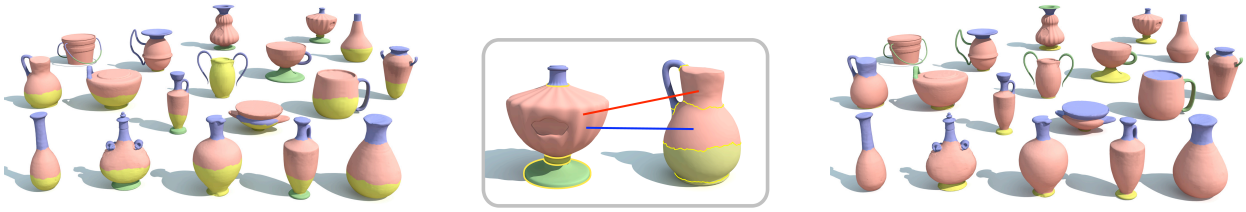
Active Co-Analysis of a Set of Shapes

Yunhai Wang* Shmulik Asafi† Oliver van Kaick‡ Hao Zhang‡ Daniel Cohen-Or† Baoquan Chen*
*Shenzhen VisuCA Key Lab/SIAT †Tel-Aviv University ‡Simon Fraser University



(a) (b) (c)

Figure 1: *Overview of our active co-analysis: (a) We start with an initial unsupervised co-segmentation of the input set. (b) During active learning, the system automatically suggests constraints which would refine results and the user interactively adds constraints as appropriate. In this example, the user adds a cannot-link constraint (in red) and a must-link constraint (in blue) between segments. (c) The constraints are propagated to the set and the co-segmentation is refined. The process from (b) to (c) is repeated until the desired result is obtained.*

Abstract

Unsupervised co-analysis of a set of shapes is a difficult problem since the geometry of the shapes alone cannot always fully describe the semantics of the shape parts. In this paper, we propose a semi-supervised learning method where the user actively assists in the co-analysis by iteratively providing inputs that progressively constrain the system. We introduce a novel constrained clustering method based on a spring system which embeds elements to better respect their inter-distances in feature space together with the user-given set of constraints. We also present an active learning method that suggests to the user where his input is likely to be the most effective in refining the results. We show that each single pair of constraints affects many relations across the set. Thus, the method requires only a sparse set of constraints to quickly converge toward a consistent and error-free semantic labeling of the set.

Keywords: semi-supervised learning, active learning

Links: DL PDF WEB VIDEO DATA

1 Introduction

Recently, there is an increasing interest in the co-analysis of sets of shapes, since current works have shown that more information can be extracted by simultaneously analyzing a set, rather than analyzing each shape individually [Golovinskiy and Funkhouser 2009; Xu et al. 2010; Sidi et al. 2011]. The main task in co-analysis is to simultaneously segment all the shapes in the set in a consistent manner, which is of great utility for modeling and texturing [Kalogerakis et al. 2010; Xu et al. 2011]. That is, besides partitioning the shapes into segments, we also obtain a labeling of the segments across the set, where the parts with the same label serve the same semantic purpose, albeit possibly being geometrically dissimilar.

Previous attempts to co-analyze a set of shapes can be classified into supervised and unsupervised. In the supervised setting [Kalogerakis et al. 2010; van Kaick et al. 2011], a training set with enough pre-analyzed shapes is assumed to be given. The training set is then used to probabilistically label a set of unknown shapes. Although supervised methods are not strictly speaking a co-analysis (since the shapes are not simultaneously analyzed), the result of the labeling is a consistent segmentation of the set. The unsupervised setting is more challenging, since no prior information is given and the entire knowledge must be extracted from the input set [Golovinskiy and Funkhouser 2009; Xu et al. 2010; Sidi et al. 2011]. In general, supervised methods have superior performance, but their performance hinges upon the relevance and quality of the training set.

In this paper, we consider the use of semi-supervised learning (SSL) for co-analysis. Semi-supervised methods [Zhu 2005] can be viewed as supervised methods with a rather small training set, but which also consider the latent information in the entire set. Alternatively, SSL methods can also be viewed as unsupervised methods assisted by rather minimal input coming out of the set (which we call *external input*). Viewed as augmented unsupervised methods, SSL methods outperform unsupervised methods at the minimal cost of requiring some educated external input. At the same time, an effective SSL method should outperform supervised methods in cases where the training set is too small, or poorly suited to the input shapes. Note that no method, be it supervised or unsupervised, can guarantee a perfect co-analysis of a set, since the geometry alone cannot always fully convey the semantics of parts. In particular, no descriptors can capture all possible geometric variations of a part.

Our contribution is the introduction of an SSL technique where the user interactively provides the external input as a means to iteratively correct and improve the result of the co-analysis (Figure 1). The external input consists of a sparse set of pairs of segments that the user marks as *must-link* or *cannot-link* constraints. Our SSL system is based on two main components. Firstly, we introduce a novel constrained clustering method based on a spring system which embeds elements to better respect both their inter-distances in a feature space and a given set of constraints. In our interactive setting, the user marks a few pairs of segments at each refinement

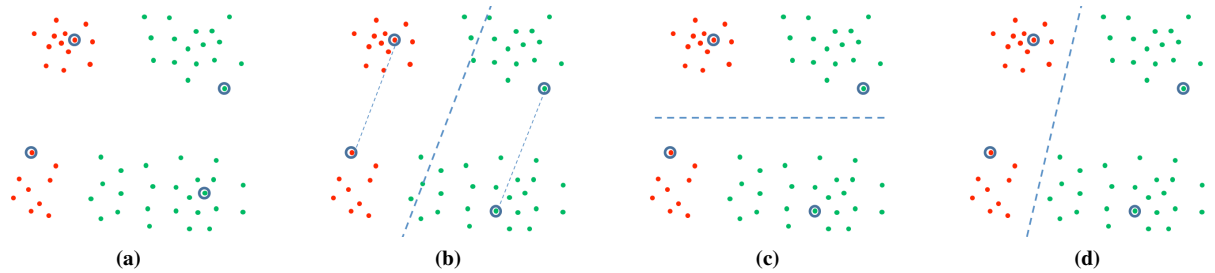


Figure 2: *Effectiveness of semi-supervised learning: (a) Consider two classes (red and green), where the data naturally forms four anisotropic clusters in a descriptor space. The sparse user-given training data is circled in blue. (b) A supervised method uses only the training data and finds the hyperplane that gives the maximum separation between training samples. Note that the classifier splits the clusters without taking their natural structure into account. (c) An unsupervised method takes the clusters into account but is oblivious to the training data. Thus, when asked for two clusters, the unsupervised method splits the samples to give the maximum cluster separation. (d) A semi-supervised method takes into account both the sparse training data and cluster structure, providing the correct result.*

step to progressively constrain the system. We show that each single pair of constraints affects many relations across the set. Thus, our proposed spring-based clustering makes efficient use of sparse constraints, as opposed to methods based on modifying only the similarities between elements, which typically require a higher number of constraints. Secondly, we develop a method that indicates to the user where his input is likely to be the most effective in refining the results. The method analyzes the clustering and identifies elements with low confidence regarding their cluster membership.

By combining these two components, we show that our SSL system makes the result quickly converge toward a consistent and error-free co-segmentation of the set. We show that this automatic analysis makes efficient use of user input, and that a sparse set of constraints typically suffices to yield a suitable co-analysis of the set. We present results on various sets of shapes exhibiting interesting geometric variations, and especially on large sets with 200 or more shapes, which would be challenging to manually segment or refine.

2 Background and related work

Our work is motivated by the strength of semi-supervised learning, as illustrated in Figure 2. In this section, we briefly present the background for this work and related research.

Semi-supervised learning. In supervised methods, a classifier is learned from labeled data (often referred to as *the training set*) and used to label unknown data. Often, labeled data is not available, and in general, it is expensive to create an effective training set. When the training set is rather small, as shown in Figure 2 (b), supervised methods typically yield poor results. Unsupervised methods, on the other hand, discover the latent structure of unlabeled data and analyze it by means of clustering. Since these methods are limited to the knowledge present in the unlabeled data, they are often unaware of any semantics, and thus do not necessarily convey the results expected by the user, as shown in (c). Semi-supervised learning (SSL) methods can be seen as a hybrid between the above two schemes, where the classification and analysis are performed based both on the labeled data and the structure of the unlabeled data. In semi-supervised methods, minimal user input can lead to a more meaningful classification of unknown data, as illustrated in (d). In many real-life problems, there is a large amount of unlabeled data, but only limited labeled data. Thus, SSL methods have become a topic of much interest [Zhu 2005].

Constrained clustering. SSL improves the performance of unsupervised clustering by considering only a small amount of labeled data or some constraints on the data. Clustering techniques where the external input is provided by constraints are referred to as constrained clustering. These techniques typically consider two types of constraints: *must-link* constraints, which specify that two samples should be in the same cluster, and *cannot-link* constraints, which specify that two samples must be in different clusters [Wagstaff and Cardie 2000]. These constraints have been incorporated into many clustering algorithms, such as K-means clustering [Wagstaff and Cardie 2000], hierarchical clustering [Klein et al. 2002], and clustering with Gaussian mixture models [Shental et al. 2004]. Since spectral clustering has gained much popularity in recent years [Shi and Malik 2000], adapting constraints to spectral clustering has also attracted considerable attention in the literature.

Many existing techniques directly modify the similarity matrix according to the given constraints and then perform spectral clustering [Kamvar et al. 2003; Kulis et al. 2005; Lu and Carreira-Perpinán 2008]. Rather than treating the constraints as hard constraints, Yu and Shi [2004], Coleman et al. [2008], and Li et al. [2009] treat them as soft constraints. By incorporating constraints into the objective function of spectral clustering, Wang and Davidson [2010b] propose a flexible constrained spectral clustering method, which can handle both binary and real-valued constraints.

Metric learning. Another important set of techniques for constrained clustering is based on *metric learning*, where the goal is to learn a distance metric that best fits a known set of data properties (such as pairwise constraints). Learning a meaningful distance metric is important to many data mining and computer vision tasks, such as content-based image retrieval, image tagging, and handwriting recognition. A complete review of the existing metric learning methods and their applications can be found in the survey of Yang and Jin [2006]. Here, we briefly describe a few representative works that are closely related to our method.

In the context of learning distances between high-dimensional features, traditional distance learning algorithms cannot be used due to overfitting and high computational complexity. Thus, Weinberger et al. [2006] make the problem tractable by first using Principal Component Analysis (PCA) to reduce the dimensionality of the data and then learning a metric in the resulting low-dimensional subspace. Torresani and Lee [2007] propose a method that unifies the objectives of dimensionality reduction and metric learning to achieve better learning accuracy.

Recently, semi-supervised metric leaning has been proposed, which exploits both the pairwise constraints and the structure of the unlabeled data [Hoi et al. 2008]. This method seeks to learn a linear mapping from the original feature space to a low-dimensional subspace. Our method follows a similar idea, but does not explicitly learn a mapping, which allows us to go beyond the linear case. Our method directly embeds the data into a new subspace where the (possibly non-linear) constraints are satisfied.

Active learning. In cases where it is expensive to obtain labeled data, active learning techniques were developed to query the user to label the most informative entries [Settles 2009]. Similarly, to minimize user effort, *active* constrained-clustering has been studied for suggesting pairwise constraints in order to maximize performance [Klein et al. 2002; Basu et al. 2004]. To build an active spectral clustering framework, Wang and Davidson [2010a] propose to choose the constraint with the largest expected error between the current cluster assignment and the user-specified input. With the assumption that the underlying clusters are nearly separated, Xu et al. [2005] identify points on the boundaries of clusters by examining the spectral clustering eigenvectors. Note that the two methods above are limited to two-class problems.

Learning from sets of shapes. Learning methods use the knowledge present in sets of shapes to solve classical problems, such as shape segmentation [Shamir 2008; Chen et al. 2009].

A variety of supervised methods have been developed in recent years and applied in many fields. In particular, the graphics community used supervised methods to segment shapes [Kalogerakis et al. 2010], establish a correspondence between two shapes [van Kaick et al. 2011], and detect geometrical features [Sunkel et al. 2011]. In these methods, the external user knowledge is captured by means of a training set. The training set is then used to learn models or classifiers that allow the method to reapply the knowledge.

Unsupervised methods for co-segmentation and co-analysis of a set have been presented in [Golovinskiy and Funkhouser 2009; Xu et al. 2010; Sidi et al. 2011; Hu et al. 2012]. Golovinskiy and Funkhouser [2009] pre-align all the shapes in the set and then cluster the shape faces according to an underlying graph. The graph links faces that are adjacent in the models and faces that are close-by after the alignment. The resulting clusters provide a natural co-segmentation of the shapes. Xu et al. [2010] factor out the scale variation in the shape parts by first classifying the shapes into different styles. In this manner, they are able to co-segment shapes with a higher level of variability. Sidi et al. [2011] pose the co-segmentation problem as that of clustering in a descriptor space, which allows their method to handle shapes with rich variations in part composition and geometry, where a rigid alignment scheme would not lead to a proper co-segmentation. Recently, Hu et al. [2012] present an alternative unsupervised co-segmentation method based on subspace clustering. Huang et al. [2011] also present a method where a set is used to assist in the segmentation of individual shapes. They can handle rich shape variations by using shape descriptors, without aiming at a consistent segmentation of the entire set. Note that, since the analysis is unsupervised in all of the methods above, it is entirely determined by the cluster structures either in the primal shape space or in a descriptor space.

3 Overview

We present an active SSL technique for co-analyzing a set of shapes, based on constrained clustering. The user interactively adds *must-link* or *cannot-link* constraints between segments, assisting the correction of the cross-shape segment clustering towards an error-free co-analysis of the set. The goal of the co-analysis is to obtain a segmentation and labeling of the shapes that is consistent across the set. The input shapes have a similar part composition and thus share a common label set. By clustering segments extracted from the shapes, we obtain a co-segmentation, since the segments are grouped into classes that represent the types of parts that exist in the set, where the number of clusters is provided by the user.

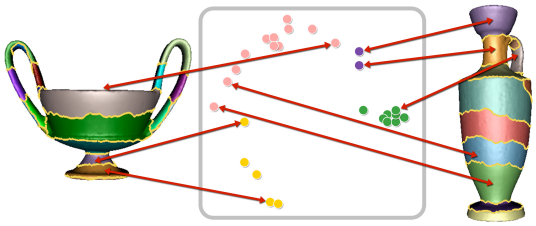


Figure 3: *Each super-face lives in two spaces: over the surface of the shapes (the primal space, shown on the shapes to the left and right) and in the feature space (the dual space, shown in the center).*

We assume that an unsupervised method already computed and labeled an initial segmentation of the given set (Figure 1 (a)). Each shape is then partitioned with K-means clustering into smaller segments, which we refer to as *super-faces*. Each such super-face is associated with a vector of shape descriptors. Thus, each super-face can also be seen as a point in a high-dimensional descriptor space. With the aid of multi-dimensional scaling (MDS), we reduce the high-dimensional descriptor space to a lower-dimensional feature space where our active SSL is carried out (Figure 3).

As shown in Figure 3, each super-face lives in two spaces: over the surface of the shapes (the primal space) and in the feature space (the dual space). The clustering is carried out in the feature space, while the primal space is where the user marks the constraints. Note that Figure 3 displays a real embedding in 2D. We see that although similar types of parts tend to be grouped together, the embedding does not form simple cluster arrangements. Thus, it is unlikely that an unsupervised method can provide an error-free analysis of the set. Since we aim at a semantic labeling of the parts, even ideal geometric descriptors cannot always fully capture the part semantics, and an active learning scheme is thus sought.

In the core component of our technique, the user interactively adds must-link or cannot-link constraints between super-faces. The introduction of every new constraint refines the clustering in the feature space and thus also the co-segmentation of the set (Figure 1 (b) and (c)). This interactive process is repeated until the user is satisfied with the result. The key challenges of this work are: (i) to maximize the effect of a single user input; and (ii) to ease the user on the task of identifying where his or her input is likely to be the most effective. To address these two challenges, we introduce: (i) a constraint re-embedding technique where the effect of adding a constraint (either must-link or cannot-link) is propagated to its environment in feature space and consequently improves the clustering (described in Section 4); and (ii) a method that analyzes the feature space and identifies elements that have high potential of being misclassified (discussed in Section 5). These elements are promptly highlighted to the user (as shown in Figure 1 (b)).

4 Sparsely-constrained re-embedding

The input to our active co-analysis is a set of super-faces extracted from the input shapes. Our method can start with the segments generated by any unsupervised co-segmentation algorithm. Specifically in our experiments, we use an algorithm similar to that of Sidi

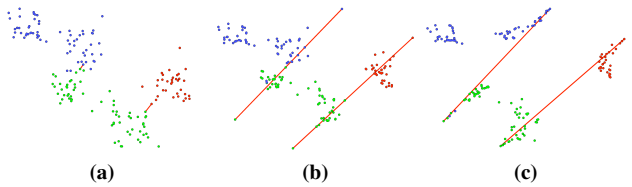


Figure 4: *Comparison between linear and non-linear spring systems: (a) Dataset with two cannot-link constraints (red lines). (b) Result of a linear spring-system, where constraint springs act like metric springs with higher spring constants. (c) Result of our non-linear spring-system, which better separates the clusters.*

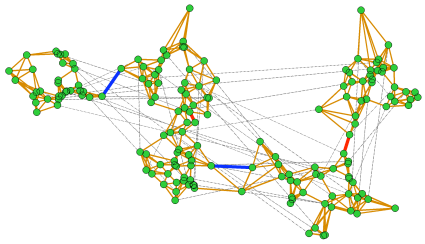


Figure 5: *The four types of springs used in our system: metric springs are shown in brown, must-link constraint springs in blue, cannot-link constraint springs in red, and repulsion springs in gray.*

et al. [2011], however, we break the segments into smaller super-faces. First, we cluster faces with K-means according to shape descriptors, where $K = 5 \times$ number of labels, to obtain an over-segmentation of the shapes. Next, we use graph cuts [Boykov et al. 2001] to refine the super-face borders. This procedure ensures that if a single segment incorporates parts that should be labeled differently, these will be split into separate super-faces (a situation that appears in the results of Sidi et al. [2011]). Thus, the user will not need to split super-faces when adding constraints. If we are given a different initial segmentation, we also break the segments into smaller super-faces with K-means. Finally, we cluster the super-faces into several classes of parts by using the descriptor-space spectral clustering approach [Sidi et al. 2011] to obtain the initial co-segmentation. The number of clusters is user-specified.

For the K-means clustering step, we use the same shape descriptors as Sidi et al. [2011], which are based on the position and orientation of faces, and also on the shape diameter function [Shapira et al. 2008]. As in Sidi et al., the descriptors are histograms that capture the distribution of these face-level descriptors across the super-faces. For computing the descriptors, we assume the shapes are in their upright orientation [Fu et al. 2008; Jin et al. 2012].

In the active co-analysis, we also use the same super-face descriptors, which result in a high-dimensional descriptor space. However, the high dimensionality of this space may result in overfitting and high computational complexity when performing the constrained clustering [Weinberger et al. 2006]. Hence, the super-faces are embedded into a feature space of reduced dimensionality with the use of multi-dimensional scaling (MDS), simplifying the complexity of the constrained clustering. We verified that the choice of dimension d for this space does not significantly affect the results of the algorithm, as long as the selected d is not too small. Thus, we choose $d = 5$ so that the space possesses enough degrees of freedom to allow for the convergence of the constrained-embedding algorithm.

Finally, at each iteration of the active co-analysis, the user interactively adds *must-link* (ML) or *cannot-link* (CL) constraints between two super-faces. This is carried out in an interface where the user clicks on two selected super-faces on the shapes. The interface also suggests pairs of super-faces that should receive a constraint to improve the clustering (described in Section 5). After new constraints are added to the current set of constraints, we execute the constrained-clustering algorithm on the feature space. The result of the clustering directly provides a co-segmentation of the set.

To perform the clustering, we first re-embed the super-faces into a space that better reveals the cluster structures and constraints. Our re-embedding method is based on the relaxation of a *spring system*, which moves the super-faces to new locations in this space. The new locations ensure that the result of applying an unconstrained clustering algorithm on the re-embedded super-faces will satisfy the constraints. Thus, we can apply any clustering algorithm on this space to obtain the final grouping of super-faces. In our case, we use the K-means algorithm with the number of clusters provided by the user. The details of our spring system follow next.

Spring system. The embedding of the super-faces is posed as the relaxation of a spring system. The spring system models the inter-element distances together with the constraints directly in the embedded space. The input to the system is a set of constraints and a matrix of pairwise distances between elements, which is derived from the positions of super-faces in the aforementioned 5D feature space. The spring system re-embeds the elements to better respect the given set of constraints. Then, any clustering algorithm (such as K-means) can be applied to obtain the final grouping of elements.

Each super-face is represented by a vertex in a fully-connected undirected graph G. The edges of the graph (E) model springs, whose relaxed lengths are set to the distance between the corresponding elements in the 5D feature space. These springs are called *metric springs* since they model the original feature space. This setting results in a spring system where each spring exerts a force when it is longer or shorter than its relaxed length (i.e., when it contains potential energy). Intuitively, a spring system "aims" to re-embed vertices in such a way that minimizes the total sum of potential energy; this re-embedding is called a *stationary state*.

Constraints are modeled by special *constraint springs* between the two constrained vertices. Cannot-link springs are set with a long relaxed length (the maximum distance between any two elements in the feature space) and they only exert a force when they are shorter than the relaxed length; Must-link springs are set with a short relaxed length (the minimum distance in the feature space) and they only exert a force when they are longer than their relaxed length.

Constraint springs must be persistent enough to overcome many metric springs and have a significant effect on the stationary state. To achieve this, we assign proper *spring constants* to each spring. The spring constant κ is a measure of how much force F a spring exerts when displaced one unit length from its relaxed length. According to Hooke's Law, $F = \kappa \Delta x$, where Δx is the spring displacement. We assign the arbitrary value $\kappa_{\text{metric}} = 1$ to metric springs; and assign a higher value of $\kappa_{\text{constraint}} = 10,000$ to constraint springs. This difference allows the constraints to have a significant effect on the stationary state of the spring system.

The objective function of the minimization can be written as a sum of the potential energy stored in each spring:

$$\begin{aligned}\text{Obj}(G) = &\sum_{\langle u,v\rangle \in M} \kappa_{(u,v)} \cdot \left(dis(u,v) - D(u,v)\right)^2 \\ + &\sum_{\langle u,v\rangle \in CL} \kappa_{(u,v)} \cdot \left(\max\left(0, D(u,v) - dis(u,v)\right)\right)^2 \\ + &\sum_{\langle u,v\rangle \in ML} \kappa_{(u,v)} \cdot \left(\max\left(0, dis(u,v) - D(u,v)\right)\right)^2,\end{aligned} \tag{1}$$

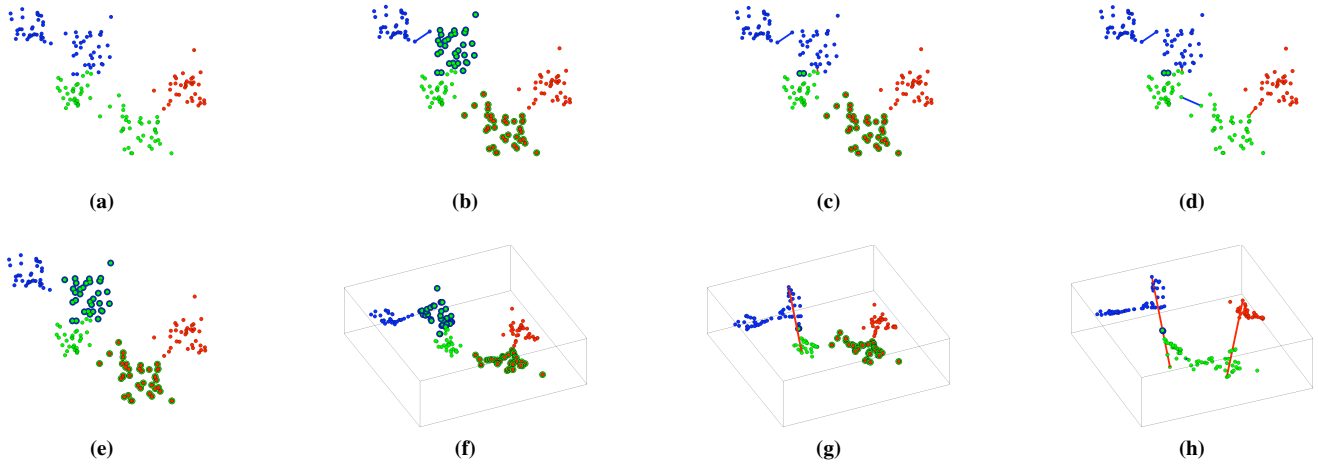


Figure 6: *Example of applying our spring-embedding clustering: (a) Input set of points, where the ground-truth is shown by the colors. (b)-(d) Four constraints are sequentially added to the spring system (must-link in blue and cannot-link in red). The number of misclassified points drastically reduces at the insertion of each new constraint. After adding four constraints in (d), the cluster assignment is close to error-free. The same steps are shown from a different viewpoint in (f)-(h). For reference, (e) shows the clustering obtained without any constraints, providing an imperfect result. Note that the line color of a point denotes the ground-truth, while the fill color indicates the clustering result.*

where $dis(u, v)$ is the distance between the positions of vertices u and v in the new embedded space; $D(u, v)$ is their distance in the original feature space; $\kappa_{(u,v)}$ is the constant of the spring between u and v; M is the set of metric springs, *CL* is the set of cannot-link springs, and *ML* is the set of must-link springs.

The conditional exertion of a force by constraint springs in (1), which can also be represented as a conditional potential energy function, makes the system non-linear and allows for a significantly greater degree of flexibility than a linear system. It also leads to a stationary state that is better at satisfying the constraints without damaging the local structure of the graph near the constrained vertices; this effect is key in achieving a successful clustering with a sparse set of constraints. This is exemplified in Figure 4: the linear system tries to exactly satisfy the constraints, so the result is more rigid (the shape of the clusters remains closer to the original), while the non-linear case allows more flexibility (the clusters are more deformed while still satisfying the metric and linking constraints).

Relaxation. The solution of the objective function is achieved by a series of relaxation iterations, where we calculate the force being exerted on the vertices by each spring according to Hooke's Law.

In order to avoid the quadratic time complexity of calculating the forces between all pairs of vertices, we build a sparse graph in which every vertex is connected to its K-nearest neighbors (KNN). We also add $O(N)$ randomly-chosen metric springs that are not included in the KNN sub-graph, which we call *repulsion springs*. We set $\kappa_{\text{repulsion}} = \kappa_{\text{metric}} = 1$, since the goal of these springs is to capture the *overall* interactions between distant vertices and avoid that they accidentally end up close to each other. We find this approximation effective since in a clustering problem we are interested in capturing exact distances mainly between proximal vertices, while the exact interactions between distant vertices are less important. Thus, the relaxation has overall time complexity $O(N) \times$ number of iterations, where there are N super-faces in the set. The types of springs used in our system are shown in Figure 5.

Note that we do not use the spring constants as "physical" factors, but rather employ them as weights in a weighted sum expressing the force exerted on each vertex v:

$$F_v = \sum_{\langle u,v\rangle \in E} \kappa_{(u,v)} \cdot (dis(u, v) - D(u, v)) \quad / \sum_{\langle u,v\rangle \in E} \kappa_{(u,v)}. \tag{2}$$

Moreover, since $\kappa_{\text{metric}} \ll \kappa_{\text{constraint}}$, it appears that the displacement of vertices affected only by metric springs is insignificant. However, in fact, if a vertex is not influenced directly by a constraint spring, then it is significantly displaced by the metric springs.

The relaxation iterations are stopped either when all the vertex displacements are below a constant threshold (convergence to a local minimum), or when the system has reached 1,000 iterations without convergence. We observed that this number of iterations is sufficient to yield good results for the co-analysis of small and large sets (Section 6), demonstrating the scalability of the relaxation. An example of the full execution of our spring embedding is shown in Figure 6. Note in (b)-(d) that, as more and more constraints are added to the spring system, the number of misclassified points drastically decreases. With four constraints, as shown in (d), the accuracy is close to error-free. On the other hand, clustering without constraints yields far from accurate results, as shown in (e).

Comparison with previous work. Previous work on constrained clustering has proposed methods that modify the matrix of pairwise similarities according to the input constraints, or methods that restrict the space of feasible solutions [Wang and Davidson 2010b]. The first class of methods provides a new similarity matrix that can be used with any clustering algorithm, while the second class restricts the choice of possible clustering solutions.

Our spring embedding clustering incorporates the constraints directly as requirements that the spring system, and thus the resulting clustering, have to satisfy. The advantage of such a re-embedding of the elements is that a small number of constraints is able to greatly improve the clustering result, while previous methods would require a larger number of constraints (on the order of hundreds for small inputs) to achieve a similar result. This advantage in the sparsity of the constraints helps to fulfill one of our design goals: minimize user effort in guiding the result toward an error-free co-segmentation. Figure 7 demonstrates the advantage of our spring

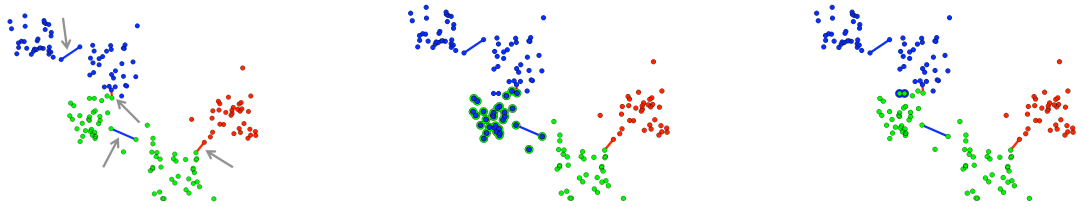


Figure 7: *Comparison between constrained-clustering techniques: (a) Dataset to which we add four constraints (highlighted by the arrows). Must-link constraints are colored blue and cannot-link constraints are red. (b) Spectral learning result (part of the green cluster is misclassified). (c) Our spring embedding result, which is more consistent with the ground-truth. Note that the line color of a point denotes the ground-truth, while the fill color indicates the clustering result.*

embedding method in comparison to the result of an alternative method (the spectral learning of Kamvar et al. [2003]).

5 Active Learning

By interactively adding constraints, the user refines the co-analysis towards an error-free co-segmentation of the set. At every iteration, new constraints are added, the spring-embedding is updated, and the super-faces are re-clustered. Moreover, to minimize the effort required from the user, at the beginning of each iteration, the system suggests pairs of points that when constrained are likely to improve the co-segmentation. The suggestions involve points that are far from their cluster centers, and which have a low confidence of belonging to their clusters, since their corresponding super-faces reside in regions of the embedding where the part class is more uncertain. The confidence is given by the *silhouette index* of a point.

Silhouette index. Let us assume that $\mathbf{x}$ is a point in the embedding which was assigned to the cluster C_k, and n_k is the number of points in this cluster, while K is the total number of clusters. Then, the *silhouette index* of $\mathbf{x}$ is defined as [Brun et al. 2007]:

$$S(\mathbf{x}) = \frac{b(\mathbf{x}) - a(\mathbf{x})}{\max[b(\mathbf{x}), a(\mathbf{x})]}, \tag{3}$$

where $a(\mathbf{x})$ is the average distance between $\mathbf{x}$ and all other points in its cluster C_k, defined as

$$a(\mathbf{x}) = \frac{1}{n_k - 1} \sum_{\mathbf{y} \in C_k, \mathbf{y} \neq \mathbf{x}} d(\mathbf{x}, \mathbf{y}), \tag{4}$$

and $b(\mathbf{x})$ is the minimum of the average distances between $\mathbf{x}$ and the points in other clusters, given by

$$b(\mathbf{x}) = \min_{\substack{h = 1, \ldots, K \\ h \neq k}} \left[\frac{1}{n_h} \sum_{\mathbf{y} \in C_h} d(\mathbf{x}, \mathbf{y}) \right]. \tag{5}$$

The silhouette index $S(\mathbf{x})$ is a value in the range $[0, 1]$ when the clustering results are produced by the K-means algorithm. $S(\mathbf{x})$ values close to 0 represent points with less confidence, since the points are located near to the cluster boundary and their membership is thus more uncertain.

Constraint suggestion. Now, given the silhouette index of each point, we can find points in the current clustering that have a low or high confidence of belonging to their assigned cluster. So, when suggesting constraints, the system asks the user to establish a link between a low-confidence point and a high-confidence point. However, some of the low-confidence points do not represent good suggestions since they already possess the correct label. This can happen when the point is far on average from the points in its cluster, but is close to its cluster center in the original feature space. So, we select as suggestions the low-confidence points that are closest to another cluster center in the original feature space. By connecting low-confidence to high-confidence points, the user can add a constraint which will connect an uncertain and incorrectly labeled region of the embedding with a region that has a definite label.

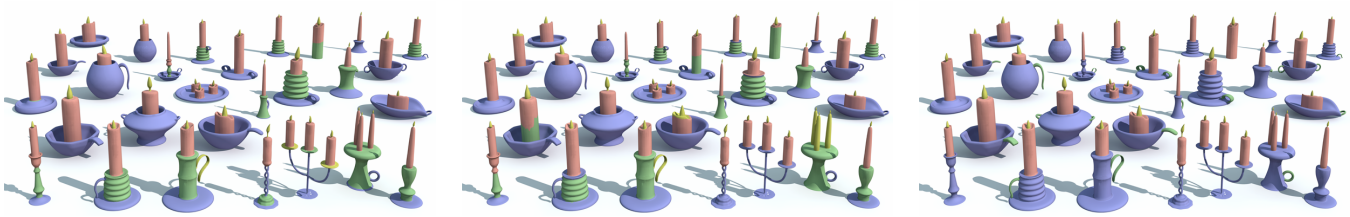
More specifically, this is carried out by sorting all the points according to their confidence $S(\mathbf{x})$. First, we select K points with high confidence (points with $S(\mathbf{x})$ values close to 1), where each point belongs to a different cluster. Next, we obtain the distances in the original feature space from all the points to the K selected points, and remove those points that are closest to their own clusters. Finally, we select several uncertain points ($S(\mathbf{x})$ values close to 0). The system then presents the shapes that contain these super-faces in the interface and asks the user to add must-link or cannot-link constraints by marking the super-faces. Notice that the user can add as many constraints as desired before re-clustering. However, since the spring embedding is fast, executing in less than 15 seconds for 3,000 super-faces, it is practical to re-cluster after adding only one or two constraints to the current set of constraints.

Interface and interactivity. In the interface, we present all the suggested super-faces organized by their confidence. We present the K certain super-faces in the center of a 4×4 grid, along with $16 - K$ uncertain super-faces disposed around the center. In this manner, the user can add must-link or cannot-link constraints between any desired pairs of super-faces and then update the clustering. The super-faces are shown highlighted on their shapes, while the shapes are colored according to the current labeling, so that the user can easily identify mislabeled super-faces. The accompanying video demonstrates the interface in more detail.

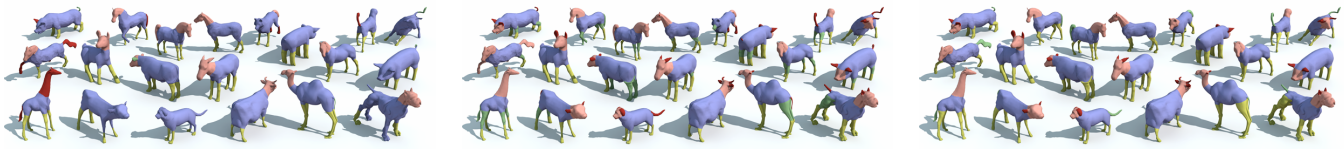
6 Results

In this section, we demonstrate the effectiveness of our active co-analysis in converging towards an error-free co-segmentation. We evaluate our spring-embedding clustering and also the active learning component of the algorithm. Furthermore, we compare to the results of both a supervised and an unsupervised method.

Datasets and ground-truth. We evaluate our active co-analysis on 11 sets of shapes which possess a ground-truth segmentation and

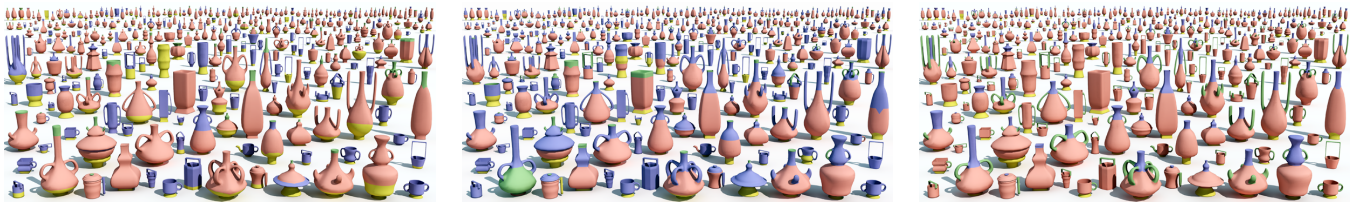


(a) *Candelabra: 28 shapes, 164 super-faces, 24 constraints*



(b) *Four-legged animals: 20 shapes, 264 super-faces, 69 constraints*

(c) *Large tele-alien set: 200 shapes, 1,869 super-faces, 106 constraints*



(d) *Large vase set: 300 shapes, 1,527 super-faces, 44 constraints*

Figure 8: *Results of our active co-analysis on various sets. For each set, we show the initial unsupervised co-segmentation (middle) and then the refined close to error-free co-segmentation (right), along with the minimal number of constraints needed to co-segment these sets. Note how the co-segmentations are consistent across the sets and how only a small number of constraints is necessary to achieve these results. For comparison, we also show the results of applying the method of Sidi et al. 2011 on these sets (left).*

labeling (Table 1). We used all the seven sets from the dataset of Sidi et al. [2011], where the labeling for the four-legged animals was prepared by Kalogerakis et al. [2010]. Note that, although our descriptors are designed for man-made shapes, our active co-analysis can still generate reasonable results for organic shapes when enough constraints are provided. We created three additional large sets, since we consider the labeling of large sets as one of the main motivations of our work, and we also created a small but challenging set of *irons*. The full dataset can be downloaded at `http://web.siat.ac.cn/~yunhai/ssl/ssd.htm`.

To compare to the ground-truth, similarly to Kalogerakis et al. and Sidi et al., we use a measure that captures the extent of a shape's area that is labeled correctly:

$$\text{Accuracy}(r, g) = \sum_i a_i \, \delta(r_i = g_i) \quad / \quad \sum_i a_i, \qquad (6)$$

were r are the labels assigned to the faces by the co-segmentation, g is the ground-truth labeling, a_i is the area of face i, and $\delta(x = y)$ is 1 only if $x = y$. The face labels r can be directly derived from the labeling of the super-faces. Next, we average the accuracies for all the shapes in the set to obtain the accuracy of the co-segmentation. However, notice that the resulting labels do not carry a semantic meaning as in the ground-truth. Thus, first we find a one-to-one mapping between the ground-truth labels and the resulting labels. This is achieved by selecting the mapping that is coherent for the whole set and gives the highest accuracy.

Active co-analysis. Figure 8 shows visual results of our active co-analysis on sets of shapes with rich geometric variations. For each set, we display the best co-segmentation obtained by one user, along with the minimal number of constraints needed to obtain the result (the best number of constraints achieved by one user in 10 trials). The results for the remaining sets as well as an enlarged view of the large sets are available in the supplementary material. Notice how close to error-free co-segmentations are obtained after adding only a small number of constraints, implying a small number of user interactions. More particularly, the results of our active co-analysis are more pronounced on large sets like the *large chairs*, *tele-aliens* and *large vases*. It would be challenging to manually segment and label these sets over 200 shapes, but they were refined in our SSL with respectively only 162, 106 and 44 constraints. The vases required only 44 constraints since, once the user added the key constraints linking the most dissimilar parts, the shape descrip-

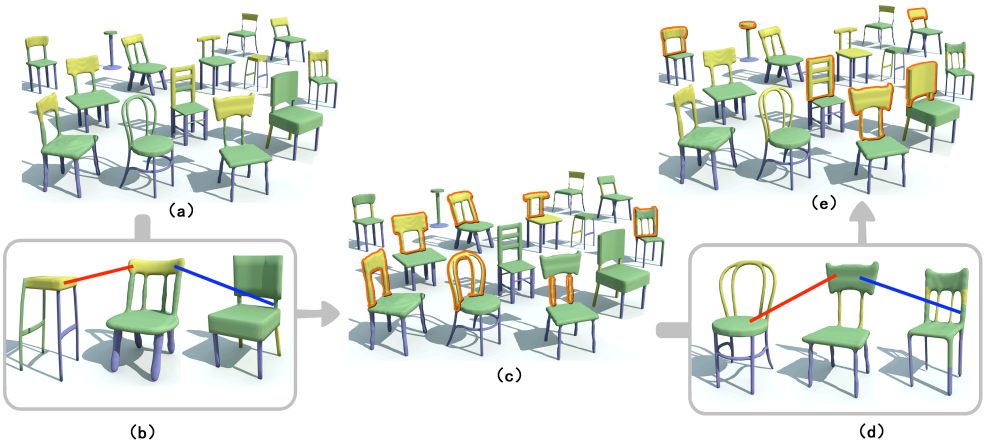


Figure 9: *Two iterations in the active co-analysis of a set, starting from the unsupervised result in (a). Notice how the constraints added by the user in (b) and (d) have a significant effect and refine the segmentation for several shapes (refined segments are marked in red).*

tors established the remaining part links. Note also that the set of tele-aliens is unfamiliar and so does not have immediate semantic labels that a user could employ to train a supervised method. On the other hand, the co-analysis does not have this semantic requirement and is able to automatically infer the common structure of the shapes. The experiments for all the sets are summarized in Table 1.

For comparison, Figure 8 also shows the co-segmentations obtained with a state-of-the-art unsupervised method [Sidi et al. 2011]. Note that, although the results of this method are superior to our initial segmentation in some cases, they are still far from error-free and are also surpassed by the results of the active co-analysis.

Moreover, Figure 9 presents two example iterations in our active co-analysis. Every constraint that is added by the user has a significant effect in improving the co-analysis of the set, refining the segmentation of several shapes in each iteration. In the supplementary material, we present example iterations for the large set of vases.

User study. To demonstrate the effectiveness of the active co-analysis independently of the specific user input, we performed a study where we asked 15 participants to co-segment two sets: candelabra (28 shapes) and a large set of vases (300 shapes). The users in our study were research assistants from diverse science backgrounds, who were never involved in this work. Before the test, we explained the goal of the co-segmentation and demonstrated how to use our system. Next, we asked the users to co-segment each set of shapes with and without suggestions. Each user repeated the two experiments for three times, without any time limit. The test did not start until the users successfully segmented the Goblet set, where four correct constraints can achieve the ground-truth. For the interface without suggestions, the users had to scroll through the models to find mislabeled segments and add appropriate constraints. However, the interface provided hints for good constraint candidates by highlighting the super-faces that are cluster centers.

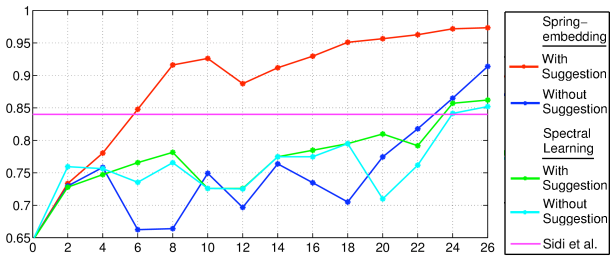
Figure 10 shows the accuracy of the co-segmentation at each constraint added, averaged for all the user sessions. We see in the graph that our active learning with suggestions has a faster convergence rate than when no suggestion mechanism is used. Thus, given the faster convergence, the active learning component reduces the time needed to improve the co-segmentation. For the candelabra, the average time needed by the users to co-segment the set with suggestions was 7 minutes, while the time was 11 minutes without suggestions. For the large vase set, the users needed 20 minutes with suggestions and 28 minutes without.

Table 1: *Characteristics of the test sets and minimal number of constraints needed to achieve a close to error-free co-segmentation.*

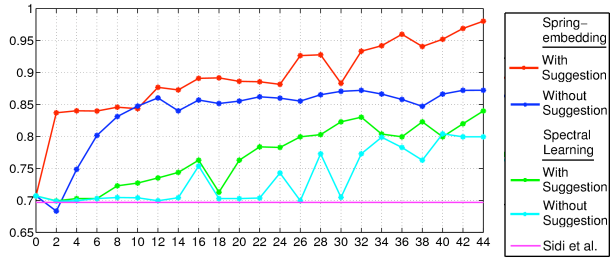
Set	#shapes	#super-fcs.	#constr.
Candelabra	28	164	24
Chairs	20	236	36
Four-legged	20	264	69
Goblets	12	49	4
Guitars	44	330	6
Lamps	20	97	2
Vases	28	169	34
Irons	18	138	26
Large chairs	400	2,832	162
Large tele-aliens	200	1,869	106
Large vases	300	1,527	44

Effectiveness of the spring embedding. In Figure 10, we also show the average co-segmentation accuracy obtained by using the same sets of input constraints, but replacing the spring embedding with the spectral learning algorithm of Kamvar et al. [2003]. In this manner, we compare our spring-embedding clustering to an alternative algorithm, and evaluate the contribution of the spring embedding to the convergence of the co-analysis. By contrasting the four curves, we see that when spectral learning is used, the introduction of a constraint has a more subtle effect on the result, while in the spring embedding, each constraint has a pronounced effect on the co-segmentation. Thus, the spring embedding also contributes to the faster convergence of our active co-analysis, since it is able to use the constraints more effectively.

Comparison to an unsupervised method. Figures 8 and 10 also allow us to compare our SSL directly to an unsupervised learning (USL) method. On the left of Figure 8, we see the co-segmentation obtained with the descriptor space clustering of Sidi et al. [2011]. In the middle, we show our initial segmentation which is also based on a similar algorithm. We can see that these results contain errors, although our initial segmentation is then corrected by the active co-analysis. As discussed in Section 1, an unsupervised method cannot guarantee a perfect co-analysis of a set, since the geometry alone cannot always convey the semantics of parts, a fact that is demonstrated by the non-trivial cluster arrangements that appear in the feature space (Figure 3). One exception is the set

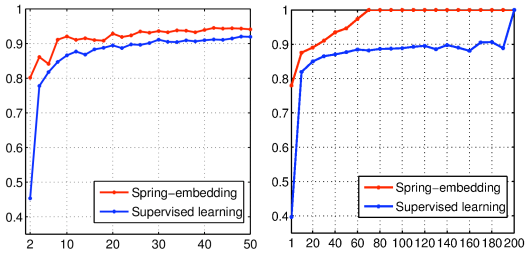


(a) *Candelabra*



(b) *Large vases*

Figure 10: *Convergence rate of the active co-analysis averaged for several users: the x-axis corresponds to the number of constraints, while the y-axis denotes the accuracy of the co-segmentation. The curves show the different constrained-clustering algorithms combined or not with the suggestion mechanism. Note that our spring embedding with suggestions displays the fastest convergence.*



(a) *Large vases* **(b)** *Large tele-aliens*

Figure 11: *Convergence rate of the spring-embedding clustering compared to the supervised learning, on two sets: the x-axis corresponds to the number of shapes used for training, while the y-axis denotes the co-segmentation accuracy measured only on the test shapes. Note the better performance of the spring embedding.*

of *lamps* which have a simpler structure, where the USL already reaches a satisfactory result. Moreover, in Figure 10, we explicitly contrast the accuracy obtained by Sidi et al. [2011] to our co-analysis on two sets. We see that, although the results of Sidi et al. are better than our initial segmentation, the active co-analysis quickly surpasses the USL accuracy with only a few constraints. Overall, the unsupervised co-segmentations have an accuracy of at most 85% for many sets, implying that they still require improvement to reach higher accuracies.

Comparison to a supervised method. Figure 11 shows the convergence rate of a supervised learning (SL) method compared to the spring-embedding. In this experiment, we use the same descriptors as in the active co-analysis to train classifiers that distinguish the different types of labels. Next, these classifiers can be used to label the super-faces of test shapes in a consistent manner. Similarly to Kalogerakis et al. [2010], we utilize the *boosting* algorithm for training. More specifically, we train a *gentleBoost* classifier for each possible label, where the number of training iterations is set to 150. We learn the classifiers on training sets of different sizes composed of shapes randomly selected from the set, and test the accuracy of the labeling on the remaining shapes. Such experiments are averaged over ten runs for each set size. To perform a direct comparison, we run the spring embedding with constraints derived from the same training sets. By taking the labels from all the training shapes, we can derive constraints from all possible combinations of super-faces. Then, we also measure the accuracy on the test shapes. For the vases, the accuracy remains constant when adding more than 50 training shapes, so we omit this portion of the graph. We see that the constrained clustering converges faster towards an error-free segmentation, exemplifying the usefulness of semi-supervised learning, which takes advantage of both the labeled data and the structure of the unlabeled data.

7 Conclusions

We have presented a semi-supervised learning method for the co-analysis of sets of shapes. The method augments the clustering in a feature space with supervised data that a user provides interactively. We introduced a graph-based technique for embedding distances while respecting prescribed constraints. The embedding technique is based on the relaxation of a spring-system that reflects the geometric knowledge latent in the set of shapes together with constraints. The technique allows for an active learning setting, where the system analyzes the embedding space and offers suggestions to the user as to where his input is likely to be effective. We have shown through extensive experiments on various sets, the effectiveness of the active learning spring-based embedding as a means to quickly converge towards an error-free consistent part labeling.

Limitations. Although the system accelerates the convergence towards an error-free co-analysis, the system has no means of knowing the ground truth, so it is not possible to guarantee convergence unless the user exhaustively constrains all possible pairs. After enough constraints have been added, the system can sense the cluster-membership confidence of points by computing their silhouette indices. These provide an empirical measure that suggests convergence. Moreover, our active learning is limited. As said above, the ground truth is unknown, and the technique that we presented is based on heuristics that rely on the embedding of the super-faces. The embedding space, of course, is subject to various errors due to the *a priori* imperfect descriptive power of the descriptors and also distortion introduced by the dimensionality reduction.

Future work. Our system analyses a static set. It will be interesting to investigate a dynamic setting where new shapes are integrated into the system in an incremental way. Likewise, the active learning framework can be applied in a collaborative setting where multiple users are involved, so that much larger sets can be handled. Another interesting research avenue is to go beyond the analysis of the feature distances, and consider also the inter-relation among the labels, maybe in the form of a smoothness term, or other means to regulate their configurations in the embedding space. Here, it is also of interest the design of new types of constraints beyond must-link and cannot-link, considering partially-labeled data or even possibly incorporating constraints at the cluster-level. Verifying the consistency of the constraints added by the users would also be a possible improvement to the method. Finally, we believe that our spring-based embedding technique can be useful for other learning applications, not necessarily in geometry analysis.

Acknowledgements

The authors would like to thank all the reviewers for their valuable comments. This work is supported in part by grants from NSFC (61202222, 61232011, 61025012), Guangdong Science and Technology Program (2011B050200007), National 863 Program (2011AA010503), Shenzhen Science and Innovation Program (CXB201104220029A, JC201005270329A), NSERC (611370) and the Israel Science Foundation.

References

BASU, S., BANERJEE, A., AND MOONEY, R. 2004. Active semi-supervision for pairwise constrained clustering. In *Proc. SIAM Int. Conf. on Data Mining (SDM)*, 333–344.

BOYKOV, Y., VEKSLER, O., AND ZABIH, R. 2001. Fast approximate energy minimization via graph cuts. *IEEE Trans. Pattern Anal. Mach. Intell. 23*, 11, 1222–1239.

BRUN, M., SIMA, C., HUA, J., LOWEY, J., CARROLL, B., SUH, E., AND DOUGHERTY, E. R. 2007. Model-based evaluation of clustering validation measures. *Pattern Recogn. 40*, 3, 807–824.

CHEN, X., GOLOVINSKIY, A., , AND FUNKHOUSER, T. 2009. A benchmark for 3D mesh segmentation. *ACM Trans. on Graphics (Proc. SIGGRAPH) 28*, 3.

COLEMAN, T., SAUNDERSON, J., AND WIRTH, A. 2008. Spectral clustering with inconsistent advice. In *ICML*, 152–159.

FU, H., COHEN-OR, D., DROR, G., AND SHEFFER, A. 2008. Upright orientation of man-made objects. *ACM Trans. on Graphics (Proc. SIGGRAPH) 27*, 3.

GOLOVINSKIY, A., AND FUNKHOUSER, T. 2009. Consistent segmentation of 3D models. *Computers & Graphics (Proc. of SMI) 33*, 3, 262–269.

HOI, S., LIU, W., AND CHANG, S. 2008. Semi-supervised distance metric learning for collaborative image retrieval. *Proc. IEEE Conf. on CVPR*.

HU, R., FAN, L., AND LIU, L. 2012. Co-segmentation of 3D shapes via subspace clustering. *Computer Graphics Forum (Proc. SGP) 31*, 5, 1703–1713.

HUANG, Q., KOLTUN, V., AND GUIBAS, L. 2011. Joint shape segmentation with linear programming. *ACM Trans. on Graphics (Proc. SIGGRAPH Asia) 30*, 6.

JIN, Y., WU, Q., AND LIU, L. 2012. Unsupervised upright orientation of man-made models. *Graphical Models 74*, 4, 99–108.

KALOGERAKIS, E., HERTZMANN, A., AND SINGH, K. 2010. Learning 3D mesh segmentation and labeling. *ACM Trans. on Graphics (Proc. SIGGRAPH) 29*, 3.

KAMVAR, S. D., KLEIN, D., AND MANNING, C. D. 2003. Spectral learning. In *International Joint Conference on Artificial Intelligence*, 561–566.

KLEIN, D., KAMVAR, S., AND MANNING, C. 2002. From instance-level constraints to space-level constraints: Making the most of prior knowledge in data clustering. In *ICML*, 307–314.

KULIS, B., BASU, S., DHILLON, I., AND MOONEY, R. 2005. Semi-supervised graph clustering: a kernel approach. In *ICML*, 457–464.

LI, Z., LIU, J., AND TANG, X. 2009. Constrained clustering via spectral regularization. In *Proc. IEEE Conf. on CVPR*, 421–428.

LU, Z., AND CARREIRA-PERPINÁN, M. 2008. Constrained spectral clustering through affinity propagation. In *Proc. IEEE Conf. on CVPR*.

SETTLES, B. 2009. Active learning literature survey. Tech. Rep. 1648, Univ. of Wisconsin-Madison.

SHAMIR, A. 2008. A survey on mesh segmentation techniques. *Computer Graphics Forum 27*, 6, 1539–1556.

SHAPIRA, L., SHAMIR, A., AND COHEN-OR, D. 2008. Consistent mesh partitioning and skeletonization using the shape diameter function. *The Visual Computer 24*, 4, 249–259.

SHENTAL, N., BAR-HILLEL, A., HERTZ, T., AND WEINSHALL, D. 2004. Computing Gaussian mixture models with EM using equivalence constraints. In *Proc. NIPS*, 465–472.

SHI, J., AND MALIK, J. 2000. Normalized cuts and image segmentation. *IEEE PAMI 22*, 8, 888–905.

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 Graphics (Proc. SIGGRAPH Asia) 30*, 6.

SUNKEL, M., JANSEN, S., WAND, M., EISEMANN, E., AND SEIDEL, H. 2011. Learning line features in 3D geometry. *Computer Graphics Forum (Proc. EUROGRAPHICS) 30*, 2, 267–276.

TORRESANI, L., AND LEE, K. 2007. Large margin component analysis. In *Proc. NIPS*, vol. 19, 1385–1392.

VAN KAICK, O., TAGLIASACCHI, A., SIDI, O., ZHANG, H., COHEN-OR, D., WOLF, L., AND HAMARNEH, G. 2011. Prior knowledge for part correspondence. *Computer Graphics Forum (Proc. EUROGRAPHICS) 30*, 2, 553–562.

WAGSTAFF, K., AND CARDIE, C. 2000. Clustering with instance-level constraints. In *ICML*, 1103–1110.

WANG, X., AND DAVIDSON, I. 2010. Active spectral clustering. In *ICDM*, IEEE, 561–568.

WANG, X., AND DAVIDSON, I. 2010. Flexible constrained spectral clustering. In *SIGKDD*, 563–572.

WEINBERGER, K., BLITZER, J., AND SAUL, L. 2006. Distance metric learning for large margin nearest neighbor classification. In *Proc. NIPS*, vol. 18, 1473–1480.

XU, Q., DESJARDINS, M., AND WAGSTAFF, K. 2005. Active constrained clustering by examining spectral eigenvectors. In *Discovery Science*, 294–307.

XU, K., LI, H., ZHANG, H., COHEN-OR, D., XIONG, Y., AND CHENG, Z. 2010. Style-content separation by anisotropic part scales. *ACM Trans. on Graphics (Proc. SIGGRAPH Asia) 29*, 5.

XU, K., ZHENG, H., ZHANG, H., COHEN-OR, D., LIU, L., AND XIONG, Y. 2011. Photo-inspired model-driven 3D object modeling. *ACM Trans. on Graphics (Proc. SIGGRAPH) 30*, 4.

YANG, L., AND JIN, R. 2006. Distance metric learning: A comprehensive survey. Tech. rep., Michigan State Universiy.

YU, S., AND SHI, J. 2004. Segmentation given partial grouping constraints. *IEEE PAMI 26*, 2, 173–183.

ZHU, X. 2005. Semi-supervised learning literature survey. Tech. Rep. 1530, Univ. of Wisconsin-Madison.