Interaction Context (ICON): Towards a Geometric Functionality Descriptor

Ruizhen Hu[1,2,3*] Chenyang Zhu[1] Oliver van Kaick[4] Ligang Liu[5] Ariel Shamir[6] Hao Zhang[1]
[1]Simon Fraser University [2]SIAT [3]Zhejiang University [4]Carleton University [5]USTC [6]IDC

Abstract

We introduce a *contextual descriptor* which aims to provide a *geometric* description of the *functionality* of a 3D object in the context of a given scene. Differently from previous works, we do not regard functionality as an abstract label or represent it implicitly through an agent. Our descriptor, called *interaction context* or *ICON* for short, explicitly represents the geometry of object-to-object interactions. Our approach to object functionality analysis is based on the key premise that functionality should mainly be derived from interactions between objects and not objects in isolation. Specifically, ICON collects geometric and structural features to encode interactions between a central object in a 3D scene and its surrounding objects. These interactions are then grouped based on feature similarity, leading to a hierarchical structure. By focusing on interactions and their organization, ICON is insensitive to the numbers of objects that appear in a scene, the specific disposition of objects around the central object, or the objects' fine-grained geometry. With a series of experiments, we demonstrate the potential of ICON in functionality-oriented shape processing, including shape retrieval (either directly or by complementing existing shape descriptors), segmentation, and synthesis.

CR Categories: I.3.5 [Computer Graphics]: Computational Geometry and Object Modeling—Geometric algorithms.

Keywords: object functionality analysis, contextual descriptor, shape similarity, shape retrieval

"The essential definition of object classes is functional."

— Stark & Bowyer [1996]

1 Introduction

Recently in shape analysis, an increasing effort has been devoted to extracting high-level and semantic information from geometric objects and datasets [Mitra et al. 2013], especially man-made shapes. It is arguable that an important goal of some of these developments is to obtain a functional understanding of objects and object categories. The *functionality* of an object usually refers to the particular use for which the object is designed, while different interpretations of this concept are possible. For example, functionality can be defined as *the application of an object in a specific context for the accomplishment of a particular purpose* [Bogoni and Bajcsy 1995].

*ruizhen.hu@gmail.com

ACM Reference Format
Hu, R., Zhu, C., van Kaick, O., Liu, L., Shamir, A., Zhang, H. 2015. Interaction Context (ICON): Towards a Geometric Functionality Descriptor. ACM Trans. Graph. 34, 4, Article 83 (August 2015), 12 pages.
DOI = 10.1145/2766914 http://doi.acm.org/10.1145/2766914.

SIGGRAPH '15 Technical Paper, August 09 – 13, 2015, Los Angeles, CA.

DOI: http://doi.acm.org/10.1145/2766914

Figure 1: *Similarity between shapes (top) vs. similarity between functionalities (bottom). A shape descriptor (LFD) considers the middle cart more similar to the desk, as shown on the left using a 2D MDS projection of the distances between objects. Our contextual descriptor, interaction context or ICON, takes into account object-to-object interactions and identifies the two carts as more similar.*

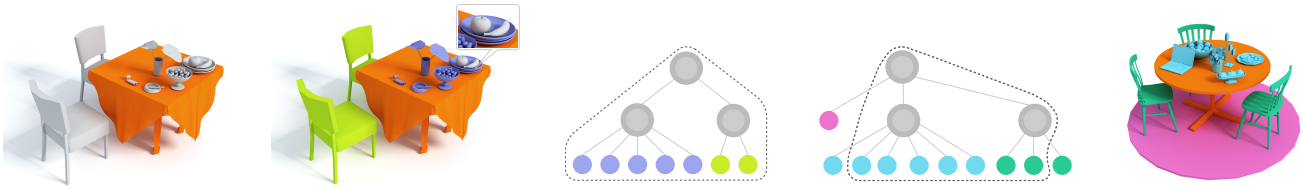
Ongoing pursuits on functional shape analysis have represented functionality in different manners. Some methods take a *category-specific* approach, relying on functionality models handcrafted for specific object categories [Sutton et al. 1994; Pechuk et al. 2008]. Another line of works characterize object functionalities *implicitly* through a human agent interacting with an object [Kim et al. 2014; Zhu et al. 2014; Liu et al. 2014] to detect its *affordances*, i.e., object properties that allow a person to perform a certain action. Finally, other works represent functionalities as labels such as "to support" and "to be held" [Pechuk et al. 2008; Laga et al. 2013].

The key premise of our approach is that object functionality should mainly be derived from *interactions* between objects and not an object in isolation [Caine 1994]. Hence, to analyze the functionality of an object, the object needs to be provided in a *context*, i.e., a surrounding 3D scene, to accomplish its functional purpose [Bogoni and Bajcsy 1995]. Moreover, given that the information that is present in 3D scenes is the *geometry* of the objects, our specific focus is to represent interactions inferable from object geometry or form, reflecting an attempt to invert the well-known notion of "form follows function" and develop a *geometric functionality descriptor.*

In this work, we introduce a contextual shape descriptor we call *interaction context* or *ICON*, for short. ICON encodes pairwise and localized interaction relations between a *central* object and its surrounding objects and organizes them in a meaningful manner. In contrast to previous works, our contextual description is not category-specific, and different from affordance analyses, the interactions we consider are not confined to those involving humans. By representing the geometry of the context of interactions of the central object, which are an important cue to infer the functionality of objects, we believe that ICON can constitute the starting point for developing a geometric functionality descriptor (Figure 1).

However, there are several challenges in defining a contextual descriptor using interactions:

- The descriptor of the central object should not be sensitive to specific counts of objects or their fine-grained geometry. For example, bookcases and curio cabinets are regarded as functionally equivalent even though there is significant variation in the number and the kinds of objects displayed.



(a) *Input scene 1* **(b)** *Interacting objects* **(c)** *ICON descriptor of (a)* **(d)** *ICON descriptor of (e)* **(e)** *Input scene 2*

Figure 2: *Overview of construction and matching of ICONs. Given an input scene with the central object (orange table) in (a), we detect interactions between the central object and other objects. The interacting objects are shown with bright colors in (b), while non-interacting objects (the apple and banana) are shown in gray. Next, we group the interactions into a hierarchical structure to obtain the ICON descriptor shown in (c). Each leaf node corresponds to an interaction and has the same color as the object in (b) that gives rise to the interaction, while internal nodes group similar interactions. (d) shows the descriptor of the scene in (e). The two ICON descriptors in (c) and (d) are matched by finding a common subtree isomorphism. We obtain the intuitive correspondence between objects on the tables and chairs, shown by the matched portions of the hierarchies selected by the dashed contours. Note that the floor and extra objects in (e) do not have a match.*

- Most objects serve multiple functions, and have several interactions, e.g., a shopping cart can both be pushed, an affordance, and hold grocery. While a single human template and its associated interactions suffice in affordance analysis, the contextual nature of ICON dictates that it must account for objects with multiple interactions of different characteristics. Moreover, these objects are not integrable into a single template; they only loosely surround the central object.

- Last but not the least, a mere totality of object-to-object interactions is insufficient to accurately characterize functionality. For example, a study desk and a dining table can both be described by one or more interactions with chairs; it is the way these interactions may be grouped and spatially arranged that tell the two objects apart. Hence, ICON must provide a structural organization of the interactions.

Given a (central) 3D object and its surrounding scene, ICON collects geometric (e.g., Voronoi boundaries between objects) and structural (e.g., symmetry) features which characterize how the central object interacts with other scene objects in close proximity. Next, cluster analysis is performed, grouping the interactions into a hierarchical structure based on their feature similarity. The result is a tree where the leaves represent object-to-object interactions with the central object, and a node at a higher level of the hierarchy represents a general interaction *type* (e.g., support or push) which characterizes the interactions belonging to its subtree (Figure 2).

The hierarchy integrates multiple interactions into a single ICON descriptor and provides a meaningful grouping that models how the central object interacts with its surroundings. To account for possible ambiguities arising from the hierarchical grouping, we allow an object to have multiple hierarchies. Multiple hierarchies are beneficial to the comparison of descriptors as considering multiple grouping hypotheses leads to a better similarity estimation. To compare two ICON descriptors, we find the best matching between the trees via subtree isomorphism. To ensure that the comparison is oblivious to specific object counts and geometry, we rely on a robust feature encoding of the object-to-object interactions, giving more importance to higher levels of the trees while matching.

We demonstrate the relevance of ICON for functionality analysis mainly through experiments on object retrieval. The reason is twofold. First, ICON is a descriptor that captures interactions, which are one possible set of cues to infer the functionality of an object. Second, a useful categorization of objects is arguably based on their functionality, hence the ability of a descriptor to accurately retrieve objects from different categories amid significant variations in object shapes, locations, counts, and semantic types would reflect well how the descriptor serves to distinguish object functionality.

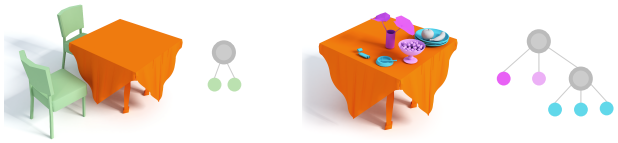
We present object retrieval results using ICON on scenes extracted from well-known databases, and compare to existing approaches including affordance analysis. We also show how ICON can complement existing shape descriptors that are focused on shape discrimination, as well as scene descriptors, to improve retrieval performance. Furthermore, we show the potential of ICON in enabling new applications, namely, segmentation and transfer of interacting regions, which are difficult to accomplish if functionality is represented only with an abstract label or an implicit description of a human pose. Finally, we extend the ICON concept to encode and analyze the context of object *parts*, and show its potential in complementing semantic part analysis.

2 Related work

Many solutions to classic problems in shape analysis, such as similarity estimation and segmentation, have been enabled by the development of appropriate shape descriptors. Recent efforts are departing from using only local geometric descriptors, placing emphasis on extracting high-level and semantic information from geometric data [Mitra et al. 2013]. These works have set a clear trend in the field towards focusing the analysis on structural aspects and even functionality of shapes.

Model-based analysis. Model-based methods derive the functionality of shapes by matching them to pre-defined models of functional requirements. Sutton et al. [1994] handcraft models to represent functional categories, where a model is a knowledge base that specifies what functional requirements or *primitives* are needed to define a specific functionality, e.g., a mug cup needs to provide containment for a liquid, stability, and a graspable handle. Rivlin et al. [1995] define a functional category by a set of parts, their corresponding functionalities, and their spatial relationships. Although parts are automatically recovered from an image with a segmentation method, the association between parts and functionalities is still given by a set of a priori functional primitives. Pechuk et al. [2008] manually construct a hierarchy of functional parts for a specific object category and learn properties of each type of functional part and their relationships from labeled 3D images. Laga et al. [2013] follow a similar idea, assigning functional labels to shape parts based on their geometry and relationships.

The main drawback of these methods is that the possible functional structure or labels of each category have to be known beforehand.



(a) *ICON 1* **(b)** *ICON 2*

Figure 3: *ICON is data-dependent: different interactions lead to different descriptors, i.e., hierarchies. We show two input scenes where different types of interactions take place with the same central object (orange table). Note how the corresponding hierarchies have a different structure in each case.*

In comparison, ICON assumes no pre-defined knowledge of the objects in question, including their categories or semantic part labels. At the same time, ICON is versatile: if required, we can always learn an association of categories to certain descriptor instances.

Another group of methods analyze shape structures to model functionality. Zheng et al. [2013] detect three-part support structures on the input models and use them to synthesize shapes that retain this particular (support) functionality. Wang et al. [2011] relate symmetry to functionality based on the observation that symmetric parts in an object tend to perform the same function. This work, along with those of van Kaick et al. [2013] and Tevs et al. [2014], all infer functional similarity between shape parts or groups of shape parts by matching structures which characterize symmetries and regularities. In contrast, ICON offers a means to explicitly represent and organize object-to-object interactions.

Robotics and agent-based analysis. In the field of robotics, there has been intensive work on modeling interactions and affordances, with the motivation of using such a model to control a robot that interacts with an environment. Many of the methods proposed in the field are agent-based, where the functionality of an object is identified with an indirect shape analysis based on interactions of an agent [Bar-Aviv and Rivlin 2006; Grabner et al. 2011; Zhu et al. 2014; Kim et al. 2014; Liu et al. 2014]. Given a template of the agent (e.g., a human), these methods find a correspondence between an interaction pose of the agent and a specific functionality, which is called an *affordance* model. With such a model, the methods can predict the interacting pose for an unknown shape and then assign a specific functionality to the shape based on the matching between the predicted pose and a functionality.

The agent may also be observed in a video sequence [Gupta et al. 2009] or obtained from skeleton tracking. In SceneGrok [Savva et al. 2014], a 3D scene is scanned and human interactions with the objects in the scene are recorded. An action map of the scene is created from this data and, through a learning procedure, is used to infer regions that can serve specific actions in unknown scenes. Although the agent (human) is captured from real interactions and not simulated, the type of agent is known beforehand. The actions also need to be labeled to enable learning.

One limitation of the agent-based methods is that they cannot model functionalities if the agent is unknown. Extending these works to more general settings by constructing templates for all types of possible agents seems unrealistic. Also, a single agent may not be sufficient to define certain object functionalities, e.g., those of a hand-truck; functionality description involving multiple agents is challenging. In comparison to these works, ICON directly captures the multiple interactions that appear in one or more available input scenes without the need to model external agents.

Contextual descriptors. Shape descriptors play a key role in all shape analysis tasks. A well-designed descriptor for a point or region over a shape is often not purely local, but captures a spatial context around the point or region in the form of a grid, a histogram, or other kinds of organizational structures. Well known examples of such *contextual descriptors* include spin images [Johnson and Hebert 1999] and shape contexts [Belongie et al. 2002].

For 3D scene analyses, Fisher et al. [2011] model object contexts based on locations of nearby objects and their semantic relations. These measures, along with object geometries, are encoded into a graph kernel representation for object and scene comparison. Similarly to [Fisher et al. 2011], our goal is to capture the context of an object in a given scene, albeit a context of interactions. One key difference is that ICON encodes the geometry of *interactions* between objects, not just their spatial or semantic relations. For example, without considering geometry, the relation "one object is to the left of another" can imply drastically different interactions between the objects. Another key difference is that ICON organizes the interactions in a hierarchical manner, which allows us to compare two object contexts based on *groups* of similar interactions. Due to the heterogeneous nature of indoor scenes, Xu et al. [2014] propose to characterize scenes by focal points, which are representative substructures that allow for scene comparisons relative to these structures. Differently from focal points, we represent the interactions of central objects, as opposed to offering a mechanism to organize sets of scenes.

Geometry of interactions. Our contextual descriptor requires a representation of the spatial region or interaction between *two or more* objects. In robotics, there has been seminal work on detecting grasp affordances, that is, regions where an agent is able to grasp or hold an object. These methods are based on analyzing a single object, e.g., with texture cues and pose information extracted from an image [Song et al. 2011], and are mainly suited for detecting specific types of interactions such as grasping. Drawing inspiration from such works, Zhao et al. [2014] propose the interaction bisector surface (IBS) to describe the geometry of the interaction between two or more objects. By defining appropriate features on this surface, it is possible to distinguish different types of interactions. In our work, we use the IBS in conjunction with our proposed interaction regions (IRs) to capture object-to-object interactions (Section 4). IRs are essential to capture the regions on the central object that correspond to the interactions with different objects. The IBSs and IRs are further organized with a hierarchy in our contextual descriptor (Section 5), enabling a meaningful comparison of descriptors coming from different objects.

3 Overview

The input to ICON construction consists of a 3D object, the central object, provided together with a surrounding scene. We assume that different objects in the scene are represented separately, for instance, by independent triangle meshes. When defining the ICON descriptor, we seek *generalization ability*, i.e., similarity between central objects in geometrically and structurally dissimilar scenes as long as the general interactions of the objects are similar. This is achieved by grouping the interactions in a hierarchy that captures their general structure and is oblivious to the number of interactions or to their specific positions; see Figure 2. At the same time, ICON is *data-dependent*, i.e., different scenes of the same object, as determined by the interactions with surrounding objects, can lead to different ICON descriptors; see Figure 3.

We start the construction by identifying the interacting objects in the scene and extracting an initial set of pairwise interactions of the central object with all interacting objects. For each pairwise in-

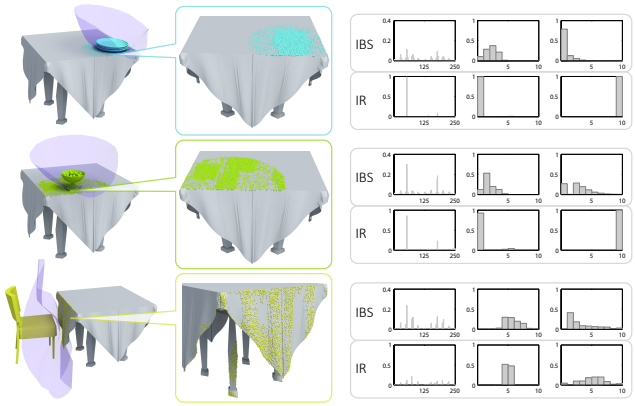
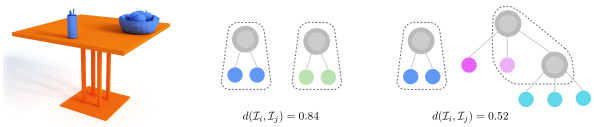


Figure 4: *Representation of interactions in ICON. Each interaction in the scene is captured by an IBS (purple surfaces) and an IR (the colored samples on the table). These two entities are described by a set of appropriate descriptors (the histograms). Note how the IR descriptors distinguish the interactions better than IBS in this case, and imply that the interactions with objects on the table are similar and can be grouped together, while the interaction with the chair is more distinct.*

teraction, we compute two entities: the interaction bisector surface (IBS) and the interaction region (IR). The IBS [Zhao et al. 2014] is a surface that captures the spatial boundary between two objects, while the IR is the region on the object corresponding to the interaction (see Figure 4 and Section 4). These entities represent one interaction and can be compared with the use of appropriate features. Note that not all objects in the scene are interacting with the central object. We compute the IBS for the entire scene first and select the objects that share a subset of the IBS with the central object as the interacting objects. That is, the interacting objects are those objects that are separated from the central object directly by the IBS. As shown in Figure 2(b), the apple and banana are not regarded as interacting with the table since they do not share an IBS with the table.

We organize the interactions of an object in a hierarchy that captures the general structure of the object's interactions. The hierarchy is a tree where each leaf node corresponds to an interaction and the internal nodes group the leaf nodes into meaningful clusters according to the similarity of their interactions. Thus, the higher levels of the hierarchy potentially describe the functionality of the object, while the lower levels capture the finer detail of specific interactions that appear in the scene (Figure 2). This provides a meaningful organization of interactions that is less sensitive to the specific numbers of objects in the scenes and their arrangement (Figure 5). Also, since in certain cases there can be ambiguity in how to group the nodes, an ICON descriptor may contain multiple hierarchies, one for each grouping hypothesis. In this manner, the comparison between two central objects in different scenes is more robust.

The hierarchies allow for cross-comparison of ICON descriptors; see Figure 2(c)-(d). The similarity of two ICON descriptors is given by the best matching score between any two of their hierarchies. The matching score for two hierarchies is derived from a tree metric where we find the common subtree isomorphism of the input trees according to the node similarities, which are computed from the similarities of features that describe their corresponding interactions. We describe the hierarchical organization and construction of the descriptor in Section 5.



(a) *Input scene* **(b)** *Matching 1* **(c)** *Matching 2*

Figure 5: *Robustness of ICON descriptors to the number of interacting objects. We show the matching of the ICON for the central object in (a) to the ICONs constructed from scenes with different numbers and types of objects in Figure 3. We also report the tree distances. Note that the table in (a) is more similar to Figure 3(b) than Figure 3(a), since both tables have objects on top.*

4 Representation of Interactions

Interactions are represented in ICON with a combination of interaction bisector surfaces (IBS) and interaction regions (IR). Each interaction i of a central object is represented by a tuple $\mathcal{I}_i = (\mathcal{S}_i, \mathcal{R}_i)$, where $\mathcal{S}_i$ is an IBS and $\mathcal{R}_i$ is the corresponding IR.

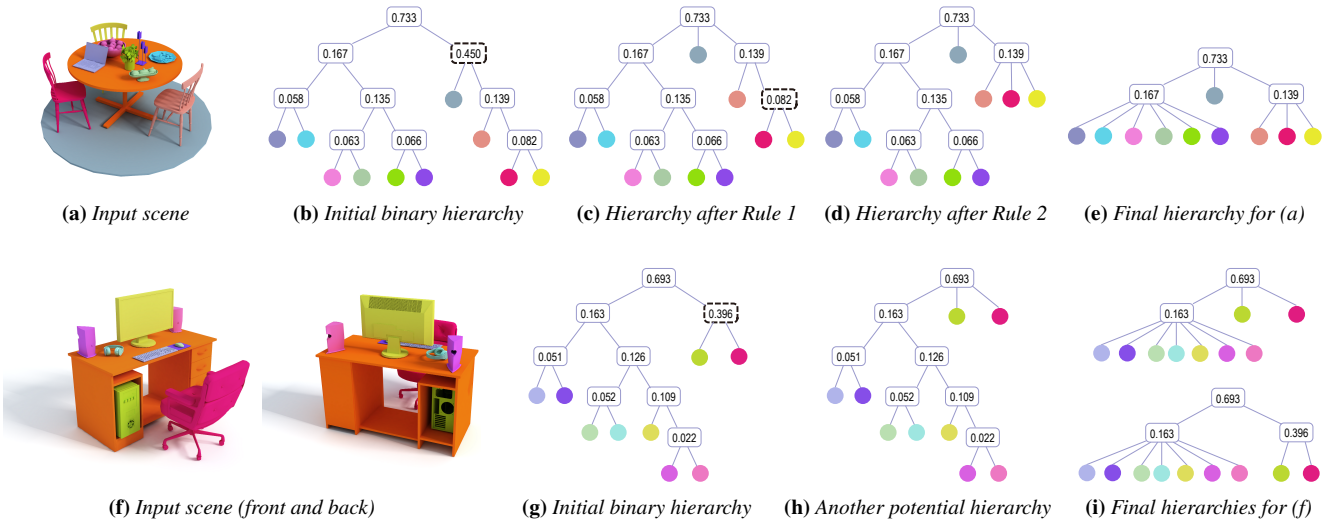
Interaction bisector surface (IBS). IBS represents a spatial region between two or more objects and is defined as a subset of the Voronoi diagram computed between the objects. By extracting an adequate set of features from the IBS, we can distinguish topological and geometric properties of the region.

Given two sets of points $\mathcal{P}_1$ and $\mathcal{P}_2$, sampled from two different objects O_1 and O_2, respectively, the IBS between O_1 and O_2 is defined as the set of points equidistant to both $\mathcal{P}_1$ and $\mathcal{P}_2$. The IBS is obtained by computing the Voronoi diagram of all the samples and selecting only the *ridges* (faces) of the diagram that bisect points from two different objects (Figure 4). Since the IBS is infinite, we truncate it by intersecting it with a bounding sphere of O_1 and O_2. Finally, we triangulate all the faces of the IBS and orient their normals consistently so that they point towards the central object.

IBS Features. To distinguish the different types of interactions captured by the IBS, we first perform an importance-based sampling on the objects and then analyze a set of *geometric features* defined on the sampled points, as proposed by Zhao et al. [2014]. We compute: 1) The point feature histogram (PFH) descriptor, which collects the relative rotation between each pair of normals in the surface and thus captures the general shape of the IBS; 2) A histogram that collects the angle between the surface normals and the $+z$ vector (Dir). Assuming that the scenes are upright-oriented, this histogram captures the general direction of the boundary between the objects; 3) A histogram that captures the distribution of distances between the IBS and the objects (Dist). We do not use the topological features of Zhao et al. [2014], since these tend to be sensitive to small differences in the surfaces.

Interaction region (IR). In addition to the IBS and its features, we extract the regions on the central object that correspond to the interactions with different objects. These regions allow us to link the interactions captured by IBS to the central object. Given the point samples $\mathcal{P}$ of the central object and a set $\{\mathcal{S}_1, \ldots, \mathcal{S}_N\}$ of extracted IBSs, we define a set of (possibly overlapping) interaction regions $\{\mathcal{R}_1, \ldots, \mathcal{R}_N\}$, where each $\mathcal{R}_i \subseteq \mathcal{P}$ corresponds to one $\mathcal{S}_i$, and includes all points that are linked to it.

These points are found as follows. Each triangle on an IBS $\mathcal{S}_i$ has a weight that indicates how important the triangle is in determining the interaction. Since $\mathcal{S}_i$ consists of the ridges of the diagram that



(a) *Input scene* **(b)** *Initial binary hierarchy* **(c)** *Hierarchy after Rule 1* **(d)** *Hierarchy after Rule 2* **(e)** *Final hierarchy for (a)*

(f) *Input scene (front and back)* **(g)** *Initial binary hierarchy* **(h)** *Another potential hierarchy* **(i)** *Final hierarchies for (f)*

Figure 6: *Illustration of hierarchy creation and merging in ICON, including the creation of multiple hierarchies. To represent the interactions of the table in (a) and desk in (f), we start with binary hierarchies (b) and (g). Note that each leaf node has the same color as the corresponding object in the scene that gave rise to the interaction. We merge nodes by the repeated application of Rules 1 and 2 (applied to the dashed nodes) to obtain the final hierarchies in (e) and (i). By applying Rule 3 to get from (g) to (h), the final ICON descriptor for the scene in (f) consists of two hierarchies. Note that the front and back of (f) show that the interactions of the desk with the computer and the chair are similar. Thus, their corresponding nodes are grouped together.*

bisect points of the central object and the interacting object, each triangle can be assigned to a single (nearest) point on the central object. Thus, each sample point $\mathbf{p}_j \in \mathcal{P}$ corresponds to one or more triangles on each $\mathcal{S}_i$. We can then assign a weight $w_{i,j}$ to a sample $\mathbf{p}_j$ for each IBS $\mathcal{S}_i$, which is given by the sum of the weights of triangles in $\mathcal{S}_i$ corresponding to $\mathbf{p}_j$. Finally, we assign each point $\mathbf{p}_j$ to all regions $\mathcal{R}_i$ where $w_{i,j} > 0$.

IR features. We also extract a set of features to describe the interaction regions. We use the PFH and Dir descriptors, defined similarly as for the IBS. In addition, we construct a histogram (HH) that collects the distribution of heights of the points on the interaction region,which captures the overall height of the region.

Interaction distance. To facilitate comparisons between interactions as well as clustering, we define a distance measure between two interactions $\mathcal{I}_i = (\mathcal{S}_i, \mathcal{R}_i)$ and $\mathcal{I}_j = (\mathcal{S}_j, \mathcal{R}_j)$ as:

$$d(\mathcal{I}_i, \mathcal{I}_j) = (1-w)\, d_{\mathcal{S}}(\mathcal{S}_i, \mathcal{S}_j) + w\, d_{\mathcal{R}}(\mathcal{R}_i, \mathcal{R}_j), \tag{1}$$

$$\text{where} \quad d_{\mathcal{S}}(\mathcal{S}_i, \mathcal{S}_j) = \begin{array}{l} u_1\, d_H(\text{PFH}_{\mathcal{S}_i}, \text{PFH}_{\mathcal{S}_j}) + \\ u_2\, d_H(\text{Dir}_{\mathcal{S}_i}, \text{Dir}_{\mathcal{S}_j}) + \\ u_3\, d_H(\text{Dist}_{\mathcal{S}_i}, \text{Dist}_{\mathcal{S}_j}), \end{array} \tag{2}$$

$$\text{and} \quad d_{\mathcal{R}}(\mathcal{R}_i, \mathcal{R}_j) = \begin{array}{l} v_1\, d_H(\text{PFH}_{\mathcal{R}_i}, \text{PFH}_{\mathcal{R}_j}) + \\ v_2\, d_H(\text{Dir}_{\mathcal{R}_i}, \text{Dir}_{\mathcal{R}_j}) + \\ v_3\, d_H(\text{HH}_{\mathcal{R}_i}, \text{HH}_{\mathcal{R}_j}). \end{array} \tag{3}$$

Since all the IBS and IR features are normalized histograms, we define d_H as the L_1-norm divided by $2\sqrt{2}$ to ensure that the distances are in the range $[0, 1]$. The constants w, u_i, and v_i weight the different terms in the distance and are listed in Section 6.

5 Interaction context (ICON)

In this section, we define our contextual descriptor and its associated distance measure, computed via a tree matching procedure. We also show how ICON can be adopted as a "part-in-object" descriptor for describing the context of a shape part.

To capture the context of interactions of a central object, the ICON descriptor of the object must not depend on specific characteristics of the scene such as the number of interacting objects (e.g., number of cups on a table or number of books on a shelf) as well as their exact spatial configuration. To achieve a generalization capability, the ICON descriptor of a central object organizes the local interactions of the object in a hierarchical manner. First, similar interactions are merged into groups (nodes), and then each group is treated as a single interaction. Second, diversity and robustness are kept by allowing several hierarchies to represent an interaction context of an object. This facilitates comparison between similar objects in different scenes. The effect of the hierarchical organization is illustrated in Figure 5.

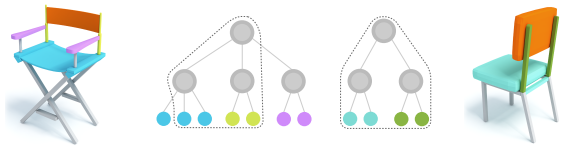
Hierarchy construction. Given the set of interactions $\mathscr{I} = \{\mathcal{I}_1, \ldots, \mathcal{I}_N\}$ of the central object, we build a hierarchy of interactions in two steps. First, we build a binary tree of interactions, and second, we merge multiple tree branches into single nodes to better reflect the natural grouping of interactions. We tested other alternatives for grouping such as self-tuning spectral clustering applied recursively, but found our approach to be more stable.

To obtain a binary tree, we group the interactions with agglomerative hierarchical clustering. The distance between two clusters is defined as the maximal distance among all the pairs of interactions in the two clusters. The output of this step is a binary tree where interactions are grouped hierarchically by their similarity.

Next, we merge multiple nodes of the tree to remove cases where the binary branching is almost arbitrary. Since the binary tree represents the hierarchical clustering of interactions, a node and all of its descendants can be seen as a cluster. Hence, we can use the cluster distance (maximal distance among all interactions) to compare two nodes. We perform the merging according to the following rules (see Figure 6 for an illustration):

(a) (b) (c)

Figure 7: *Effect of symmetry in the construction of ICON hierarchies. When the pattern of the shelves in (a) is taken into account in the construction, the resulting hierarchy in (b) groups objects according to the symmetry structure of the central object. In (c), we see that the construction without symmetry consideration groups objects more arbitrarily (arrows in (a) and red boxes in (c)).*



(a) *Central part 1* **(b)** *ICONs and matching* **(c)** *Central part 2*

Figure 8: *Construction and matching of the part-level ICON descriptor. Given the central parts (in orange) in (a) and (c), we detect the parts (shown with bright colors) that interact with the central parts. Next, we build the ICON descriptors that can be used to match the interactions of two parts as shown in (b).*

Rule 1. If the maximal distance among the children of a given node is large, we delete the parent node and merge all of its children with the root node by making them children of the root. This case indicates that the cluster defined by the node and its children is not meaningful. To perform this test, we use a threshold of $\theta_{up} \cdot \text{MaxDist}$, where $\text{MaxDist} = \max_{\mathcal{I}_i,\mathcal{I}_j \in \mathcal{S}} d(\mathcal{I}_i, \mathcal{I}_j)$.

Rule 2. If the distance between a node and one of its children is small, we merge the child with the parent by attaching the child to the node's parent. This case indicates arbitrary branching as the parent and child could be reversed. The threshold used for the test is set to $\theta_{low} \cdot \text{MaxDist}$.

Rule 3. When the distance between a node and one of its children lies in an intermediate range (between $[\theta_{mid}, \theta_{up}] \cdot \text{MaxDist}$), the grouping of the nodes also has the potential of being arbitrary. However, it is not trivial to decide whether the nodes should be merged or not. Thus, instead of creating a possibly incorrect hierarchy, we build multiple candidate hierarchies capturing different merge options. We take all pairs of nodes in the intermediate range and generate all possible merge combinations. A combination is invalid if a node is merged while any of its ancestry (except the root) is not merged, since this violates the coherence of the tree. For each valid combination, we generate a candidate hierarchy and then use Rules 1 and 2 to further merge nodes.

Thus, at the end of the merging process, we obtain a set of hierarchies that represent an ICON descriptor (Figure 6). Finally, we recompute the interactions and their features for each hierarchy by considering all the objects corresponding to a node as a single object. In this manner, every node is associated with a single interaction and its features, which facilitates the descriptor comparison.

ICON comparison. The distance between two ICON descriptors is defined as the minimum distance among every pair of their hierarchies. The distance between two hierarchies $\mathcal{T}_i$ and $\mathcal{T}_j$, is defined as a normalized version of the tree distance proposed by Torsello et al. [2005], which is proven to be a metric:

$$d(\mathcal{T}_i, \mathcal{T}_j) = 1 - W(\mathcal{T}_i, \mathcal{T}_j)/(|\mathcal{T}_i| + |\mathcal{T}_j| - W(\mathcal{T}_i, \mathcal{T}_j)), \quad (4)$$

where $|\mathcal{T}_i|$ denotes the sum of all node weights in the tree $\mathcal{T}_i$ and $W(\mathcal{T}_i, \mathcal{T}_j) = \sum_{n \in \mathcal{T}_i} \frac{w_n + w_{\phi(n)}}{2} d(n, \phi(n))$, with ϕ being the maximum similarity common subtree isomorphism between the two trees. The isomorphism can be computed in $O(bN^2M)$ time for trees with N and M nodes and maximum branching factor b. We use the interaction distance (Eq. 1) as the distance metric between two nodes, and weight each node n according to its depth in the hierarchy as $w_n = \tau^{\text{depth}(n)}$, to give more importance to higher-level nodes in the sum of node similarities, where τ is a constant.

With the hierarchical organization of interactions and the distance measure defined above, we obtain a robust procedure to compare the contexts of two objects. First, variable numbers of objects surrounding the central object do not interfere significantly with the representation, since the interactions are grouped into nodes with variable degree and the most important interactions are captured in higher levels of the hierarchy. Moreover, the maximum common subtree isomorphism ensures that we obtain the best matching between interactions according to their properties. Finally, the descriptor does not require the definition of labels or classes, but directly encodes the interactions of the central object in the hierarchy.

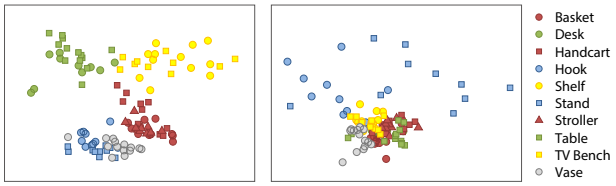
Interaction correspondence. Note that as a side product of computing the distance between two ICON descriptors, we also obtain information on how the interactions represented in the two descriptors may be matched. As each interaction induces an interacting object, ICON comparison also leads to a correspondence between the objects (or object groups). Such a matching can be useful to compute a many-to-many correspondence between scenes with different numbers and types of objects.

Symmetry structure. We also consider the symmetry structures of the central object in the representation of interactions, whenever the object possesses any such structures. Two interactions are deemed more similar if they are part of the same repeating pattern, which can include rotations as well as 1D and 2D translation groups of a repeating geometric primitive, e.g., repeating shelves in a cabinet. The effect of incorporating this consideration is shown in Figure 7, where we see that the resulting hierarchies group objects according to symmetry patterns of the central object. In our current implementation, the regular structures are manually annotated, but they can be detected automatically with methods such as the one proposed by Mitra et al. [2006]. After detecting the symmetry structures, we assign an IR to a group if its spatial extent overlaps with the repeating primitive of the group.

We then take the symmetry groups into consideration when computing the distance between interactions. We modify Eq. 1 into the symmetry-aware distance $d_{\text{sym}}(\mathcal{I}_i, \mathcal{I}_j) = d(\mathcal{I}_i, \mathcal{I}_j)/(\rho^\sigma)$, where σ is a symmetry factor set as follows: $\sigma = 2$ if $\mathcal{R}_i$ and $\mathcal{R}_j$ fall into the same repeated primitive, $\sigma = 1$ if $\mathcal{R}_i$ and $\mathcal{R}_j$ fall into the same pattern but not the same primitive, and $\sigma = 0$ if there is no common structure or no symmetries were detected on the shape. ρ is a constant that decreases the distance according to the symmetry factor σ. In this manner, we regulate the similarity of two interactions depending on how related they are by symmetry. Note that the symmetry-aware distance boils down to the original distance when no symmetry is present ($\sigma = 0$).

Part-level descriptor. Finally, we investigate the potential of ICON as a contextual descriptor of shape *parts*, to show the gen-

Figure 9: *An example scene from the dataset of Fisher et al. [2011], from where we extract subscenes (shown with the boxes) to compose our dataset. Each subscene consists of a central object (shown in orange) and its close-by objects (shown in blue).*



(a) *ICON MDS* **(b)** *LFD MDS*

Figure 10: *Embeddings based on descriptor distances among all central objects in our dataset for ICON and LFD, obtained with multidimensional scaling (MDS). Note how the embedding derived from ICON better groups objects according to their functionality, and classes with overlap have similar or related functionalities.*

erality behind ICON's design. That is, we would like to show that the idea of capturing interactions and organizing them in a meaningful manner provides a useful descriptor that can be applied to different situations. Given a segmented shape, we describe each segment or part with an ICON descriptor that captures the interactions of the part with all of its neighboring parts. The construction is similar to that of the object-level descriptor and is illustrated in Figure 8. However, we extend the segment boundaries to the volume of the shape so that the interactions between two parts can be captured with the computation of the IBS and IR.

6 Results, evaluations, and comparisons

In this section, we present results of using ICON towards object recognition to show ICON's potential in functionality analysis. We also evaluate the different aspects of the descriptor and compare ICON to existing descriptors, discussing ICON's potential in enabling new shape analysis applications. For the evaluation, we group the central objects in our dataset of scenes according to their primary function and perform shape retrieval experiments. Although reducing our evaluation to object recognition has disadvantages (objects with multiple functionalities can only be assigned to a single class), we are able to objectively measure the recognition of similar functionalities, and explore the class overlaps that appear when objects have similar functionalities.

Datasets. We test ICON on a dataset of scenes composed of man-made objects. We use known databases such as Trimble 3D Warehouse and the scenes made available by Fisher et al. [2011]. To compose our dataset, we extract subscenes from these scenes because objects far away from a central object hardly interact with the central object. Specifically, we extract central objects with their close-by objects, as shown in Figure 9. This is almost equivalent to using the full scenes as ICON first detects the interacting objects, except that we also remove some objects that differ too much in scale, e.g., the floor, walls and plants in Figure 9. We group these subscenes into 10 different categories (each category has 10-15 models) according to the primary functionality of the central object. We obtain the classes Basket, Desk, Handcart, Hook, Shelf, Stand, Stroller, Table, TV Bench, and Vase. The full dataset can be seen in the supplementary material. We use this dataset to evaluate the different aspects of ICON and also to perform a comparison to other descriptors.

In addition, we use the dataset of human poses of Kim et al. [2014], which is composed of 6 categories: Bicycle, Bipedal device, Cart, Chair, Cockpit, and Gym equipment. We use this dataset specifically to compare our method to the use of human poses.

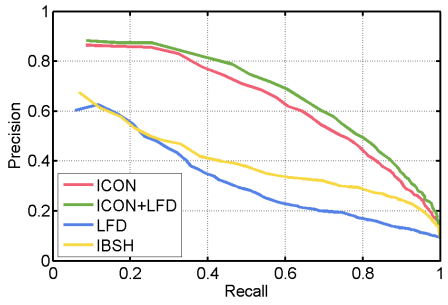
ICON results. In Figure 10(a), we present an embedding based on the distances of ICON descriptors between all pairs of objects in our dataset, obtained with multidimensional scaling (MDS). For contrast, we show the embedding obtained by using a shape descriptor (LFD) computed only on the central objects. We see that ICON provides a meaningful grouping of the classes in our dataset according to the primary functionality of the objects. A few classes that have significant overlap (we display them with the same color but different marker) in fact possess highly related functionalities, such as Hook and Stand that can be used for hanging objects. Among highly distinct functionalities (shown with different colors), there is almost no class overlap.

Moreover, to perform a quantitative evaluation, we assess the performance of ICON in an object retrieval experiment to evaluate its potential for functionality-based retrieval. For the experiment, we take each central object in our dataset and use it as a query to retrieve models with similar descriptors. Next, we verify if the retrieved models have the same label as the query and compute the precision and recall for the result. We present the average precision and recall for all categories of shapes in Figure 11(a), and the supplementary material presents the individual curves for each class. Figure 12 shows examples of retrieved shapes for selected queries.

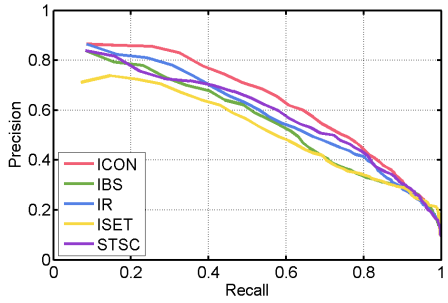
We see that ICON returns 20% of objects in the same category with a precision of almost 90%. The first two rows in Figure 12 show examples of how the top shapes retrieved by ICON have a similar functionality, even if their geometry as well as the number and location of interacting objects vary significantly. In the third row we see a failure case where several false positives (strollers) are returned for a handcart query. This result clearly occurs because of the ambiguity among these two classes, as also seen in the MDS in Figure 10(a), since both types of objects are used for carrying "items" and are pushed by humans.

Comparison to alternative descriptors. In Figure 11(a), we also compare the retrieval performance of ICON against alternative approaches, which we describe below.

Shape descriptor. As a baseline comparison to existing shape descriptors, we evaluate the retrieval performance of the light-field descriptor (LFD) [Chen et al. 2003], which is a popular descriptor for shape retrieval. LFD estimates the similarity of two shapes purely from their geometric appearance, without considering the interactions with other objects. We compute LFD only for the central object in each scene. We see that the performance of LFD is inferior to ICON, with a precision under 60% for a recall of 20%. However, we noticed that for a few classes such as Desk and Table, the precision rates for LFD are higher (the results for all classes are available in the supplementary material). Thus, a natural question



(a) *Comparison to other descriptors*



(b) *ICON's design aspects*

Figure 11: *Evaluation of ICON in terms of precision and recall for retrieval on our dataset of scenes. Each curve shows the average over all 10 categories in our dataset. (a) Note the better performance of ICON and ICON combined with LFD, compared to a shape descriptor (LFD) or scene hierarchies built with IBSH. (b) Note that the full ICON is 5% to 10% better than using only its individual components (IBS or IR), no hierarchy (ISET), or an alternative hierarchy (STSC).*

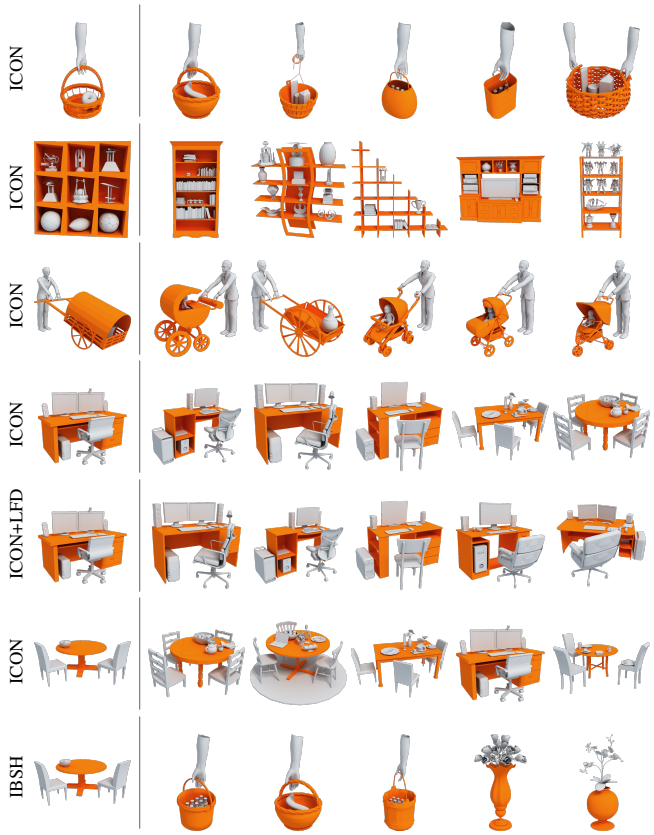


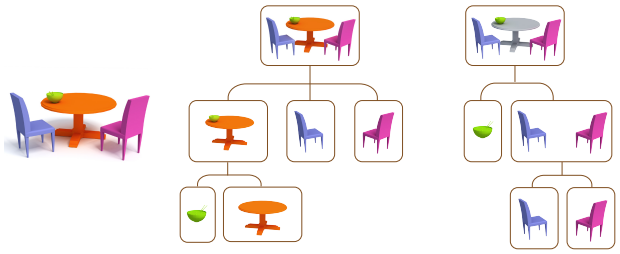
Figure 12: *Examples of retrieval on our dataset with ICON and other descriptors. In each row, the query is to the left, while the top-5 results are on the right. The central object is colored in orange. Note in the last four rows how combining ICON and LFD improves the accuracy of the results for the desk, and the results for the dining table are more accurate for ICON than for IBSH.*

is whether combining the power of our contextual descriptor with a shape descriptor can yield the best of both approaches. We see in Figure 11(a) that when ICON and LFD are combined with a weight of 0.5 each, we indeed obtain overall better results. This is also illustrated by comparing the 4th and 5th rows in Figure 12, where we see that the top-5 results for a desk query are improved by the combination of descriptors. Thus, this is certainly the best option for retrieval of scenes when invariance to the geometry of the central objects is not necessary.

IBS hierarchy. Zhao et al. [2014] propose to use the IBS to hierarchically group objects in a scene, a method we denote IBSH in our results. The hierarchy provides a more complete descriptor for scene comparison than using only the IBS between pairs of objects. In their work, they build a hierarchy iteratively in a bottom-up fashion. Two objects are clustered together depending on the fraction of an IBS shared between the objects. Next, to compare two central objects, they find a path from the leaf node containing the central object to the root node of the hierarchy. The central object is then represented by several sets of IBSs on different levels along this path. Given a number of levels, the two central objects are then compared by computing the distance between the features of the IBSs at each level of the tree.

Figure 11(a) shows the results of performing retrieval based on two (the maximum number of) levels in the IBS hierarchies. We observe that the performance of ICON is overall better, as IBSH retrieves models with a precision under 60% for a recall of 20%. By comparing the 6th and 7th rows in Figure 12, we see an example of how the objects retrieved by ICON for a dining table are in the same category, while the objects returned by IBSH are not meaningful. Although we also build an initial hierarchy in a bottom-up fashion, we cluster interactions based on a series of features, such as the type of interaction and features of the interaction regions. Our focus is in organizing the interactions rather than the objects themselves, which yields a descriptor more suitable for functionality comparison. In Figure 13, we show examples of the ICON and IBS hierarchies built for the same example scene, to better contrast this difference.

Agent pose. We compare ICON to the use of *poses of a human agent* as an alternative representation of functionality. Specifically, we compare to the ground-truth agent poses in the dataset of Kim et al. [2014], which represent the case where the human pose would be optimally recovered by an affordance model. We do not compare ICON directly to agent-based approaches since the assumptions of these approaches on the input are different. To retrieve models that are similar to a query, we order the objects according to the similarity of the agent pose in the scene to the agent pose in the query. The similarity between two poses is estimated from the average distance between joint locations (after optimal rigid pose alignment), as described by Kim et al. [2014]. To evaluate ICON, we process the scenes to fit a geometric model of a human agent to each pose. In this manner, we are able to extract the interactions between the objects and the agent.



(a) *Input scene* **(b)** *IBS hierarchy* **(c)** *ICON hierarchy*

Figure 13: *Comparison of the hierarchy generated for the input scene in (a) with IBSH in (b) and ICON in (c). Notice how ICON clearly groups similar interactions of the central object together, while IBSH groups the objects by the extent of the interaction, yielding a less meaningful grouping for functionality analysis.*

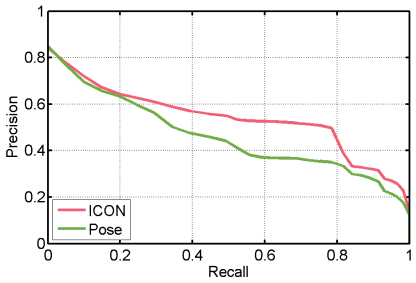


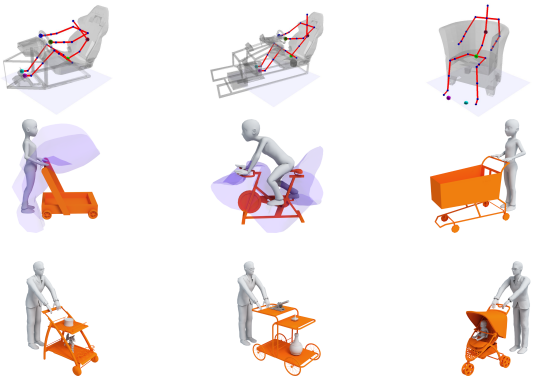
Figure 14: *Precision and recall for retrieval on the dataset of Kim et al. [2014], where each scene consists of a human agent interacting with a single object. Each curve shows the average over all 6 categories in the dataset. Note how ICON is comparable to poses.*

Figure 14 shows the results of this comparison as an average for all classes. We note that the two approaches are comparable in practice, since the difference on the performance of both methods is not too significant, especially up to a recall of 20%. The curves for each individual class are shown in the supplementary material, where we see that ICON performs better in 4 of the 6 classes in their dataset, although with a small difference. To provide some insight on where the two methods differ, we present in the first two rows of Figure 15 examples of shapes retrieved by both methods, picked from the two categories that have the most difference in performance for each method (Cockpit and Cart). We see that the similarity of two poses can sometimes retrieve models from the wrong class, while the IBS computed for ICON is sometimes not discriminative enough (both surfaces have a saddle-like shape in the example) and thus can lead to incorrect results.

We also compare the use of the agent pose to retrieve objects in our dataset. For this experiment, we fit a pose to the human agents in our scenes to enable the comparison of poses. The last row of Figure 15 shows examples of the retrieved shapes. We see that an agent pose can only capture one interaction and leads to incorrect results when the characterization of the object depends on multiple interactions. Thus, the advantage of ICON is that it can be more general by capturing interactions beyond those of an agent, although it requires the interactions to appear in the scene.

ICON's evaluation. We evaluate now how the different aspects of ICON contribute to capturing the interactions of the central object.

Interaction representation. Interactions in ICON are represented with a combination of IBS and IR. Since Zhao et al. [2014] evaluated the different features that describe the IBS and showed that each one is necessary for accurate discrimination between different types of surfaces, here we verify mainly the impact that the IR has in the representation. More specifically, we evaluate the retrieval performance when only IBS or IR are used with ICON. We see in Figure 11(b) that combining both IBS and IR leads to better retrieval results in general. Thus, we conclude that the IR complements the description of interactions captured by the IBS.



(a) *Query* **(b)** *ICON* **(c)** *Pose*

Figure 15: *Examples of objects retrieved with ICON and agent poses for the queries in (a). The objects in (b) appear before (c) in the ranking of ICON, while the objects in (c) appear before (b) in the ranking of poses. Note how ICON retrieves objects in the same class before poses (rows 1 and 3), but sometimes also leads to incorrect results where the agent pose is more effective (row 2). The purple surfaces in the second row are the IBSs computed by ICON, which are similar for these two shapes.*

Interaction organization. Interactions in ICON are organized in a hierarchical manner to allow a cross-comparison of the descriptors. We study the performance in retrieval when the interactions are stored with alternative structures. First, we evaluate the case where the interactions are stored simply as an unordered set. For comparing two sets of interactions $\mathscr{I}$ and $\mathscr{I}'$, we use the similarity measure proposed by Zhao et al. [2014]:

$$\text{Similarity}(\mathscr{I},\mathscr{I}') = \frac{K(\mathscr{I},\mathscr{I}')}{\max[K(\mathscr{I},\mathscr{I}),K(\mathscr{I}',\mathscr{I}')]},$$
$$\text{with } K(\mathscr{I},\mathscr{I}') = \sum_{\mathcal{S}\in\mathscr{I}}\sum_{\mathcal{S}'\in\mathscr{I}'} d_{\mathcal{S}}(\mathcal{S},\mathcal{S}'). \tag{5}$$

Figure 11(b) shows that the use of a hierarchical representation leads to overall better results compared to the use of a set.

Next, we compare our hierarchy construction to an alternative: we build a hierarchy by applying self-tuning spectral clustering (STSC) [Zelnik-Manor and Perona 2004] to define the branching at each level of the tree. STSC automatically scales the data and selects the number of clusters by analyzing the data, and is thus a good comparison to our approach that incorporates some problem-specific knowledge in the thresholds for hierarchy construction. We perform the comparison only on classes where the scenes involve three or more interactions, since only then the resulting hierarchies are different. In Figure 11(b) we see that our construction leads to slightly better performance. Thus, we conclude that the hierarchical organization of interactions is an important component to ensure the generality of the descriptor, although the specific construction algorithm may still be open to improvements.

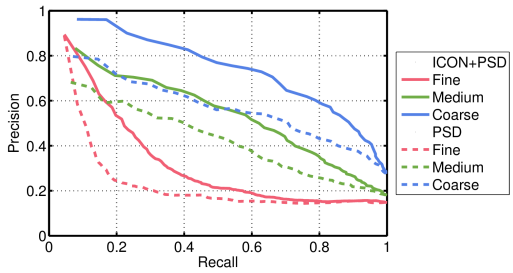


Figure 16: *Precision and recall for part retrieval on a dataset of chairs when considering different segmentation levels. Note the better performance of ICON combined with a part-level shape descriptor (PSD), compared to using only PSD.*

Figure 17: *Segmentation of interacting regions and shapes that support multiple interactions (hybrids). Given the input scenes in (a), we obtain the segmentations of the central objects in (b). Next, we match the parts that support similar interactions (shown in blue) and transfer the other regions (green) from each shape on the left in (b) to each shape on the right in (b), to obtain the hybrids in (c). Note how the hybrids resemble the real-world designs shown in the red boxes.*

Parameters. The key parameters used by our method are the following. The weights in the interaction distance are set to $w = 0.6$, $u_1 = v_1 = 0.3$, $u_2 = v_2 = 0.4$, and $u_3 = v_3 = 0.3$, the thresholds in the tree merging are $\theta_{up} = 0.6$, $\theta_{mid} = 0.5$ and $\theta_{low} = 0.2$, the tree comparison is based on $\tau = 0.5$, and the symmetry-aware distance uses $\rho = 4$. All results shown in the paper and used for comparison are produced using the same parameter setting.

Statistics and timing. We describe some relevant statistics of our data and the time needed to run the different components of our method with an Intel i7 CPU with 3.4GHz and 16GB of RAM. The average number of interacting objects in all of our scenes is 7, while the Desk, Shelf, Table, and TV Bench categories contain more complex scenes with 12 interacting objects on average. The average depth of the trees in ICON is 2.77 for all scenes and 3.77 for the complex categories. Building an ICON descriptor takes around 1min for our most complex scene involving 12 interacting objects, where 90% of the time is spent on the IBS computation that involves around 90K sample points. The average time for computing a single IBS is 7s. For simpler scenes, like a basket involving 4 interacting objects, computing an ICON takes 13s. Comparing two ICON descriptors with the tree matching is then relatively fast, taking on average only 2ms for all of our scenes.

Part-level descriptor. To show the potential of our preliminary extension of ICON to the domain of shape parts, we perform an evaluation of the part-level descriptor on a set of shapes from the same class. We segment a set of chairs into three segmentations with increasing level of detail. Note that in the fine-level segmentation, all the parts such as legs and armrests are captured by separate segments. In the medium-level segmentation, all these parts are treated as a single segment, while the coarse-level segmentation has less segment labels. Next, we compute the precision and recall of part-level retrieval using the part-level shape descriptors in the work of Laga et al. [2013] combined or not with ICON. We perform the experiment at the three different segmentation levels, to ensure that the evaluation is not biased by the size of the segments.

Figure 16 shows the result of this experiment summarized for all part labels. We present the curves for each label and segmentations in the supplementary material. We observe that ICON combined with part-level shape descriptors leads to overall better results than when only descriptors are used. Thus, our conclusion is that combining ICON with more elaborate part-level comparison schemes, such as one considering the context of parts as proposed by Laga et al. [2013], should also lead to improved results for these methods.

Segmentation of interacting regions. The goal of shape segmentation is typically to partition shapes into meaningful semantic parts. Here, we explore ICON's potential in enabling a different type of segmentation, where we segment a shape into regions that serve different interactions. Such type of segmentation cannot be inferred from an object alone, since it requires information on how the object interacts with other objects, which is the type of information captured by ICON. Such a segmentation can then facilitate subsequent tasks where we wish to modify the interactions supported by a shape, e.g., transferring interacting regions from one shape to another.

To perform the segmentation for a given shape, we use ICON to infer the number of interacting regions, and then refine the regions with the use of graph cuts [Boykov et al. 2001]. To select the number of regions, we first pick one ICON hierarchy, specifically the one that matches most of the hierarchies for the other objects in the same category. Next, we take each node in the first level of the tree as one possible label in the segmentation. We use the IR weights defined on the surface of the shape for these labels as the unary data term. To select the initial labels, we assign the label with highest weight to each triangle. For those triangles whose weights for all the labels are zero, we assign the label of its nearest labeled triangle. We define the pairwise smoothness term as the dihedral angle between two triangles. The cost between two different labels follows the Potts model, being set to 1 for each pair of different labels, or 0 otherwise. We then refine the initial labeling with α-expansion iterations to obtain the final segmentation.

In Figure 17(b) we show a few preliminary segmentations on different shapes. We see how the computed segments are regions of the shapes that enable certain interactions, such as *holding*, *grasping* and *pulling*, although only one region is obtained if the object supports one type of interaction. Moreover, the segmentation has a similarity to the regions extracted by SceneGrok [Savva et al. 2014], although their approach involves human agents and is based on supervised learning that requires a significant amount of training data.

Transfer of interacting regions and shape synthesis. The segmentation of interacting regions can potentially be used in a synthesis context to transfer regions from one shape to another to create novel shape variations. We explore this aspect to create objects that support different interactions as shown in Figure 17. To create these hybrid objects, we first match two regions that support the same type of interaction in two different shapes, and then transfer

(a) *Data dependence* **(b)** *Positions of interactions*

Figure 18: *Limitations of ICON. (a) Data dependence is a limitation when the input scenes have interactions that are scarce or inadequately describe the central objects, e.g., the bike and stand are recognized as similar because the only interactions present in the scenes are hanging bags. (b) We do not encode the relative positions of interactions on the central object, to render ICON oblivious to specific geometries of objects. However, in some cases the position is the only cue to distinguish the type of object, e.g., the basket and wheelbarrow are recognized as similar, while the position of the hands would be able to tell them apart.*

the remaining regions from a source shape to a target shape. The transferred regions can either be combined with the remaining regions of the target or substitute them. The final stitching of the regions is manually improved by an artist. Note how we are able to transfer the wheels of a handcart to a basket, to create a basket that can be moved in multiple manners, resembling a design found in supermarkets. The use of ICON in these examples allows us to discover the regions that serve different interactions, so that these can be transferred from one shape to another while still preserving the associated functionality. For example, the multiple wheels in the handcart are described by a single region that can be transferred as a whole to the cabinet and preserve the motion functionality.

7 Discussion, limitations, and future work

We introduce a novel contextual descriptor, the interaction context (ICON), that is designed to describe the interactions of an object in the context of a surrounding scene. Our description of interactions is geometric; specifically, it encodes the geometry of interactions between the central object and the surrounding objects and organizes these interactions into a hierarchy. This contextual descriptor of interactions has much potential in being used to analyze the functionality of an object. However, in reality, the majority of shape collections provide objects in isolation, and datasets like the one we used are not as common. Nevertheless, we foresee that as ongoing efforts in modeling continue, these datasets will increase in complexity and incorporate more realistic details in the future.

We have demonstrated that ICON can serve several applications that either necessitate or can benefit from functional analyses of objects, such as object retrieval, and has the potential of constituting the basis of a geometric functionality descriptor. However, ICON is not designed to encompass all functionalities, like those offered by a smart phone (non-geometrical) or a human hand (with dynamic movements). In our setting, we focus on interactions that can be derived from the geometry of multiple objects in a scene.

Additional features. We represent interactions with a combination of IBS and IR and showed that this representation leads to good discrimination between different objects. However, there are still cases where either these entities or their features are not descriptive enough to distinguish certain types of interactions, as seen in Figure 18(b). Hence, incorporating additional features to describe these regions could lead to improvements, e.g., encoding the relative position of the interactions around or on the surface of the central object, although it has to be balanced with ICON becoming more sensitive to the specific geometry of the shapes.

Functionality abstraction. We note again that ICON is a contextual descriptor for a given object, not an abstract description such as "to support" or "to cover". Nevertheless, we believe that ICON can also be used to learn a model for abstract interactions. A standard approach would be to label interactions (pairs of interacting objects) in a set of scenes with abstract labels, and then learn a model that predicts labels based on ICON descriptors. However, an indirect inference would also be possible, where we only provide the labels of interactions that appear in the scene and, besides learning a predictive model, we also infer what objects give rise to the different interactions, similarly to annotation of image regions from keywords [Duygulu et al. 2002].

Co-analysis of interactions. ICON should also be applicable in a co-analysis setting. Instead of relying on geometric or structural descriptors, like in the works of Sidi et al. [2011], Huang et al. [2011], etc., we can complement or replace the descriptors with ICON to achieve an analysis of common interactions in a set. Moreover, we demonstrated that ICON is data-dependent and built from a single scene with interactions. The use of a set may also allow us to derive a single ICON descriptor that aggregates the different types of interactions of a central object present in various scenes. The interactions could then be represented with a probabilistic model, which would be beneficial for circumventing the limitation shown in Figure 18(a).

Part-level ICON. We presented a preliminary version of a part-level ICON to demonstrate its potential as a descriptor of shape parts. However, there are further challenges specific to part representations. For example, the description of interactions with IBS and IR is suitable for object-to-object interactions, while in the case of parts we would like to extend the concept of IBS to represent the interactions of multiple neighboring parts in a unified surface.

Future work. Besides the directions for future work suggested above, there are other aspects of ICON that could be investigated in further detail. For example, we proposed one hierarchical representation that leads to a meaningful grouping of interactions. However, this may not be the ultimate representation; other clustering algorithms can be investigated. Furthermore, we showed the potential of ICON for shape synthesis by creating objects with hybrid functionality. In future work, the object-level and part-level ICON descriptors could be combined to yield a more complete system for suggesting novel shape designs. Finally, we would also like to further explore the matching provided by the ICON descriptors, which could enable applications such as functional correspondence of parts and objects across shape and scene collections.

Acknowledgements

We sincerely thank the reviewers for their comments, suggestions, and the tremendous effort they put in during iterations of the paper in the revision phase, leading to its final functional form. Thanks also go to Hadar Averbuch-Elor for being the voice in the video and for her careful proofread of the paper. This work was supported in part by grants from NSERC (No. 611370), GRAND NCE, NSFC (61232011, 61222206), National 973 Program (2014CB360503), and Shenzhen Key Lab (CXB201104220029A).

References

BAR-AVIV, E., AND RIVLIN, E. 2006. Functional 3D object classification using simulation of embodied agent. In *British Machine Vision Conference*, 32:1–10.

BELONGIE, S., MALIK, J., AND PUZICHA, J. 2002. Shape matching and object recognition using shape context. *IEEE Trans. Pat. Ana. & Mach. Int. 24*, 4, 509–522.

BOGONI, L., AND BAJCSY, R. 1995. Interactive recognition and representation of functionality. *Computer Vision and Image Understanding 62*, 2, 194–214.

BOYKOV, Y., VEKSLER, O., AND ZABIH, R. 2001. Fast approximate energy minimization via graph cuts. *IEEE Trans. Pat. Ana. & Mach. Int. 23*, 11, 1222–1239.

CAINE, M. 1994. The design of shape interactions using motion constraints. In *IEEE Conference of Robotics and Automation*, vol. 1, 366–371.

CHEN, D.-Y., TIAN, X.-P., SHEN, Y.-T., AND OUHYOUNG, M. 2003. On visual similarity based 3D model retrieval. *Computer Graphics Forum 22*, 3, 223–232.

DUYGULU, P., BARNARD, K., DE FREITAS, N., AND FORSYTH, D. 2002. Object recognition as machine translation: Learning a lexicon for a fixed image vocabulary. In *Proc. Euro. Conf. on Comp. Vis.*, 97–112.

FISHER, M., SAVVA, M., AND HANRAHAN, P. 2011. Characterizing structural relationships in scenes using graph kernels. *ACM Trans. on Graph (SIGGRAPH) 30*, 4, 34:1–12.

GRABNER, H., GALL, J., AND VAN GOOL, L. 2011. What makes a chair a chair? In *Proc. IEEE Conf. on Comp. Vis. and Pat. Rec.*, 1529–1536.

GUPTA, A., KEMBHAVI, A., AND DAVIS, L. S. 2009. Observing human-object interactions: Using spatial and functional compatibility for recognition. *IEEE Trans. Pat. Ana. & Mach. Int. 31*, 10, 1775–1789.

HUANG, Q., KOLTUN, V., AND GUIBAS, L. 2011. Joint shape segmentation with linear programming. *ACM Trans. on Graph (SIGGRAPH Asia) 30*, 6, 125:1–12.

JOHNSON, A., AND HEBERT, M. 1999. Using spin-images for efficient multiple model recognition in cluttered 3D scenes. *IEEE Trans. Pat. Ana. & Mach. Int. 29*, 5, 433–449.

KIM, V. G., CHAUDHURI, S., GUIBAS, L., AND FUNKHOUSER, T. 2014. Shape2Pose: Human-centric shape analysis. *ACM Trans. on Graph (SIGGRAPH) 33*, 4, 120:1–12.

LAGA, H., MORTARA, M., AND SPAGNUOLO, M. 2013. Geometry and context for semantic correspondence and functionality recognition in manmade 3D shapes. *ACM Trans. on Graph 32*, 5, 150:1–16.

LIU, Z., XIE, C., BU, S., WANG, X., HAN, J., LIN, H., AND ZHANG, H. 2014. Indirect shape analysis for 3D shape retrieval. *Computer & Graphics 46*, 110–116.

MITRA, N. J., GUIBAS, L., AND PAULY, M. 2006. Partial and approximate symmetry detection for 3D geometry. *ACM Trans. on Graph (SIGGRAPH) 25*, 3, 560–568.

MITRA, N., WAND, M., ZHANG, H. R., COHEN-OR, D., KIM, V., AND HUANG, Q.-X. 2013. Structure-aware shape processing. In *SIGGRAPH Asia 2013 Courses*, 1:1–20.

PECHUK, M., SOLDEA, O., AND RIVLIN, E. 2008. Learning function-based object classification from 3D imagery. *Comput. Vis. Image Underst. 110*, 2, 173–191.

RIVLIN, E., DICKINSON, S. J., AND ROSENFELD, A. 1995. Recognition by functional parts. *Comput. Vis. Image Underst. 62*, 2, 164–176.

SAVVA, M., CHANG, A. X., HANRAHAN, P., FISHER, M., AND NIESSNER, M. 2014. SceneGrok: Inferring action maps in 3D environments. *ACM Trans. on Graph (SIGGRAPH Asia) 33*, 6, 212:1–10.

SIDI, O., VAN KAICK, O., KLEIMAN, Y., ZHANG, H., AND COHEN-OR, D. 2011. Unsupervised co-segmentation of a set of shapes via descriptor-space spectral clustering. *ACM Trans. on Graph (SIGGRAPH Asia) 30*, 6, 126:1–10.

SONG, H. O., FRITZ, M., GU, C., AND DARRELL, T. 2011. Visual grasp affordances from appearance-based cues. In *ICCV Workshops*, 998–1005.

STARK, L., AND BOWYER, K. 1996. *Generic Object Recognition Using Form and Function*. World Scientific.

SUTTON, M., STARK, L., AND BOWYER, K. 1994. GRUFF-3: generalizing the domain of a function-based recognition system. *Pattern Recognition 27*, 12, 1743–1766.

TEVS, A., HUANG, Q., WAND, M., SEIDEL, H.-P., AND GUIBAS, L. 2014. Relating shapes via geometric symmetries and regularities. *ACM Trans. on Graph (SIGGRAPH) 33*, 4, 119:1–12.

TORSELLO, A., HIDOVIC-ROWE, D., AND PELILLO, M. 2005. Polynomial-time metrics for attributed trees. *IEEE Trans. Pat. Ana. & Mach. Int. 27*, 7, 1087–1099.

VAN KAICK, O., XU, K., ZHANG, H., WANG, Y., SUN, S., SHAMIR, A., AND COHEN-OR, D. 2013. Co-hierarchical analysis of shape structures. *ACM Trans. on Graph (SIGGRAPH) 32*, 4, 69:1–10.

WANG, Y., XU, K., LI, J., ZHANG, H., SHAMIR, A., LIU, L., CHENG, Z., AND XIONG, Y. 2011. Symmetry hierarchy of man-made objects. *Computer Graphics Forum (Eurographics) 30*, 2, 287–296.

XU, K., MA, R., ZHANG, H., ZHU, C., SHAMIR, A., COHEN-OR, D., AND HUANG, H. 2014. Organizing heterogeneous scene collection through contextual focal points. *ACM Trans. on Graph (SIGGRAPH) 33*, 4, 35:1–12.

ZELNIK-MANOR, L., AND PERONA, P. 2004. Self-tuning spectral clustering. In *NIPS*, vol. 17, 1601–1608.

ZHAO, X., WANG, H., AND KOMURA, T. 2014. Indexing 3D scenes using the interaction bisector surface. *ACM Trans. on Graph 33*, 3, 22:1–14.

ZHENG, Y., COHEN-OR, D., AND MITRA, N. J. 2013. Smart variations: Functional substructures for part compatibility. *Computer Graphics Forum (Eurographics) 32*, 2pt2, 195–204.

ZHU, Y., FATHI, A., AND FEI-FEI, L. 2014. Reasoning about object affordances in a knowledge base representation. *Lecture Notes in Computer Science (Proc. ECCV) 8690*, 408–424.