.), and are often repeated. Our interactive framework creates partial scene reconstructions based on cuboid-proxies with minimal user interaction. It subsequently allows a range of intuitive image edits mimicking real-world behavior, which are otherwise difficult to achieve. Effectively, the user simply provides high-level semantic hints, while our system ensures plausible operations by conforming to the extracted non-local relations. We demonstrate our system on a range of real-world images and validate the plausibility of the results using a user study.

Keywords: image manipulation, non-local relations, scene understanding, cuboid proxies, coupled optimization.

Links: DL PDF WEB VIDEO DATA CODE

1 Introduction

Images are the most popular medium for quickly capturing our surrounding 3D world. This is not surprising given the simplicity of the representation and the ease of acquiring high-resolution inputs. A natural desire is to manipulate such images, mimicking real-world interactions. Traditionally, however, image editing has largely focused on local pixel- or patch-level 2D operations. Only recently have large-scale region-based manipulations have been enabled by exploiting image-space structures and non-local repetitions [Oh et al. 2001; Wang et al. 2008; Barnes et al. 2009; Cheng et al. 2010].

Direct object-level manipulation of images, however, remains elusive. The primary bottleneck is the lack of underlying 3D scene information. Specifically, absence of depth cues makes it challenging to alter perspectives or plausibly resolve inter- and intra-object occlusion and illumination changes that are necessary for realistic object-level manipulations. While an understanding of the scene can remove many such restrictions, automatic 3D reconstruction from a single image unfortunately is ill-posed in absence of priors or user-assistance (see [Sinha et al. 2008; Jiang et al. 2009; Jain et al. 2010]). Hence, existing methods either heavily rely on the availability of suitable priors (e.g., symmetry, 3D models, etc.), or require moderate to heavy user assistance.

In contrast, we humans find it second nature to reason about complex scenes even from single images [Biederman et al. 1982]. One hypothesis is that we recover non-local relations and structures from the underlying scenes and use the information to abstract a plausible mental model of the 3D scene [Norman 1990]. Especially in the context of man-made objects, e.g., tables, chairs, shelving units, etc., such relations and regularities are abundant and often relate to object function and affordance [Gibson 1979]. Our goal is to make images *interactive* using minimal user intervention. To this end, we need to robustly extract non-local relations across scene objects. The main challenge is to extract such relations and calibrate the scene from a single input image simultaneously.

Although high-quality 3D reconstruction from a single image remain difficult, we semi-automatically extract a collection of 3D cuboid proxies to plausibly approximate images of man-made objects where structure and non-local relations are abundant. Intuitively, these proxies along with their non-local mutual geometric relations provide sufficient regularization to recover a partial scene model with minimal user input. While the proxies do not provide an accurate scene reconstruction, we demonstrate that they are sufficient to enable many plausible and powerful object-level image manipulations. For example, in Figure 1, the replaced furniture are automatically adjusted to conform to the relations detected in the input image, e.g., coplanarity across seating surfaces and the tabletop.

First, in the *analysis* phase, the user draws a few strokes to annotate the objects of interest in an input image. We then automatically extract an initial camera calibration and potential proxy edges corresponding to each segment, thus creating an initial collection of upright 3D cuboid-based scene approximations. In a key step, we resolve occlusion and stacking ambiguities using a Markov Random Field (MRF) formulation and propose a joint 2D-3D optimization to conform to the detected non-local relations and repetitions. Note that we *simultaneously* optimize the camera parameters, the proxy attributes, and their orientations. For complex objects, e.g., tables, we propose a decomposition to obtain tighter fitting proxies. Finally, we use the proxies with user hints to estimate a light position and extract shadow regions, and decompose the input image into a background layer and textured 3D proxies along with their mutual relations.

Subsequently, in the *edit* phase, we enable different *smart* object-level manipulations, e.g., translate, rotate, deform, on the input image using the extracted 3D proxies, thus recreating real-world interactions. Specifically, based on the extracted geometric proxies, we expose *only* the useful degrees of freedom. We combine the geometric constraints with human understanding: while the 3D proxies greatly simplify image editing by maintaining non-local relations, the user manipulates the image using her semantic understanding of the scene (see supplementary video and demo). In our interactive framework, user assistance, when needed, is mainly restricted to high-level hints like annotating a missed relation (e.g., indicating that two proxies share a hinge joint in Figure 10). Such relations are easy for humans to infer and greatly increase edit possibilities.

We demonstrate our framework on a range of images and present a number of applications (see supplementary material). We evaluate the plausibility of the manipulations on real-world scenes with ground truth data and also using a user study.

Contributions. In summary, we introduce:

- an interactive image manipulation framework enabling object-level interactions mimicking real-world behavior using image-based scene understanding;
- a joint image- and scene-space optimization to initialize and refine 3D proxies, resolve their occlusion and stacking ordering, and extract their non-local mutual relations and repetitions; and
- intuitive move, deform, and delete operations on input images of man-made scenes while maintaining the extracted relations.

2 Related Work

Traditional image editing. Retouching photographs has a long history starting from traditional darkroom techniques to modern digital touchup tools. Most commercial image editing tools now provide basic support for removing scars, user-guided segmentation, illumination adjustment, patch reshaping or replacement, etc. Complex edits, however, are still tedious and can take hours of manual efforts. State-of-the-art image editing techniques include seamless editing of image regions [Pérez et al. 2003], appearance manipulations [Shapira et al. 2009], image-space retargeting [Rubinstein et al. 2009]. These techniques work on local/regional image features and are (largely) oblivious to underlying image structures in the absence of depth. Hence, plausible handling of perspective effects, occlusion changes, or shading variations for moving and deforming objects remain challenging.

Semantic-based image editing. Preservation of image-space semantics, often identified by user annotations, can lead to powerful edits. For example, Oh et al. [2001] allow users to provide depth annotations and extract layers from a single image and then to use the information for distortion-free cloning and other operations. Hoiem et al. [2005] label image regions of outdoor scenes into ground, vertical, or sky, and then cut-and-fold the images to create 3D photo popups. In order to edit repeated objects in a single image, the RepFinder system [Cheng et al. 2010] extracts a template element from user markings, searches for image space repetitions, and produces plausible depth layering among them. The extracted information is then used towards non-local interactions across the repeating elements.

In a notable effort, the Photo Clip Art system [Lalonde et al. 2007] allows users to directly insert new objects into existing photographs using an image-based approach. Instead of tackling the problem of manipulating objects to change the view, the system simply retrieves a suitable object with desirable attributes from a massive image-based object library. Missing 3D information, however, prevents operations involving changes in perspective, e.g., due to rotation, or handling 3D relations.

Zhou et al. [2010] and Jain et al. [2010] propose data-driven approaches by fitting 3D human models to images and videos, respectively. Subsequently, they map low-parameter morphable models of humans to image-space operations to allow intuitive edits. The approaches are difficult to generalize as they require availability of appropriate 3D models. Instead, we demonstrate that, by working with simple proxies and their geometric inter-relations as constraints, intuitive edits can be performed for a large range of images of man-made scenes.

In another effort, working directly with images, Carroll et al. [2010] treat user annotated lines as projective constraints, e.g., vanishing points, to compute as-shape-preserving-as-possible 2D warps for a variety of powerful image level manipulations (see also the latest Adobe Photoshop vanishing-point plugin). Our focus, however, is to enable scene-level manipulations.

Image-based 3D reconstruction. Reconstructing 3D scenes from images and videos has been intensively studied both in the computer graphics and vision communities (see [Seitz et al. 2006] for a survey). The methods, however, are not applicable for scene reconstruction from single images, which is an ill-posed problem. Instead, single image-based modeling methods rely on detection of vanishing points, availability of user-provided depth annotation, or scene objects being symmetric to perform camera calibration (see [Zisserman et al. 1999; Sinha et al. 2008; Xue et al. 2010]). Such methods are either sensitive to noise or require extensive user interactions.

Alternately, geometric primitives such as parallelepiped or pyramid frustums have been used to calibrate the camera pose and obtain proxy-based 3D reconstruction [Wilczkowiak et al. 2005; Jiang et al. 2009]. In our framework, we use a similar approach for initial calibration and refine the extracted proxies using a joint optimization (see Figure 4). While such 3D proxies are too coarse for accurate scene reconstruction, we demonstrate that they are sufficient, especially with the extracted intra- and the inter-object relations, to act as substitutes for various non-local image edits.

Image understanding. Our work is inspired by the recent advances in image-based scene understanding (e.g., [Saxena et al. 2009; Gupta et al. 2010; Hedau et al. 2010; Gupta et al. 2011]). For example, starting from a single image, the Make3D framework [Saxena et al. 2009] learns the orientation and parameters for a set of planar proxies to best explain the input image. In another attempt, Gupta and colleagues use simple user annotations to model a scene as a collection of candidate axis-aligned boxes. In the context of indoor scenes, they perform a joint analysis of the extracted box-based scene geometry and human motion data to predict human poses

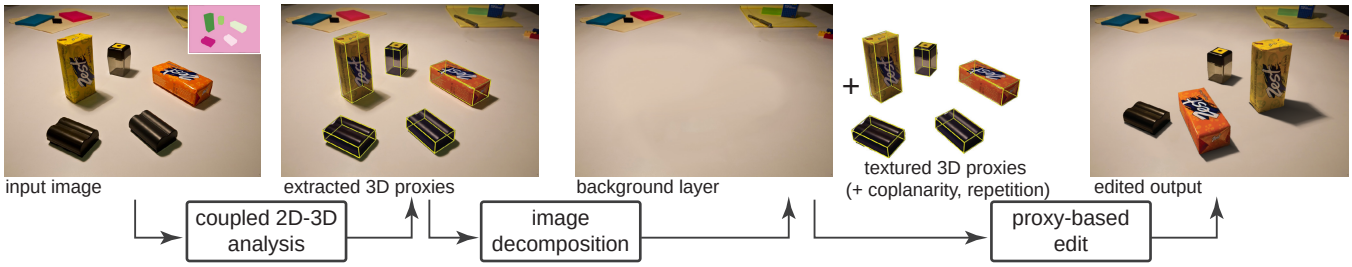

Figure 2: *Interactive images: We create cuboid-based proxies to represent segmented objects of a single image, decompose the input into a background image and textured proxies, and enable a range of intuitive interaction possibilities.*

and region annotation from a single image, e.g., a person sitting on a couch. In the context of outdoor scenes, they use physical constraints, e.g., statics, volumetric, occlusion, etc., to decompose a facade image into an axis-aligned box layout. The algorithms look for coarse axis-aligned boxes or planar proxies that are sufficient for their target applications, but such proxies are too coarse for our target manipulations (see Figure 15 and supplementary material).

Although significant progress has been made in image understanding, the findings have rarely been used for computer graphics applications. In one exception, Karch et al. [2011] use box-level proxies to extract appearance properties from a single image using a non-linear optimization. Subsequently they demonstrate powerful illumination effects by rendering synthetically introduced 3D objects using the estimated boxes to evaluate shading and shadow effects. They focus on appearance and rendering effects, while we focus on geometric edits (e.g., move, deform, and delete objects) that require much more structured 3D information. Further, we do not need a Manhattan assumption requiring objects to be axis aligned.

Image completion. Starting from early influential efforts on texture synthesis [Efros and Leung 1999] and image inpainting, various methods have been proposed for image completion [Hays and Efros 2007; Wei et al. 2009]. While most approaches focus on pixel- or patch-level operations, Sun et al. [2005] use manual annotations of image-space structures as guides to achieve significantly improved completion results. In our framework, we use the extracted 3D structures for better image completion and occlusion handling while using PatchMatch [Barnes et al. 2009] for image completion.

3 System Overview

Our system takes as input a single image and is applicable to those regions that can be approximated using a collection of simple 3D primitives, e.g., office furniture, other man-made objects. Further, we assume that: (i) objects rest on some dominant horizontal surface (e.g., a tabletop or floor plane) or they are stacked on the surface, (ii) one or more cuboid-like objects are available to help with initial calibration, (iii) the scene objects are opaque, and (iv) shadows appear primarily due to a single-point (or directional) light source.

User experience. Our goal is to create an image manipulation system that is fast, simple, robust, and, most importantly, *smart*, i.e., mimics real-world experience in response to user interactions. For this purpose, we exploit non-local relations and repetitions across objects. The user loads an image and segments out respective regions using GrabCut [Rother et al. 2004]. Then, the system automatically extracts a background layer, textured 3D proxies, and their mutual relations. Although the proposed method is automatic, we support *override* possibilities, i.e., when any stage fails, the user can intervene to resolve ambiguities using corrective high-level annotations, e.g., deleting false edges, adjusting corner points, etc. (see Section 7). Subsequently, the user interacts with the image in real-time (see supplementary video and demo).

Our framework runs in three stages (see Figure 2): (i) scene analysis, (ii) image decomposition, and (iii) interactive editing.

(i) Scene analysis. We start with user-annotated rough regions of interest $\{R_1, \ldots, R_n\}$ of an input image such that each region corresponds to one object of interest. For each region R_i, we extract candidate corner points by analyzing (the convex hull of) the segmentation boundary of R_i. Using the extracted corner points, we automatically create initial cuboid proxies for the image objects along with a camera calibration. Next, in an important phase, we propose an iterative joint optimization to progressively align the projected 3D proxies to the image-level edges and use an MRF formulation to resolve occlusion and stacking, while also extracting and conforming to extracted scene-level relations, e.g., placement, coplanarity, repetition, etc. Specifically, we refine the proxy parameters, their orientations, and also the camera parameters, while simultaneously resolving ambiguities using a coupled proxy analysis. For example, in Figure 6, we resolve the occlusion ordering and stacking of the blocks, while concurrently optimizing the proxy attributes and camera parameters.

The extracted proxies can still have large voids for non-convex objects, e.g., chairs, shelves, etc. Based on the observation that man-made objects are usually fabricated using parts aligned to object boundaries (i.e., bounding box faces), we propose an automatic decomposition procedure (see Figure 7). We use a heuristic to generate a set of cutting planes based on the analysis of the segmentation contours aligned with the vanishing points to decompose the 3D proxies using parallel faces. Subsequently, we greedily remove sub-cuboids that lie outside the respective segmentation regions. In ambiguous cases, we allow the user to intervene.

Certain important functional constraints of man-made objects are often invisible, e.g., hinges of a door, or the sliding mechanism of a drawer. Such relations, however, amount to identifying the proxies that are connected, and then reasoning in 3D to parameterize the connections. We allow the user to annotate such semantic relations by simply identifying proxy pairs, while the system optimizes for the respective joint attributes (see Figure 10 and supplementary video). Intuitively, the user resolves the ambiguity in the data through simple strokes, while the underlying optimization responds smartly to create a cuboid-based scene representation.

(ii) Image decomposition. We expect the user to roughly indicate shadow correspondences for two (or more) objects. We then use the extracted 3D proxies to recover a light position and determine the shadow boundaries. Finally, we decompose the scene into (a) a background layer by removing the proxies and their estimated shadows and filling in the missing parts, (b) 3D textured proxies for the individual regions with hidden parts plausibly filled in using symmetry, repetitions, etc., and (c) a set of relations among the proxies and with the ground plane.

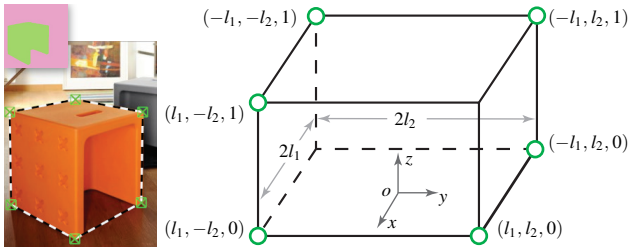

Figure 3: *Based on detected corners (green crosses) of the extracted hexagon boundary (left), we assign correspondences to a 3D upright cuboid parameterized by l_1 and l_2. The information is used to obtain an initial camera calibration and then to create initial proxies for other segmented regions in the input (see Figure 4).*

(iii) Interactive editing. The extracted 3D proxy-level scene understanding enables several possibilities: (a) moving objects using translation or rotation while conforming to placement constraints, e.g., a box is moved on the table, (b) deleting objects, (c) performing coupled non-local edits, and (d) introducing objects from other images (see [Karsch et al. 2011] who introduce synthetic 3D objects). During interactions, we propagate constraints across the proxies to preserve both the intra-object characteristics as well as their non-local mutual relations (see also [Gal et al. 2009; Yang et al. 2011]). For example, when the face of a table is lifted up, all the objects lying on it are accordingly lifted up, while the sizes of the individual objects remain unchanged (see Figure 12). Note that effects due to contacts, occlusion, shadow, and perspective are naturally handled in our system.

4 Coupled 2D-3D Analysis

Many image manipulation tasks are greatly simplified by having even an approximate 3D model (of relevant parts) of the underlying scene. Although significant progress has been made in automatic scene understanding from a single image, we need 3D proxies with higher accuracy for image manipulations. We now introduce an algorithm to extract robust cuboid-based 3D geometry for images of man-made objects with minimal user interaction.

a) Initial camera calibration. For any segment R_i, the goal is to fit a 3D box-proxy. Since the scene objects are assumed to be opaque, we expect the convex hull of the boundary $CH(\partial R_i)$ to have six corner points. We use dynamic programming to break $CH(\partial R_i)$ into six parts such that the resultant curves are best approximated by line segments (see Figure 3). Let the corresponding end points of the curves be $p_i^0, \ldots, p_i^5$, which we then approximate as a hexagon H_i. Note that outliers from occlusions are detected and handled later.

We align the edges of H_i to the image segmentation to reduce the fitting error in 2D using an iterated closest point (ICP)-like method. Specifically, for each point on $CH(\partial R_i)$ we find its closest edge in H_i. Thus, each edge in H_i becomes associated with a set of segmentation boundary points. We then apply least-squares line fitting to the points associated with the hexagon edges to refine the estimate for H_i. We iterate for a fixed number of steps (3 in our setting). Next, we solve for a cuboid-proxy along with a camera calibration such that the projection fits the image information, i.e., it agrees with H_i.

For initial camera calibration, we use geometric primitives (see also [Wilczkowiak et al. 2005]). Specifically, we assume that each segment is the image of a cuboid. Hence, we use the extracted corner points of segment R_i to fit a cuboid of unknown dimensions l_1, l_2 (assuming unit height). We take the world origin as the center of the base-face of the cuboid and the world axes to be aligned with the cuboid axes (see Figure 3). We use the possible correspondence (note there are multiple possible assignments) between the extracted corners in 2D and the cuboid corners in 3D to determine the camera parameters, and l_1, l_2. A common camera projection matrix model is described as $M_{3\times4} := K[R|t]$, where K is the intrinsic camera matrix ($K \in \mathbb{R}^3$, since we vary u, v, f) and $R, t \in \mathbb{R}^3$ represent the camera position. Working with homogeneous coordinates, the projection matrix maps each cuboid corner P^j to a corresponding image point p_i^j, i.e., $p_i^j \simeq MP^j$, where $\simeq$ means equality up to a scale. Using the six correspondence pairs $P^j \rightarrow p_i^j$, we solve for 11 unknowns in K, R, t, l_1, l_2 using a linear program (see [Hartley and Zisserman 2006; Jiang et al. 2009] for details). We retain the best solution, provided the system of equations is non-degenerate. Note that each region can independently produce one camera calibration. Next, under each camera, we create initial estimates of cuboid-proxies for the other scene objects. For each camera along with the estimated proxies, we perform a joint optimization to construct an aligned scene and select the one with the least fitting error.

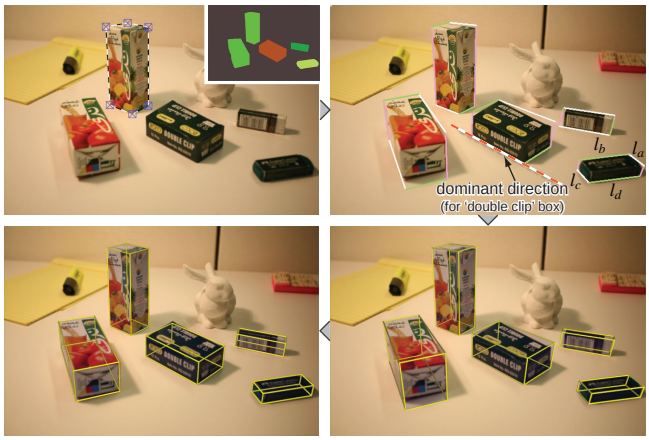

Figure 4: *Starting from a rough camera calibration that is extracted based on a segment boundary (top-left), for each region R_i we identify a dominant line direction (top-right). We use these directions along with the current xy-vanishing line estimate to produce initial cuboid proxies (bottom-right). We use a joint optimization to simultaneously refine the camera parameters, the proxy attributes, and the extracted non-local relations across the proxies (bottom-left). Note that the proxies need not be axis-aligned.*

b) Initial scene estimation. Using the camera projection matrix, we compute the vanishing points along the z-direction as well as the vanishing line corresponding to the xy-plane [Zisserman et al. 1999]. We observe that any *upright* cuboid is uniquely determined once we have three pairs of parallel edges on its silhouette. Further, we make use of the assumption that most objects lie on the ground plane, i.e., xy-plane. Next, we extract proxies for *all* the individual segments in the image (with respect to each candidate camera).

For hexagon H_i of each segment, we identify the line segments along the vertical direction. Among the remaining four line segments, say ordered as l_a, l_b, l_c, l_d, we select the pair (l_a, l_c) or (l_b, l_d) that better agrees with the xy-vanishing line as the dominant direction d_1 for the particular object. Using d_1, we then determine the other direction d_2 on the xy-plane as the direction orthogonal to d_1. Finally, along the directions d_1, d_2, and $d_3 = z$, we get rays, two for each direction, that touch the object segmentation boundary (see Figure 4). We use these rays to extract the object boundary as the refined hexagon.

If the object in R_i is (approximately) a cuboid without any occlu-

sion, then the hexagon agrees with $CH(\partial R_i)$. Objects, however, can be occluded. We use the hexagon to estimate such occluded corners. Based on the observation that any triplet of adjacent edges of a hexagon corresponds to cuboid axes and uniquely determines a candidate cuboid, we iterate through all such triplets to create a set of candidate cuboids, say $\{P_i\}$ (see supplementary video). Note that at this stage, all candidate proxies are assumed to rest on the ground and possible stackings are handled later. As an initial estimate, we retain the best matching proxy based on the (minimum) deviation from the hexagon edges H and (the convex hull of) the segmentation R_i measured as (see Figure 5):

$$E_f(R_i, H) := \frac{\#(R_i \setminus H) + \#(\text{void } H)}{\#(R_i)},$$

where $\#(\cdot)$ denotes the area of the polygon and $(\text{void } H)$ denotes the area inside hexagon H that is not covered by any segment.

Thus, based on each segment R_i, we get a camera calibration and a set of scene proxies. We select the combination with the minimum deviation score. Note when estimated from an occluded region, the camera matrix poorly explains the other regions and is discarded. Even then, any error in the initial camera calibration results in the deviation of the fitted proxies (see Figure 4). Next, to reduce bias from the selection of the base proxy, we jointly optimize the camera and the box parameters.

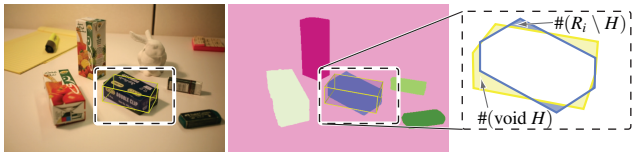

Figure 5: *Measuring error $E_f(R_i, H)$ between region R_i (in blue) and hexagon H (in yellow), i.e., projected proxy $\tilde{P}_i$.*

c) Joint optimization. In order to minimize the total fitting error (i.e., $\sum_i E_f(R_i, H_i)$), we introduce an iterative approach to progressively align the 3D scene to the image scene. Let us denote the new hexagon (in 2D) for each cuboid B_i as H_i^*. We now refine the scene estimate by accounting for both image features and the 3D scene structure. Each cuboid has six parameters, namely $\{l_1^i, l_2^i, l_3^i, t_x^i, t_y^i, \theta^i\}$. Note that, for the base proxy, we have $l_3^0 = 1.0, t_x^0 = t_y^0 = \theta^0 = 0$, while for the others (t_x^i, t_y^i, θ^i) we denote the offset and orientation of the cuboid B_i vector with respect to the base cuboid B_0 on the xy-plane. Thus, each cuboid adds 11 3D→2D constraints (six constraints from corners with scale ambiguity). We then optimize the camera parameters together with the parameters of each 3D cuboid in a non-linear optimization whose objective function is the reprojection error of each (visible) cuboid point with respect to its corresponding image point, i.e., corners of H_i. Since we have a good estimate of the scene (see part-b), we optimize using the Levenberg-Marquardt method for one or two iterations using Levmar [Lourakis 2004] (see Figure 4).

d) Resolving ambiguities. We now consider mutual relations across proxies (e.g., occlusion, stacking) and resolve the ambiguities using a joint formulation. We describe how to resolve occlusions by extracting a plausible ordering by jointly reasoning in 2D and 3D (note that the camera remains fixed in this stage).

Candidate generation. Let the set of initial scene proxies be $S := \{\{P_1\}, ..., \{P_n\}\}$ corresponding to the n regions, assuming that all the proxies rest on the ground plane (later we describe how to handle stacking). Note that each region can produce multiple candidate proxies $\{P_i\}$ as described in part-b. We mark regions R_i and R_j as potentially *occluding*, if, for some $a \in \{P_i\}$ and $b \in \{P_j\}$, the image-space projections of a, b overlap. This significantly prunes

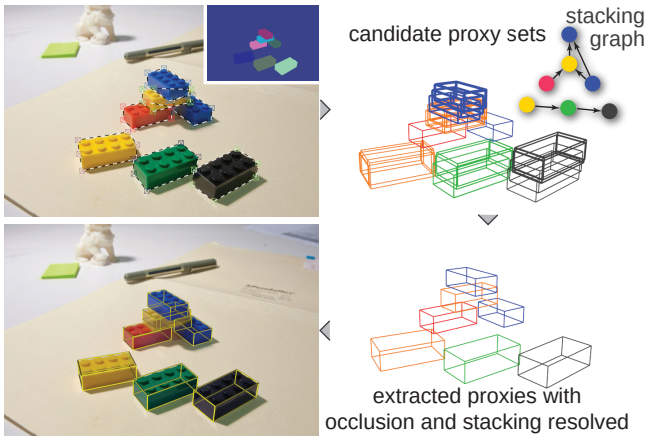

Figure 6: *In the case of occlusion and stacking among objects in segmented regions, we reason with pairs of regions, leading to a binary term, and solve for a consistent assignment via an MRF formulation.*

the candidate set (see supplementary video). Let $\mathcal{P}$ denote the set of all such pairs of proxies.

MRF formulation. For each region R_i only a single proxy should be selected from $\{P_i\}$. The selection, however, depends on global effects, and hence all the regions have to be considered simultaneously. Effectively, we have a labeling problem in which the selection of each proxy from $\{P_i\}$ depends on a certain data cost, i.e., how well the proxy explains the corresponding region, and the selection of pairs depends on a mutual-cost due to occlusions. Since the general labeling problem is difficult, we simultaneously optimize over the label possibilities in a Markov Random Field (MRF) formulation. We define the optimization energy as follows:

$$E_l := \sum_i u(R_i \to l_i) + \beta \sum_{(i,j)\in\mathcal{P}} c(R_i \to l_i, R_j \to l_j), \qquad (1)$$

where $u(.)$ is the unary cost, $c(.)$ is the pairwise cost, β is the weighting factor (set to 10 in our tests) and l_i is the labeling function denoting which proxy among $\{P_i\}$ is selected for R_i.

The unary term measures the cost of selecting a candidate among the $|\{P_i\}|$ possibilities. We measure it based on how well a proxy fits the underlying segmentation boundary, with a lower score denoting a better fit. In particular, for each $l_i \in \{1, \ldots, |\{P_i\}|\}$, we compute $u(\cdot)$ as follows:

$$u(R_i \to l_i) := E_f(R_i, \tilde{P}_i), \qquad (2)$$

where $\tilde{P}_i$ denotes the proxy P_i projected to the image space using the camera information.

For the pairwise cost, we measure if the occlusion ordering derived from the proxies agrees with that observed in the image, i.e., if at pixel p, $\tilde{P}_i$ occludes $\tilde{P}_j$, then $p \in R_i \setminus R_j$. We capture this as,

$$c(P_i \to l_i, P_j \to l_j) := e^{r_1} e^{r_2} - 1 \quad \text{where,} \qquad (3)$$

$$r_1 = \frac{\#(R_i \cap (\tilde{P}_j \text{ occ } \tilde{P}_i))}{\#(R_i)}, \quad r_2 = \frac{\#(R_j \cap (\tilde{P}_i \text{ occ } \tilde{P}_j))}{\#(R_j)},$$

and $(\tilde{P}_j \text{ occ } \tilde{P}_i)$ denotes the pixels where $\tilde{P}_j$ occludes $\tilde{P}_i$ (using corresponding depth information). Again, a lower value denotes better agreement with 3D proxies, i.e., $r_1 = r_2 = 0$ if the projected proxies agree with the corresponding region labels. Note that unlike scene understanding scenario (see [Gupta et al. 2011] and references therein), we benefit from having segmentation information and hence can obtain tighter 3D proxies without restriction to axis-aligned scenes.

Stacking proxies. Until now, we assumed that all the proxies rest on the ground. In order to handle stacking, we perform a simple modification: When regions R_i and R_j are marked to interfere, we spawn new candidate proxies appropriately. For example, for each $b \in \{P_j\}$, we create new proxies for R_i assuming that they rest on b, i.e., their base is coplanar to the top of b, and add them to $\{P_i\}$. Similarly, we create new candidates assuming that they rest on top of $a \in \{P_i\}$. The optimization energy is computed similarly as before (see Equation 1).

When there are more than three objects in a stack, the size of the generated candidates can become very large (it is exponential to number of stacking objects). We use a simple heuristic to trim the candidate set: Given two stacked objects, we leverage the corresponding image segments to eliminate ambiguity since we have the camera information. In particular, suppose that A lies on B. Then, any pixels of segment of A will not be occluded by B (using their hexagons). This excludes the option of B being on A. We build a graph with each directed edge $A \rightarrow B$ indicating the possibility that A lies on B. With this graph, we can remove such obviously bad candidates and also infer the maximum level of stacking (see Figure 6).

Label assignment. We extract the best labeling by minimizing Equation 1 over all possible choices, i.e., $\Pi_i |\{P_i\}|$. In our examples, since the number of such possibilities is limited, we exhaustively search the solution space, e.g., in the toy bricks example, we test 80 labeling possibilities (after pruning). Advanced optimization techniques like genetic programming or MRF optimization can alternately be used.

e) Global optimization. Having extracted a consistent set of 3D cuboid proxies, we next look for relations across them. We consider placement, coplanarity among faces, equal attributes, and also repetitions (other relations can be incorporated, if desired). Since we work with cuboids, such relations are easily extracted by comparing the length, width, and height of the proxies, with equivalence established using a default threshold of 1% of the box's diagonal length. Although this worked well in our examples, user intervention may be necessary in other images. We add the extracted relations as (non-local) constraints and re-estimate the scene and camera as in part-c. Thus, at this stage, we have a (globally) calibrated camera along with a consistent set of optimized proxies with the inter-proxy occlusion and stacking resolved and their mutual relations extracted.

f) Removing void sub-cuboids. In order to model non-convex man-made objects, certain sub-volumes of the corresponding box-proxies should be empty. Hence, we first decompose a cuboid into sub-cuboids by using cutting planes aligned with the faces of the cuboid, and then remove the redundant sub-cuboids (see Figure 7).

Cutting plane identification. For each region R_i, we obtain three sets of vanishing points using the projected box proxy $\tilde{P}_i$. We then split the boundary ∂R_i (instead of its convex hull) into line segments using dynamic programming. We align each such line segments to the closest extracted direction (measured using the image-space angle). For example see yellow/blue lines in Figure 7-left.

We use the extracted line segments to generate a set of potential candidate cutting planes aligned to the proxy faces. Each extracted line segment l_j produces cutting planes along two directions. We ignore those planes that coincide with a cuboid face since they do not cut the cuboid. For each extracted edge l_j, if there are multiple candidate planes along a particular direction, we retain the one that covers the largest area of R_i. The cutting planes decompose the cuboid into sub-cuboids, say Ω.

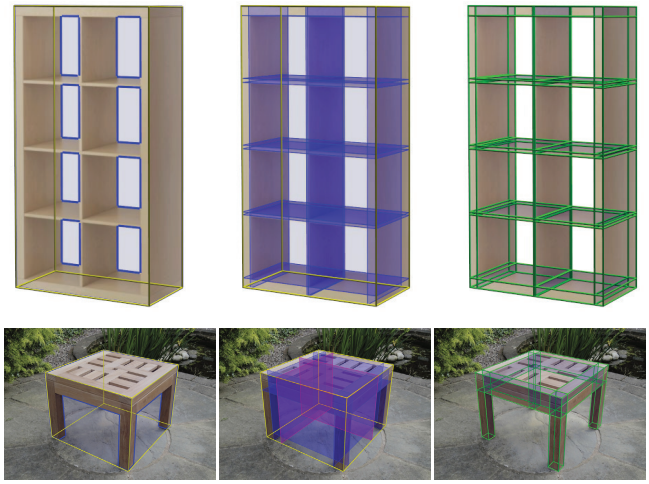

Figure 7: *For non-convex objects, we use the boundary segment lines to generate cutting planes aligned to the proxy faces to decompose the proxy. We then remove void sub-cuboids using a greedy strategy. In complex cases, user intervention may be required.*

Redundancy removal. We use a greedy algorithm to remove redundant sub-cuboids from the candidate set. We assign a score to each sub-cuboid in Ω based on the number of object segmentation pixels the projected sub-cuboid covers: the larger the score, the smaller the possibility of the sub-cuboid being redundant. Let Ω_o denote the unremovable sub-cuboids, each of which *solely* covers certain parts of R_i. We then greedily select the element of $\Omega \setminus \Omega_o$ with the smallest score and delete it if the removal does not split the existing sub-cuboids into disconnected components. Finally, we perform non-local relations detection to identify sub-cuboids that are coplanar, regularly spaced, or of equal thickness and that conform to the realtions as described in part-e. Again, since we only have box proxies, we simply detect such relations by comparing their attributes.

For complex objects, e.g., sofa, or objects with curved profiles, the automatic decomposition can fail. We then allow the user to manually add or remove some cutting planes (e.g., the user removes the red planes in Figure 7; see also the supplementary video).

5 Image Decomposition

In this stage, we use the extracted 3D proxies to create image layers. First, we estimate the light position and inter-proxy shadows. We then use the information to identify texture regions to fill in (due to occlusion or shadows), complete such regions, and decompose the input image into a background layer and textured 3D proxies along with their extracted mutual relations (see Figure 9).

The user marks two or more 3D proxies and shadow correspondences, and then we automatically estimate the light region and shadow polygons, as described next. Note that although automatic (data-driven) shadow detection algorithms are available [Guo et al. 2011], we found them not to be suitable for complex scenes.

Note that any user-marked shadow correspondence is of the form (proxy corner $\rightarrow$ ground plane point). Each such correspondence defines a ray in 3D. For each annotated region R_i, we use two such rays to estimate a 3D light position, say l_i, as the point with the minimum sum of the squared distance to the rays (in the least squares sense). We use the light position l_i to get a shadow boundary S_i (assumed to be on the ground) for the 3D proxy P_i in each region R_i. Now we look for a single light position l_p that best ex-

Figure 8: *The collection of cuboid-proxies provides intuitive manipulation possibilities. (Left-to-right) We can translate, rotate, scale, free-form deform a proxy; change their stacking and relative ordering; or perform discrete edits or insert new scene objects, while preserving original non-local relations, e.g., objects remain on the ground plane (see Figures 6 and 7 for original images).*

plains all S_i-s. Based on P_i and l_p, we parameterize the shadow polygon on the ground as $f(P_i, l_p)$. We refine the light estimate l_p such that $f(P_i, l_p) \approx S_i$ (in a least squares sense) using an iterative ICP-like method, alternately assigning correspondence between $f(P_i, l_p) \leftrightarrow S_i$ and estimating l_p. We run the optimization initialized by $l_p \leftarrow l_i$ and take the best solution. Experimentally, we found this approach to provide satisfactory shadow and light estimates even for complex scenes.

To create the background layer, we take the union of the projected proxy and the refined shadow, dilate (by 10-20 pixels) and remove the region, and perform image completion [Barnes et al. 2009] (see Figure 9). For each of the proxies, we perform two steps: (i) Based on the light position l_p, we estimate the inter-proxy shadow effects and perform image completion to recover the parts in the shadows. (ii) Further, for invisible faces (due to occlusion or backfacing), we use symmetric regions to fill in missing texture. Note that for symmetry we simply use information from the same proxy (reflectional symmetry of the cuboid), although more complex texture-based similarity detection can be performed.

Finally, along with the textured proxies, we also store the mutual relations among them (e.g., resting on the ground, repetitions, coplanarity, etc.) and use the information for constrained editing.

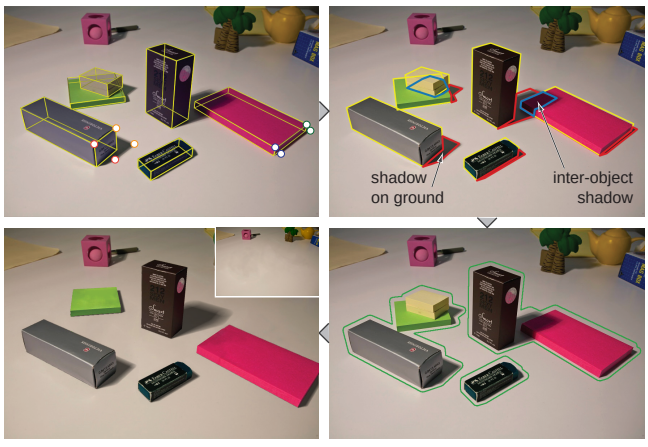

Figure 9: *The user marks a few correspondences across proxy corners and shadow corners (top-left), which are then used to initialize and optimize the light position and shadow boundaries. The extracted shadows (top-right) can either be ground plane shadows (in red), or inter-proxy shadows (in blue). We extract the object segments and associated shadows (in green, bottom-right) and use image completion to create a completed background layer and restore parts of the proxies in the shadows (bottom-left).*

6 Proxy-based Edits

We enable smart object-level image manipulations using the extracted 3D proxies and their relations. Recall that each region R_i is represented by a single cuboid, or a set of sub-cuboids $\{P_i\}$ with their intra-cuboid relations. Further, we have a set of non-local relations across proxies from different regions. In our framework, we consider placement, repetition, alignment (e.g., cuboids are vertically aligned), coplanarity, and parallelism, prioritized in decreasing order.

Interaction possibilities. Given the extracted 3D proxies, we realize basic operations with objects, e.g., translate, rotate, scale, or stack, to mimic real-world experience (see Figure 8). We support real-time interactions while optional collision handling (using the open-source physics simulator Box2D) and displaying simple shadow effects (see the supplementary video). Advanced appearance and rendering effects can possibly be integrated (see [Karsch et al. 2011]).

Importantly, we allow the user to manipulate objects while respecting the extracted relations. The user initiates an edit using a selected single cuboid, and then we automatically propagate the edit to the other proxies in the scene. We use the propagation strategy of [Gal et al. 2009; Zheng et al. 2011], which is fast and suitable for interactive performance. Note that during any interaction, the ground plane remains fixed, and objects placed on the ground are constrained such that $z = 0$ for their base plane, unless we lift the objects. The edit propagation happens in two stages: (i) for intra-object propagation among the sub-cuboids, say $\{P_1\}$, we directly use the iWires framework [2009]; (ii) for inter-object propagation, we design a simple symmetry-and-proximity based extension, as described next.

Let, $\Phi \leftarrow \{P_1\}$ be the set of sub-cuboids that are already handled in the intra-object propagation stage. Next, we find the proxy set $\{P_i\}$ with the closest proxy to elements in Φ and update all the elements

Figure 10: *Users can provide high-level annotations: (top) the two proxies are marked to share a hinge joint; (bottom) the drawers are marked to be sliding inside the bed frame. Our framework computes the necessary joint attributes based on proxy geometry, making the images interactive (see the supplementary video and demo).*

Figure 11: *Repeated edits on building examples showing both discrete and continuous changes.*

of $\{P_i\}$ according to their relations to elements in Φ. (Conflicts, if any, are resolved using the priority ordering, and ties are broken arbitrarily.) For example, if a face $f \in \Phi$ is coplanar to $g \in \{P_i\}$, then lifting f in the intra-object propagation lifts g in the inter-proxy stage to restore coplanarity. Effectively, edits affecting Φ act as deformation handles to proxies in $\{P_i\}$ and then we again apply intra-object propagation to the remaining elements in $\{P_i\}$ to maintain their original shape characteristics. Note that unless indicated by users, we do not modify proxy sizes. We then add $\Phi \leftarrow \Phi \cup \{P_i\}$ and continue the propagation until there is no remaining proxy set to handle. Note that the process stops as soon as all related proxies are touched once. Further, for proxy sets that are repeated, the edits are copied over up to corresponding repetition transforms (see Figures 10, 12, and the supplementary video).

7 Evaluation

We tested our implementation on a variety of input images. We use OpenCV for basic image processing operations and PatchMatch for image completion as necessary. The core implementation efforts were in the analysis stage, which involves simultaneous optimization over multiple variables (using Levmar). Typically, for a scene involving 5-10 annotated objects, the optimization runs in a couple of seconds, since we have good initialization extracted from the individual regions (see parts 4a and 4b). The MRF optimization for resolving occlusion and stacking, however, can take longer depending on the complexity of the scene and the size of the candidate sets.

Figure 12: *Edit propagation: When we lift the tabletop, the other tables also are vertically scaled to maintain coplanarity, while also retaining contact to the ground. Subsequently, objects resting, i.e., the placement relation, on the tables also are raised. However, when a table is rotated, only the objects stacked on it are affected.*

We achieved interactive results even with up to 100 candidates (see the supplementary video and demo).

User assistance. Starting from (roughly) segmented images, our system works in the automatic mode for simple scenes. For complex scenes or regions with non-boxy shapes, we expect user intervention. Such interactions can be categorized as: (i) adjusting initial estimates for the cuboid corners (Figures 1, 11-left, 14, 16-bottom), (ii) helping with identifying image lines for decomposition (Figures 1, 7-bottom, 15-bottom-left, 14), and (iii) annotating types of joints between proxies (see Figure 10). Note that due to our joint optimization and MRF formulation, rough user strokes are sufficient to initialize the global optimization, which then adjusts the annotations. Even in complex examples (Figures 1 and 14), user interaction was limited to 1-2 minutes, including GrabCut strokes (leaving out the image browsing time).

Comparison with ground truth. We compared our manipulation results with ground truth to evaluate the accuracy and plausibility of the manipulated images. In Figure 13, we set up a scene with objects of various difficulty depending on how well they can be approximated by cuboids and on how symmetric their textures are. When the objects are close to cuboids, we get high-quality models and shadows, which are difficult to quickly differentiate from ground truth photographs. In the case of the skull, however, the shadow is visibly boxy (under harsh light) even though the shape itself follows the ground truth quite well. If we rotate the skull, however, then the illusion quickly breaks down. Further, in the case of semi-transparent objects, we found that although translations and small rotations work plausibly, with moderate rotations the warped transparent interior causes artifacts (see the sharpener in Figure 2).

Image comparison. In the above example, the original images

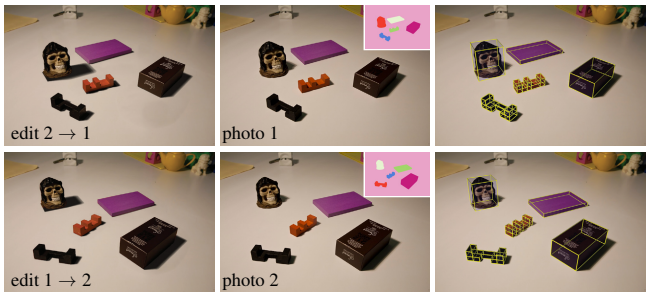

Figure 13: *Comparing proxy-based editing of images to ground truth manipulation in 3D. Note that the object poses and their shadows are realistic with user interaction being limited to GrabCut-based segmentation in the analysis phase and shadow correspondence annotation. The skull is badly approximated by a cuboid proxy resulting in the blocky shadow under strong light.*

Figure 14: *Virtual shopping and room redesign. Starting from a single image of a room, and images collected from product catalogs, the user can quickly visualize how the objects will look in her room. Note that objects are automatically resized based on original scene proxies, while individual objects can be rotated, translated as desired. In this example, the user spends less than a minute to rig up the proxies.*

have high pixel-level differences, i.e., the mean squared error $\text{MSE}(1,2) = 0.17$ (with color channels in the range of $0-1$). In contrast, $\text{MSE}(1, 2 \to 1) = .018$ and $\text{MSE}(2, 1 \to 2) = .02$ indicating that content of the two images are very similar if we factor out (ground) plane translations and rotations that do not affect the scene-space relations. This hints at the possibility of a new image-space similarity measure to factor out variations due to geometric arrangements, closer to our semantic perception of scenes.

User interaction. Since we preserve extracted relations across the proxies, effectively, only the useful degrees of interaction are exposed to the user. For example, a translation is mapped to translating the selected object while preserving coplanar relations, or (optional) handling of collisions as we move parts around, or translating a drawer results in only opening/closing the same (see the supplementary video and demo).

User study. In order to evaluate the effectiveness of our system, we asked users to distinguish between original images and our editing results. We prepared 13 image pairs of original image and a corresponding editing results (see the supplementary material). Note that the edited results were direct outputs of our system and were *not* rendered offline. Further, some of the input images were rendered scenes obtained from online scenes (e.g., in product catalogs). Each user was shown a random selection of 13 images, one from each pair, and given a maximum of 5 seconds to classify the image as *real* or *fake*. The user study comprised of 44 participants mostly computer science graduates, with many having backgrounds in computer graphics. On average, users recognized 63.2% real images as real, and 44.5% fake images as fake.

In Figure 15, we compare our system with state-of-the-art image analysis techniques. Note that the semi-automatic methods produce

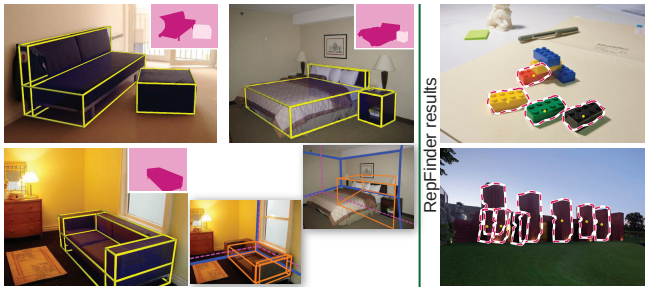

Figure 15: *State-of-the-art algorithms (e.g., [Gupta et al. 2011]) assume axis-aligned objects, produce coarse proxies (insets), or fail when assumptions are violated (e.g., on the left image). Having access to segmentation allows us to produce tighter cuboid-proxies.*

proxies that are not ideal for manipulating the objects, although they are sufficient for inserting new synthetic 3D objects [Karsch et al. 2011]. Further, the methods are restricted to axis-aligned objects. In our context, however, simple segmentation strokes are sufficient to enable proxy-based segmentation. Alternately, a purely image-based method like RepFinder [Cheng et al. 2010] is also unsuitable, especially when the scene contains perspective distortions and occlusions, or for interactions involving perspective changes.

Repeated edit. Figure 11 shows examples of repeated edits. Note that we factor out rotations and translations across objects to detect repetitions (see also Figures 6 and 12). We detect repetitions at the level of the proxies, but when transferring texture information, we verify color consistency before consolidating color/texture across proxies. Note that the facade example is comparable to state-of-the-art symmetry-based image resizing [Wu et al. 2010]. They consider an image as a whole and apply insertion and removal of repeated patterns to accommodate the resizing operator, we apply the resizing operator directly to the image object while leaving the other scene objects untouched. In the future, it will be interesting to continue exploring other summarization possibilities.

Virtual shopping. Figure 1 shows a typical application of our system. The user selects a set of images and roughly marks regions of interest. Then, we recover cuboid-based scene abstractions for the selected objects — note that our inputs are just images and *not* 3D objects. More importantly, we identify relations across the objects in each individual image (e.g., table tops are aligned to seating areas). Now, the user can *move* objects across scenes — in our framework, we restore the original relations, e.g., the table height is adjusted, the sofas are scaled anisotropically to fit in the original setting. Note that although the original scene has multiple light sources, we calibrate to only one source, and hence obtain a single directional shadow. Figure 14 shows another example. The user can also individually edit (e.g., rotate) objects to fine tune the layout (see the supplementary video). Note that in addition to coplanar and parallel relations, it can be interesting to consider relative heights between parallel surfaces, say between a chair seat and a desk. We leave this to future efforts possibly enabling function-aware scene understanding.

Limitations. When the input scene assumptions listed in Section 3 are violated, different artifacts can arise. Limitations include: (i) Failing to automatically detect good cuboid-proxies for scenes with curved chairs or sofas, etc., or cluttered workspaces with cables, bags, or heaped objects. The user can, however, manually adjust the estimated hexagon corners, while we jointly optimize the extracted constraints to regularize the results (see Figure 16); (ii) Non-boxy objects in the scenes (e.g., the teapot in Figure 11, or

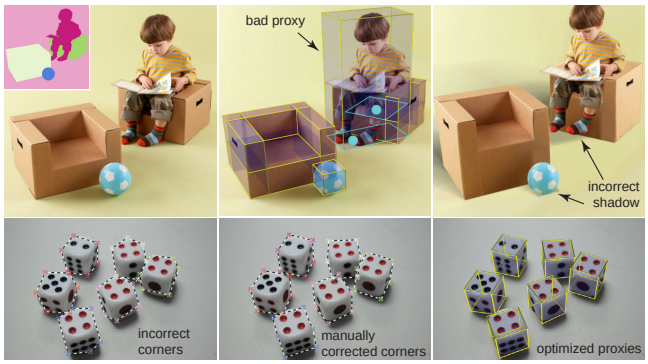

Figure 16: *Limitations. Poor proxies produce implausible shadows (top); objects with soft edges and occlusions lead to bad initial hexagon estimates, which have to be manually refined (bottom).*

the skull in Figure 13) produce noticeable distortions under rotation due to imprecise proxies. Semi-transparent objects (e.g., the pencil sharpener in Figure 2) and their shadows can appear distorted under rotations; (iii) Texturing artifacts arise when our symmetry-based texture copying fails, e.g., the back of the laptop, parts of the building blocks, or artifacts on the plants in the living room scenes (see the supplementary video); (iv) Finally, we fail to automatically infer relations without sufficient geometric clues, e.g., drawers or hinge joints have to be annotated (see Figure 10). Similarly, in absence of sufficient image edges, room walls are not reconstructed.

8 Conclusion

We presented an interactive system for smart editing of images of man-made scenes. In the analysis phase, based on user-provided segmentations, we propose a joint optimization to simultaneously recover camera calibration, generate cuboid-based proxies, and extract their non-local relations. We show that although the cuboids provide only a partial abstraction of the scene, they are sufficient to decompose the image into a background layer and textured proxies linked via non-local relations. In the interaction phase, the representation can then be used towards smart image editing mimicking real-world experiences.

In the future, we plan to explore the following directions: (i) A natural continuation is to jointly estimate geometry and appearance parameters in order to obtain quality scene understanding. Initial leads are provided by the recent work of Karsch et al. [2011] who demonstrate that convincing appearance modeling is possible with synthetic 3D object insertion. (ii) Although simple scene-level changes can result in large errors in image-level similarity measurements, such images often have very similar proxy-based representations (see Figure 13), thus providing new image comparison possibilities. One challenge, however, is how to automatically segment the images into meaningful regions. (iii) Although we focused on cuboid-proxies, other proxies like spheres, cylinders, and cones can possibly be incorporated. Further research is needed to evaluate the merits of such a generalization. (iv) Finally, in the context of 3D models, co-location priors have been shown to greatly simplify content retrieval and scene modeling tasks [Fisher et al. 2011]. We plan to investigate the use of such priors directly in the context of images using proxy-based representations, while making use of associated image level segmentation and keywords. Another interesting direction to pursue is how to automatically animate images of mechanical assemblies [Mitra et al. 2010].

Acknowledgements. We thank the anonymous reviewers, Danny Cohen-Or, and Peter Wonka for their many useful comments and feedback; Duygu Ceylan, Hung-Kuo Chu, and Yongliang Yang for proof-reading the paper draft; and Sawsan Alhalawani for the video voiceover. The work was partially supported by a KAUST visiting student scholarship, NSFC grant 60825201, the 973 Program grant 2011CB302205, and the Marie Curie Career Integration Grant 303541. We thank Tuhin for sharing his toys used in Figure 6.

References

BARNES, C., SHECHTMAN, E., FINKELSTEIN, A., AND GOLDMAN, D. B. 2009. Patchmatch: a randomized correspondence algorithm for structural image editing. *ACM TOG (SIGGRAPH) 28*, 3, 24:1–11.

BIEDERMAN, I., MEZZANOTTE, R., AND RABINOWITZ, J. 1982. Scene perception: Detecing and judging objects undergoing relational violations. *Cognitive Psychology 14*, 143–177.

CARROLL, R., AGARWALA, A., AND AGRAWALA, M. 2010. Image warps for artistic perspective manipulation. *ACM TOG (SIGGRAPH) 29*, 4, 127:1–127:9.

CHENG, M.-M., ZHANG, F.-L., MITRA, N. J., HUANG, X., AND HU, S.-M. 2010. RepFinder: Finding approximately repeated scene elements for image editing. *ACM TOG (SIGGRAPH) 29*, 4, 83:1–83:8.

EFROS, A. A., AND LEUNG, T. K. 1999. Texture synthesis by non-parametric sampling. In *IEEE ICCV*, 1033–1038.

FISHER, M., SAVVA, M., AND HANRAHAN, P. 2011. Characterizing structural relationships in scenes using graph kernels. *ACM TOG (SIGGRAPH) 30*, 34:1–34:12.

GAL, R., SORKINE, O., MITRA, N. J., AND COHEN-OR, D. 2009. iWIRES: an analyze-and-edit approach to shape manipulation. *ACM TOG (SIGGRAPH) 28*, 33:1–33:10.

GIBSON, J. J. 1979. *The Ecological Approach to Visual Perception*. MIT Press.

GUO, R., DAI, Q., AND HOIEM, D. 2011. Single-image shadow detection and removal using paired regions. In *IEEE CVPR*, 2033–2040.

GUPTA, A., EFROS, A. A., AND HEBERT, M. 2010. Blocks world revisited: Image understanding using qualitative geometry and mechanics. In *ECCV*, 119–153.

GUPTA, A., SATKIN, S., EFROS, A. A., AND HEBERT, M. 2011. From 3d scene geometry to human workspace. In *IEEE CVPR*, 1961–1968.

HARTLEY, A., AND ZISSERMAN, A. 2006. *Multiple view geometry in computer vision (2. Ed.)*. Cambridge University Press.

HAYS, J., AND EFROS, A. 2007. Scene completion using millions of photographs. *ACM TOG (SIGGRAPH) 26*, 3, 87–94.

HEDAU, V., HOIEM, D., AND FORSYTH, D. 2010. Thinking inside the box: Using appearance models and context based on room geometry. In *ECCV*, 224–237.

HOIEM, D., EFROS, A. A., AND HEBERT, M. 2005. Automatic photo pop-up. *ACM TOG (SIGGRAPH) 24*, 3, 577–584.

JAIN, A., THORMÄHLEN, T., SEIDEL, H.-P., AND THEOBALT, C. 2010. Moviereshape: Tracking and reshaping of humans in videos. *ACM TOG (SIGGRAPH Asia) 29*, 5, 148:1–148:9.

JIANG, N., TAN, P., AND CHEONG, L.-F. 2009. Symmetric architecture modeling with a single image. *ACM TOG (SIGGRAPH Asia) 28*, 5, 113:1–113:8.

KARSCH, K., HEDAU, V., FORSYTH, D., AND HOIEM, D. 2011. Rendering synthetic objects into legacy photographs. *ACM TOG (SIGGRAPH Asia) 30*, 6, 157:1–157:12.

LALONDE, J.-F., HOIEM, D., EFROS, A. A., ROTHER, C., WINN, J., AND CRIMINISI, A. 2007. Photo clip art. *ACM TOG (SIGGRAPH) 26*, 3 (August), 3.

LOURAKIS, M., 2004. levmar: Levenberg-marquardt non-linear least squares algorithms in C/C++. [web page] `http://www.ics.forth.gr/~lourakis/levmar/`.

MITRA, N. J., YANG, Y.-L., YAN, D.-M., LI, W., AND AGRAWALA, M. 2010. Illustrating how mechanical assemblies work. *ACM TOG (SIGGRAPH) 29*, 4, 58:1–58:12.

NORMAN, D. 1990. *Design of Everyday Things*. MIT Press.

OH, B. M., CHEN, M., DORSEY, J., AND DURAND, F. 2001. Image-based modeling and photo editing. In *ACM SIGGRAPH*, 433–442.

PÉREZ, P., GANGNET, M., AND BLAKE, A. 2003. Poisson image editing. *ACM TOG (SIGGRAPH) 22*, 3, 313–318.

ROTHER, C., KOLMOGOROV, V., AND BLAKE, A. 2004. "grabcut": interactive foreground extraction using iterated graph cuts. *ACM TOG (SIGGRAPH) 23*, 3, 309–314.

RUBINSTEIN, M., SHAMIR, A., AND AVIDAN, S. 2009. Multi-operator media retargeting. *ACM TOG (SIGGRAPH) 28*, 3, 23:1–23:11.

SAXENA, A., SUN, M., AND NG, A. 2009. Make3D: Learning 3D scene structure from a single still image. *IEEE PAMI 31*, 5, 824 –840.

SEITZ, S. M., CURLESS, B., DIEBEL, J., SCHARSTEIN, D., AND SZELISKI, R. 2006. A comparison and evaluation of multi-view stereo reconstruction algorithms. In *IEEE CVPR*, 519–528.

SHAPIRA, L., SHAMIR, A., AND COHEN-OR, D. 2009. Image appearance exploration by model-based navigation. *CGF 28*, 2, 629–638.

SINHA, S. N., STEEDLY, D., SZELISKI, R., AGRAWALA, M., AND POLLEFEYS, M. 2008. Interactive 3D architectural modeling from unordered photo collections. *ACM TOG (SIGGRAPH Asia) 27*, 5, 159:1–159:10.

SUN, J., YUAN, L., JIA, J., AND SHUM, H. 2005. Image completion with structure propagation. *ACM TOG (SIGGRAPH) 24*, 3, 861–868.

WANG, Y.-S., TAI, C.-L., SORKINE, O., AND LEE, T.-Y. 2008. Optimized scale-and-stretch for image resizing. *ACM TOG (SIGGRAPH Asia) 27*, 5, 118:1–118:8.

WEI, L.-Y., LEFEBVRE, S., KWATRA, V., AND TURK, G. 2009. State of the art in example-based texture synthesis. In *EG-STAR*, 93–117.

WILCZKOWIAK, M., STURM, P. F., AND BOYER, E. 2005. Using geometric constraints through parallelepipeds for calibration and 3D modeling. *IEEE PAMI 27*, 2, 194–207.

WU, H., WANG, Y.-S., FENG, K.-C., WONG, T.-T., LEE, T.-Y., AND HENG, P.-A. 2010. Resizing by symmetry-summarization. *ACM TOG (SIGGRAPH Asia) 29*, 6, 159:1–159:9.

XUE, T., LIU, J., AND TANG, X. 2010. Object cut: Complex 3d object reconstruction through line drawing separation. In *IEEE CVPR*, 1149–1156.

YANG, Y.-L., YANG, Y.-J., POTTMANN, H., AND MITRA, N. J. 2011. Shape space exploration of constrained meshes. *ACM TOG (SIGGRAPH Asia) 30*, 6.

ZHENG, Y., FU, H., COHEN-OR, D., AU, O. K.-C., AND TAI, C.-L. 2011. Component-wise controllers for structure-preserving shape manipulation. *CGF 30*, 2, 563–572.

ZHOU, S., FU, H., LIU, L., COHEN-OR, D., AND HAN, X. 2010. Parametric reshaping of human bodies in images. *ACM TOG (SIGGRAPH) 29*, 4, 126:1–126:10.

ZISSERMAN, A., REID, I. D., AND CRIMINISI, A. 1999. Single view metrology. In *IEEE ICCV*, 434–441.