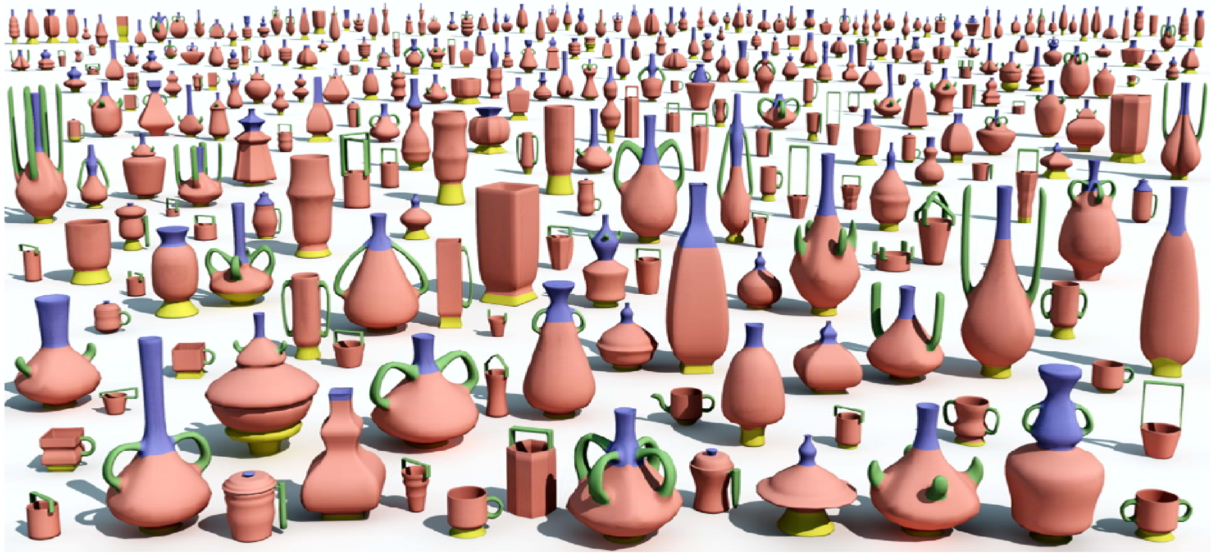
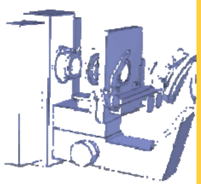
Shape Co-analysis and constrained clustering



Daniel Cohen-Or

Tel-Aviv University

High-level Shape analysis

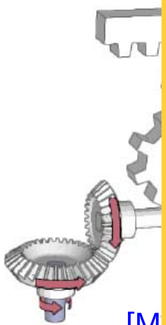


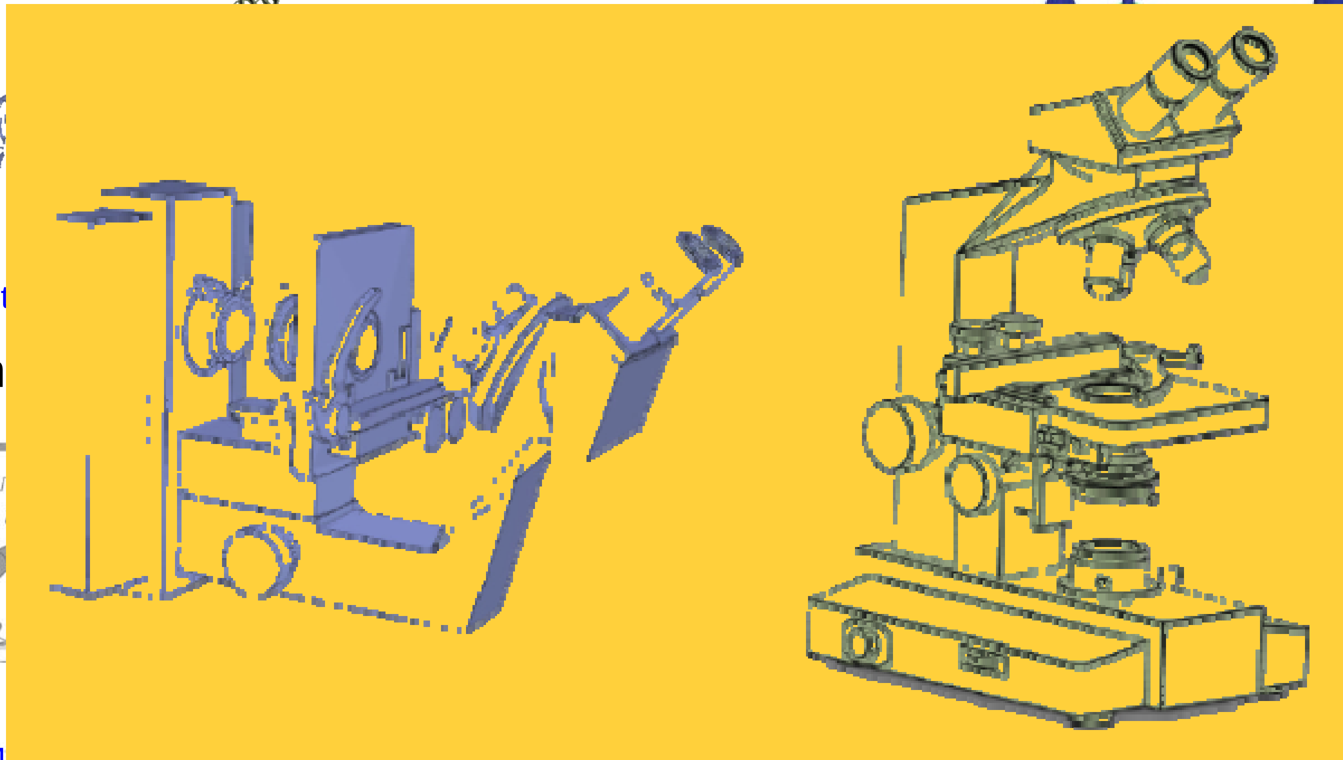
[Fu et

Uprigh

10]

tation





[Mitra et al. 10]

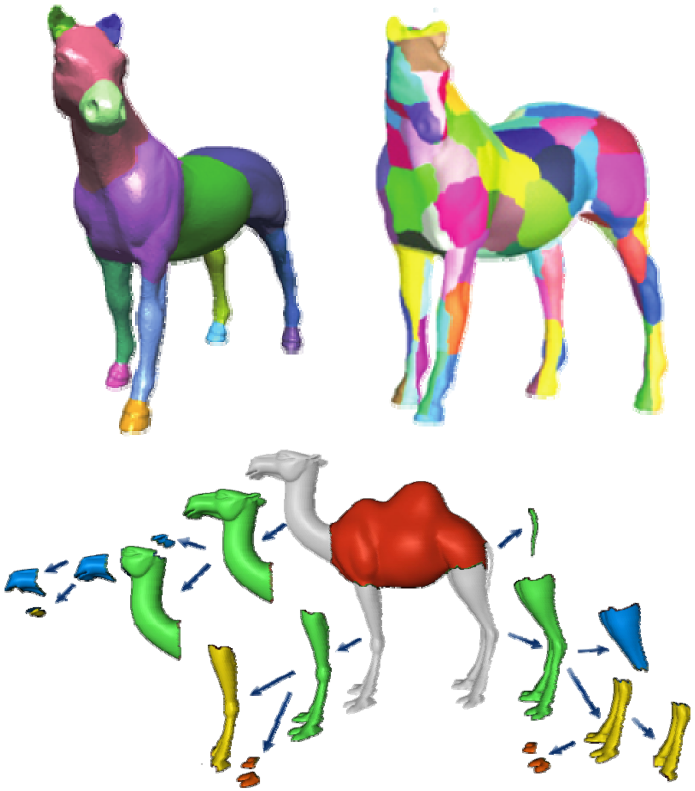
Illustrating assemblies

[Wang et al. 11]

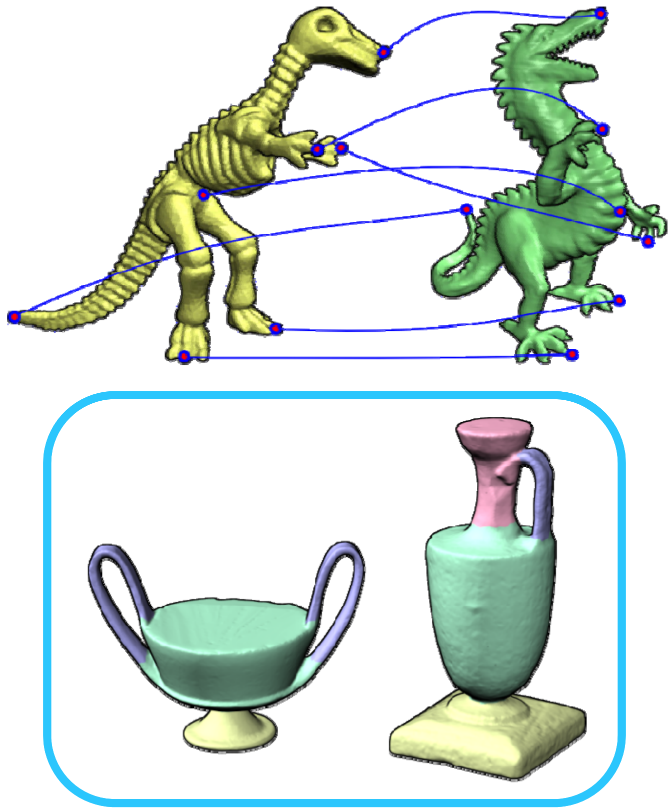
Symmetry hierarchy

Exploration of shape collections

Segmentation and Correspondence



Segmentation

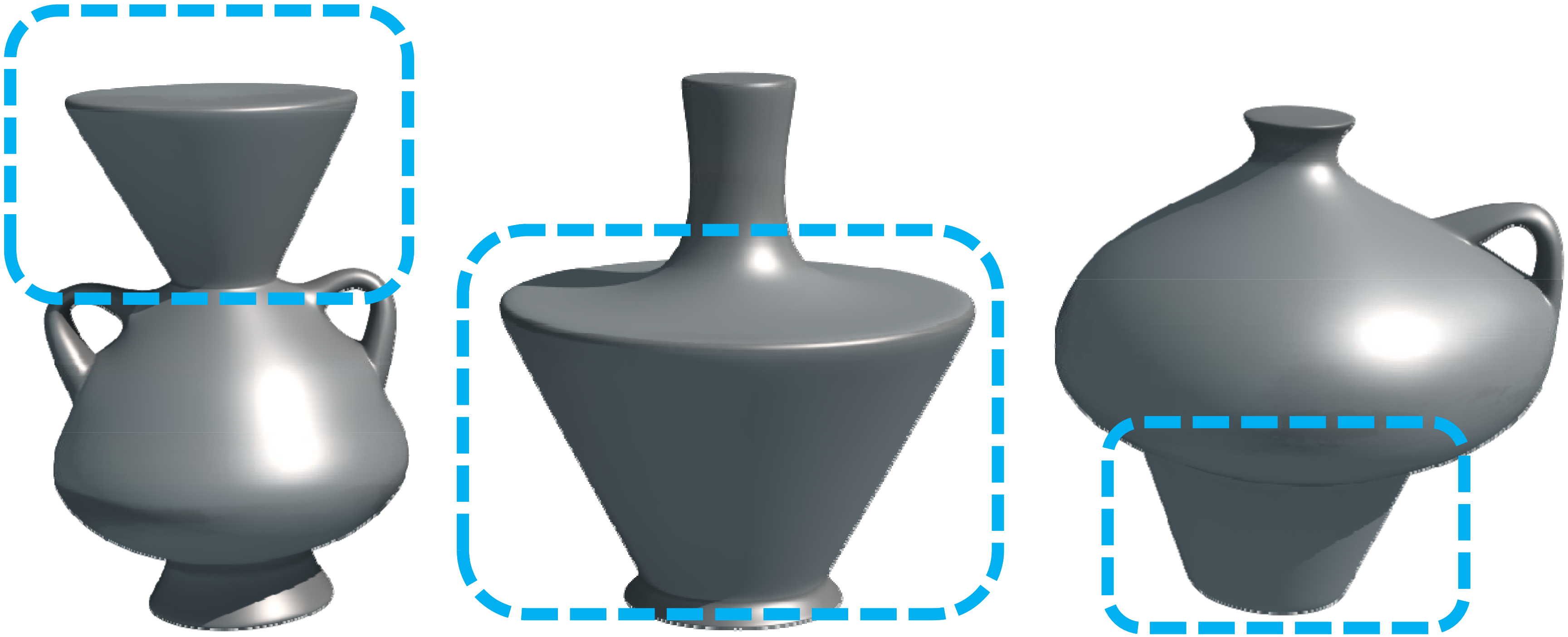


Correspondence

Individual vs. Co-segmentation

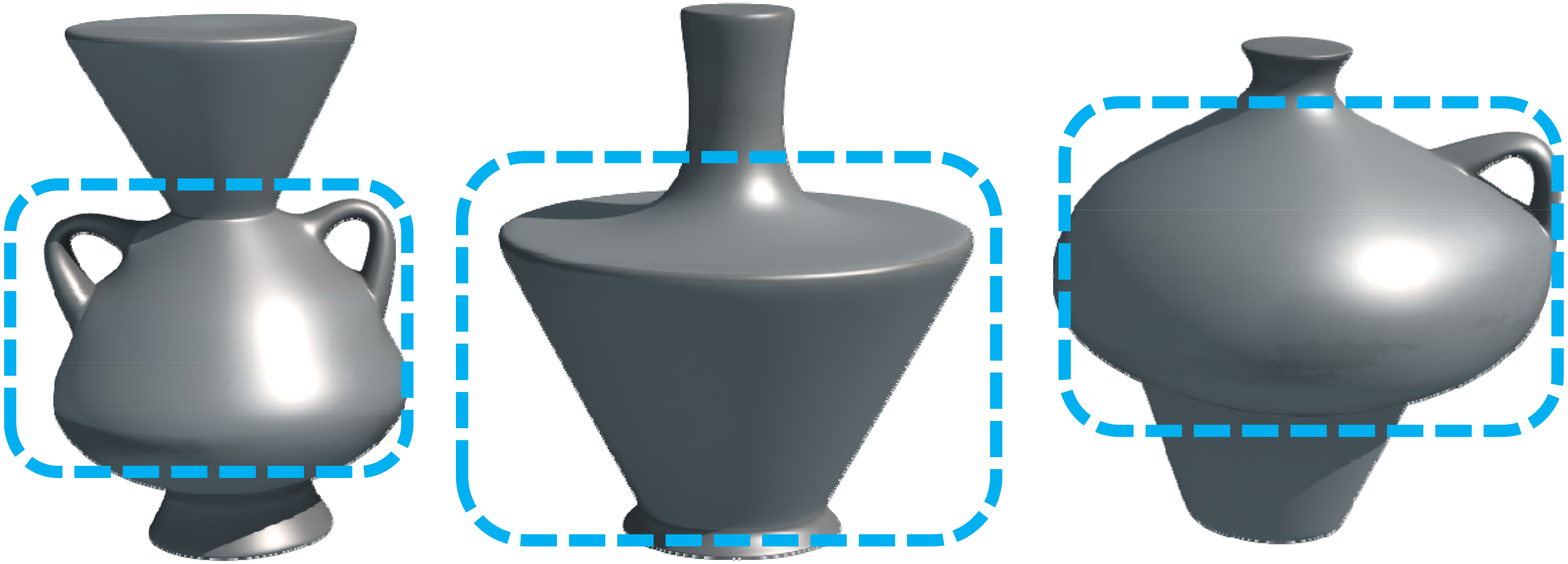
Individual vs. Co-segmentation

Challenge



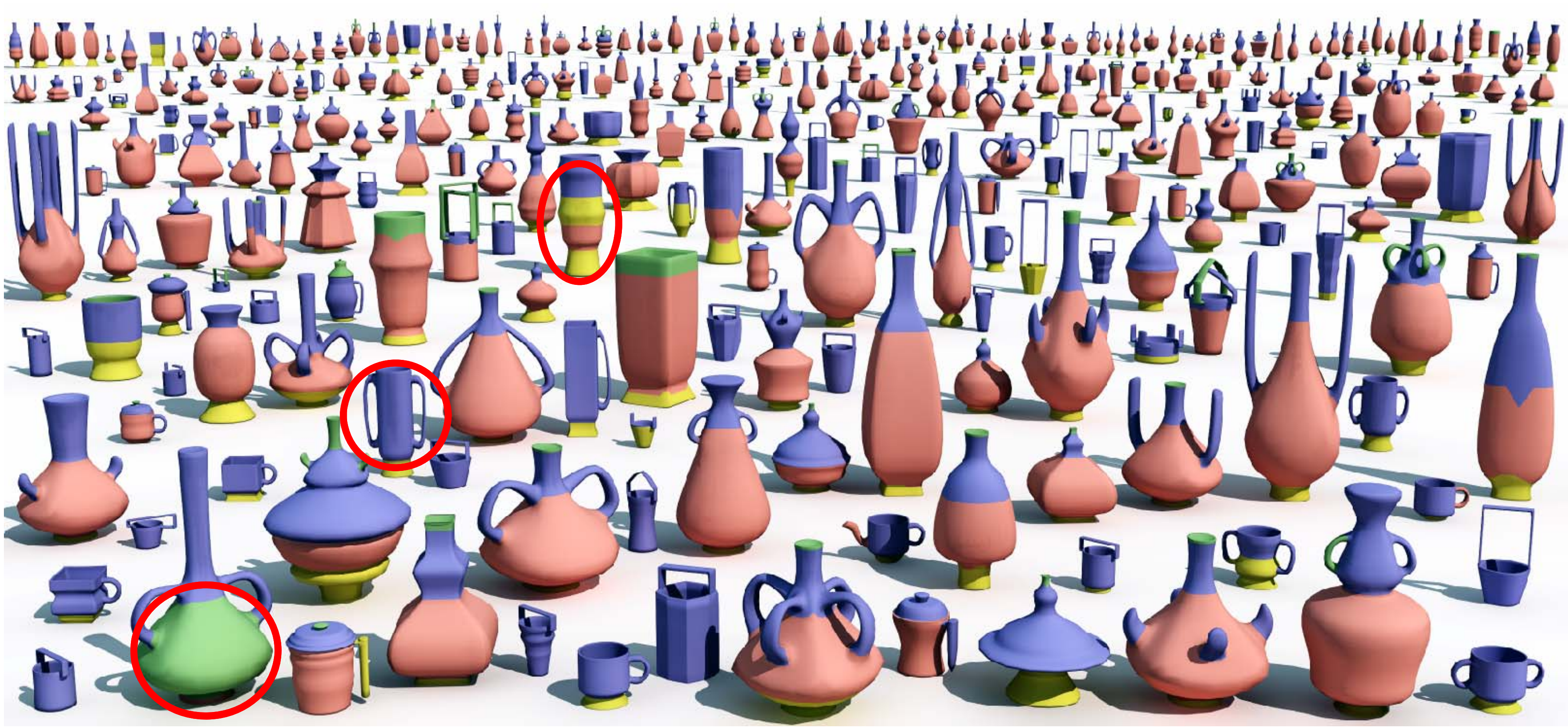
Similar geometries can be associated with different semantics

Challenge



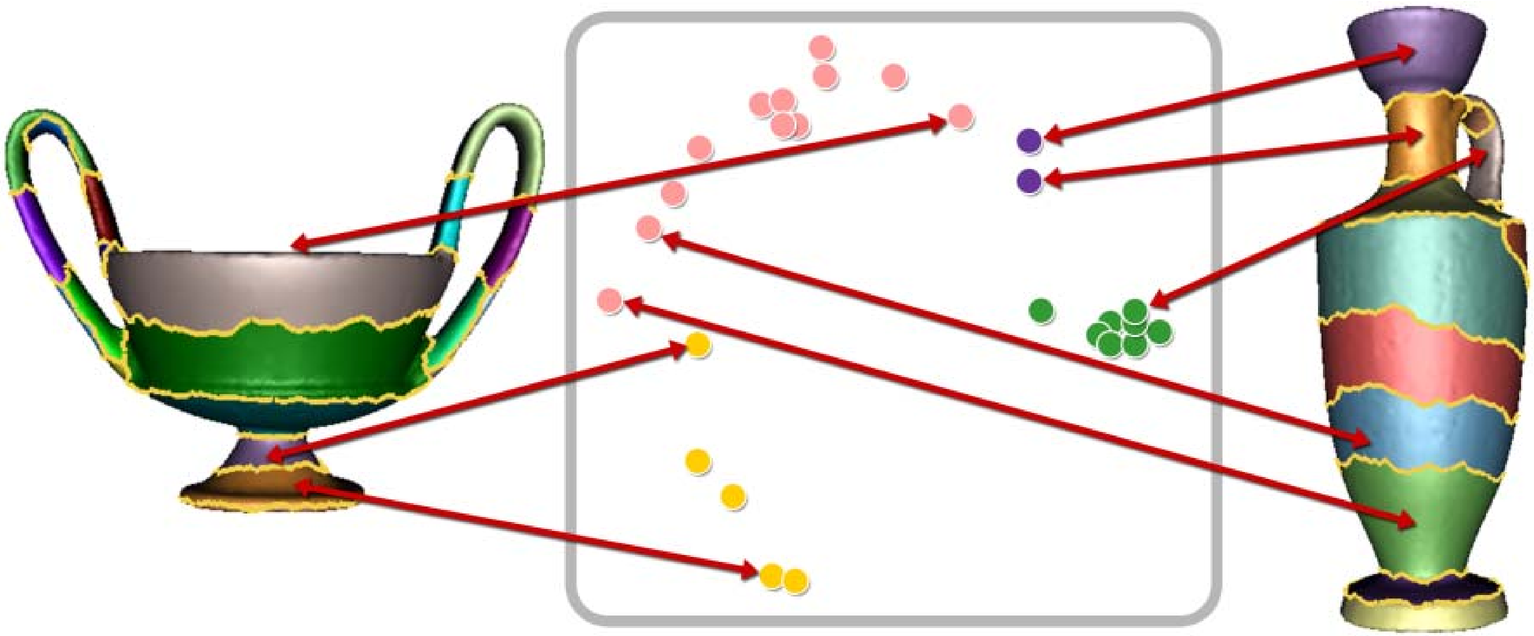
Similar semantics can be represented by different geometries

Large set are more challenging



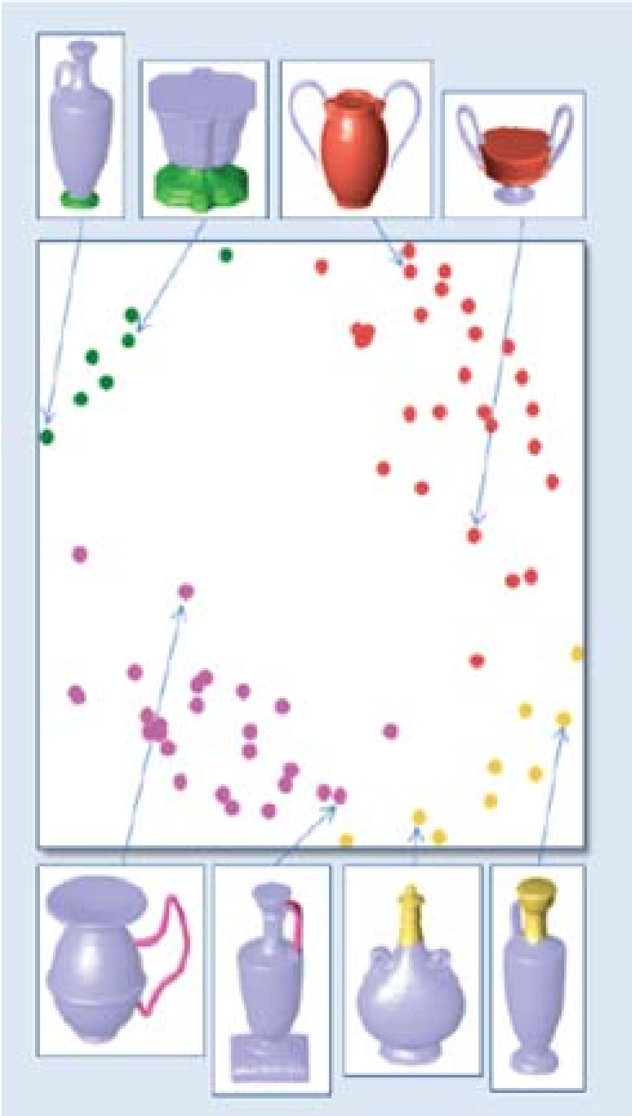
Methods do not give perfect results

Descriptor-based unsupervised co-segmentation



[Sidi et al.11]

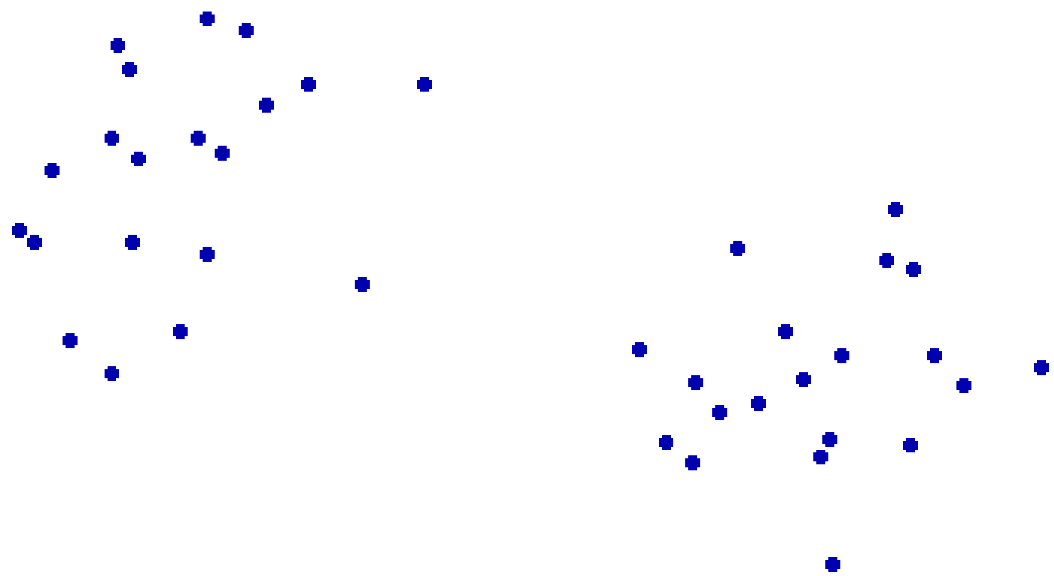
co-segmentation



Clustering in feature space

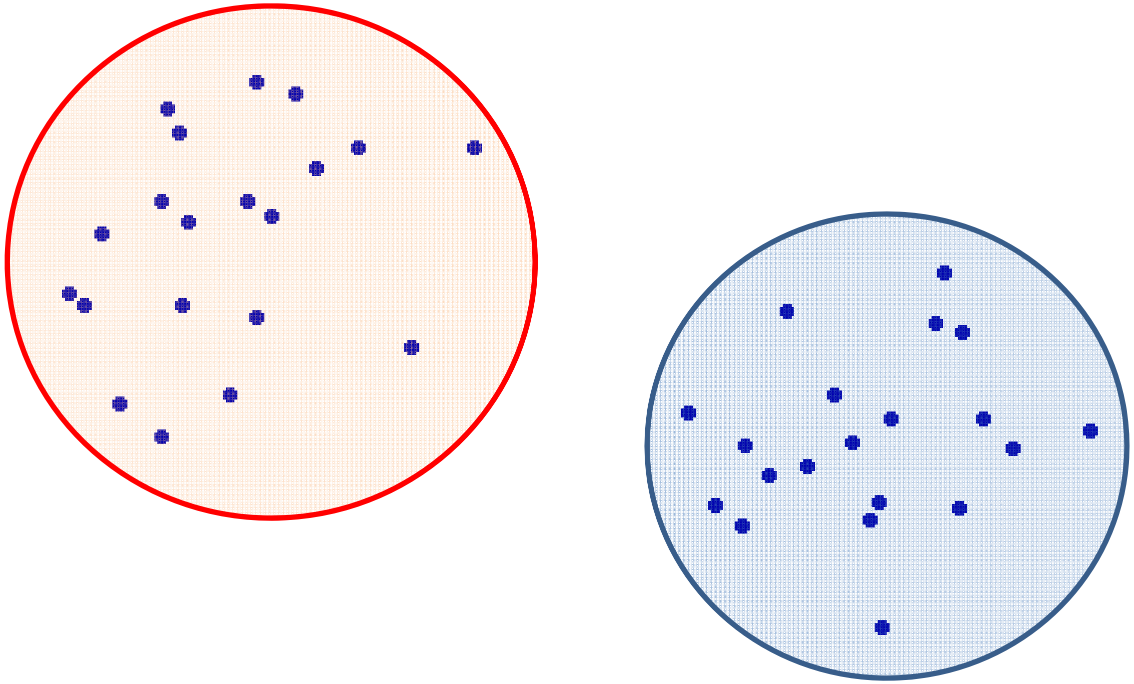
Clustering (basic stuff)

Takes a set of points,



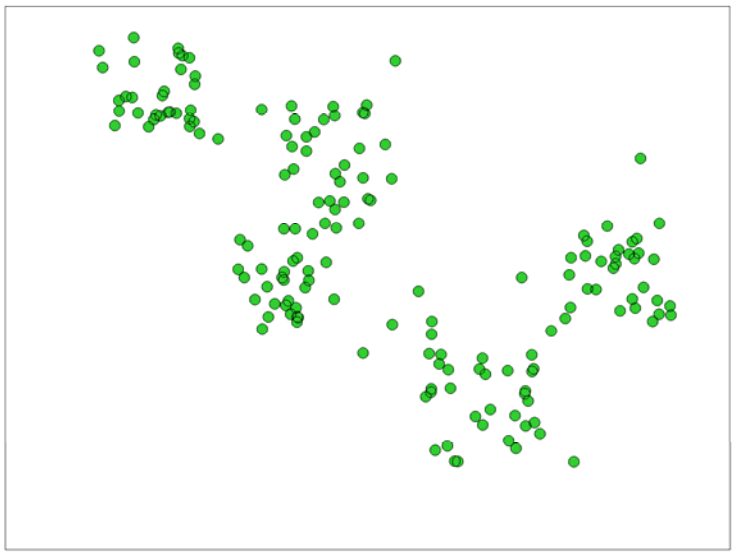
Clustering (basic stuff)

Takes a set of points, and groups them into several separate clusters

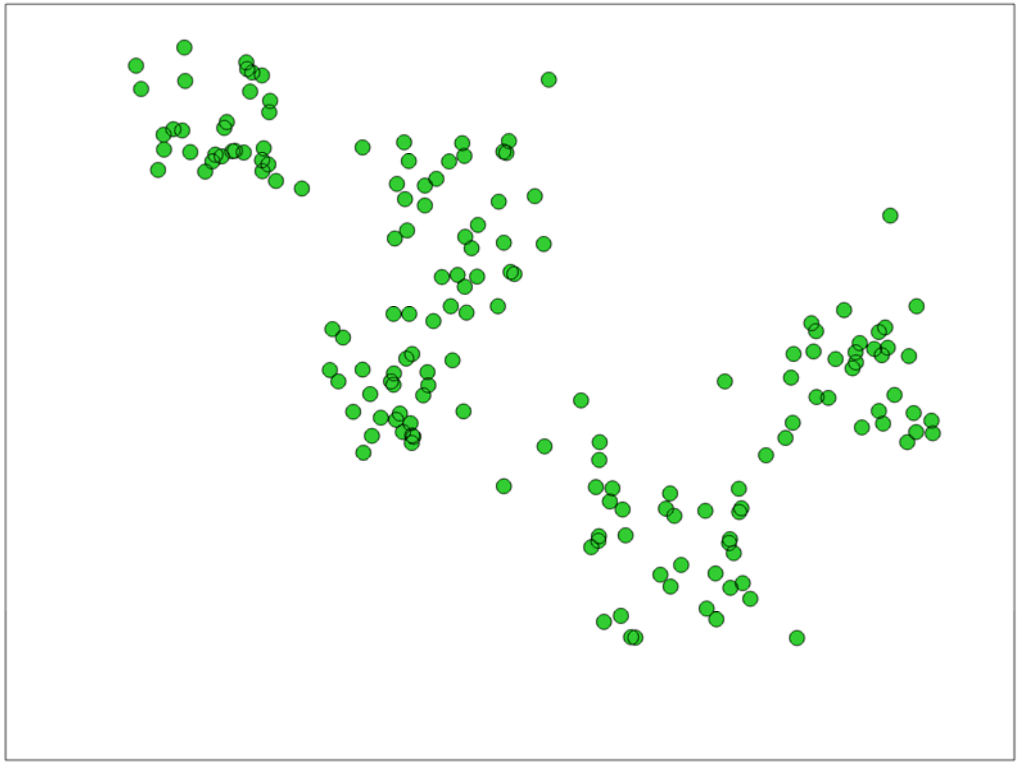


Clustering is not easy…

- Clean separation to groups not always possible
- Must make "hard splitting" decisions
- Number of groups not always known, or can be very difficult to determine from data

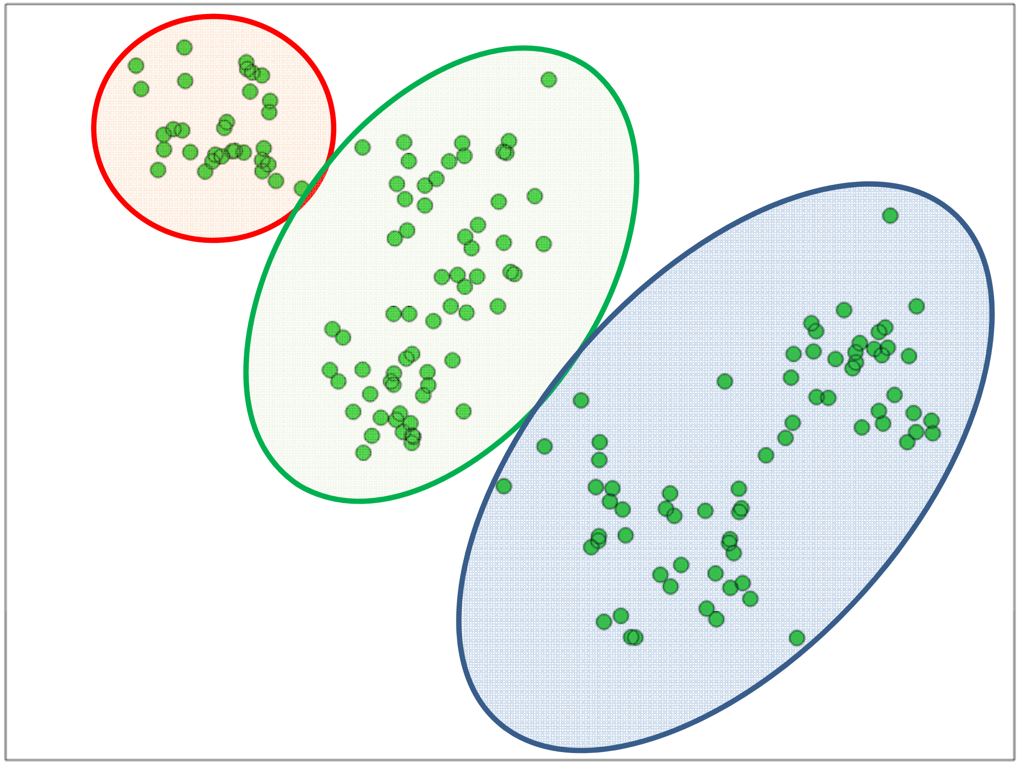


Clustering is hard!



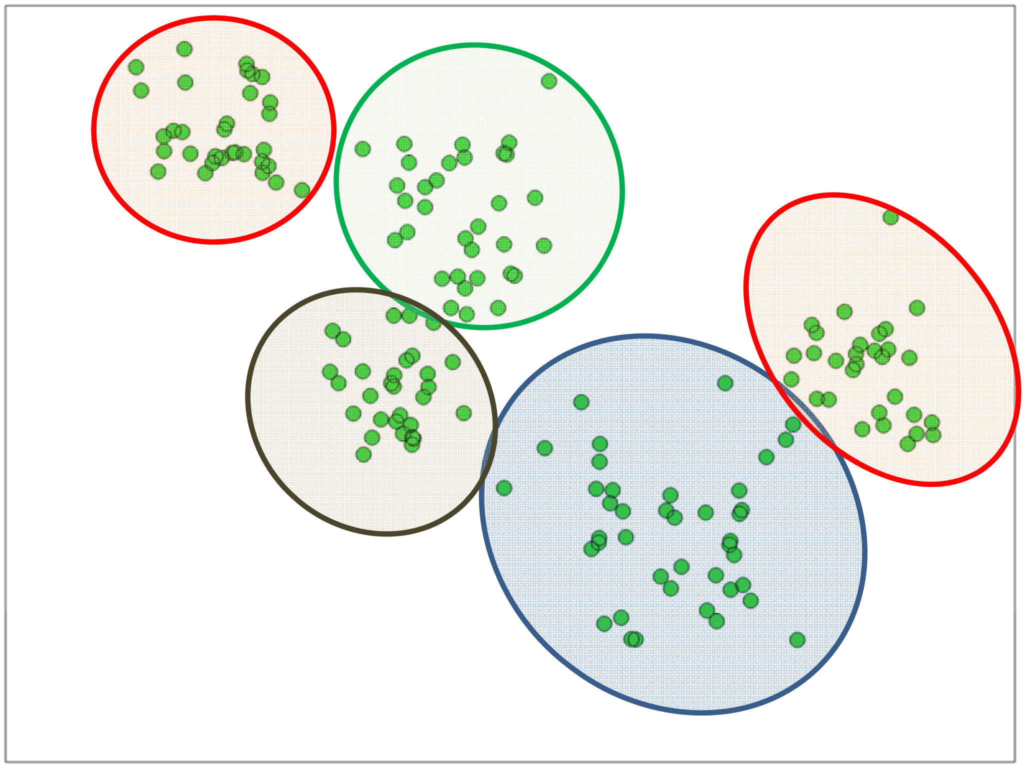
Clustering is hard!

Hard to determine number of clusters



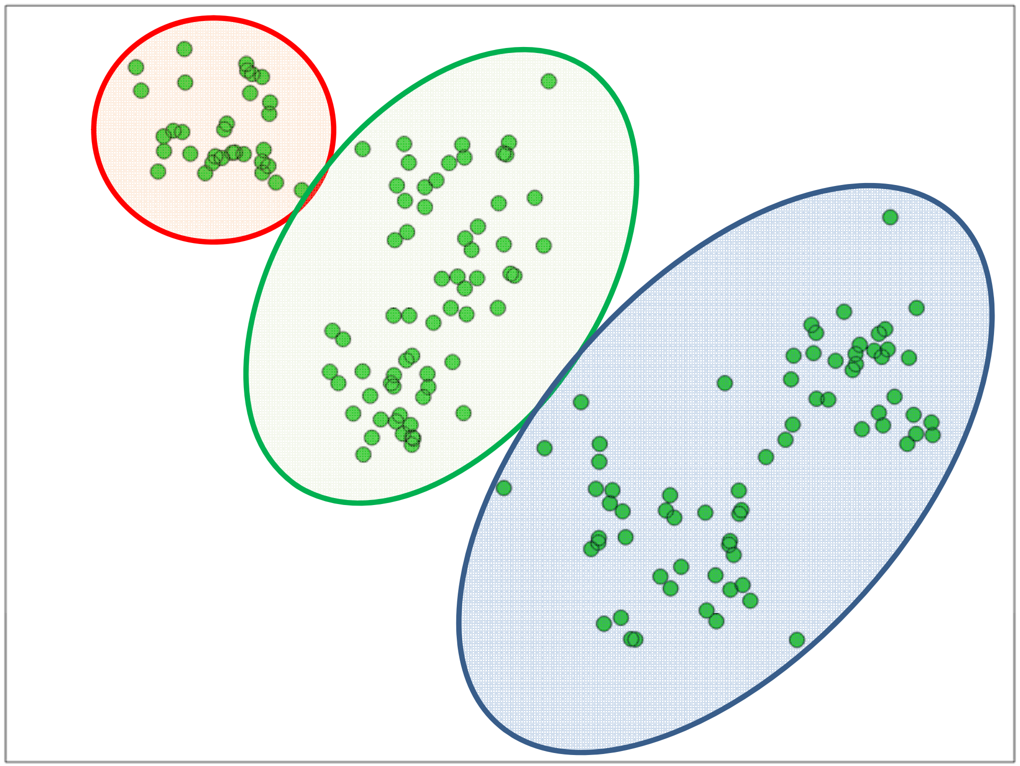
Clustering is hard!

Hard to determine number of clusters



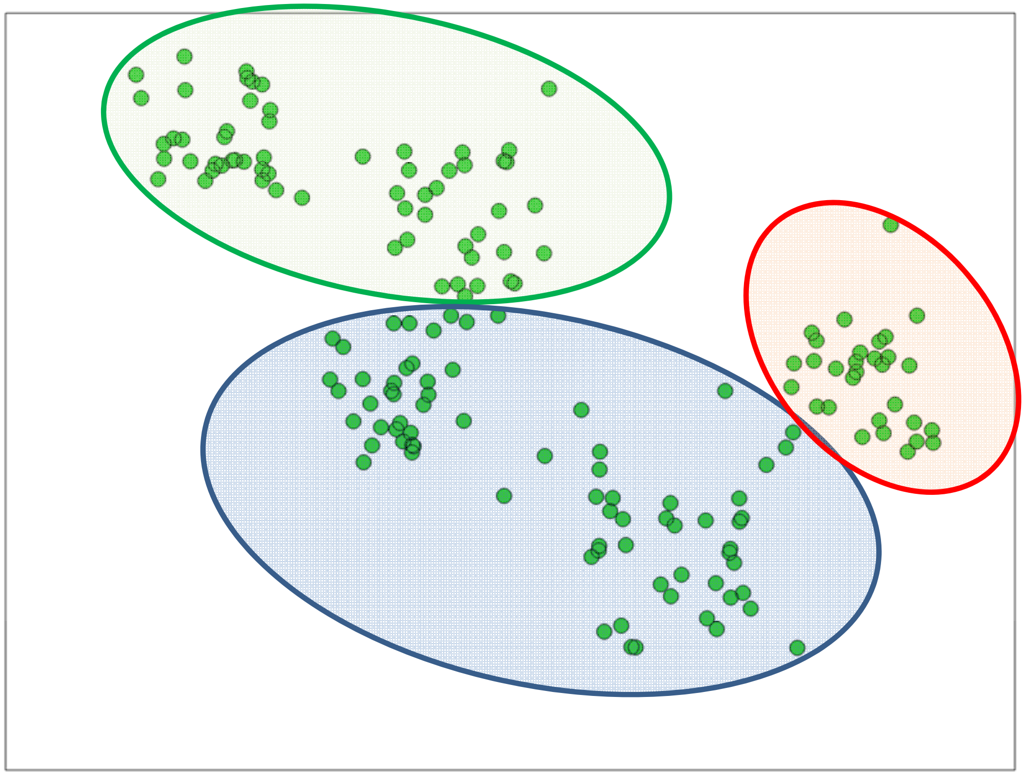
Clustering is hard!

Hard to decide where to split clusters



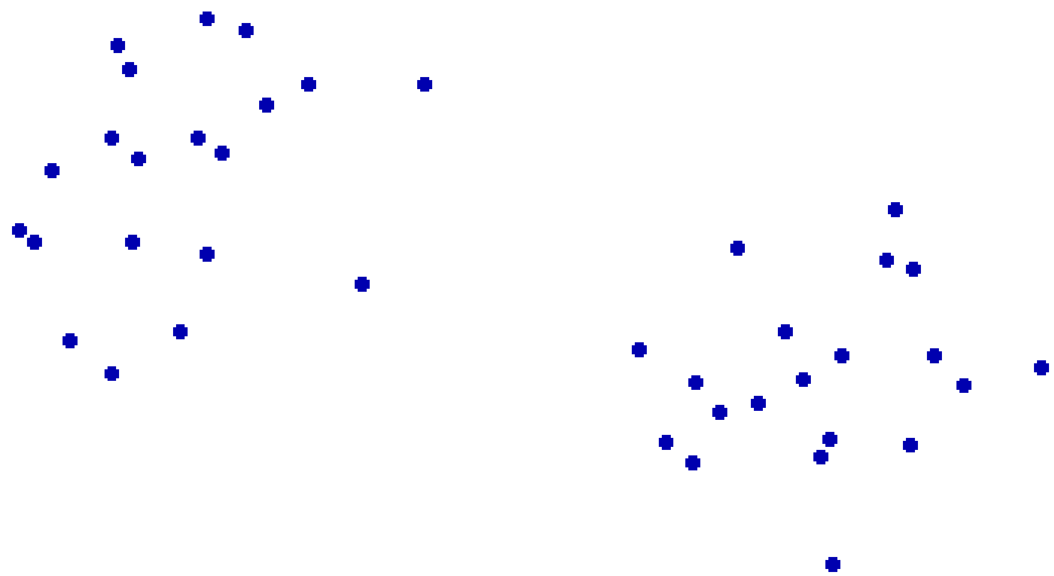
Clustering

Hard to decide where to split clusters



Clustering

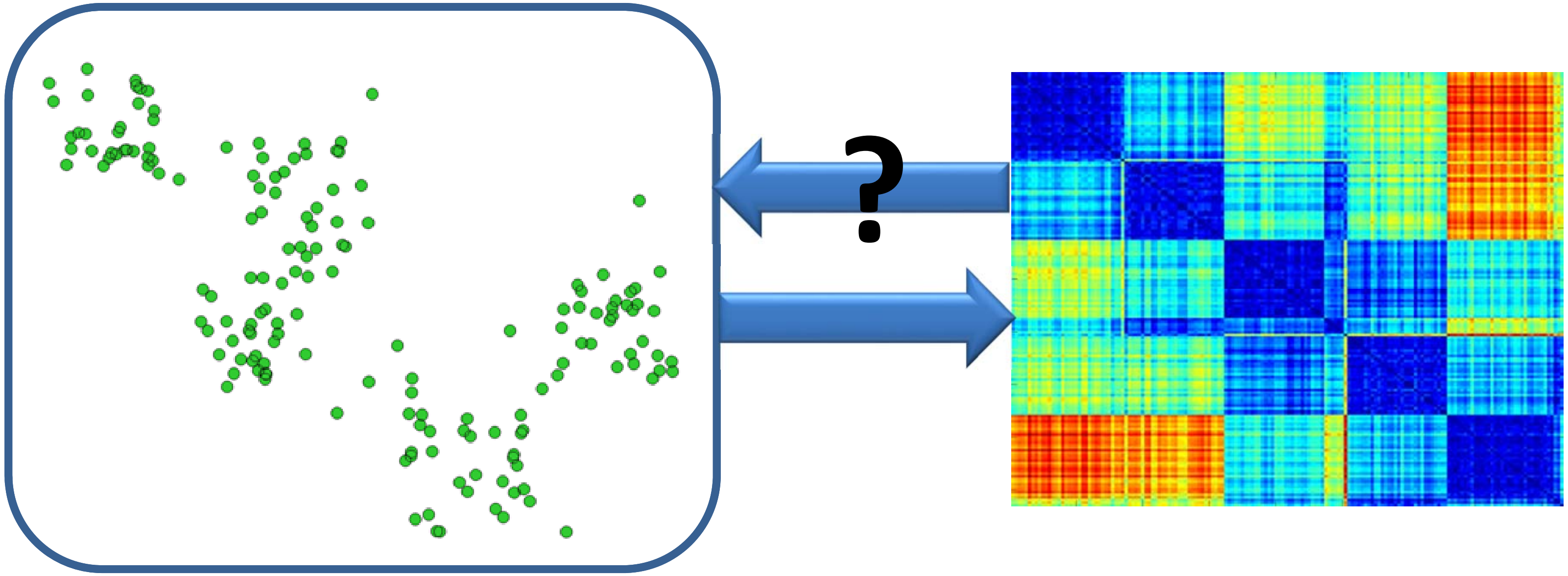
- Two general types of input for Clustering:
  - **Spatial Coordinates (points, feature space), or**
  - Inter-object Distance matrix



Clustering

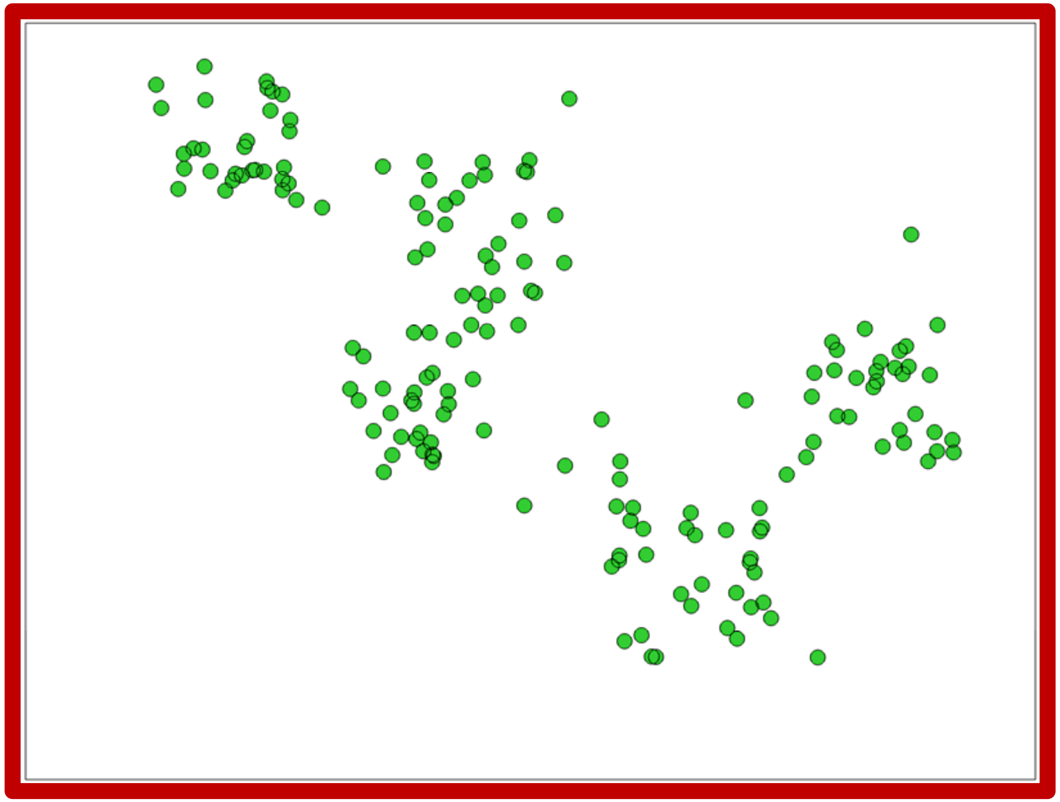
Spatial Coordinates (points, feature space), or

Inter-object Distance matrix



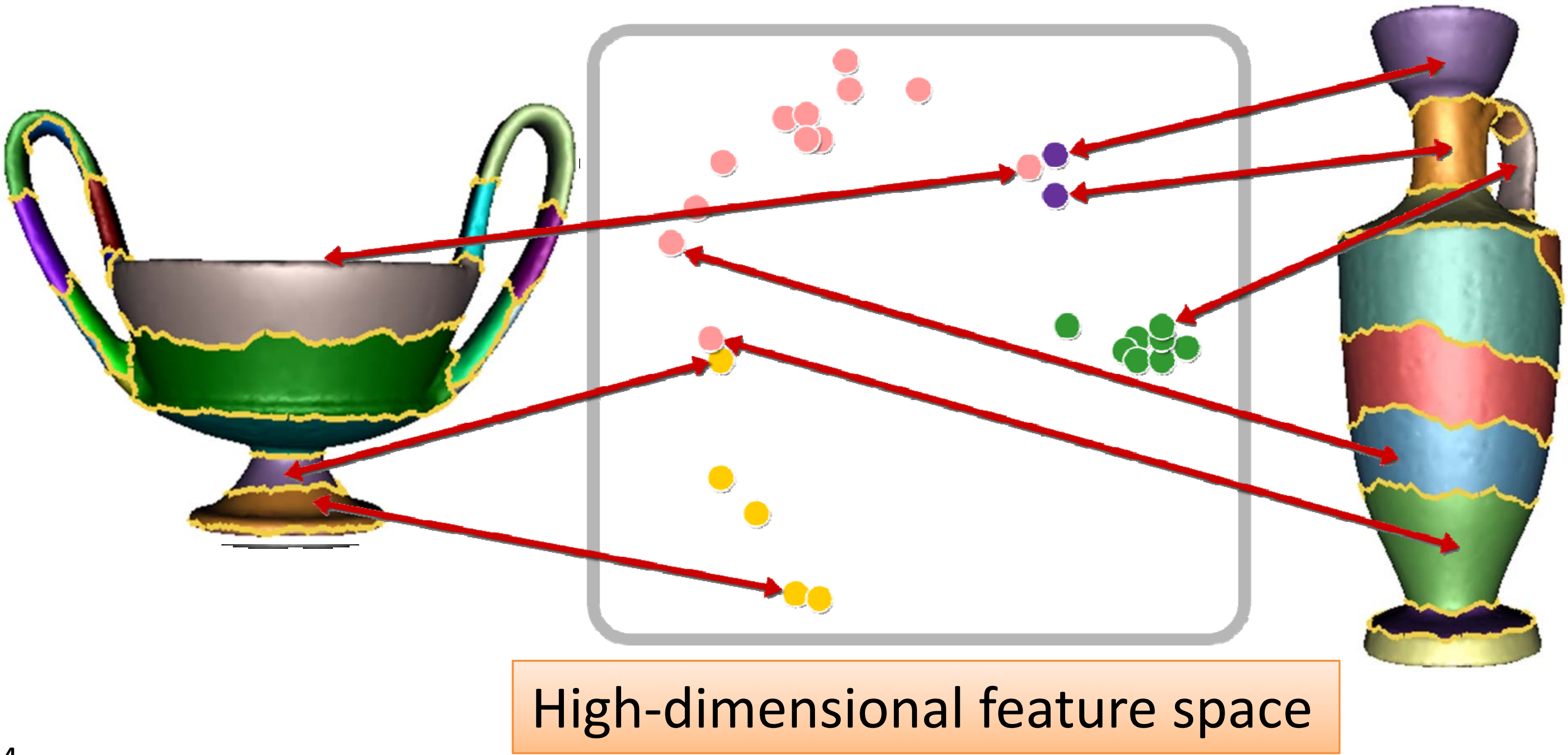
K-Means, EM, Mean-Shift, Linkage, DBSCAN, Spectral Clustering

Clustering 101



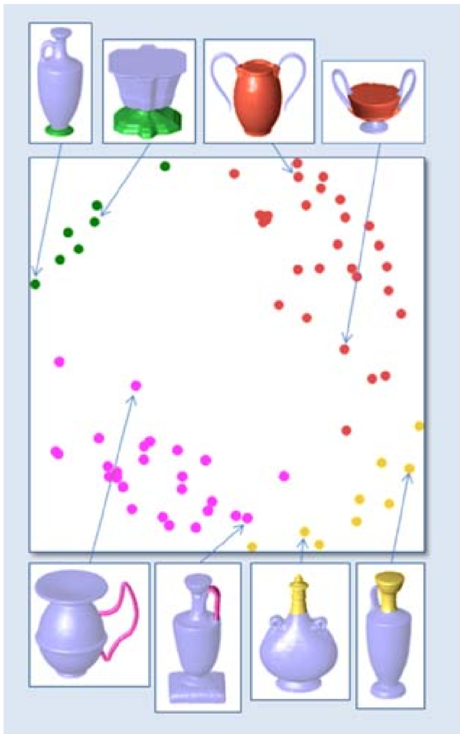
Initial co-segmentation

- Over-segmentation mapped to a descriptor space (geodesic distance, shape diameter function, normal histogram)



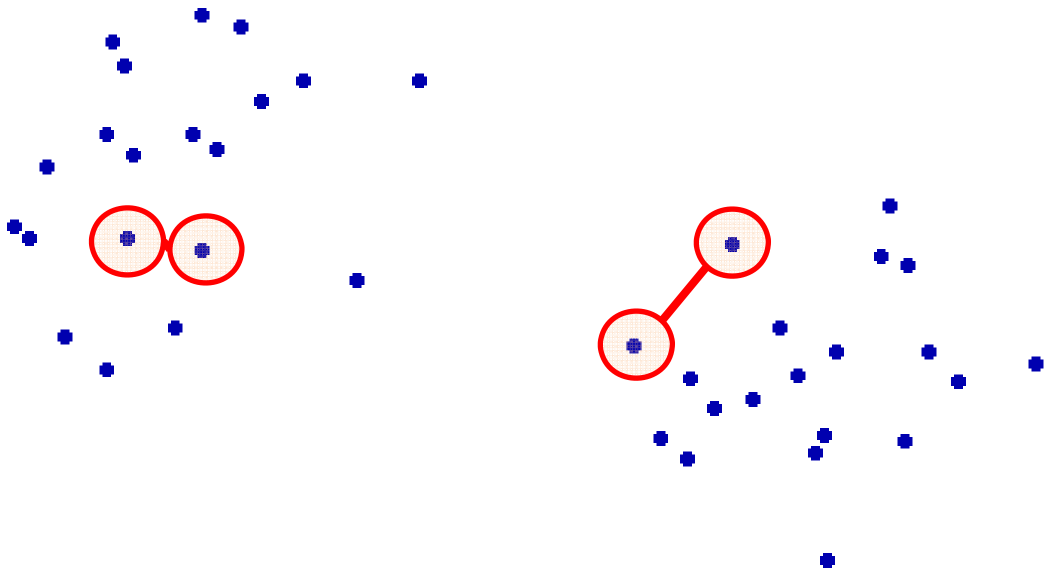
Co-segmentation

Points represent some kind of object parts, and we want to cluster them as means to co-segment the set of objects



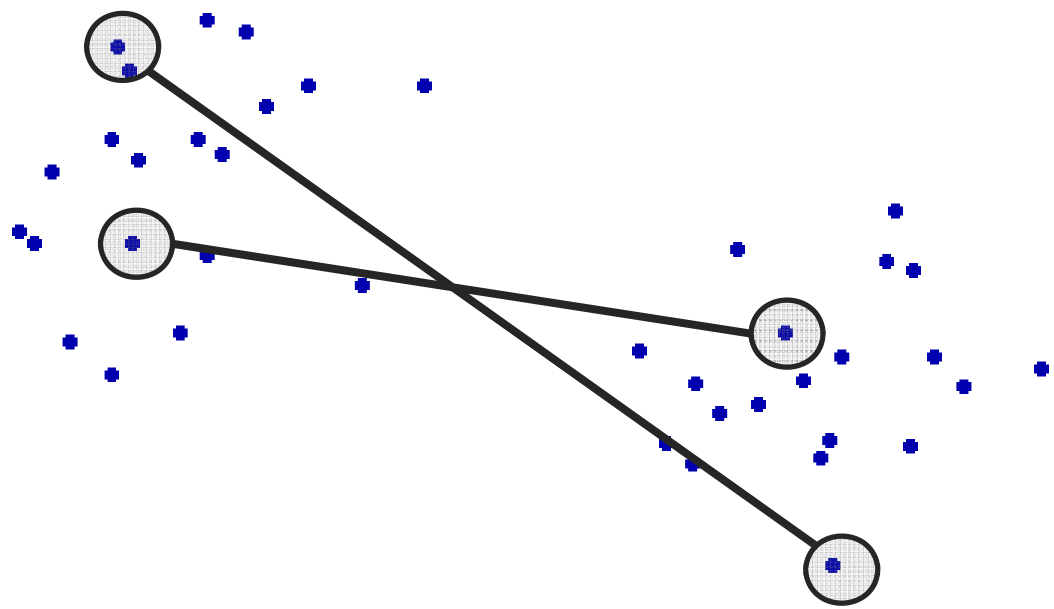
Clustering

- Underlying assumptions behind all clustering algorithms:
  - **Neighboring points imply similar parts.**



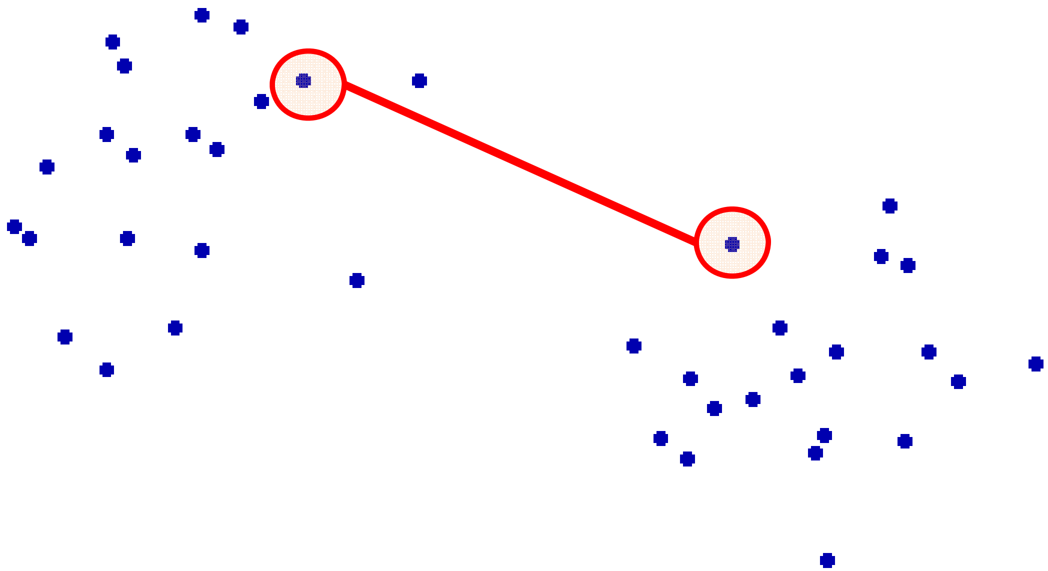
Clustering

- Underlying assumptions behind all clustering algorithms:
  - **Distant points imply dissimilar parts.**



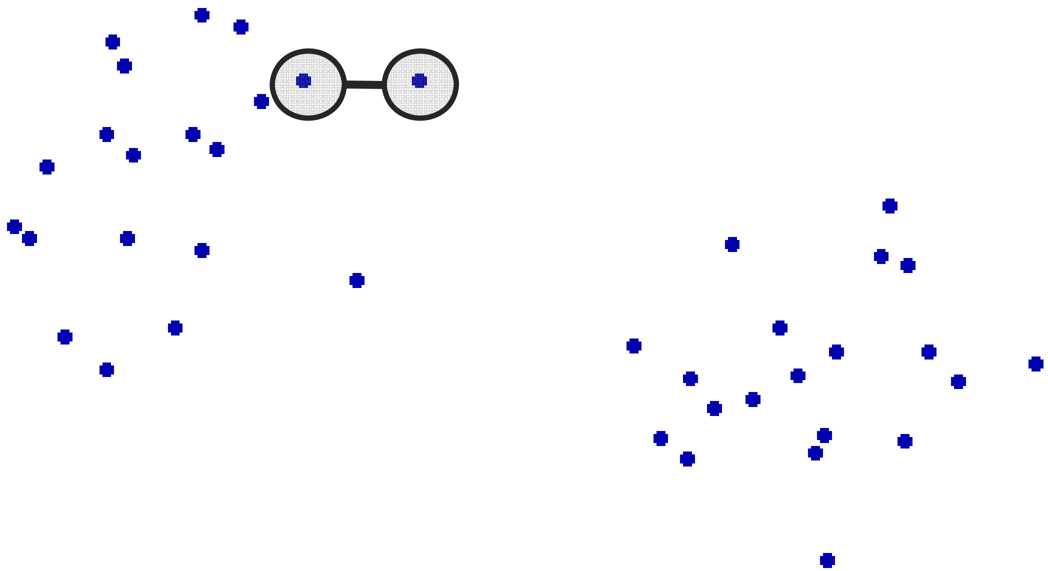
Clustering

- When assumptions fail, result is not useful:
  - **Similar parts are distant in feature space**



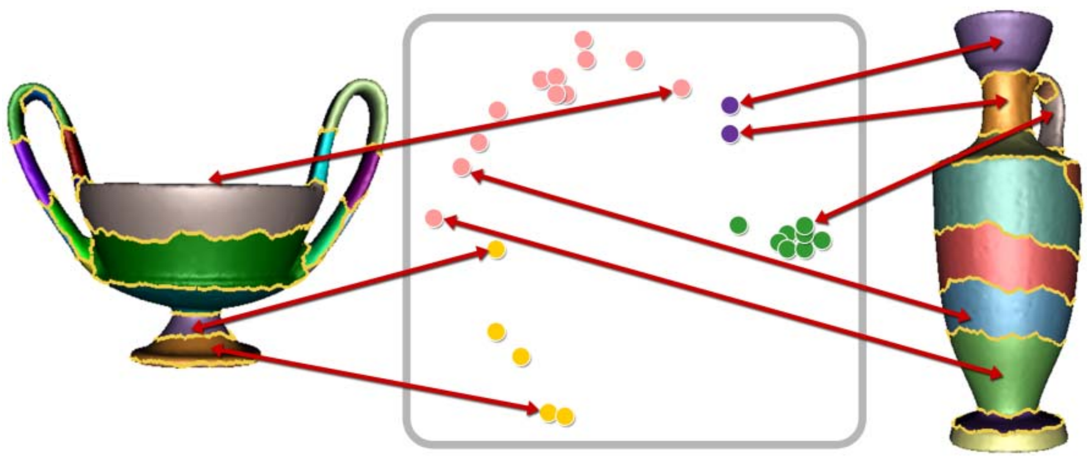
Clustering

- When assumptions fail, result is not useful:
  - **Dissimilar parts are close in feature space**



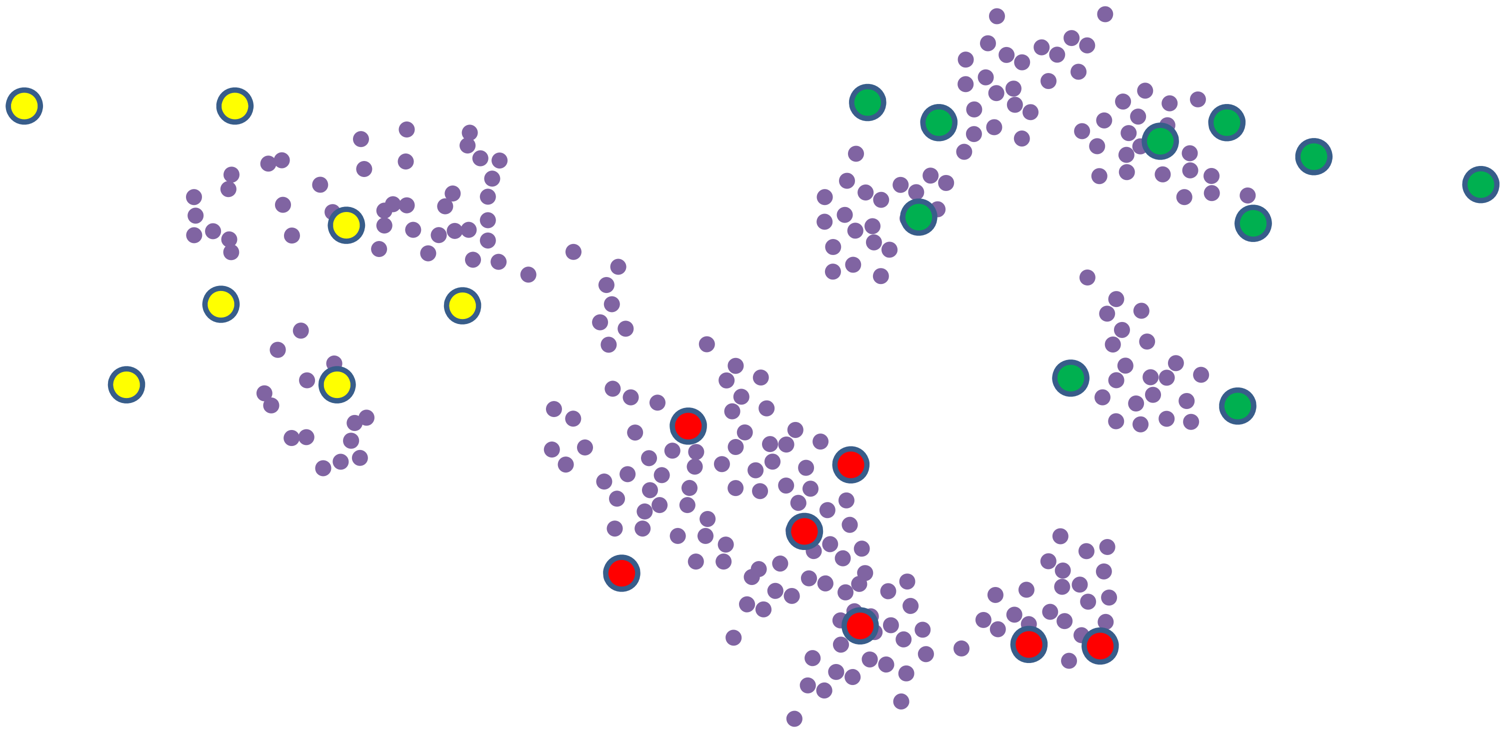
Clustering

- Assumptions might fail because:
  - Data is difficult to analyze
  - Similarity/Dissimilarity of data not well defined
  - Feature space is insufficient or distorted



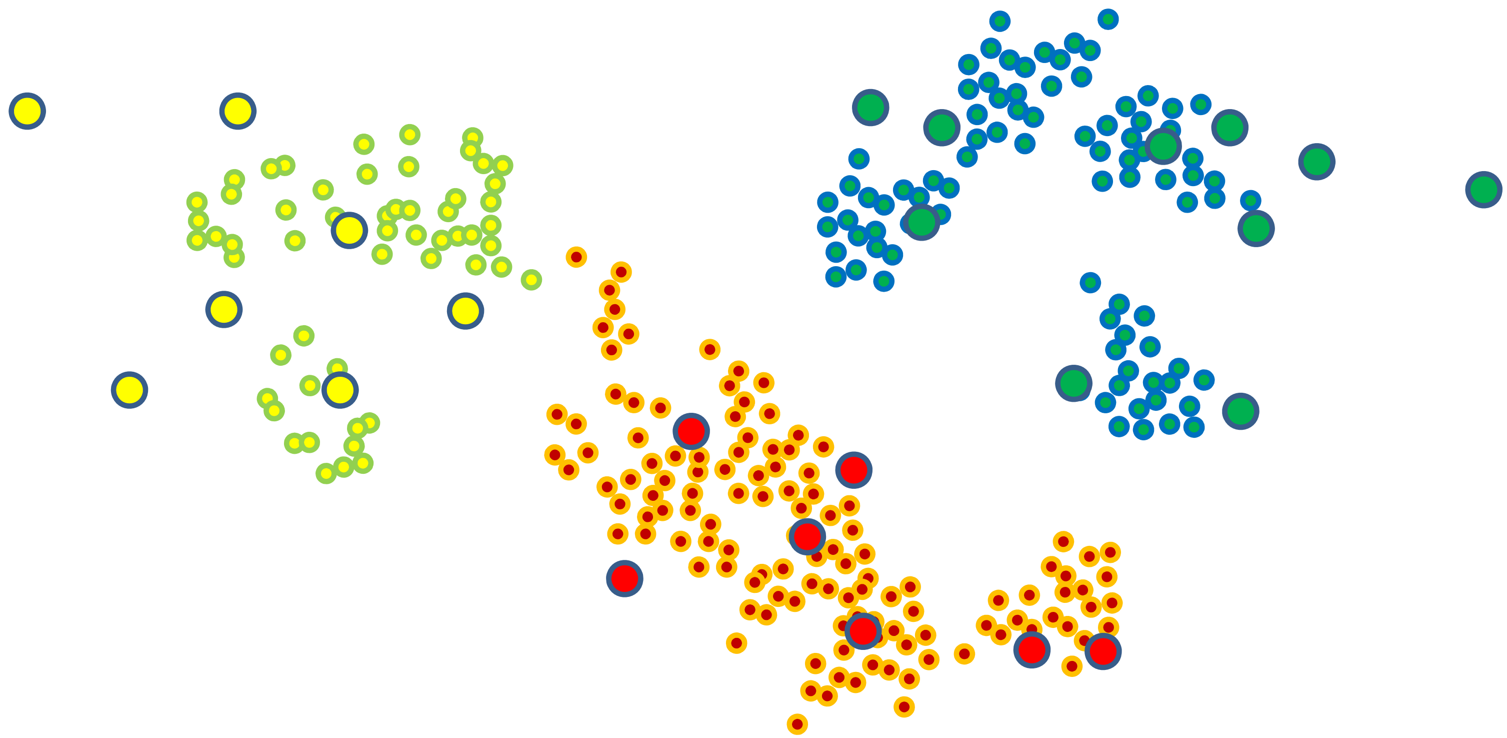
Supervised Clustering

Add training set of labeled data (pre-clustered)



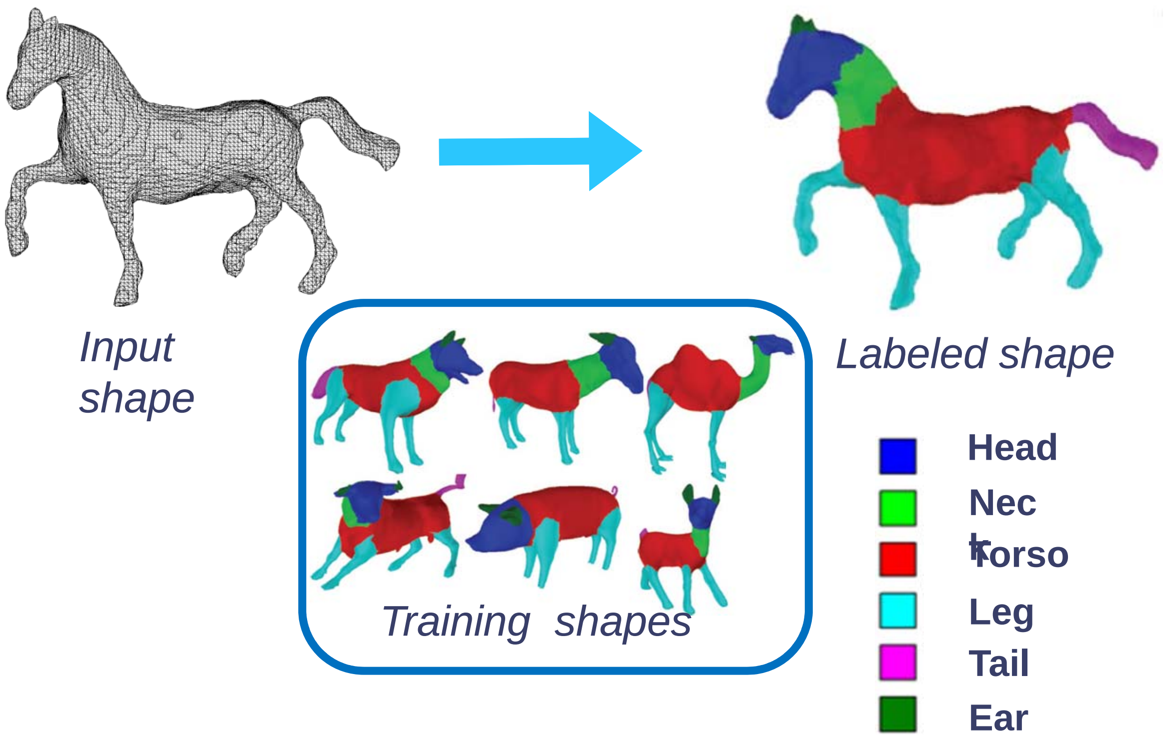
Supervised Clustering

Then clustering becomes easy...



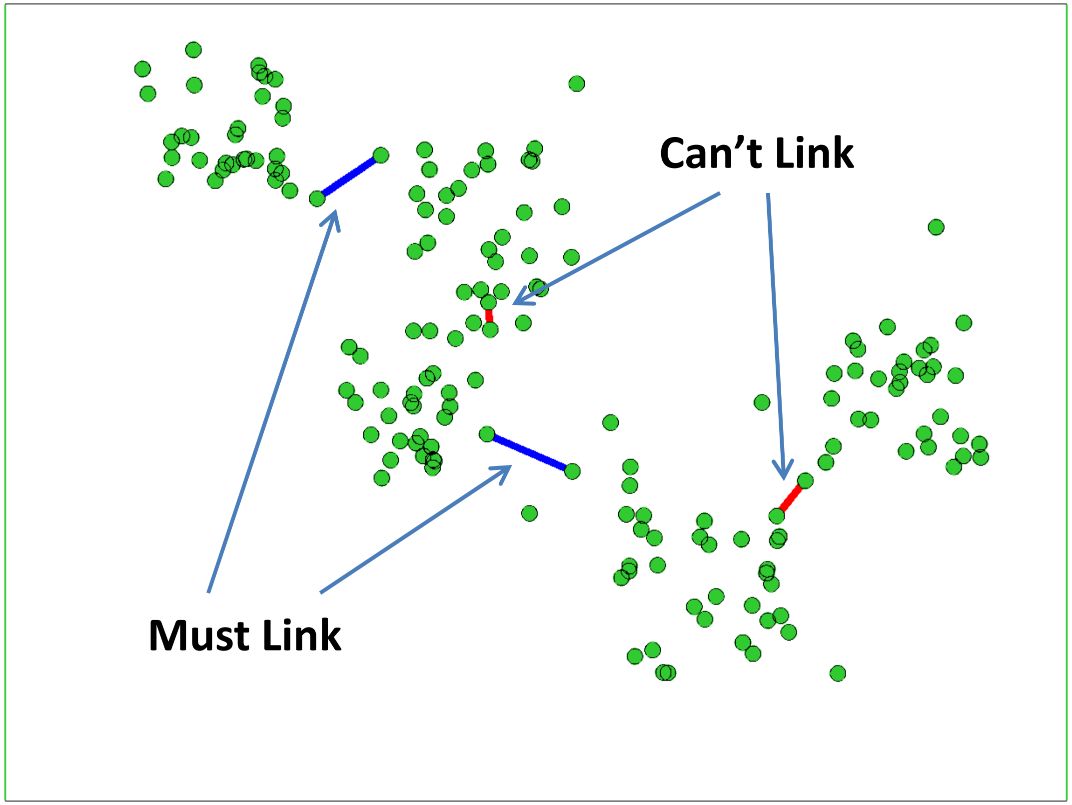
Supervised segmentation

[Kalogerakis et al.10, van Kaick et al. 11]



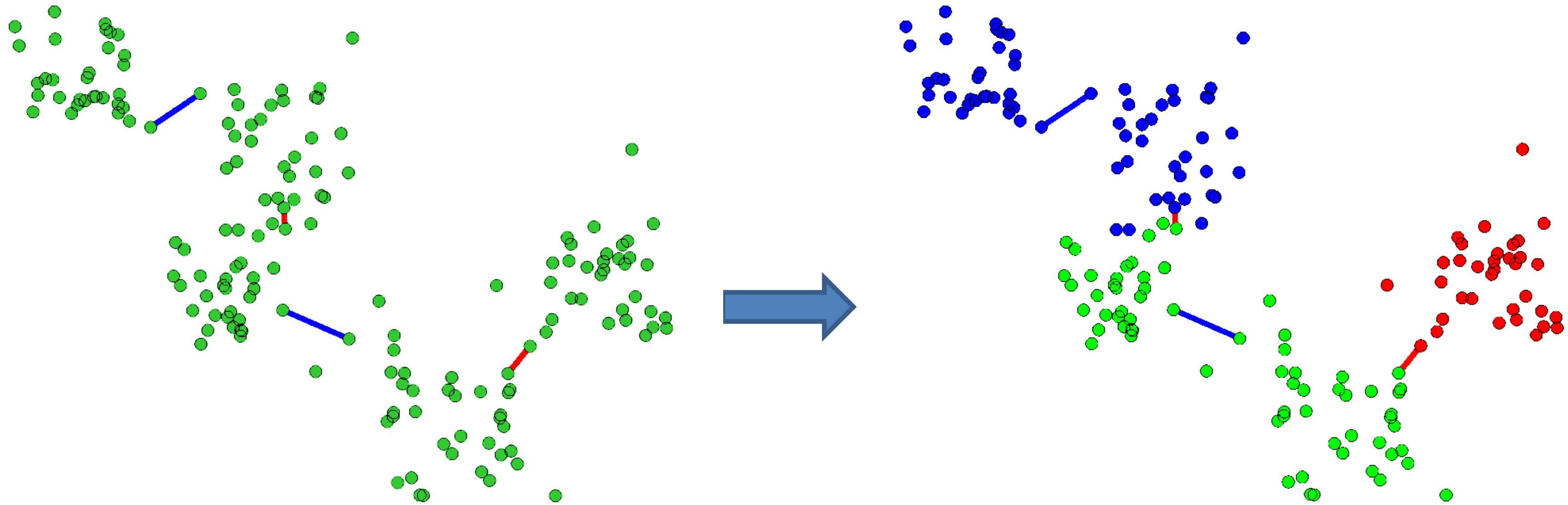
Semi-Supervised Clustering

- Supervision as pair-wise constraints:
  - **Must Link and Cannot-Link**

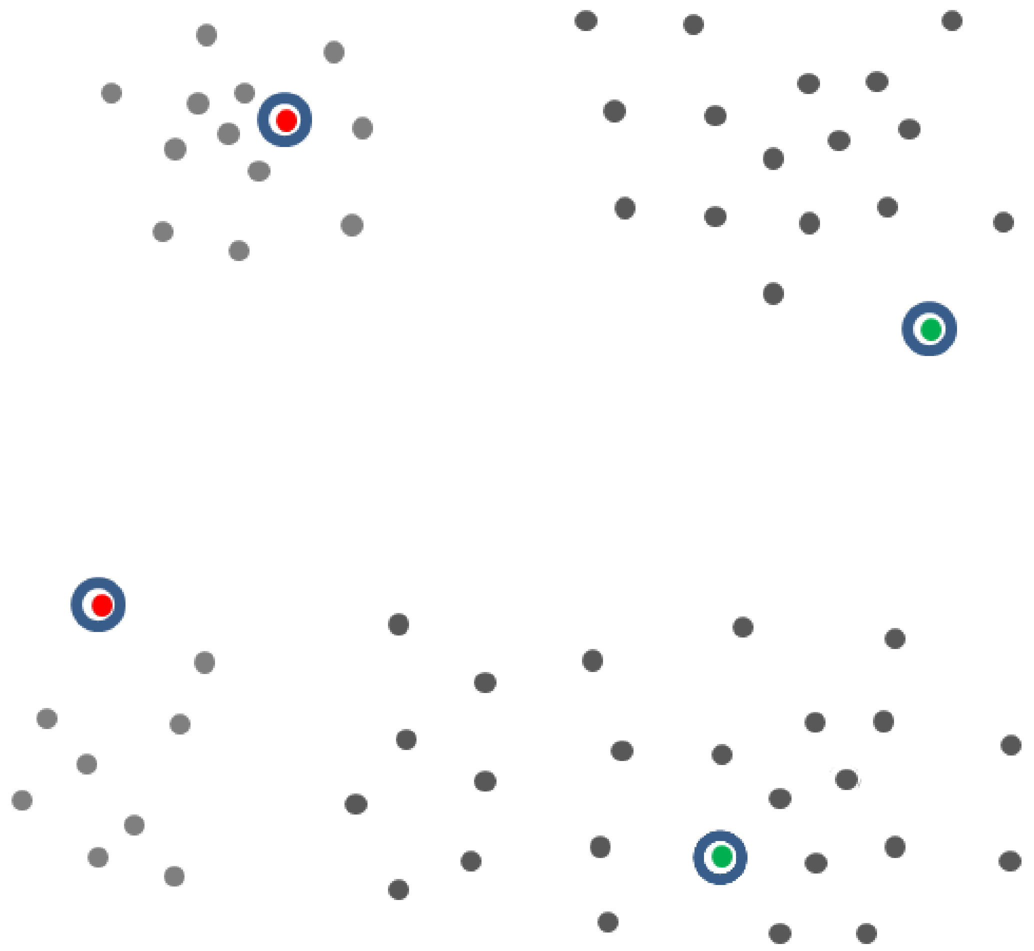


Semi-Supervised Clustering

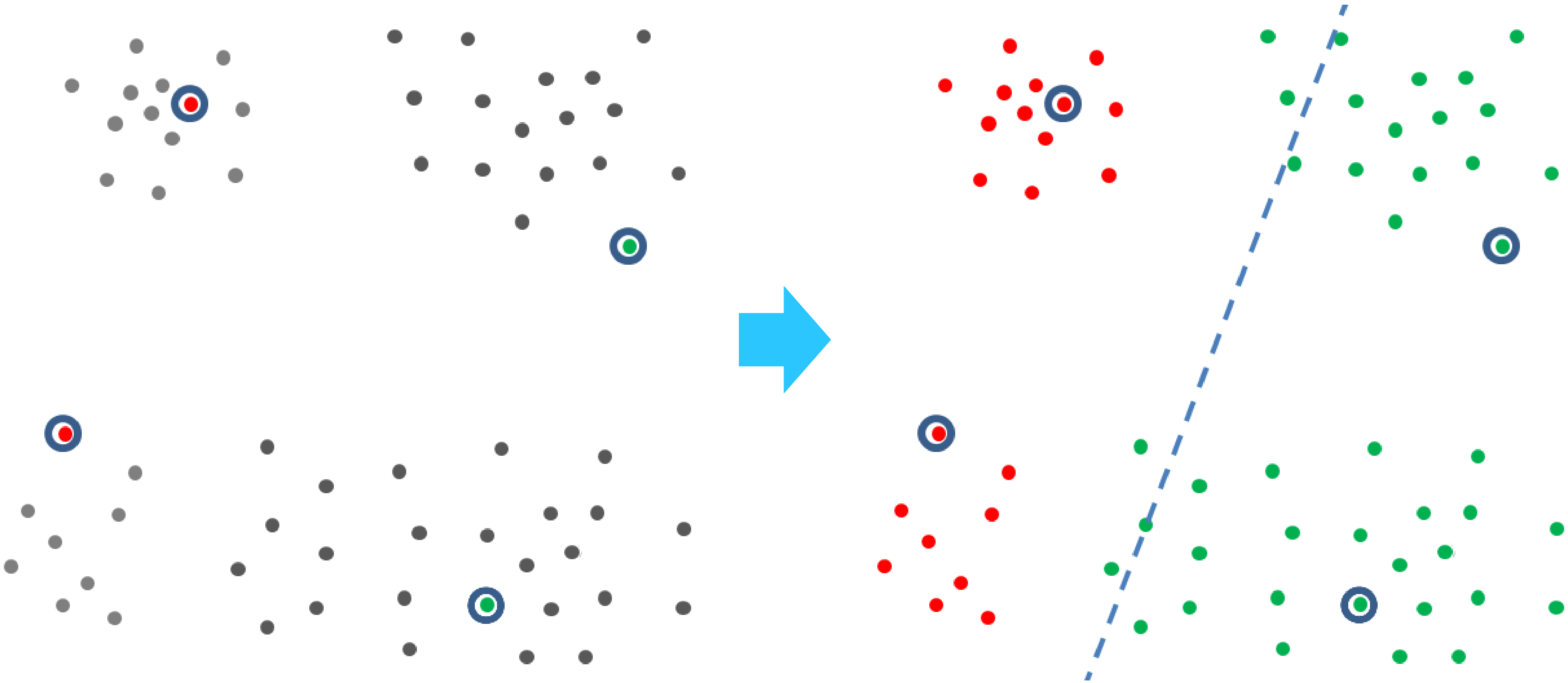
- Cluster data while respecting constraints



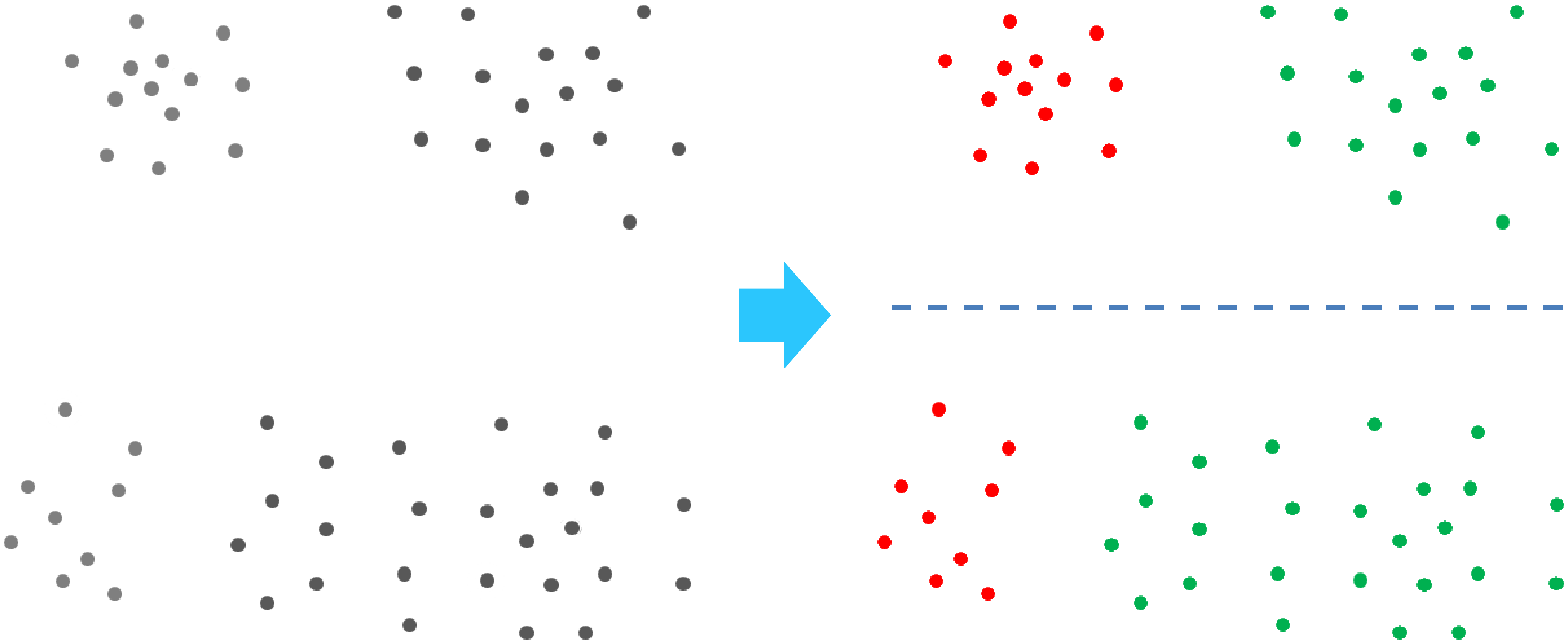
Learning from labeled and unlabeled data



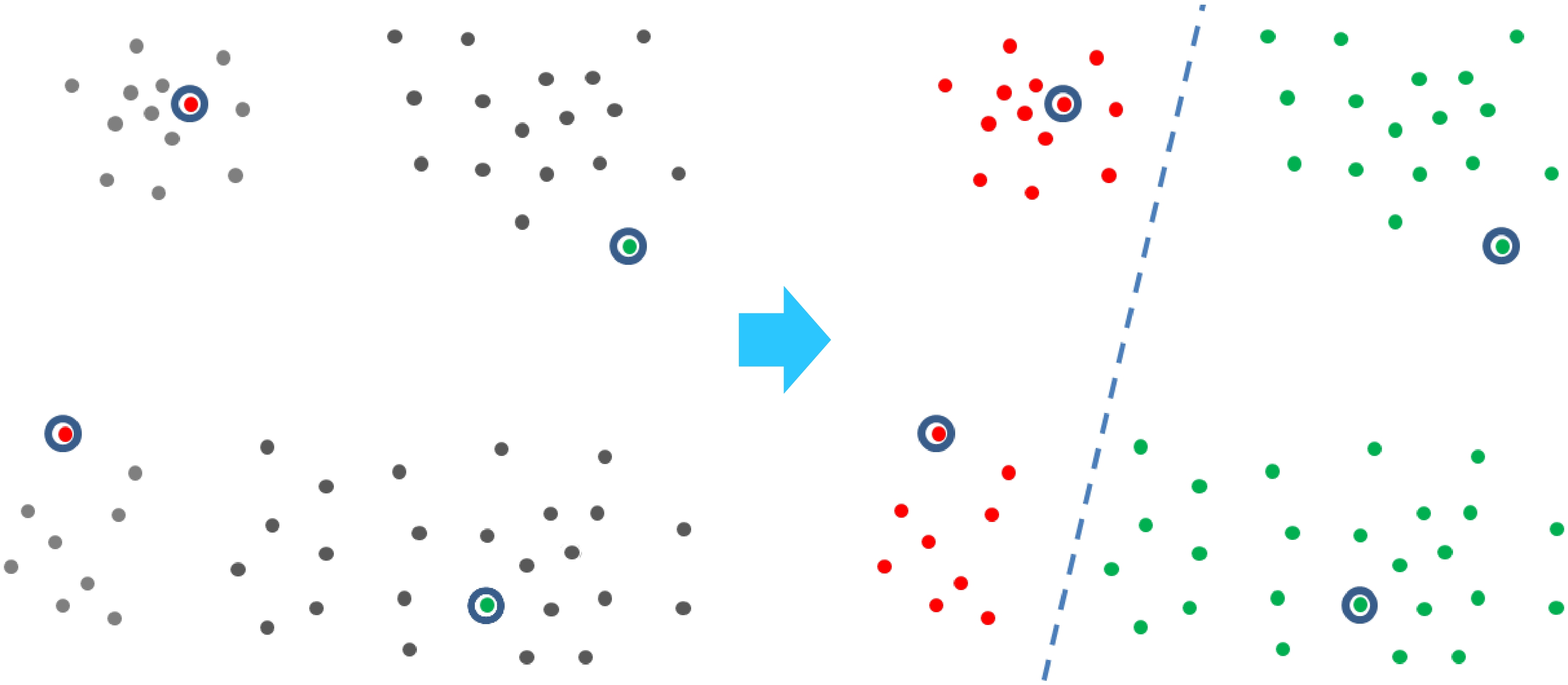
Supervised learning



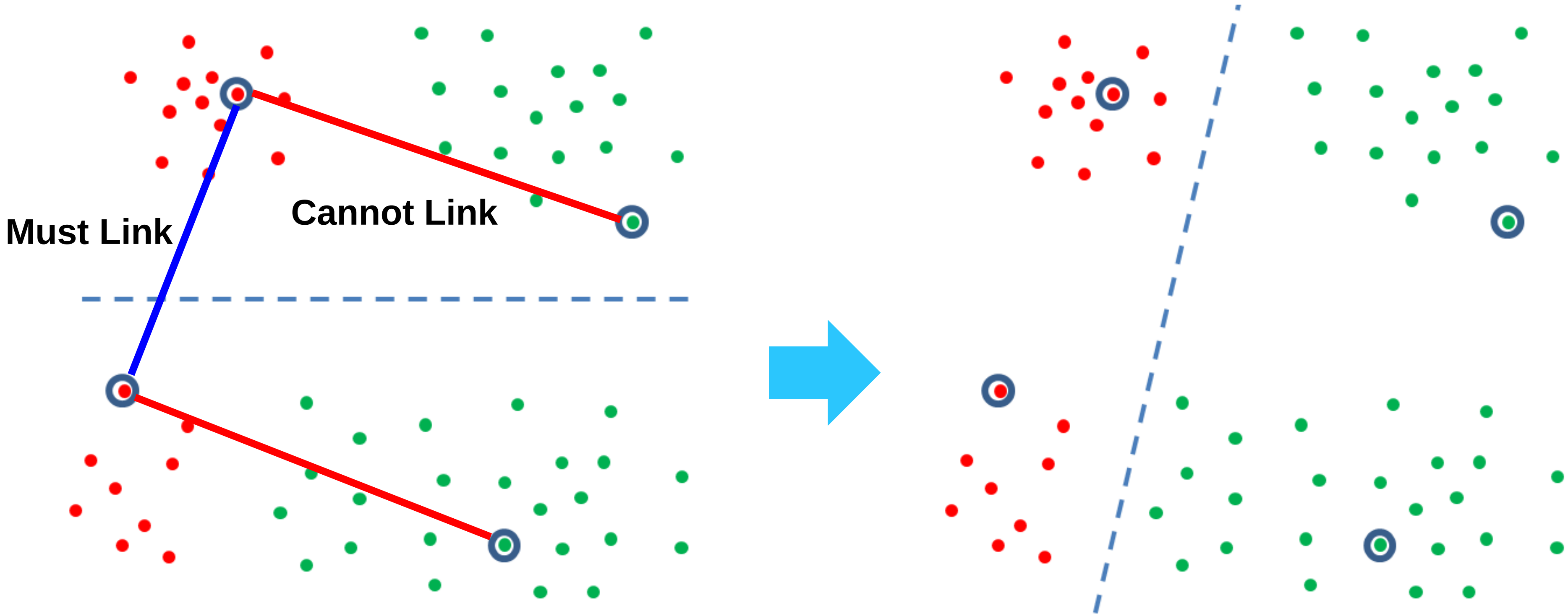
Unsupervised learning



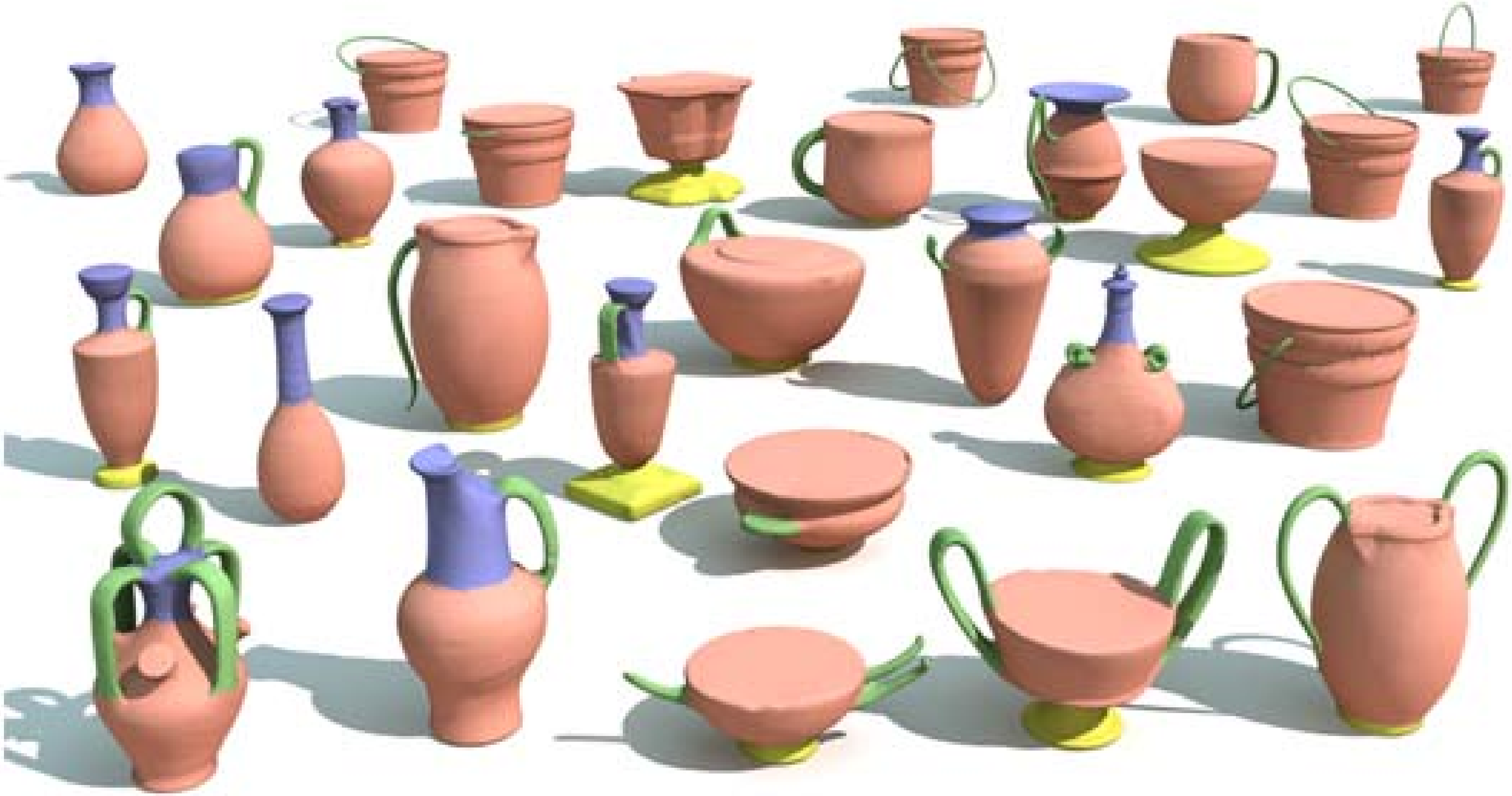
Semi-supervised learning



Constrained clustering

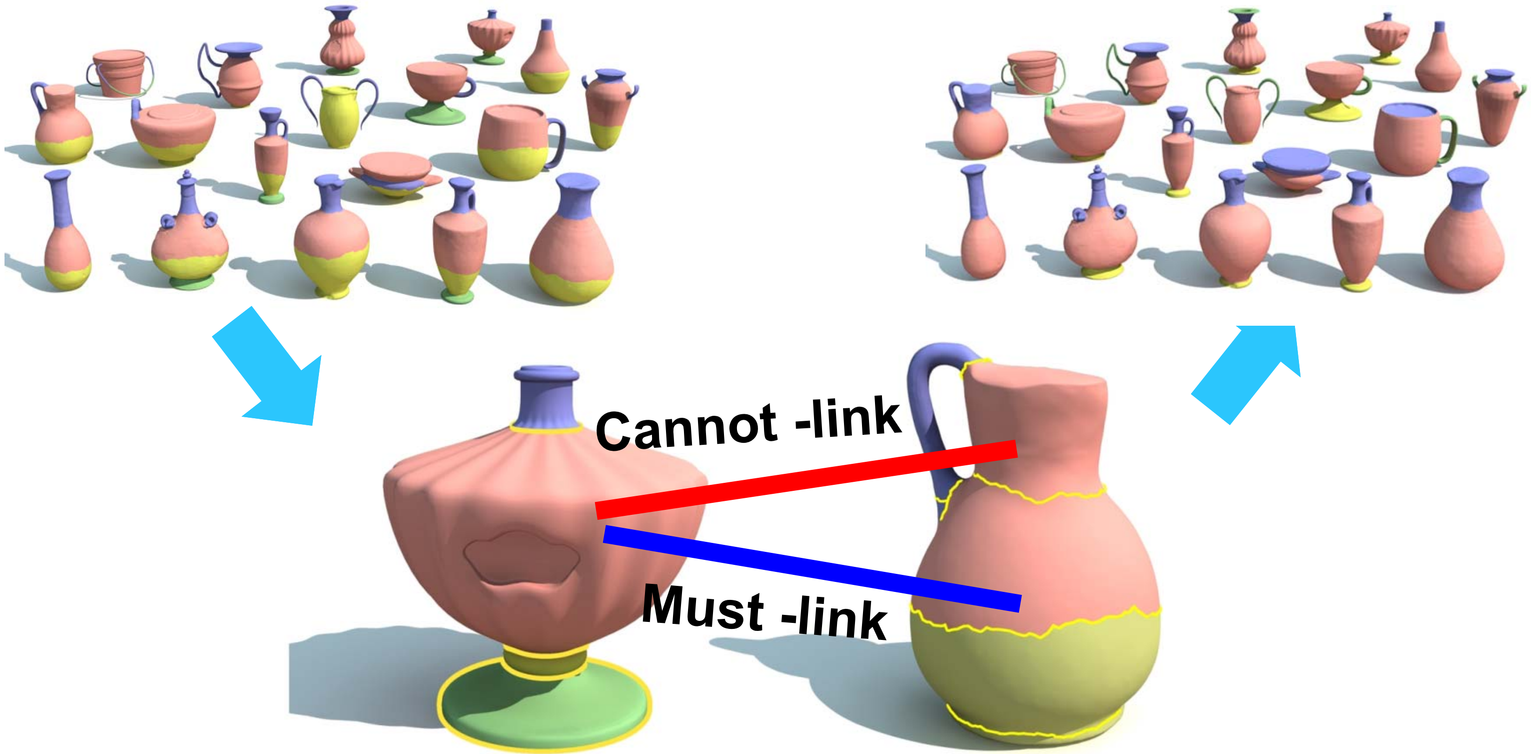


Active Co-analysis of a Set of Shapes
Wang et al. SIGGRAPH ASIA 2012

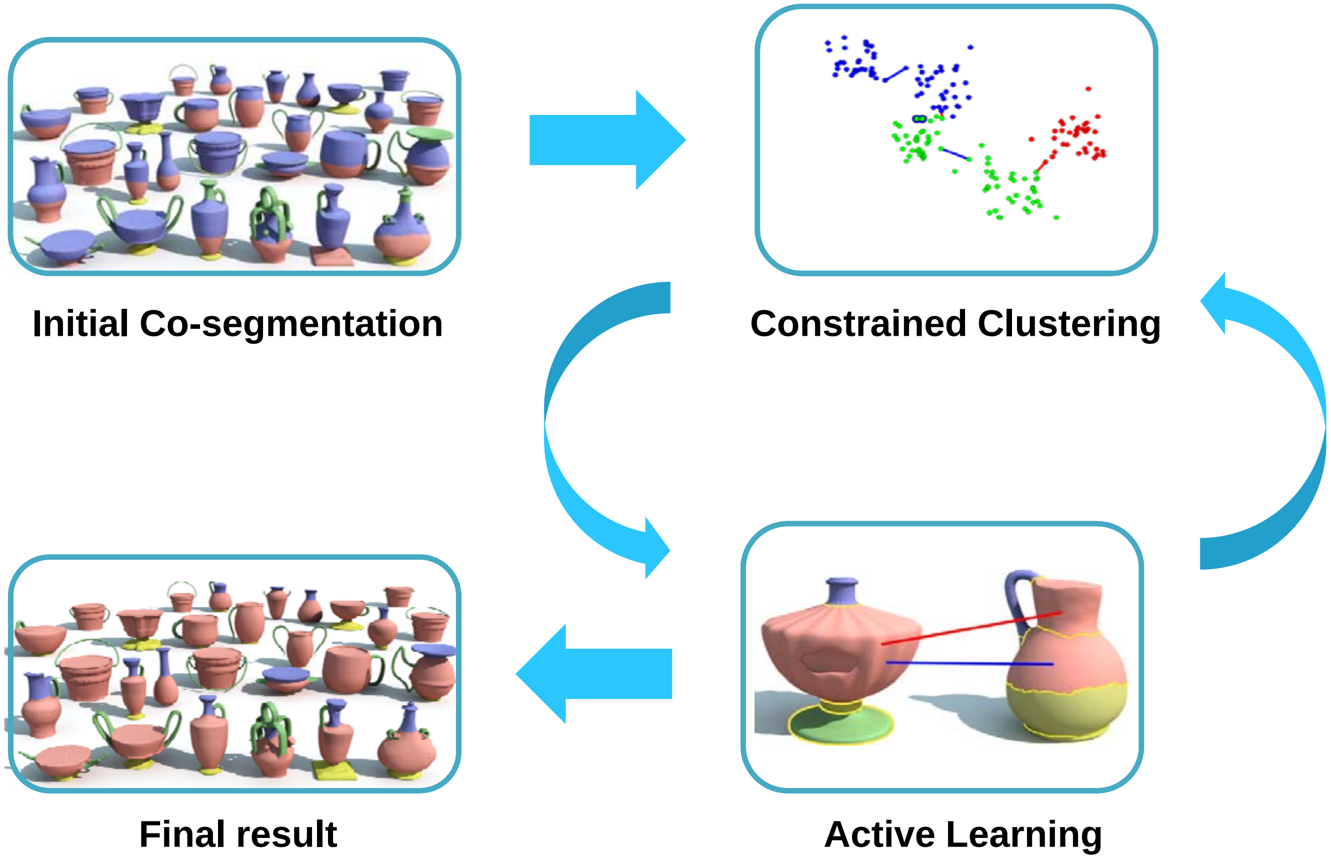


Active Co-Analysis

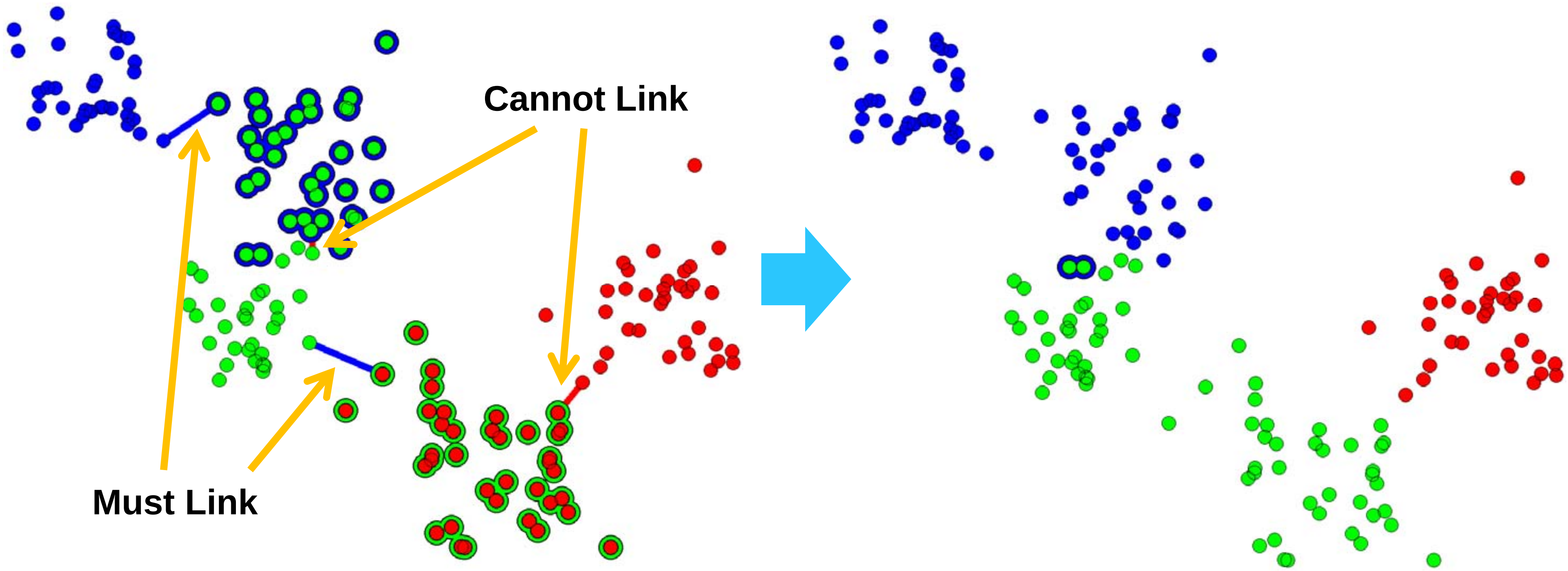
- A semi-supervised method for co-segmentation with *minimal* user input



Automatically suggest the user which constraints can be effective

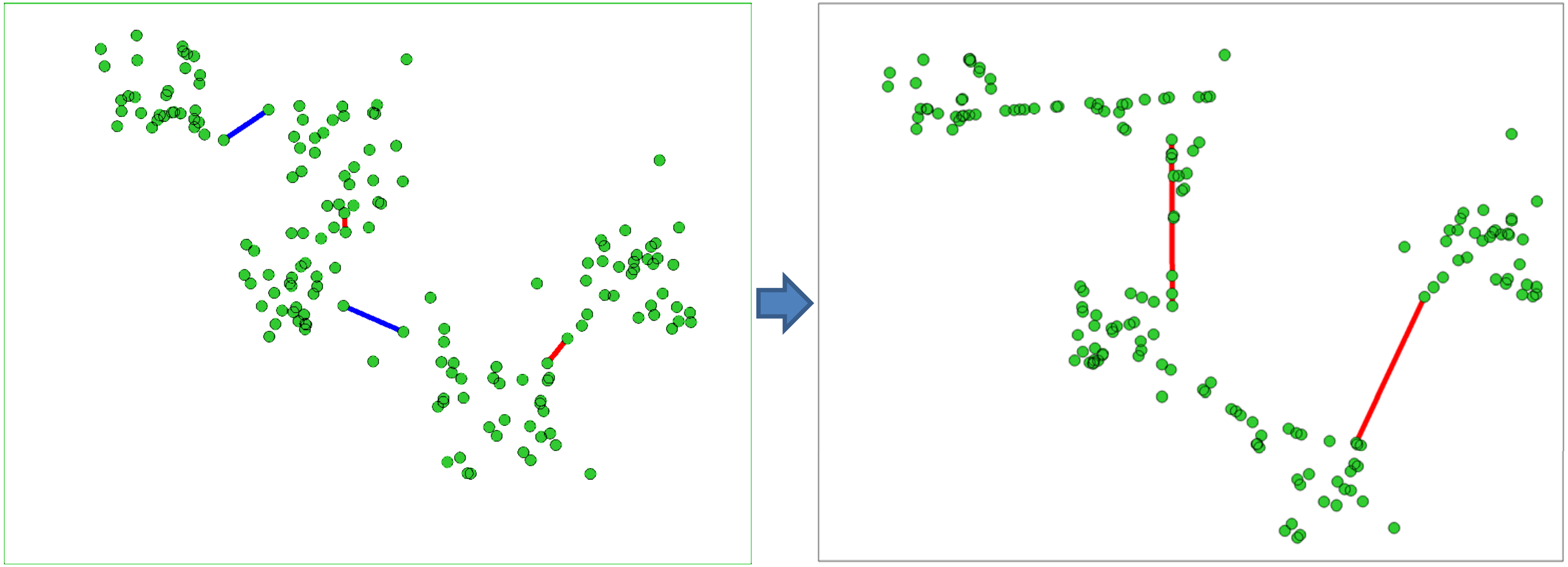


Constrained Clustering



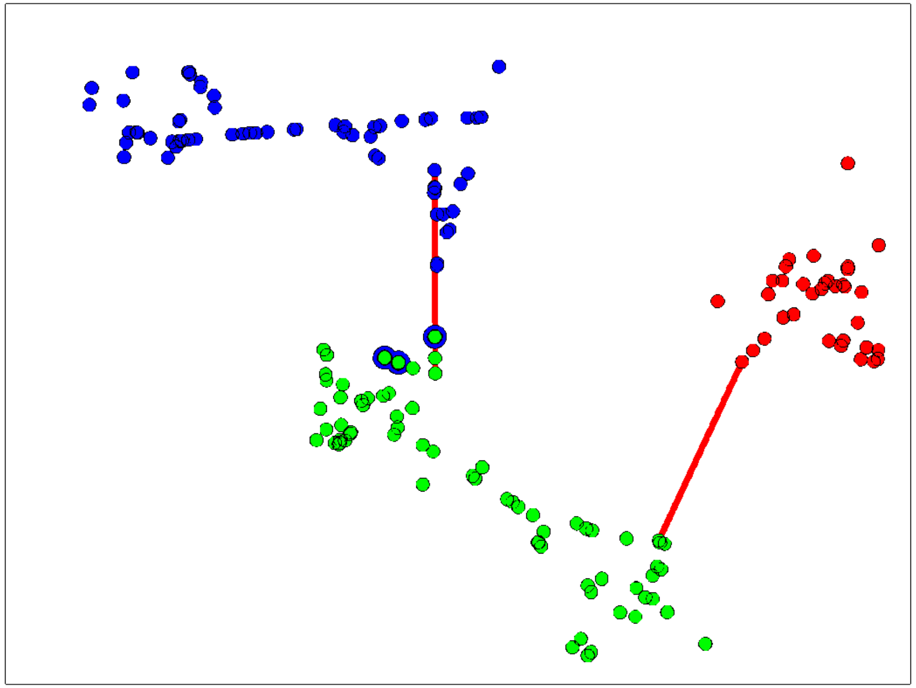
Spring System

- A spring system is used to re-embed all the points in the feature space, so the result of clustering will satisfy constraints.

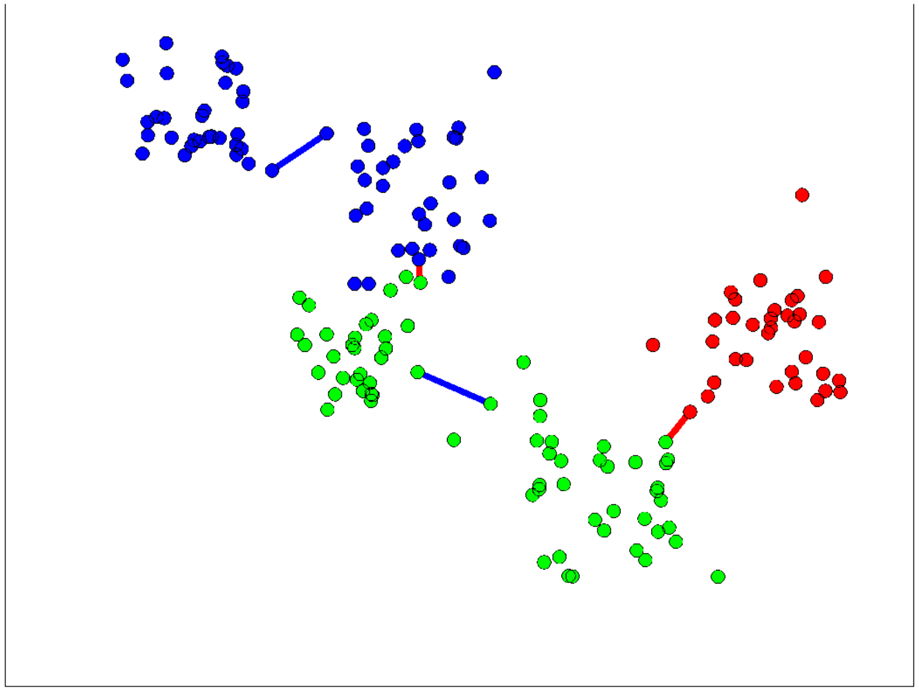


Spring System

- Result of clustering after re-embedding (mistakes marked with circle):

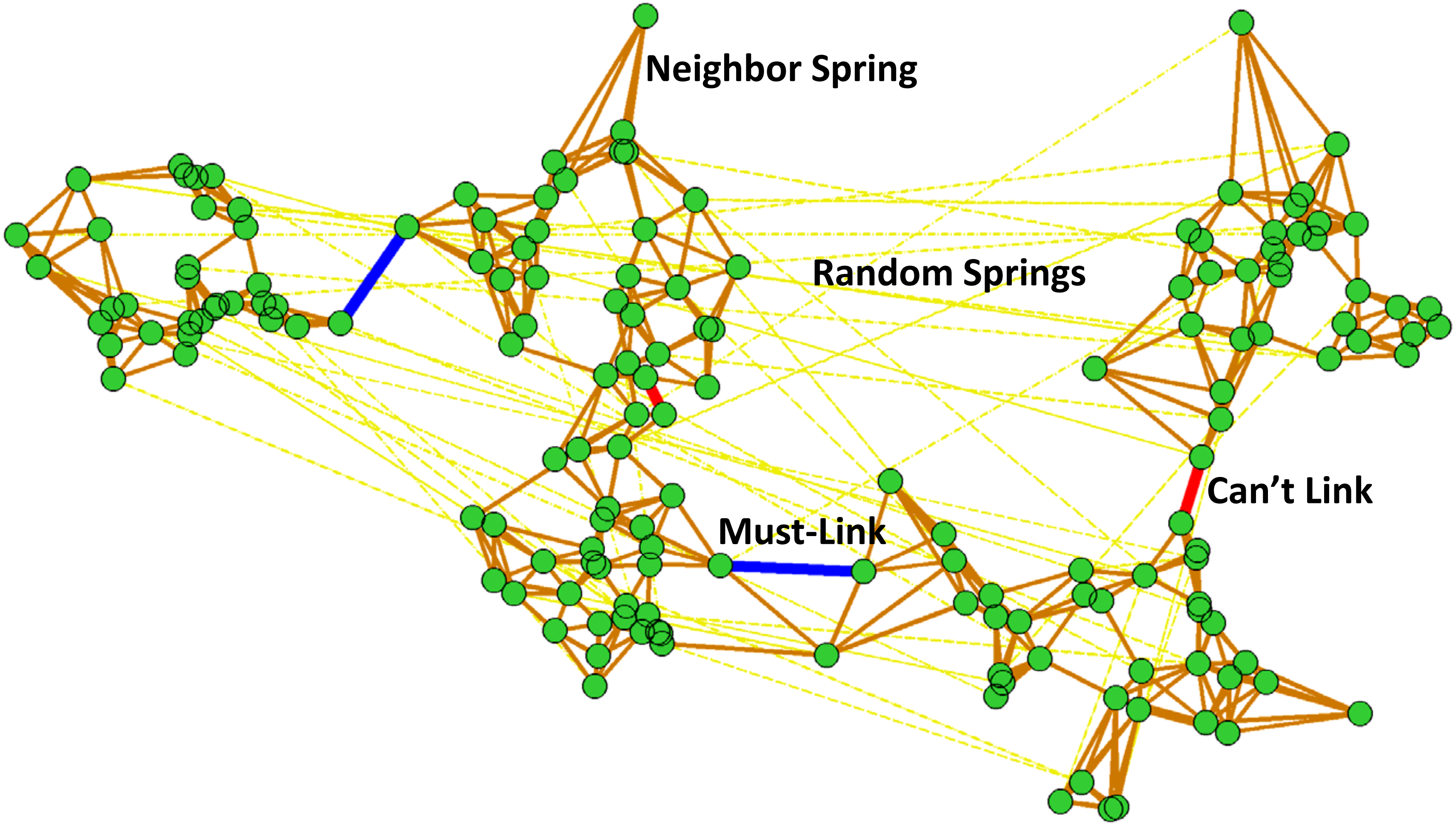


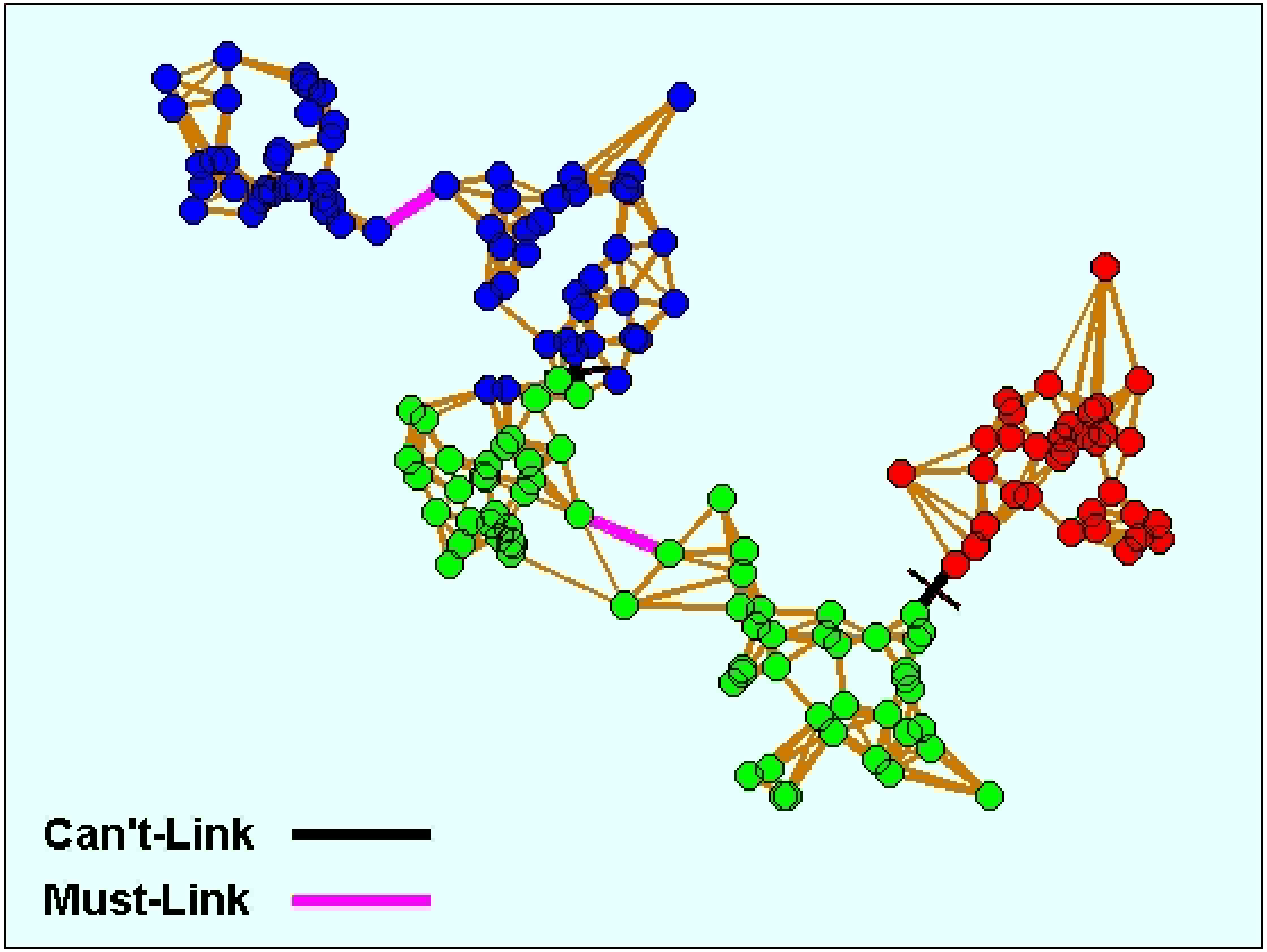
Result of Spring Re-embedding and Clustering

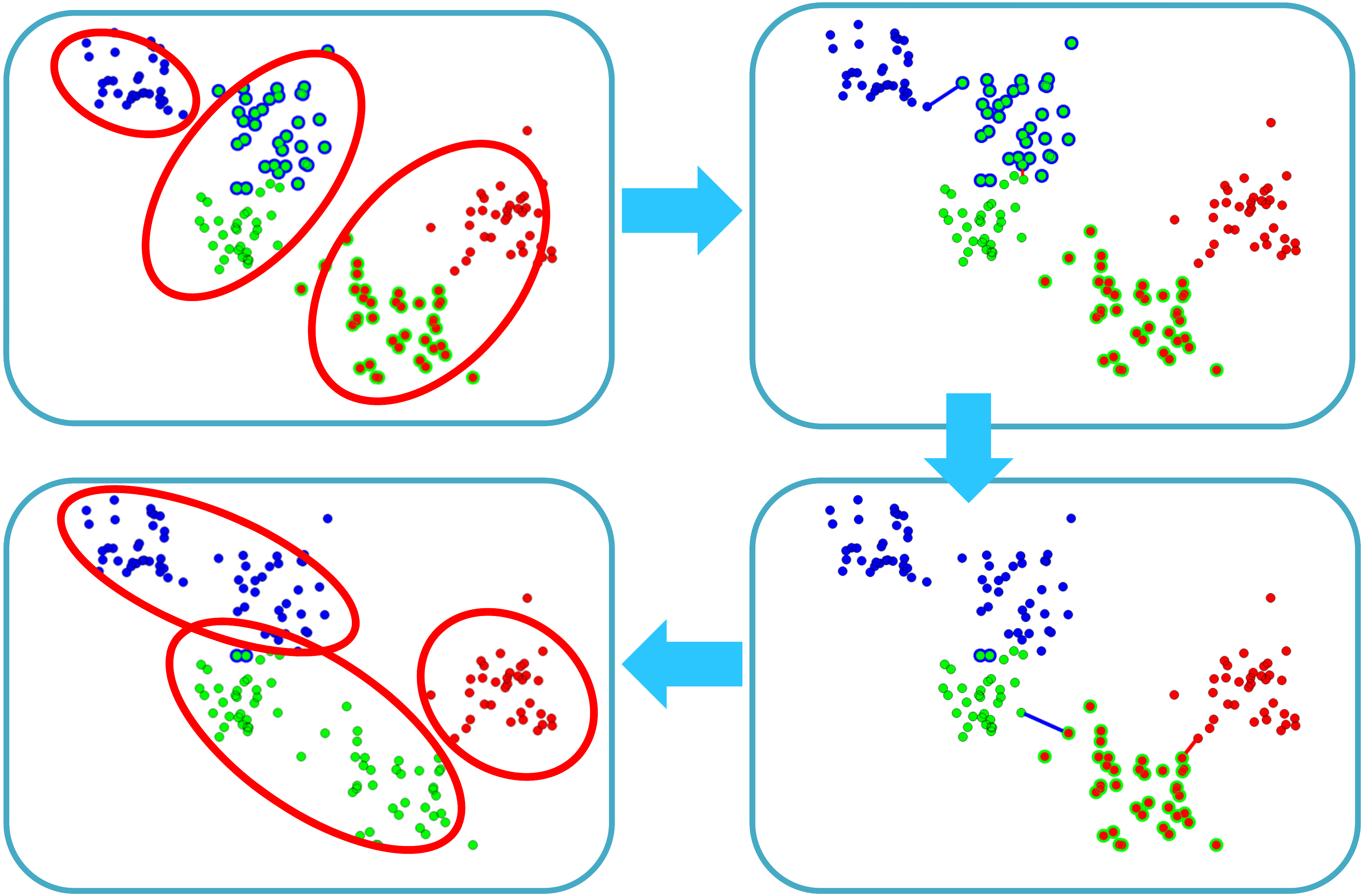


Final Result

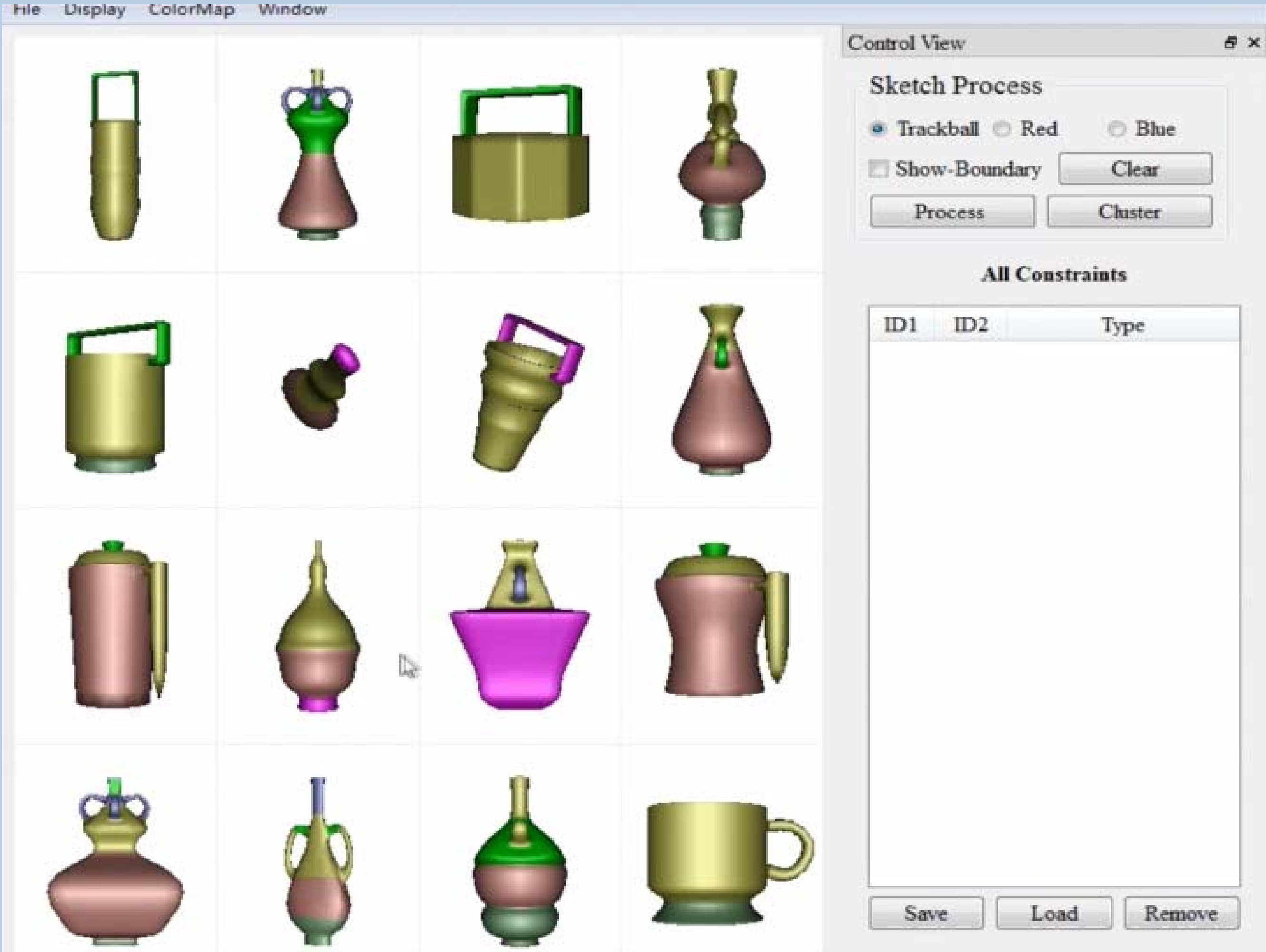
Spring System




Can't-Link
Must-Link

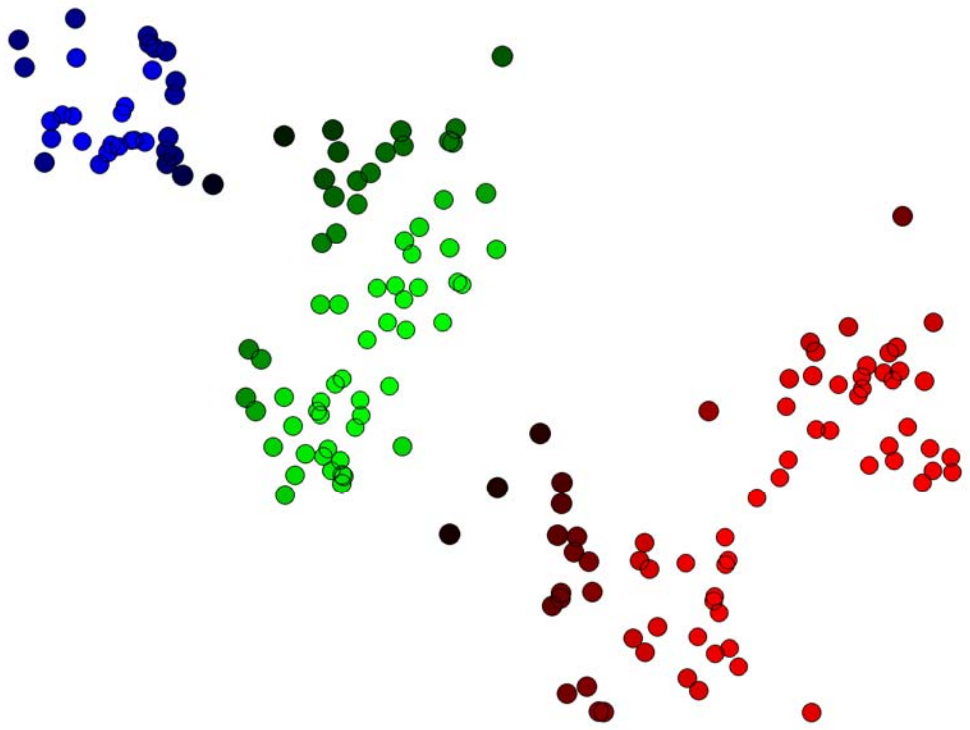


Constrained clustering & Co-segmentation



Uncertain points

- “Uncertain” points are located using the Silhouette Index:

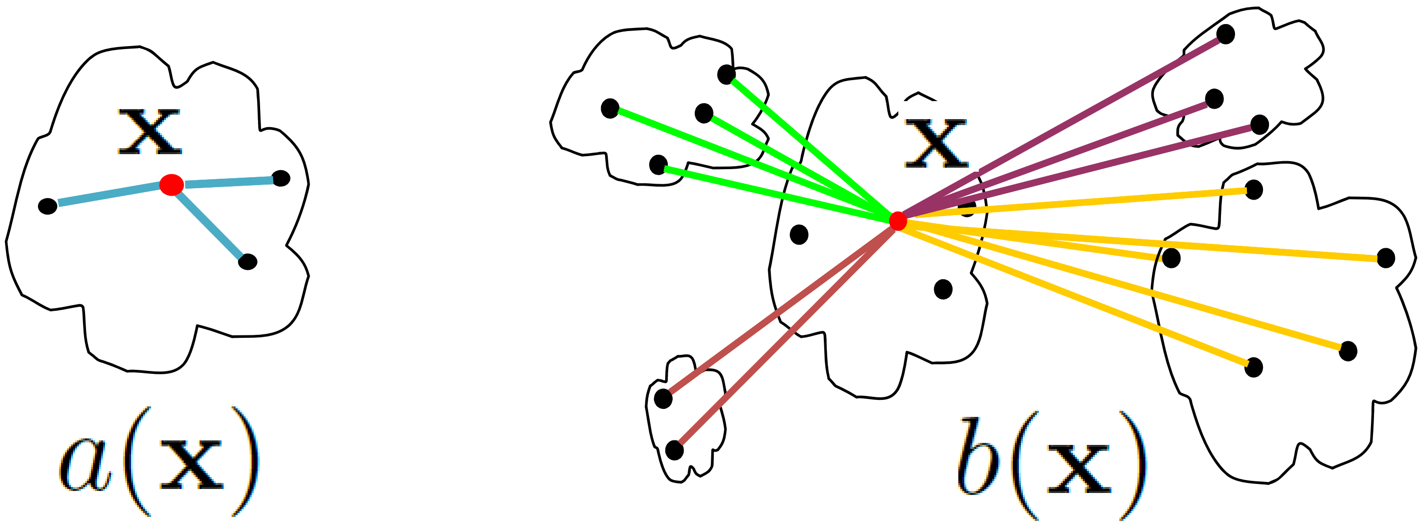


Darker points have lower confidence

Silhouette Index

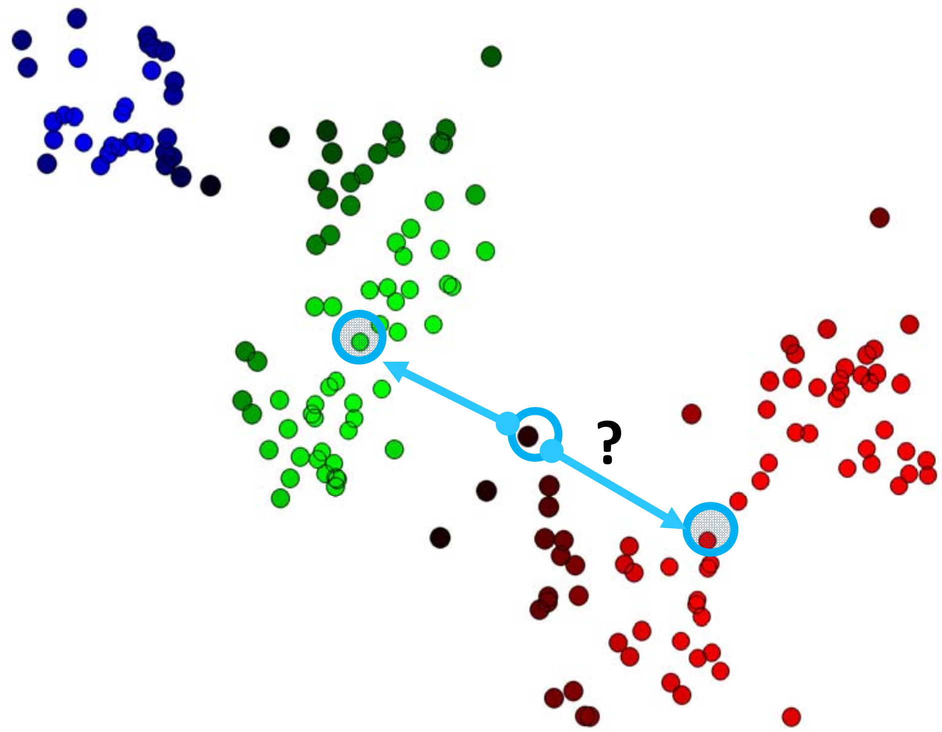
- Silhouette Index of node **x**:

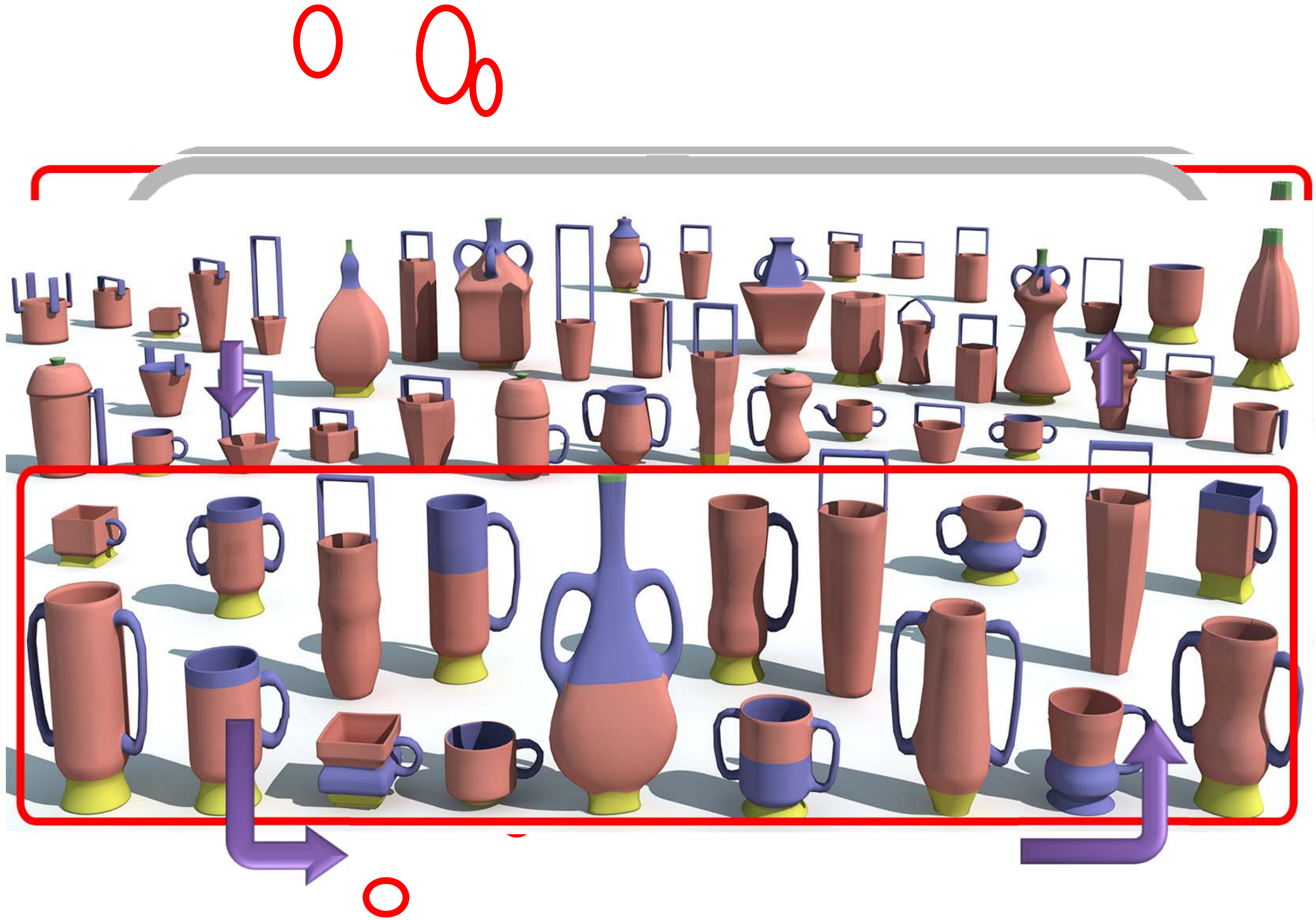
$$S(\mathbf{x}) = \frac{b(\mathbf{x}) - a(\mathbf{x})}{\max[b(\mathbf{x}), a(\mathbf{x})]}$$



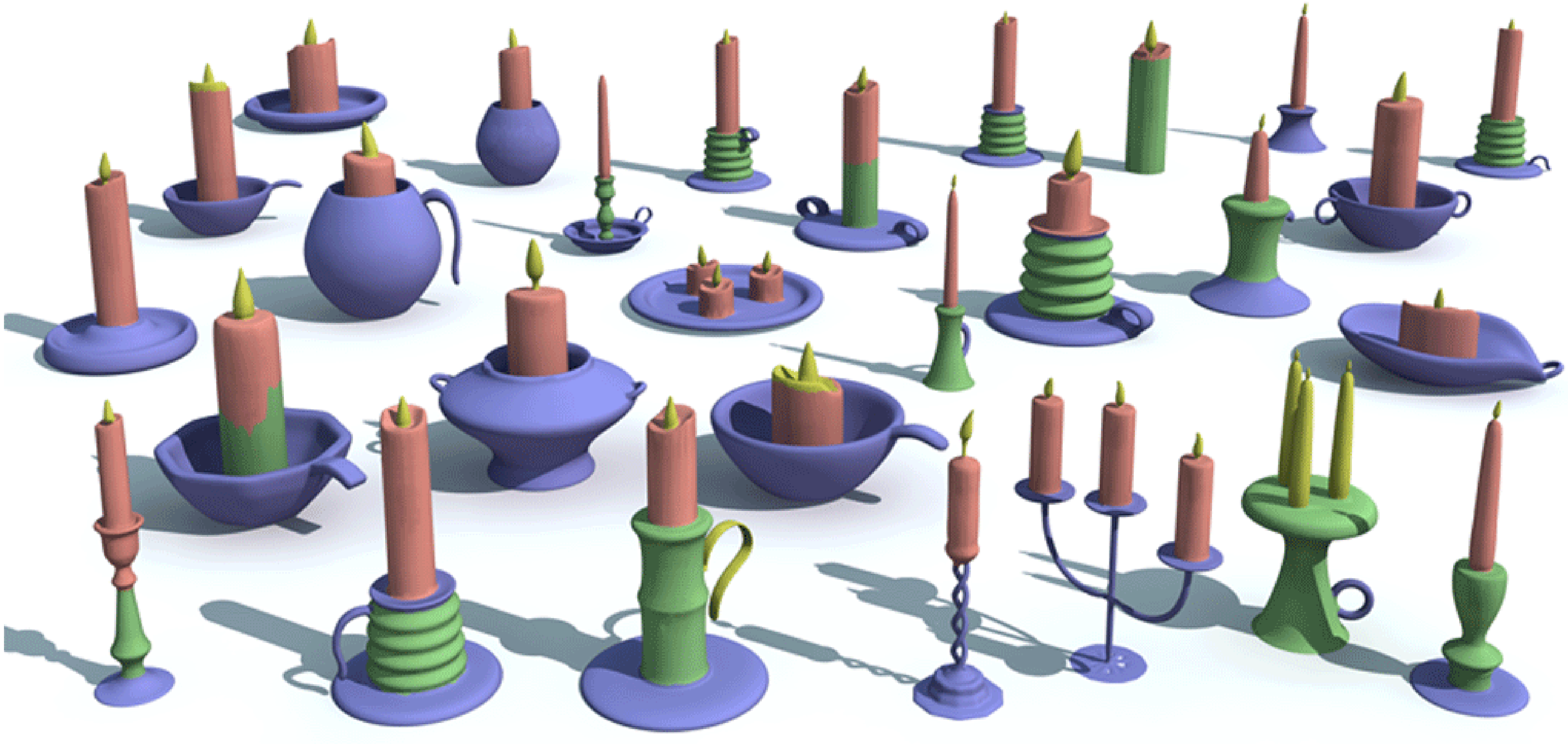
Constraint Suggestion

- Pick super-faces with lowest confidence
- Pick the highest confidence super-faces
- Ask the user to add constraints between such pairs



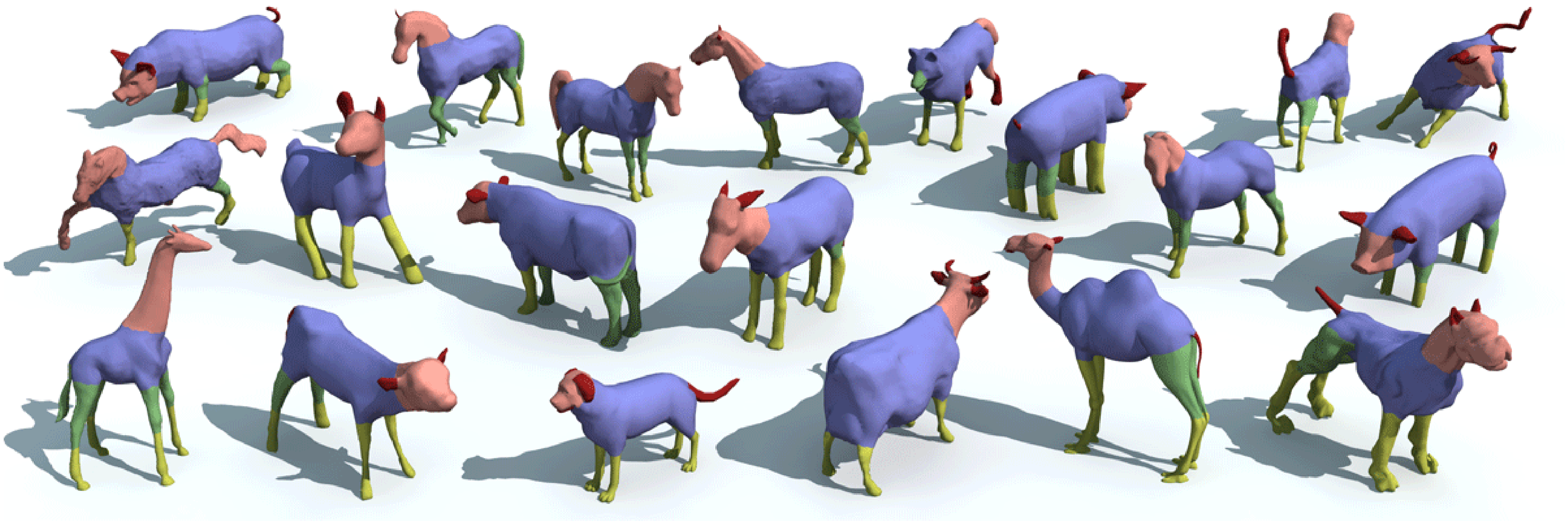


Candelabra: 28 shapes, 164 super-faces,24 constraints



Initial

Fourleg: 20 shapes, 264 super-faces,69 constraints

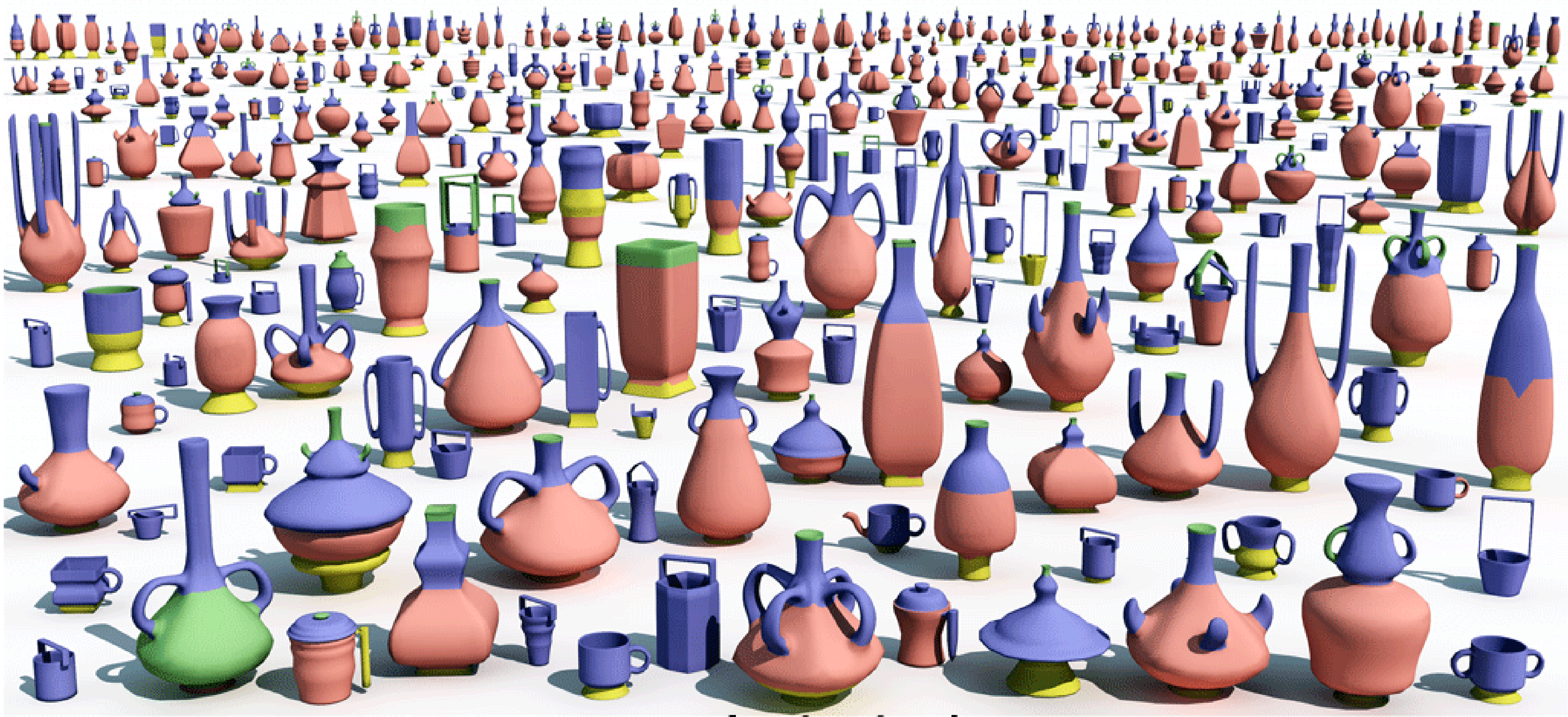


Initial

Tele-alien: 200 shapes, 1869 super-faces,106 constraints

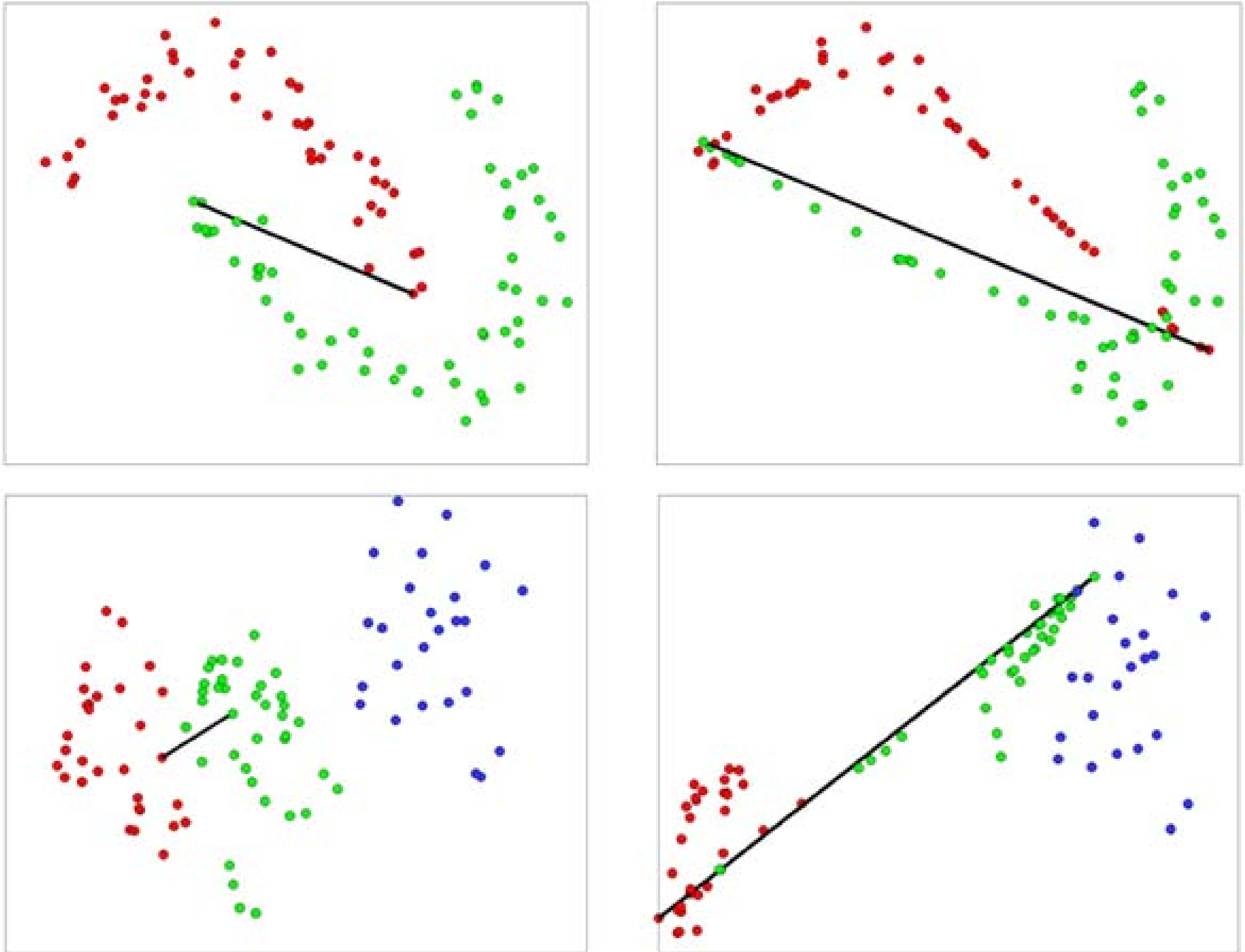
Initial

Vase: 300 shapes, 1527 super-faces,44 constraints



Initial

Cannot-Link Springs

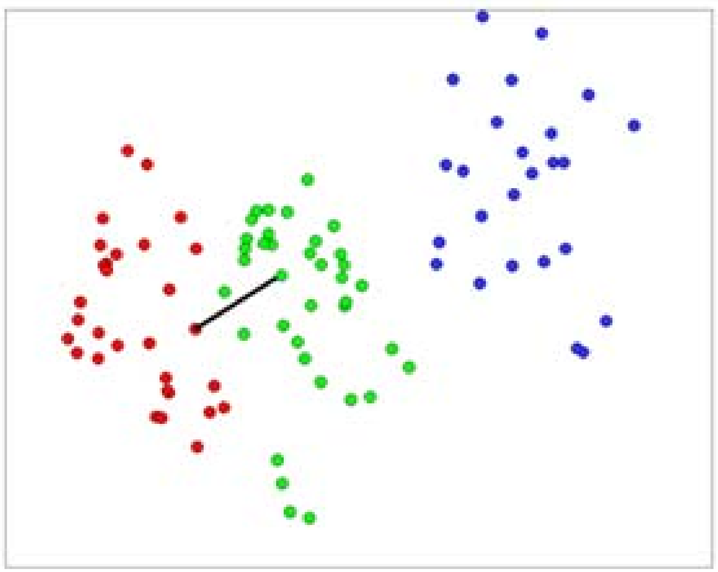


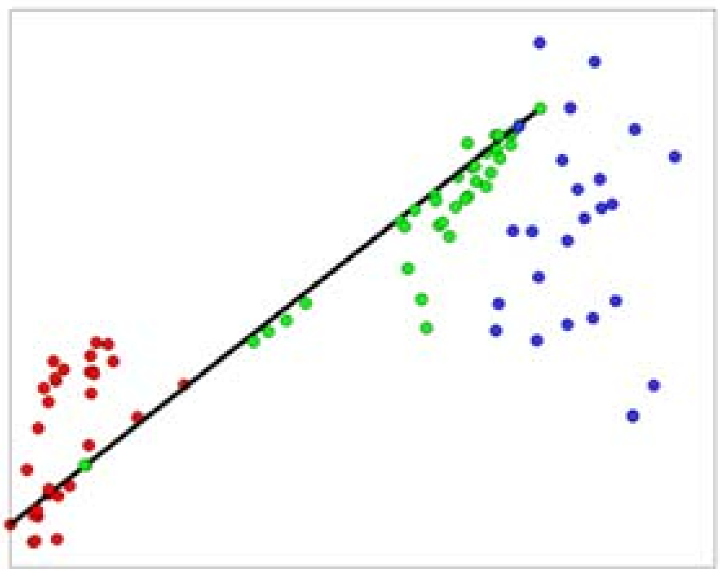
Constraints as Features
CVPR 2013

- **Goal**: Modify data so distances fit constraints
- **Basic idea**:
  - Convert constraints into extra-features that are added to the data (augmentation)
  - Recalculate the distances
  - Unconstrained clustering of the modified data
  - Clustering result more likely to satisfy constraints
- Apply this idea to Cannot-Link constraints
- Must-Link constraints handled differently

Cannot-link Constraints

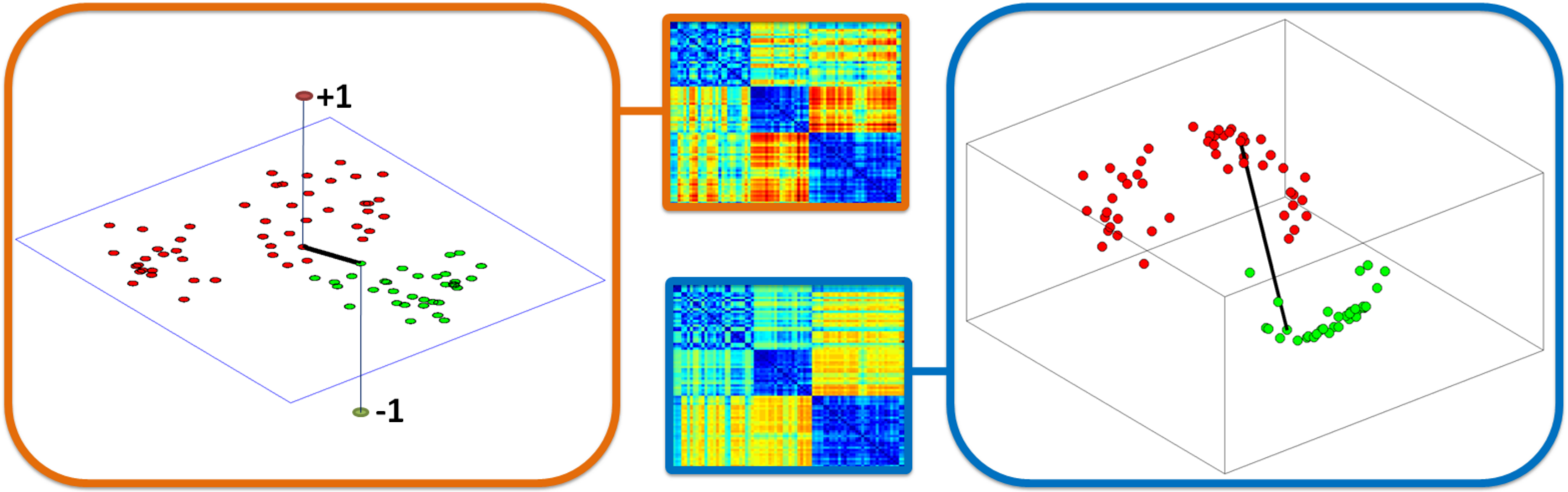
- Points should be distant.
- What value should be given: $D(c_1, c_2) = X$?
  - Should relate to max(D(x, y)), but how?
- If modified, how to restore triangle-inequality?





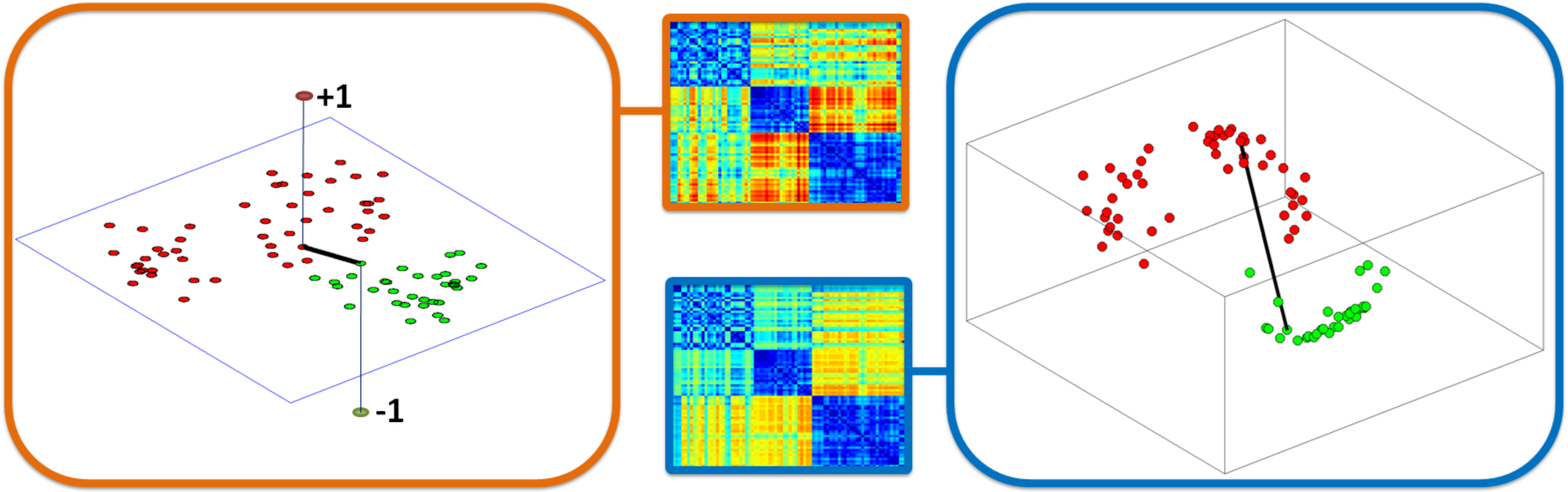
Constraints as Features

- **Solution:**
  – Add extra-dimension, where Cannot-Link pair is far away (±1):



Constraints as Features

- **Solution:**
  - Add extra-dimension, where Cannot-Link pair is far away (±1):
  - **What values should other points be given?**



Constraints as Features

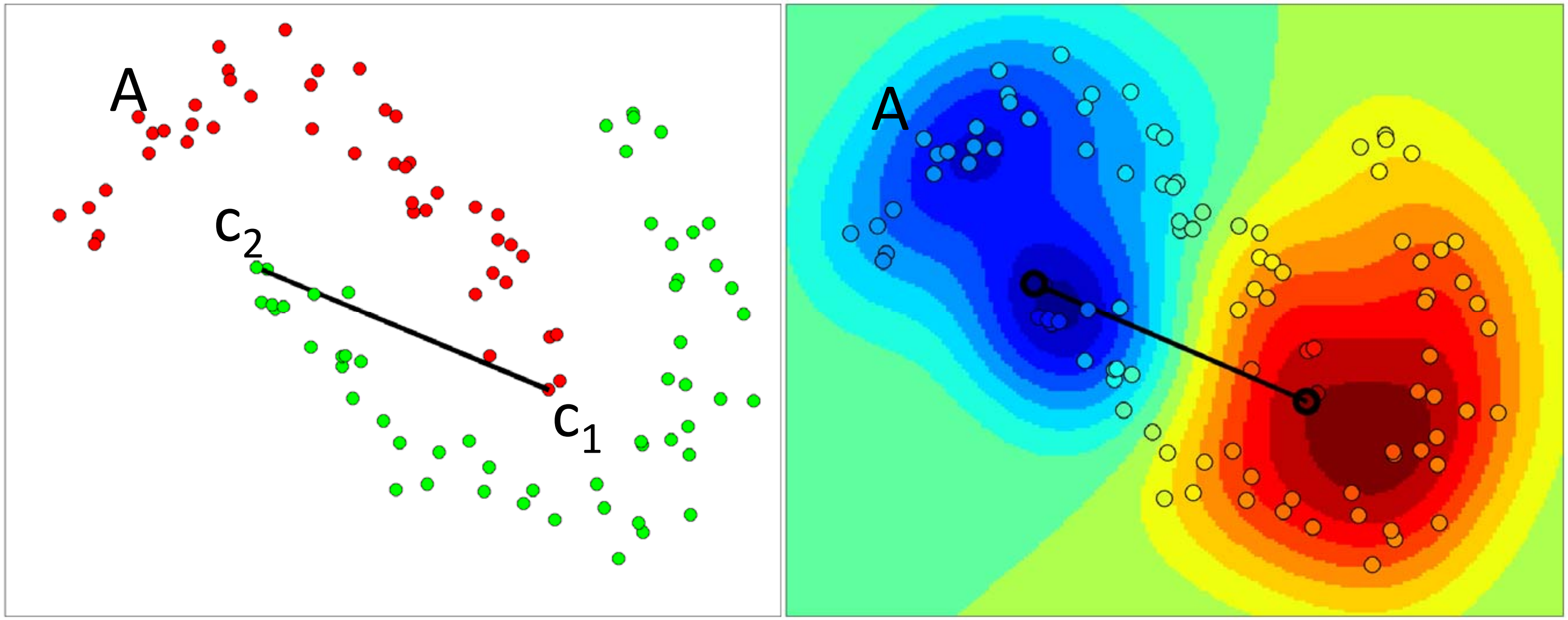
- Values of other points:
  - Points closer to c_1 should have values closer to +1,
  - Points closer to c_2 should have values closer to -1
- Formulation:

$$v_i = \frac{(\varphi(i, c2) - \varphi(i, c1))}{(\varphi(i, c2) + \varphi(i, c1))}$$

- Simple distance $\varphi(i, c1)$ does not convey real closeness.

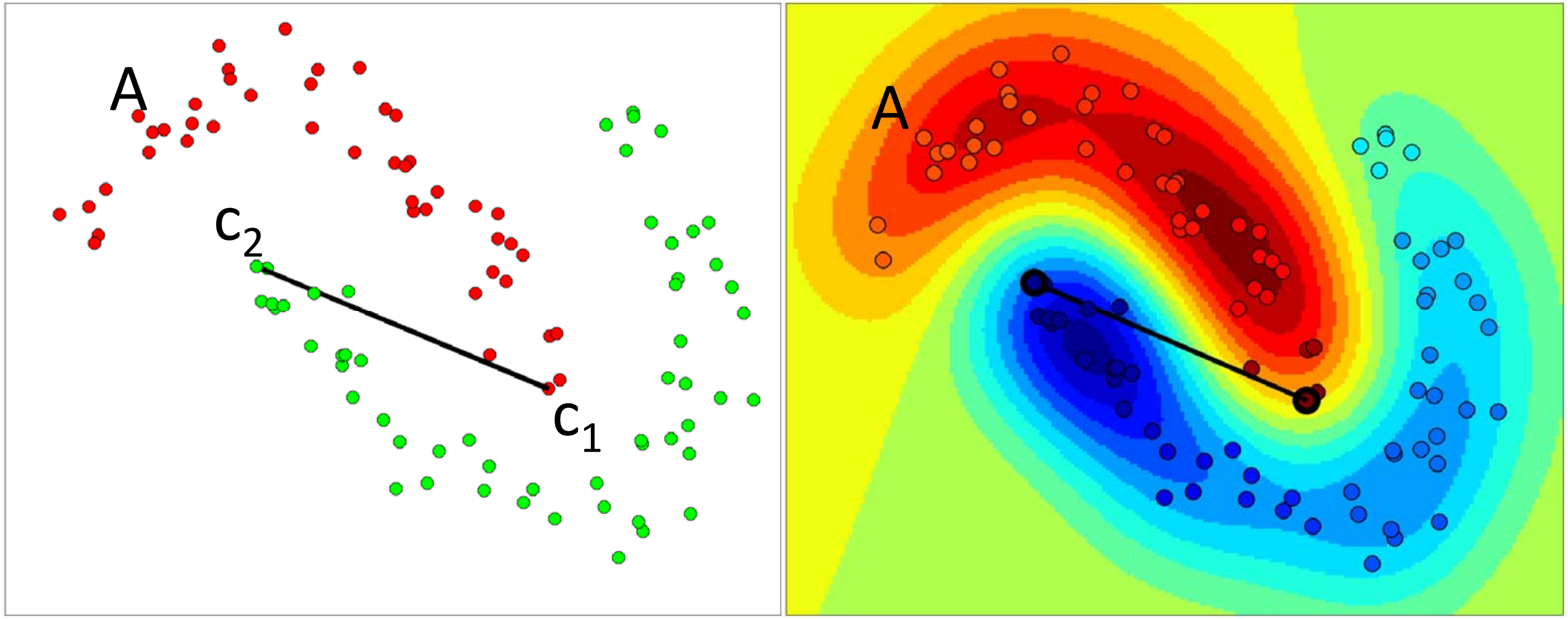
Constraints as Features

- Point A should be "closer" to c1, despite smaller Euclidean distance.



Constraints as Features

- Use a Diffusion Map, where this holds true.



Constraints as Features

- Diffusion Maps related to random walk process on a graph
- Affinity Matrix:

$$A_{i,j} = e^{-\frac{D_{i,j}^2}{\sigma^2}}$$

- Eigen-Analysis of normalized A forms a Diffusion Map:

$$\Psi_t(x) = (\lambda_1^t \psi_1(x), \lambda_2^t \psi_2(x), \ldots, \lambda_K^t \psi_K(x))$$

Constraints as Features

- Use Diffusion Map distances:

$$\varphi(x, y) = \left|\Psi_t(x) - \Psi_t(y)\right|$$

- Calculate value of each point in new dimension:

$$v_i = \frac{(\varphi(i, c2) - \varphi(i, c1))}{(\varphi(i, c2) + \varphi(i, c1))}$$

Constraints as Features

- Create new distance matrix, of distances in the new extra dimension:

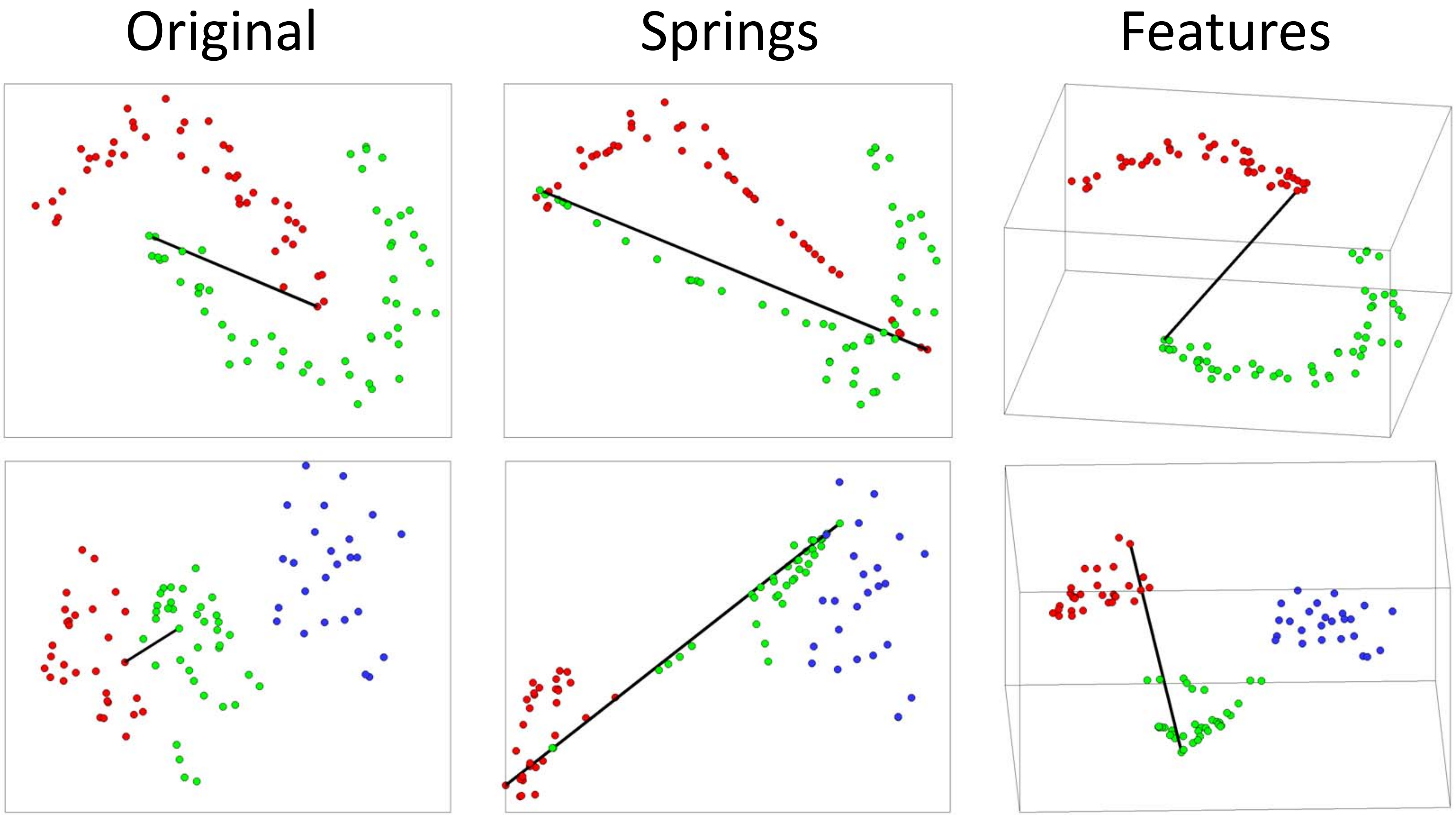
$$D_{i,j}^{(c)} = |v_i - v_j|$$

- Add distance matrix per Cannot-Link:

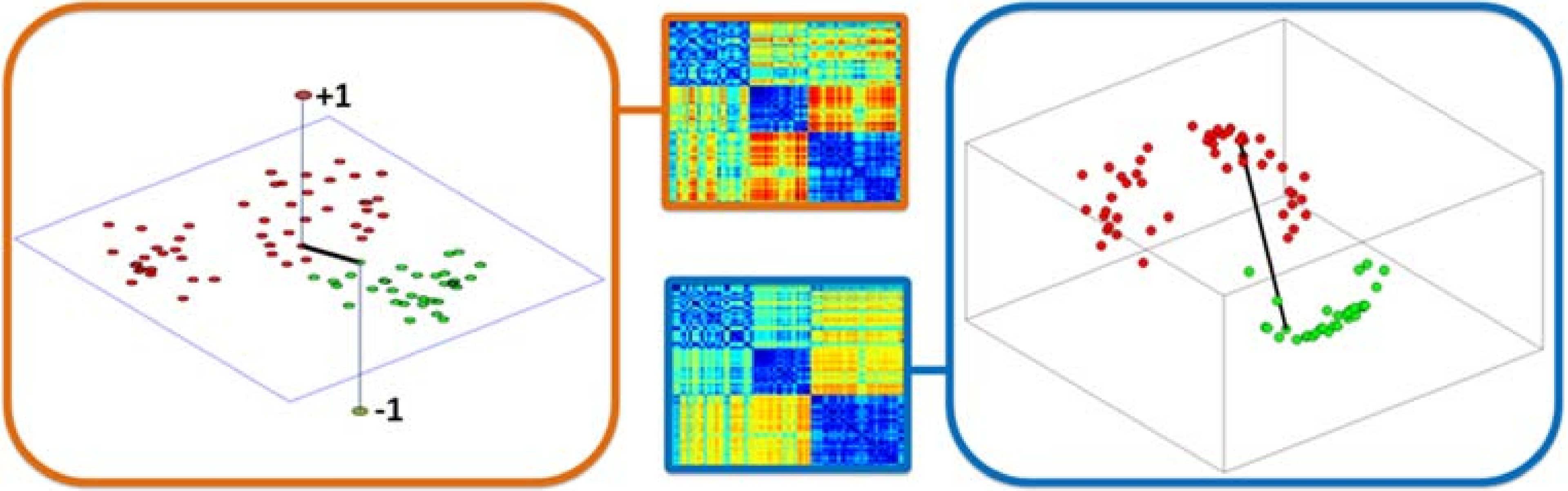
$$\tilde{D}_{i,j}^{p} = \hat{D}_{i,j}^{p} + \sum_{c \in [1,N]} (\alpha \cdot D_{i,j}^{(c)})^{p}$$

- Cluster data by modified distance matrix $\tilde{D}$

Constraints as Features

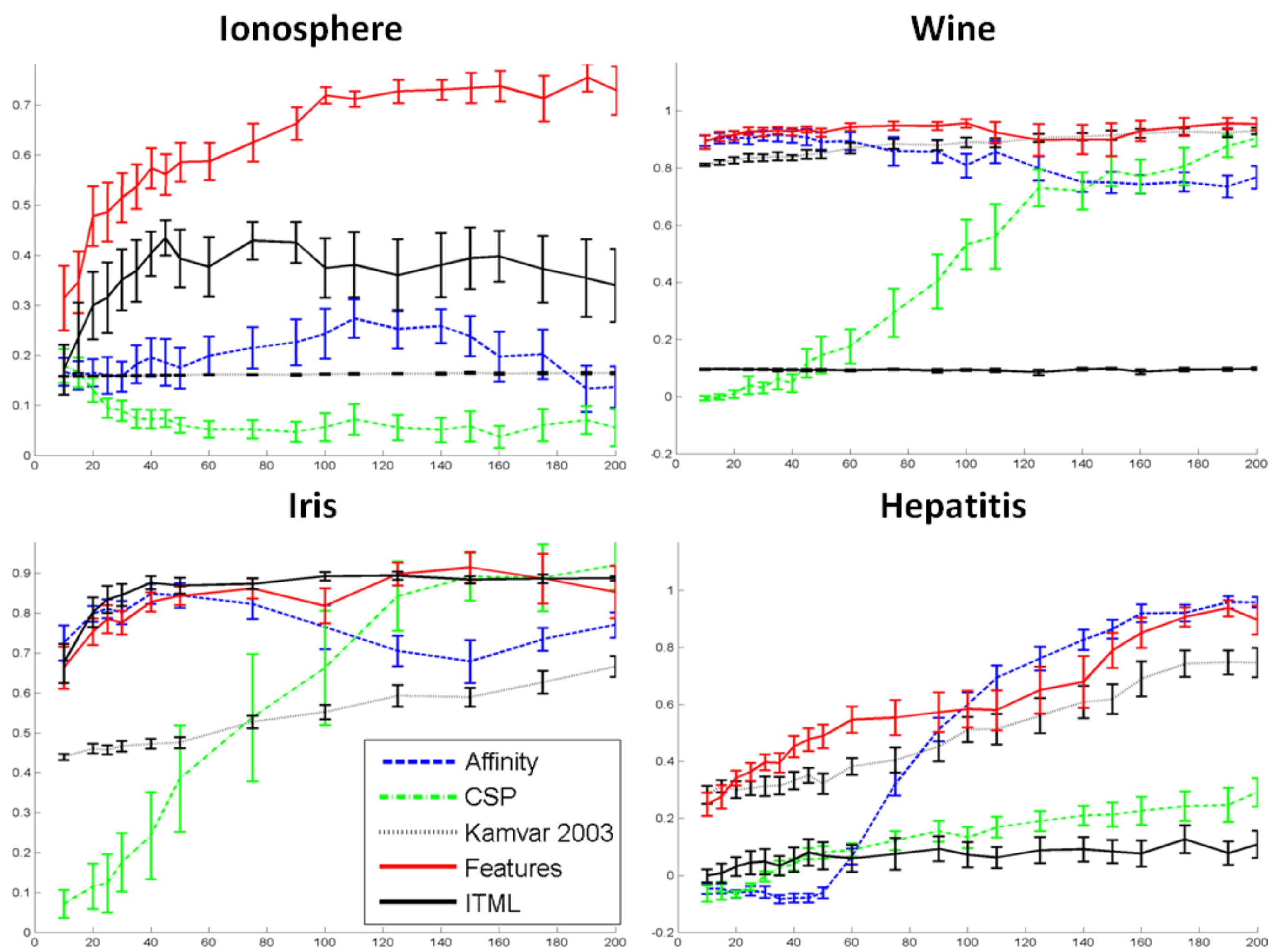


Constraints as Features!!!



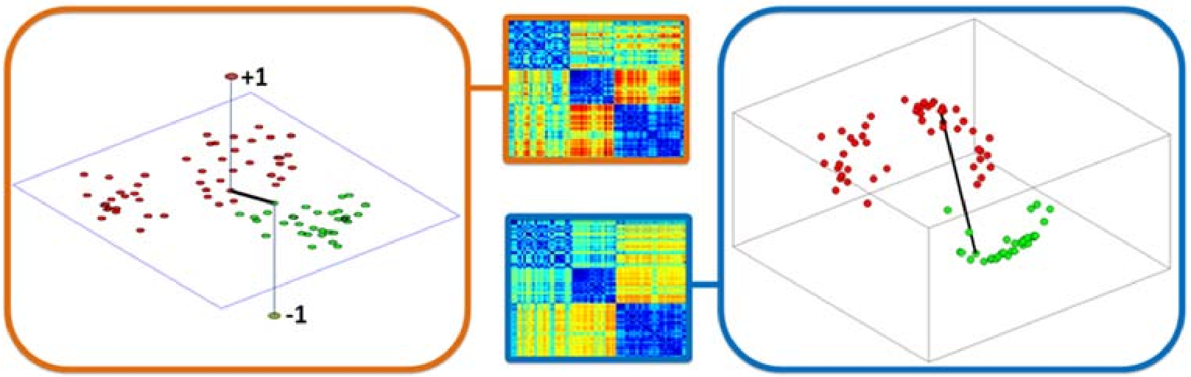
Unconstrained clustering of the modified data

Results – UCI (CVPR 2013)

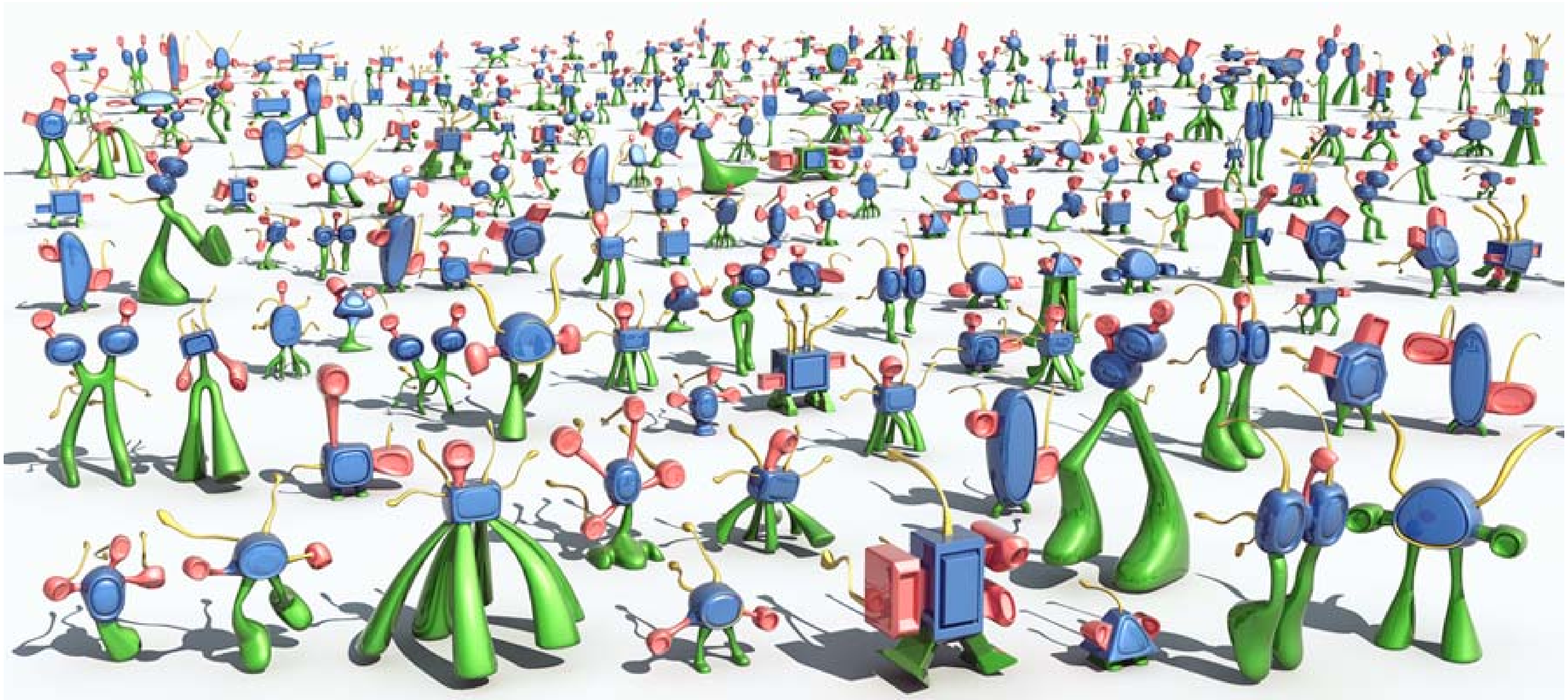


Summary

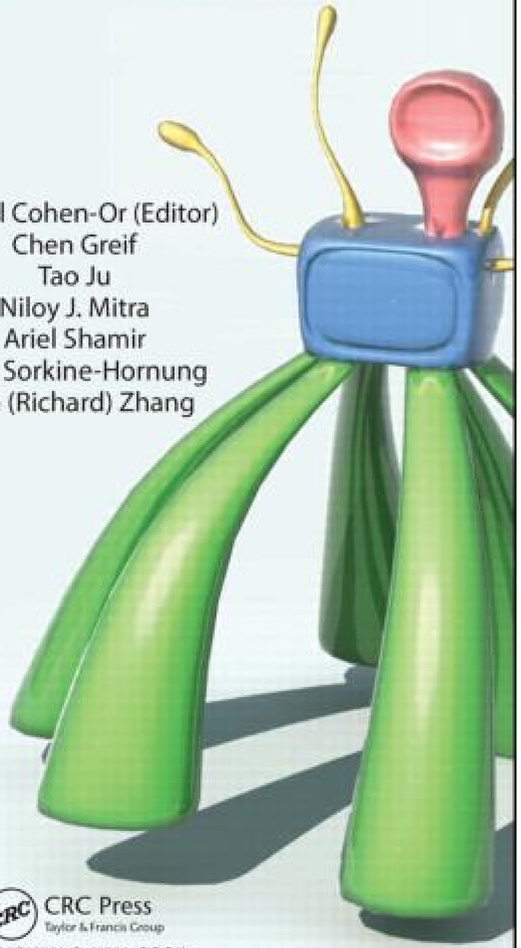
- A new semi-supervised clustering method.
- Constraints are embedded into the data, reducing the problem to an unconstrained setting.



Thank you!



A Sampler of Useful Computational Tools for Applied Geometry, Computer Graphics, and Image Processing

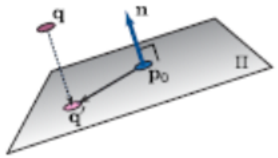


Daniel Cohen-Or (Editor)
Chen Greif
Tao Ju
Niloy J. Mitra
Ariel Shamir
Olga Sorkine-Hornung
Hao (Richard) Zhang

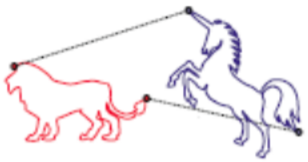
CRC Press
Taylor & Francis Group
A CHAPMAN & HALL BOOK

Book Contents

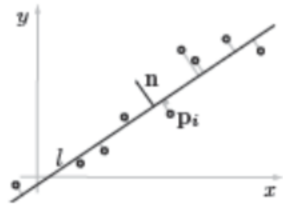
In the first chapter, we will familiarize ourselves with some basic geometric tools and see how we can put them to practical use to solve several geometric problems. Instead of describing the tools directly, we do it through an interesting discussion of two possible ways to approach the geometric problem at hand: we can employ our geometric intuition and use geometric reasoning, or we can directly formalize everything and employ our algebraic skills to write down and solve some equations. The discussion leads to a presentation of linear geometric elements (points, lines, planes), and the means to manipulate them in common geometric applications that we encounter, such as distances, transformations, projections and more.



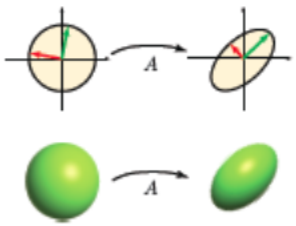
In this chapter, we will review basic linear algebra notions that we learned in a basic linear algebra course, including vector spaces, orthogonal bases, subspaces, eigenvalues and eigenvectors. However, our main goal here is to convince the readers that these notions are really useful. Furthermore, we will see the close relation between linear algebra and geometry. The chapter will be driven by an important tool called singular value decomposition (SVD), to which we will devote a separate full chapter. To understand what an SVD is, we first need to understand the notions of bases, eigenvectors, and eigenvalues and to refresh some fundamentals of linear algebra with examples in geometric context.



When dealing with real-world data, simple patterns can often be submerged in noise and outliers. In this chapter, we will learn about basic data fitting using the least-squares method, first starting with simple line fitting before moving on to fitting low-order polynomials. Beyond robustness to noise, we will also learn how to handle outliers and look at basic robust statics.

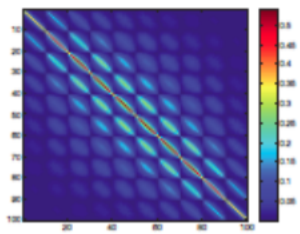


In this chapter, we introduce two related tools from linear algebra that have become true workhorses in countless areas of science: principal component analysis (PCA) and singular value decomposition (SVD). These tools are extremely useful in geometric modeling, computer vision, image processing, computer graphics, machine learning and many other applications. We will see how to decompose a matrix into several factors that are easy to analyze and reveal important properties of the matrix and hence the data, or the problem in which the matrix arises. As in the whole book, the presentation is rather light, emphasizing the main principles without excessive rigor.

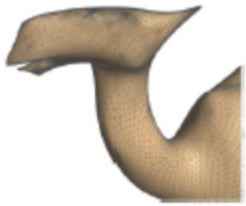


The use of signal transforms, such as the discrete Fourier or cosine transforms, is a classic topic in image and signal processing. In this chapter, we will learn how such transforms can be formulated and applied to the processing of 2D and 3D geometric shapes. The key concept to take away is the use of eigenvectors of discrete Laplacian operators as basis vectors to define spectral transforms for geometry. We will show how the Laplacian operators can be defined for 2D and 3D shapes, as well as a few applications of spectral transforms including geometry smoothing, enhancement and compression.

In the solution of problems discussed in this book, a frequent task that arises is the need to solve a linear system. Understanding the properties of the matrix associated with the linear system is critical for guaranteeing speed and accuracy of the solution procedure. In this chapter, we provide an overview of linear system solvers. We describe direct methods and iterative methods, and discuss important criteria for the selection of a solution method, such as sparsity and positive definiteness. Important notions such as pivoting and preconditioning are explained, and a recipe is provided that helps in determining which solver should be used.

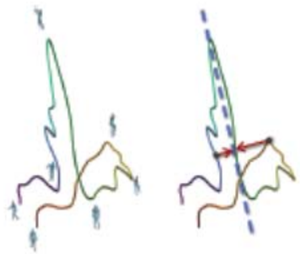


In this chapter, we make use of the well-known equations of Laplace and Poisson. The two equations have an extremely simple form, and they are very useful in many diverse branches of mathematical physics. However, in this chapter, we will interpret them in the context of image processing. We will show some interesting image editing and geometric problems and how they can be solved by simple means using these equations.

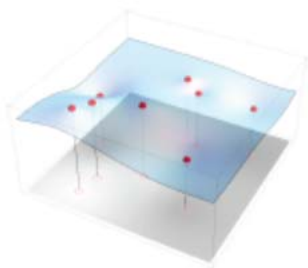


Local surface details, e.g., how "flat" a surface is locally, carry important information about the underlying object. Such information is critical for many applications in geometry processing, ranging from surface meshing, shape matching, surface reconstruction, scan alignment and detail-preserving deformation, to name only a few. In this chapter, we will cover the basics of differential geometry, particularly focusing on curvature estimates with some illustrative examples as an aid to geometry processing tasks.

In this chapter, we will learn the concept, usefulness, and execution of dimensionality reduction. Generally speaking, we will seek to reduce the dimensionality of a given data set, mapping high-dimensional data into a lower-dimensional space to facilitate visualization, processing, or inference. We will present and discuss only a sample of dimensionality reduction techniques and illustrate them using visually intuitive examples, including face recognition, surface flattening and pose normalization of 3D shapes.



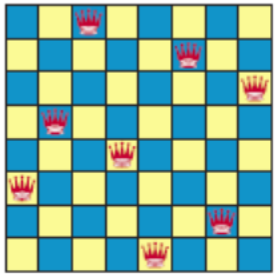
In this chapter, we visit the classical mathematical problem of obtaining a continuous function over a spatial domain from data at a few sample locations. The problem comes up in various geometric modeling scenarios, a good example of which is surface reconstruction. The chapter will eventually introduce the very useful radial basis functions (RBFs) as a smooth and efficient solution to the interpolation problem. However, to understand their usefulness, the chapter will go through a succession of methods with increasing sophistication, including piecewise linear interpolation and Shepherd's method.



In Chapter 8, we learned about local differential analysis of surfaces. In this chapter, we focus on global aspects. We will learn about what is meant by orientable surfaces or manifold surfaces. Most importantly, we will learn about the Euler characteristic, which links local curvature properties to global connectivity constraints, and comes up in a surprising range of applications.

Graphs play an important role in many computer science fields and are also extensively used in imaging and graphics. This chapter concentrates on image processing and demonstrates how images can be represented by a graph. This allows translating problems of analysis and manipulation of images to well-known graph algorithms. Specifically, we will show how segmentation of images can be solved using region-growing algorithms such as watershed or partitioning algorithms using graph cuts. We will also show how intelligently changing the size and aspect ratio of images and video can be solved using dynamic programming or graph cuts.

In this chapter, we will show an example of the usefulness of number theory, or at least one of its known theorems. We will discuss mappings of numbers to a lattice, a problem that has practical applications in systems that require simultaneous, conflict-free access to elements distributed in different memory modules. Such mappings are also called *skewing schemes* since they skew the trivial mapping from element to memory. To understand these mappings, we will visit the notions of relatively prime numbers, and the greatest common divisor (gcd).



Thank you!