Approximate 3D Partial Symmetry Detection Using Co-Occurrence Analysis

Chuan Li
University of Mainz
Institute of Computer Science
chuli@uni-mainz.de

Michael Wand
University of Mainz
Institute of Computer Science
wandm@uni-mainz.de

Xiaokun Wu
Max-Planck-Institut für Informatik
Computer Graphics Department
xwu@mpi-inf.mpg.de

Hans-Peter Seidel
Max-Planck-Institut für Informatik
Computer Graphics Department
hpseidel@mpi-inf.mpg.de

Abstract

This paper addresses approximate partial symmetry detection in 3D point clouds, a classical and foundational tool for analyzing geometry. We present a novel, fully unsupervised method that detects partial symmetry under significant geometric variability, and without constraints on the number and arrangement of instances. The core idea is a matching scheme that finds consistent co-occurrence patterns *in a frame-invariant way. We obtain a canonical partition of the input shape into building blocks and can handle ambiguous data by aggregating co-occurrence information across both all building block instances and the area they cover. We evaluate our method on several benchmark data sets and demonstrate its significant improvements in handling geometric variability, including scanning noise, irregular patterns, appearance variation and shape deformation.*

1. Introduction

Symmetry detection [22] is a fundamental tool for analyzing geometric shapes. In this paper we focus on partial symmetry, where non-trivial parts of a shape can be mapped to each other under admissible transformations such as rigid transformations [20, 10, 5]. Our objective is to recognize recurring parts, which we refer to as *building blocks*. We refer to the set of all mutually matching instances as one and the same *class* of building blocks, and permit general, irregular placements of instances.

Symmetry detection for noise-free geometry is well understood (see Section 2). Uncorrelated measurement noise and partial acquisition (as in 3D scans) have also been handled successfully in literature [20, 5, 23]. However, it remains very difficult to detect geometry that is similar in terms of semantics or functionality, but deviates substantially in geometry. For example, transformation voting [20, 24] is limited to coarse scale geometric structures due to the common voting space. Scenes with many instances can only be handled with assumptions such as regularity or hierarchy. As an alternative, feature-graph matching [3, 4] handles more complex transformations, but suffers from unreliable low level features, hence achieve limited success with strong shape variability. Spectral diffusion in matching networks [15] helps to cope with ambiguous data, but does not explicitly identify instances (geometric covering) and does not perform well with partial symmetry.

The objective of our paper is conceptually simple: We extract repetitive objects as a collection of consistently co-occurring features. The challenge is to make it work for 3D data with geometric variability in a fully unsupervised fashion. There are two key ingredients that make the detection work: invariant features and aggregation of features response using co-occurrence.

The necessity of having geometric invariance at the feature level is for handling appaerance variation. However, it comes at the price of reduced specificity and increased false positives. We therefore argue the necessity of aggregating information over larger quantities of data to improve the detection.This is our core contribution: We employ a novel EM algorithm that iteratively integrates signal over the mappings between instances (estimation) and over the area covered by each instance (maximization). In this way we consolidated the features with the true repetition, which significantly improved the detection.

In comparison to transformation voting [20, 24, 23] the new approach has no restriction in the cardinality and placement of the symmetric elements. Comparing to feature-graph matching [3, 4], it is able to handle much stronger geometric deformation. Comparing to symmetry factored

978-1-4673-8332-5/15 $31.00 © 2015 IEEE
DOI 10.1109/3DV.2015.55

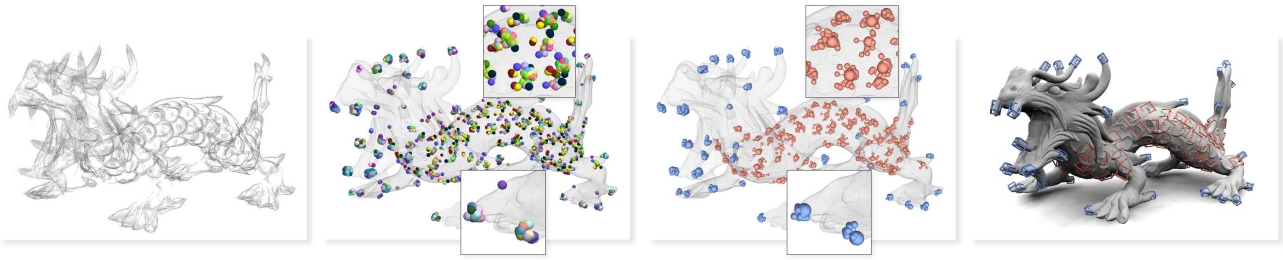


Figure 1. System overview: Input (preprocessed) — Feature Detection (distribution on model and close up look) — Building blocks detection (constellation visualization) — Detection results (bounding box visualization)

embedding [15], our spectral clustering embeds the distance between *spatial patterns* of matched features rather than single correspondences. We demonstrate that this comes with significant performance benefits. Unlike most of the previous methods, our approach optimizes for coherent constellations, therefore outputs building block instances of different classes rather than correspondences (which are often split into pieces using simple greedy region growing). In summary, our paper makes two main contributions:

- We introduce consistent co-occurrence patterns as a novel invariant for improving feature matching in 3D point clouds.
- Based on this, we develop a novel unsupervised partial symmetry detector, which outperforms know methods in case of strong geometric variability.

2. Related Work

Since its introduction to the community by a series of seminal papers [24, 20, 27, 10], a lot of researches have been devoted to symmetry detection and its applications. A recent survey is provided by Mitra et al. [22]. Here we briefly discuss three categories of related work.

Transformation voting collects pairs of matching features and casts votes for transformations that relate them [19, 24, 17]. Voting methods excel at recognizing approximate symmetry, as exemplified by symmetrization methods [21]. However, the use of a single transformation space (marginalizing out locality information) limits detection to coarse structures. Symmetries of fine details or complex patterns of many generally placed instances can not be recognized against structured background noise. The problem can be alleviated by a restrictive hierarchical decomposition [19] or by applying additional regularity priors [23]. Our method avoids marginalization of locality information. We demonstrate in several test scenes that it handles large quantities and arrangements of instances and recognizes symmetries in smaller details.

Feature-graph matching avoids the restrictions of the common transformation space [2, 5]. These methods are essentially brute-force alignment schemes that use feature matching for quick rejection of non-matching geometry. Final validation is obtained via ICP alignment [8]. However, the rigidity prohibits recognizing instances that differ by systematic deformation. Attempts have been made to handle deformable feature graphs [4]; however its performance were rather limited because the lack of an *efficient* way to validate if the conjectured high order patterns have significant support in the data. By contrast, we use simple pairwise relations to aggregate co-occurrence information while avoiding over-fitting of untrustworthy hypothesis.

Spectral clustering: Lipman et al. [15] use spectral embedding to recover the latent equivalence relation from noisy data. This method does not recognize symmetry in fine details as it relies on larger-scale and fixed-size matching (not optimized to instance-size) as its source of correspondence information. Spectral clustering is extended by Kim et al. [12] towards shape collections using diffusion in a sparse matching graph to avoid unreliable long-distance matches. Mattausch et al. [18] cluster geometric patches and incorporate co-occurrence to compensate for missing data. However, they do not aggregate signal from multiple instances, and the feature design is optimized for indoor architecture (planar patches). Kalojanov et al. [13] detect building blocks (named "microtiles"); however, their model cannot handle approximate matching.

Our approach is also related to Liu et al. [16] and Li et al. [14]. Both detect partial 2D symmetry using co-occurrence analysis. Our method is different from [16] in two major ways: First, feature dictionaries are not learned via direct unsupervised clustering but based on a more sophisticated EM optimization. Second, we use spectral clustering instead of greedy searching to link features into meaningful constellations. Li et al. [14] only consider translational building blocks from front-view, which permits using global cross-correlation for detection. This is clearly problematic in 3D, where co-occurrence must be detected in local frames that differ from each other rotationally. Further, we observe in practice that associating features with local frames introduces "rotational noise" that increases ambiguity. This renders the unsupervised discriminative learning scheme in [14] unreliable. We handle rotational noise by aggregating information from a whole constellation of

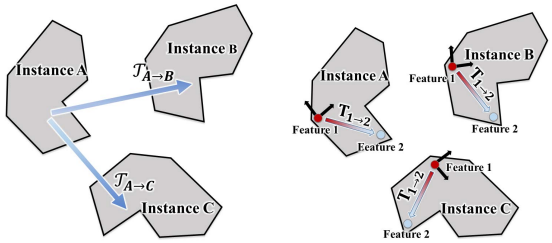


Figure 2. Left: $\mathcal{T}_{source \to target}$ maps instances of the same building block class. Right: $\mathbf{T}_{source \to target}$ maps co-occurring features in the same instance. The black arrows indicate the local frame of the source feature.

features in our EM scheme, utilizing the redundancy of the co-occurring features for improving robustness.

3. The Detection Method

Our method takes a 3D point cloud as input and detects its partial symmetry (Figure 1). We start by building a feature abstraction of high-curvature regions [5], with spatial and orientational pooling to gain invariance against local shape deformation. However, such invariance comes at the price of reduced specificity, i.e., increased false positives. We therefore need to integrate information over larger quantities of data to improve the signal-to-noise level. The idea is straight forward: we extract each building block instance as a collection of consistently co-occurring features. While matching individual features is unreliable, aggregating many feature matches improves the detection. Conceptually, two sources of information are useful: we can (i) integrate over the mappings between instances (Figure 2 left), and (ii) integrate over the volume covered by each instance (Figure 2 right). Knowing either the mappings or the instances simplifies the search of the other. However, in practice we know neither, so this is a chicken-and-egg dilemma.

To solve this problem, we employ an *Expectation-Maximization* scheme to find optimal co-occuring *constellations* of features. We first build a dictionary of relevant features by k-means clustering, pruned by a sliding-window detection. We then iterate between (i) the *expectation* step, which finds pairwise co-occurring features using current dictionary entries (words); and (ii) the *maximization* step, which updates the words so that the affinity between co-occurring features is maximized. The output of this EM process is a dictionary of reliable features (Figure 1, feature detection). Next, the location of the building blocks instances and the volume they cover are found by a spectral clustering algorithm (Figure 1, building block detection).

3.1. Notation

Here we define the notation of building blocks: Let $\mathbf{Q} = \{\mathbf{Q}^i\}_{i=1}^M$ denotes different classes of building blocks (different repetitive objects). Each class $\mathbf{Q}^i = \{\mathbf{q}_j^i\}_{j=1}^N$ consists of N instances that can be *approximately* mapped to each other under admissible transformations. It can be represented by transforming any of its j-th instance $\mathbf{Q}^i = \{\mathcal{T}_{j\to k}^i \cdot \mathbf{q}_j^i\}_{k=1}^N$ (Figure 2, left). Without loss of generality, we denote the set of transformations $\mathcal{T}^i = \{\mathcal{T}_{1\to k}^i\}_{k=1}^N$ with $\mathbf{q}_1^i$ as the template instance.

Building blocks consist of coherently co-occurring features: these features show up in the same constellation for all instances of the same building blocks class. Such a *consistent co-occurrence pattern* is an invariant of each building block class. Let $\mathbf{D} = \{\mathbf{D}^i\}_{i=1}^K$ denotes a feature dictionary, where each word $\mathbf{D}^i = \{\mathbf{d}_j^i\}_{j=1}^N$ is a list of matched features. A pair of features co-occur if their lists of matches, $\mathbf{D}^i$ and $\mathbf{D}^j$, are identical after applied the *same* local transformation $\mathbf{T}_{i\to j}$ to every member of $\mathbf{D}^i$ (Figure 2, right). That is, $\mathbf{D}^j = \{\mathbf{T}_{i\to j} \cdot \mathbf{d}^i\}_{i=1}^N$. Applying the mapping locally makes it frame invariant to different instances.

To detect building blocks, one need to identify co-occurring feature pairs across the data, then assemble the consistent pairs into instances. Doing so aggregates co-occurrence information over the set of transformations $\mathcal{T}^i$ and the target geometries $\mathbf{Q}^i$. Although we know neither the transformation nor the target geometry, we show that in practice this problem can be handled by an iterative Expectation-Maximization scheme.

3.2. Pre-processing

Our input data is a oriented point cloud. We re-sample the input point cloud using a sample space of ϵ, which controls the processing resolution and the scale of the detection (many parameters in this paper have a prefixed ratio to ϵ, see Section 4 for details). We sample points with strong (maximum principal) curvature as salient features. We denote this feature point cloud by Ω. We then set up a local frame for each feature for computing its descriptor.

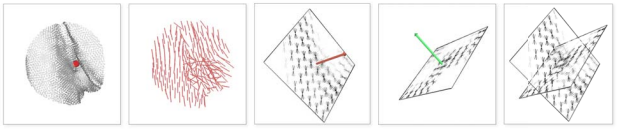


Figure 3. HOC descriptor. From left to right: local geometry, curvature, HOC projected to the tangent plane, HOC projected to a secondary plane, the final descriptor.

Our descriptor is a 3D variant of the Histogram of Oriented Gradient descriptor [9]: we use curvature as the analog of 2D-image gradients for 3D-surfaces. Such a *histogram of oriented curvature* descriptor (HOC) was first introduced by Kerber et al. [11]. Figure 3 shows an example of this descriptor, and Section 4 provides implementation details. Notice our core detection algorithm is independent of the descriptor design; so it is possible to use different types of features for different tasks [28, 25].

Next, we build a dictionary $\mathbf{D}$ of frequently appearing

features using K-means clustering. Each word (cluster) $\mathbf{D}^i = \{\mathbf{d}^i_j\}_{j=1}^N$ consists a list of similar features. We use the term "word" and "list of matches" interchangeably. The template $\mathbf{t}^i$ for each word has the smallest distance to all other members. We purify each word using a sliding window filter: features that are too close (less than half of the HOC descriptor) are excluded.

Figure 4 (left) shows the sliding window response of a single feature. The response map is imposed onto the original geometry. Note the resulting feature response is noisy (red arrow points at the query feature, and blue arrow points at a false positive). This is because the strong appearance variation of the reptile scales is beyond the invariance of a single HOC descriptor. Simply increasing the size of the descriptor won't help, since the feature response will be blurred. For this reason such a naive feature dictionary is not reliable for computing building blocks.

3.3. Robust Co-occurring Feature Detection

The core contribution of our paper is to use an iterative EM scheme to enforce pairwise co-occurrence constraints in the search of reliable features. Importantly, we use pairwise relation instead of higher order co-occurrence for avoiding over-fitting in the unsupervised setting. Our input is the initial feature dictionary $\mathbf{D}$. At each iteration, we first estimate the co-occurrence relation using exist words, then update the words so the affinity between the distributions of co-occurring features is maximized. A summary of our algorithm is listed in Algorithm 1.

Estimation. The aim is to estimate the probability of co-occurring features. Given the current feature dictionary, we first compute the best local transformation $\mathbf{T}_{i\to j}$ that maps $\mathbf{D}^i$ to $\mathbf{D}^j$, which essentially matches the spatial pattern of the two words. A successful match is found for each mapped feature $\mathbf{T}_{i\to j}\cdot\mathbf{d}^i_k$ that has a close-by neighboring feature in $\mathbf{D}^j$. We test all transformations that map between near-by feature pairs in $\langle\mathbf{D}^i,\mathbf{D}^j\rangle$, and choose $\mathbf{T}_{i\to j}$ as the one that gives the maximal number of matches. We denote $\Psi^{i,j}$ the set of the source features from the matches.

Next, we compute $\mathbf{p}(\mathbf{d}^i,\mathbf{d}^j)$, the probability density function (PDF) for a pair of features $\langle\mathbf{d}^i,\mathbf{d}^j\rangle$ to co-occur in Ω. We first use a voting based method to compute $\mathbf{p}(\mathbf{d}^j|\mathbf{d}^i)$: For each $\mathbf{x}\in\Omega$, we predict its counterpart using $\mathbf{T}_{i\to j}\cdot\mathbf{x}$, then add the prediction quality to the PDF. Specifically, we find all $\mathbf{y}\in\Omega$ that satisfy $||\mathbf{y}-\mathbf{T}_{i\to j}\cdot\mathbf{x}||<\sigma_s$, and vote for each of them with the following probability:

$$\mathbf{p}(\mathbf{d}^j{=}\mathbf{y}|\mathbf{d}^i{=}\mathbf{x}){=}\exp\left(-\frac{||\mathbf{y}-\mathbf{T}_{i\to j}\cdot\mathbf{x}||}{\sigma_s^2}m(\mathbf{t}^j,\mathbf{y})\right) \quad (1)$$

Here σ_s controls the tolerance of mis-match, and $m(\mathbf{t}^j,\mathbf{y})$ computes the descriptor distance between the template and the prediction. Finally we compute the joint probability: $\mathbf{p}(\mathbf{d}^i,\mathbf{d}^j)=\mathbf{p}(\mathbf{d}^j|\mathbf{d}^i)\mathbf{p}(\mathbf{d}^i)$. Here $\mathbf{p}(\mathbf{d}^i)$ is the prior of seeing $\mathbf{d}^i$ in Ω. It is approximated as the spatial distribution of the word:

$$\mathbf{p}(\mathbf{d}^i=\mathbf{x})\approx\begin{cases}1 & \text{if } \mathbf{x}\in\mathbf{D}^i,\\ 0 & \text{if } \mathbf{x}\notin\mathbf{D}^i.\end{cases} \quad (2)$$

Notice this approximation significantly reduces the computational cost, as we only need to compute $\mathbf{p}(\mathbf{d}^j|\mathbf{d}^i)$ for $\mathbf{x}\in\mathbf{D}^i$ instead of $\mathbf{x}\in\Omega$.

Algorithm 1 Detection of Pairwise Co-occurrence Features

Require: Input feature dictionary $\mathbf{D}$, size of dictionary K, number of iteration M

for $t\leftarrow 1$ to M **do**
  for $i\leftarrow 1$ to K **do**
    for $j\leftarrow 1$ to K **do Estimation**:
      $\mathbf{T}_{i\to j}$ = Match$(\mathbf{D}^i_t,\mathbf{D}^j_t)$
    for $j\leftarrow 1$ to K **do**
      $\mathbf{p}(\mathbf{d}^j_t|\mathbf{d}^i_t)$ = Vote$(\mathbf{t}^j_t,\mathbf{T}_{i\to j})$
      $\mathbf{p}(\mathbf{d}^i_t,\mathbf{d}^j_t)=\mathbf{p}(\mathbf{d}^j_t|\mathbf{d}^i_t)\mathbf{p}(\mathbf{d}^i_t)$
  for $i\leftarrow 1$ to K **do Maximization**:
    $\mathbf{p}(\mathbf{d}^i_{t+1})=\sum_{j=1,j\neq i}^K\mathbf{p}(\mathbf{d}^i_t,\mathbf{d}^j_t)$
    $\mathbf{d}^i_{t+1}$ = FindPeak$(\mathbf{p}(\mathbf{d}^i_t))$
    $\mathbf{t}^i_{t+1}$ = UpdateTemplate$(\mathbf{d}^i_{t+1})$

Maximization. In this step we maximize the co-occurrence in the detection by updating words. To do so we recompute $\mathbf{p}(\mathbf{d}_i)$ as the integration of all hypothetical pairwise outcomes: $\mathbf{p}(\mathbf{d}^i)=\sum_{j=1,j\neq i}^K\mathbf{p}(\mathbf{d}^i,\mathbf{d}^j)$, where K is the size of the dictionary. We then update $\mathbf{D}^i$ as a new set of features detected as the local peaks in $\mathbf{p}(\mathbf{d}_i)$. Again, non-maximum suppression is used for detecting the peaks. Finally, we update the template $\mathbf{t}^i$ using the new median feature of the word. The output of this EM process is a set of more reliable features. Figure 4 compares the feature response before and after the EM process: one iteration (Figure 4 middle) already improves the feature response, and the result is further improved after five iterations (Figure 4 right).

3.4. Instance Detection

Having found reliable features, our next task is to detect building blocks instances. From a global perspective, we identify building blocks using spectral clustering: we first perform spectral embedding using co-occurrence measurement of pairwise features, then use unsupervised clustering to extract different classes of building blocks and their supporting regions. Figure 5 shows an example of the process.

We define a co-occurrence matrix $\mathcal{M}$, where each entry is the affinity between the spatial distributions of a pair of words ($|\cdot|$ denotes cardinality):

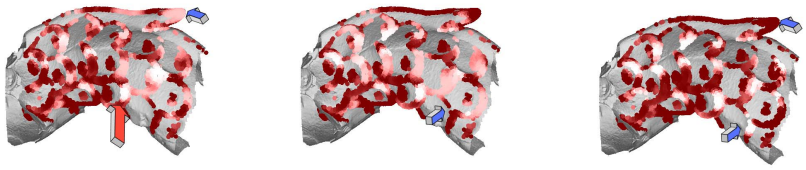


Figure 4. Feature matching can be improved by the proposed EM algorithm. The heat map encodes the probability of finding a match (white indicates high probability). Left: single feature based detection (red arrow points at the query feature) is unreliable. For example there is false positives (blue arrow). Middle: The output after one iteration. The false positive is removed. But missing detection still exists (blue arrow). Right: The output after five iteration.

Figure 5. Left: Co-occurrence features in the embedded space. Right: The median of each cluster (blue and red dot) is imposed onto the input model.

$$m_{i,j} = \frac{|\Psi^{i,j}|}{\max(|\mathbf{D}_i|, |\mathbf{D}_j|)} \tag{3}$$

From $\mathcal{M}$ we create a low dimensional embedding using classical multidimensional scaling, where co-occurring features are close to each other. We extract modes from this embedding using mean-shift. This results in M clusters ($M \leq K$), each represents a unique building block class $\mathbf{Q}_i, i = 1...M$. We use the median of each cluster $\overline{\mathbf{Q}}_i$ as an outlier-robust representation of the mode (Figure 5 right).

Finally, we determine the supporting region of each instance. We build a "star model", which uses $\overline{\mathbf{Q}}_i$ as the centers of the instances, and links co-occurring features to the centers. To do so, we add an edge for each matched pair of features from $\overline{\mathbf{Q}}_i$ and $\mathbf{D}_j$, where $\mathbf{D}_j \in \mathbf{Q}_i \backslash \overline{\mathbf{Q}}_i$. The size of each instance is approximated by the bounding box of its internal linked features. Figure 8 show examples of our detection. The bounding boxes of all instances from the same class are averaged to get a more robust estimation.

4. Implementation Details

Here we describe implementation details of our algorithm. We re-sample the input point cloud using a sample space of ϵ, which controls the processing resolution and consequently the scale of the detection. In practice, we use different ϵ for data from different sources. It is set as an absolute value (0.075) for the *Qmulus&TerraMobilita* city scan dataset, and relatively (0.0005 of diagonal length of the input shape) for the Stanford 3D scanning repository and the Aimatshape repository. We estimate surface curvature by applying quadratic moving-least-square [1] to local neighborhoods of radius 4ϵ and estimating the curvature tensor [7], from which we extract the direction of maximum principal curvature $\mathbf{t}_1$ and its magnitude κ_1.

Feature point extraction. We sample points with strong curvature as salient features: A non-maximum suppression process is performed with a local search radius of 2ϵ. We use a threshold on the magnitude of the maximum principle curvature $\kappa_1 > 0.15$ to preclude specious local maxima.

Local feature frame. To set up a local frame for each feature, we use the normal of the query feature as the local X axis, and the maximum principal curvature as the local Y axis. The local Z axis is the cross-product of the two. To resolve the sign ambiguity of the maximum principal curvature, we set up two different local frames as one is the 180 degree rotation of the other.

Descriptor. We computed the curvature statistics around the query feature: each nearby sample point is projected onto the tangent plane. We collect the projected curvature by 8 orientation bins and 8×8 spatial bins, where each spatial bin has an edge length of 8ϵ. We also project curvature onto a second plane that uses local Y axis as the normal. Such bi-directional projection increases the discriminative performance with just a modest increase of computational cost. In total our descriptor has 1024 dimensions. We normalize the descriptor so its L1 norm is one.

Feature Dictionary. The initial dictionary is built using K-means clustering(K set to 100). During sliding window detection, we compute the matching score from the L1 distance between two descriptors: $m(\mathbf{x}, \mathbf{y}) = \exp\left(\frac{-|HOC(\mathbf{x}) - HOC(\mathbf{y})|}{\sigma^2}\right)$. Here $\sigma = 0.1$ is a standard deviation parameter that models matching noise. The same matching function is used in Equation 1. Matches with score below 0.5 are removed during the non-maximum suppression to avoid spurious local maxima. The threshold is intentionally set to a relative low value to permit high noise scenarios and significant geometric variability.

Co-occurring Feature Detection. While matching the distribution of two features, we restrict the maximum spatial distance between the prediction $\mathbf{T}_{i \to j} \cdot \mathbf{d}_k^i$ and its nearest neighboring feature in $\mathbf{D}^j$ to be 8ϵ (one HOC cell). The matching process can be speed up by precluding features that are too far away in the input data: we require the spatial distance between $\mathbf{d}^i$ and $\mathbf{d}^j$ to be no larger than 64ϵ. However, it is still possible to detect instance that are larger than this size, due to the diffusion power of spectral clustering. The σ_s in Equation 1 is set to 16ϵ. In the maximization

step, we need to ensure that only reliable pairwise relations are kept in the EM process: First, we discard weakly co-occurring feature pairs via a threshold on the cardinality of $\Psi^{i,j}$. We let this threshold depend on the number of building blocks expected from the shape: 5 is used for individual shapes such as the Stanford dragon and 10 is used for city scans. Second, we filter out weak feature responses during the updating of words (maximization step). In practice we normalize $\mathbf{p}(\mathbf{d}_j)$ so the highest scored feature has confidence of 1, and weak detections are excluded with confidence below 0.25. For efficiency, we use up to 5 iterations for EM, and terminate the process as long as the dictionary does not have significant change (number of total features change is less than 1%).

Spectral Clustering. We perform a 5-dimensional embedding for the affinity matrix $\mathcal{M}$ using the classical multidimensional scaling [26]. For unsupervised clustering, we use mean-shift with bandwidth of 0.5.

In practice we observe small building block classes (with fewer instances) can drift towards bigger building classes due to the template update in Algorithm 1. To solve this problem, we use an incremental detection scheme based on a minimal description length (MDL) strategy. The task is to find the minimal number of building blocks classes that can describe most of the input data. To do so we run a sequence of building block detections. For each round, only the largest building block class in the result is kept. We remove features that are covered by this class from the feature point cloud Ω, so they would have no influence in the future detection. In this way, the input model is iteratively decomposed into building blocks of cardinality from big to small. The result in Figure 6 is produced in this way. In total we found five different building blocks that can explain the input model in an efficient way.

We implemented our detection algorithm in C++, and run with a 2.5Ghz quad-core Intel Core-i7 processor. The biggest computation bottle neck is the iterative EM process, and more complex symmetry leads to words of larger cardinality, hence higher computational cost. In practice our implementation takes less than a minute to process each model in Figure 8, 62 seconds to process "dragon" and 132 seconds to process the model in Figure 6.

5. Evaluation

We conduct a quantitative evaluation on the *Qmulus&TerraMobilita* city scan dataset [6] and on a set of non-man-made shapes from the Stanford 3D scanning repository and the Aimatshape repository. These data contains scan noise, irregular structure and/or strong geometric variability. We compare our method (EM + SC) with rigid ICP (RigidICP), symmetry factor embedding (Symmetry Embedding) [15], and our implementation of [16] in 3D (Grasp). To show the importance of the proposed EM scheme, we also compare to spectral clustering without the EM optimization (SC), and to spectral clustering using [14]'s discriminatively learned features (DL + SC). We provide precision-recall curves for both datasets (Figure 9 a-b). Like [16][14], we conduct a pressure test by varying the minimum overlapping ratio for instance matching and report the results in Figure 9 c-d.

5.1. Dataset

Previous work on partial symmetry detection mainly use qualitative evaluation as very few dataset provides ground truth annotations. We use our own iterative tool to create ground truth annotation: a user first paint a piece of geometry as the template for query, then find other instances using rigid ICP. Since rigid ICP does not work with strong geometric variability, the user often first over-decompose the data into rigid pieces and then merge them by semantics. For example, windows of different sizes and styles will first be labeled as different classes, then merged into a single "window" class. We uniformly partition the entire *Qmulus&TerraMobilita* dataset into 96 scenes, and perform annotation for each scene. There are three common building block classes across the entire dataset: window, car and balcony. Other smaller classes represents unique street furniture and building decorations. For the non-man-made shapes we annotate individual model, and the classes vary from model to model: for example the model "Dragon" has classes "scale" and "craws"; and the model "Buddha" has a single class for the decorative pattern on the cloth.

5.2. Methodology

We quantify the results using the standard precision and recall (PR) analysis and an additional stress test described in [14]. Like [28] we match 3D bounding boxes of the instances from the detection and the ground truth annotation. Let $|\mathbf{D}|$ and $|\mathbf{G}|$ denote the number of different building block classes detected/annotated. We define a symmetric matching score between two classes $\mathbf{D}^j \in \mathbf{D}$ and $\mathbf{G}^i \in \mathbf{G}$:

$$\text{score}(\mathbf{D}^j, \mathbf{G}^i) = \min(|\mathbf{D}^j \cap \mathbf{G}^i|, |\mathbf{G}^i \cap \mathbf{D}^j|) \qquad (4)$$

where $\mathbf{X} \cap \mathbf{Y}$ denotes the set of bounding boxes in $\mathbf{D}^j$ that overlap with $\mathbf{G}^i$. Taking the minimum makes the score symmetric and avoids over-counting objects that have multiple intersections. This matching score can be computed for all possible assignments between the detected classes and the annotated classes. We define precision as the number of matched instances in $\mathbf{D}$ divided by the total number of instances in $\mathbf{D}$:

$$\text{precision} = \frac{\sum_{j=1}^{|\mathbf{D}|} \max_{i=1..|\mathbf{G}|}(\text{score}(\mathbf{D}^j, \mathbf{G}^i))}{\sum_{j=1}^{|\mathbf{D}|} |\mathbf{D}^j|} \qquad (5)$$

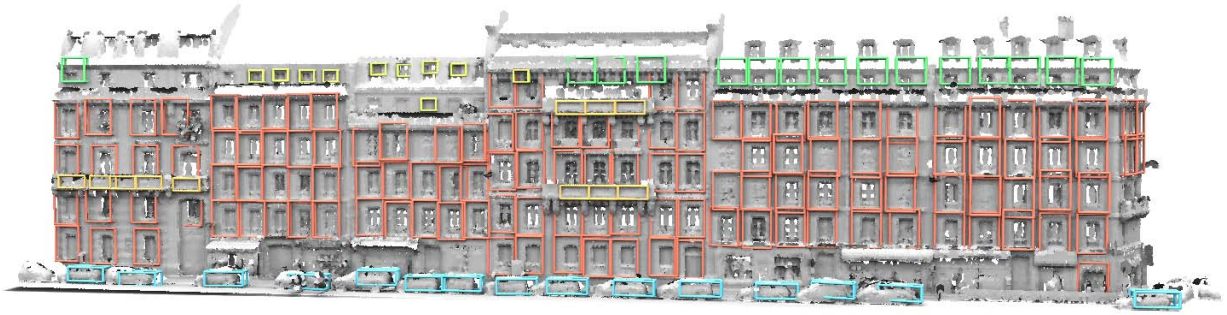


Figure 6. Here we find windows (red) and cars (blue) in the initial detection. We also find balconies (dark yellow) and two classes of smaller windows (with and without the dormer) as the green and the bright yellow building blocks in the late MDL iteration.

Notice each class $\mathbf{G}^i$ has its own matching score for $\mathbf{D}^j$, and only the highest score, $\max_{i=1..|\mathbf{G}|}(\text{score}(\mathbf{D}^j, \mathbf{G}^i))$, is kept for $\mathbf{D}^j$. We compute recall in a symmetric way by swapping the $\mathbf{D}$ and $\mathbf{G}$ terms in Equation 5. Like [14], we generate a PR curve by successively removing building block classes from the detection and simultaneously updating the recall and precision. Starting from the largest class, we remove one class at a time until no class remains. The final PR curve in Figure 9 is the average of all scenes.

We perform a stress test by incorporating a threshold of minimum overlapping ratio in Equation 4. Figure 9 a-b are generated using threshold 0.125, meaning there is on average half overlapping for each bounding box dimension. Increasing the threshold from zero to 0.5 creates the F-measurement v.s. pressure curves as shown in Figure 9 c-d.

5.3. Results

We first dicuss the PR curves of different algorithms (Figure 9 a-b). The baseline is a supervised detection using rigid ICP (purple). For each class in the ground truth, we perform a sliding window detection using the template from user annotation. The resulting low F-scores (0.55 and 0.48 on different datasets) indicate strong geometric variability can not be captured by rigid matching.

Symmetry factored embedding (pink) [15] computes the symmetry factored distance between the input data and its transformed copy. We use the symmetry factored distance to embed features and apply spectral clustering (with 10 clusters). Doing so produces an feature point cloud segmentation. We "cut out" individual instance using a 26-connected 3D region growing. This also produces very low F-scores (0.48 and 0.45 on different datasets), indicating local diffusion can fail in detecting complex partial symmetry.

Next we test our 3D implementation of [16] (Orange), which uses stochastic search to find correlated features. We implement their search algorithm with our 3D HOC features, and observe a significant improvement (F-score 0.63 and 0.60) over the baseline. However, there are still many false detections. The reasons are three folds: First, unsupervised clustering usually produces inaccurate low level features; second, inaccurate features can not be traced due to the lack of iterative refinement; third, their search algorithm is greedy and the use of *average affinity* as the optimizing objective biases the result towards false negatives.

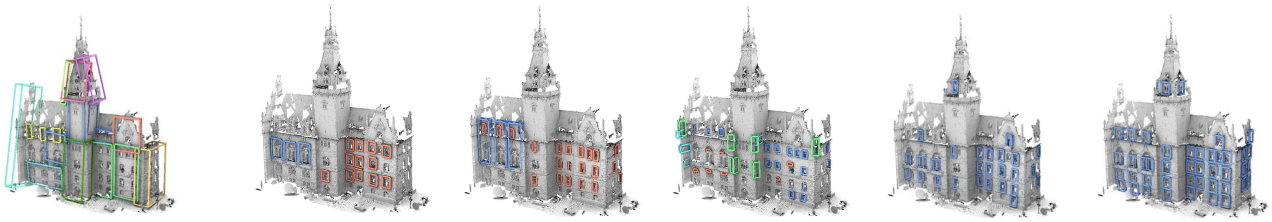
In contrast, our algorithm with EM optimization and less greedy spectral clustering (red) is able to significantly improve the performance. It achieves F-score of 0.74 on the Qmulus&TerraMobilita dataset and 0.68 on the non-man-made dataset. We also test intermediate results of our algorithm by dropping the EM optimization (SC, blue). Doing so significantly drops the performance (0.58 and 0.54). This indicates spectral clustering indeed requires reliable input features. We also test the discriminative learning scheme from [14] (DL + SC, green), which trains a one-vs-all linear SVM for each feature. However, in our experiment this only gives marginal improvement over [16], due to the additional frame noise for computing the feature descriptors. Figure 7 show a qualitative comparison between different methods.

The same conclusion can be seen from the pressure tests (Figure 9 c-d). This again proofs aggregating consistent co-occurrence information from data improves the signal-to-noise level in the feature matching, and consequently benefits the final detection of building blocks.

Figure 8 shows our detection handles different challenges. Figure 8-a shows it handles scanning noise; Figure 8-b shows it detects irregularly placed instances. Figure 8-c is a very challenge case, where the windows have very strong geometry variability. Nonetheless, our algorithm is able to identify most of them as the same class of building blocks. Figure 8 d-f are examples of our detection on non-man-made shapes.

6. Limitation and Future work

Our method could be improved at both the feature level and the structural level. Figure 10-a shows missing detection due to strong deformation that is beyond the invariance of HOC features. Figure 10-b shows the lack of curvature in the data leads to non-discriminative features. In this case our method failed to separate building blocks from the background noise. Using the entire shape for symmetry detec-



a) Symmetry Embedding b) Rigid ICP c) Grasp d) SC e) DL + SC f) EM + SC

Figure 7. Detection results of different methods.

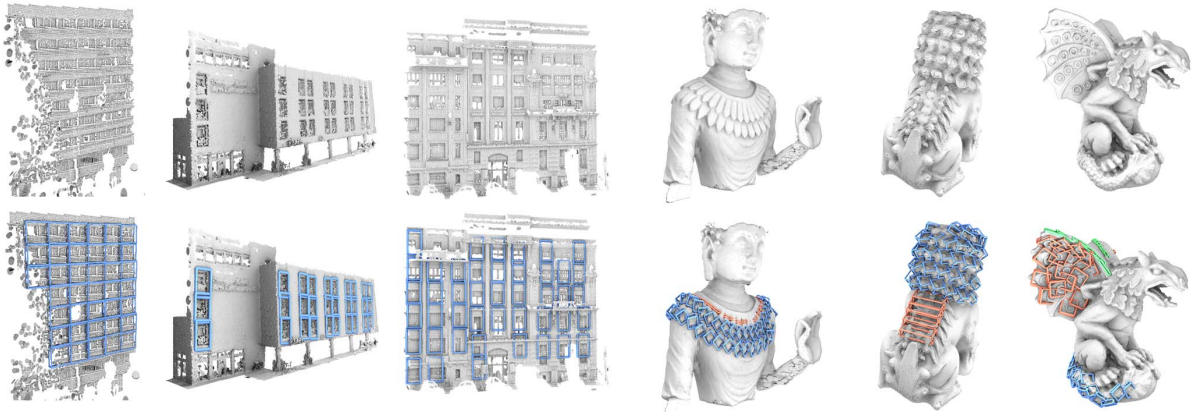
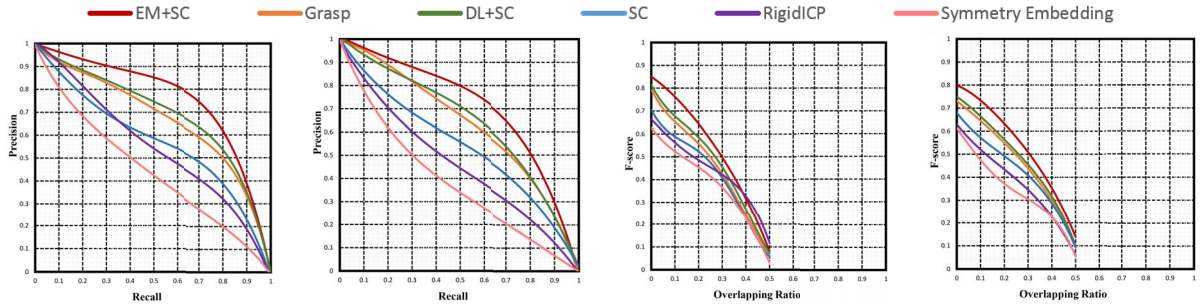


Figure 8. Detection results of different challenges.



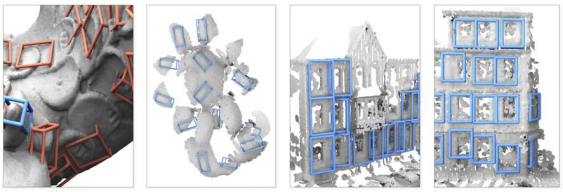
a) Qmulus&TerraMobilita b) Non-man-made shapes c) Qmulus&TerraMobilita d) Non-man-made shapes

Figure 9. Quantitative evaluation of our method. a) and b) are precision-recall curves on different datasets. c) and d) are pressure tests with different settings of overlapping ratio.

tion works better here as shown in [15].

On the structural level, our method detects general re-occurrence patterns instead of any specific forms of symmetry, such as reflective symmetry or lattice structure. This has been proved to be flexible while, to our best knowledge, none of the other specific methods works as general as ours. However, due to the lack of high level constraints, our method is less accurate when the noise level is too high: Figure 10-c shows miss-detections due to strong variance of data sampling density, and Figure 10-d shows mis-aligned building blocks. In these cases, reflective symmetry or lattice structure can be used to improve the results.

Last but not the least, our current implementation is not optimized for directly running on a big dataset. There are two main bottle necks: First, it does not scale well with the size of the scene due to the heavily used pairwise matching. Second, as the complexity of the scene increases, the matching of feature distributions also needs to be performed in a higher dimensional space. The consequence is that co-occurrence signal becomes more difficult to detect due to the curse of dimensionality. For example, small classes may drift towards big classes. In this paper we use a MDL based approach to handle such problem for partitions of big data. In the future we will investigate the scalability of our method in a more principled way.



a) Deformation b) Weak Feature c) Reflection d) Lattice

Figure 10. Limitations of our methods.

7. Conclusion

We have presented a new method for a classical problem – partial symmetry detection. The main contribution is to use consistent co-occurrence patterns as a novel invariant for improving feature matching, and based on it a novel detection algorithm that achieves robustness against ambiguous data by aggregating co-occurrence information across all building block instances and the volume they cover. Experiments show that our detection outperforms previous methods on data with strong geometric variability and irregular instance distributions. Our approach suggests a number of future work for improvement at both the feature level and the structural level. It also opens new application possibilities, for example structural aware shape synthesis.

Acknowledgments

This work has been partially supported by the Intel Visual Computing Institute and by the International Max Planck Research School for Computer Science.

References

[1] M. Alexa, J. Beht, D. Cohen-Or, S. Fleishman, D. Levin, and C. T. Silva. Computing and rendering point set surfaces. *TVCG*, 9(1):3–15, 2003.

[2] A. Berner, M. Bokeloh, M. Wand, A. Schilling, and H.-P. Seidel. A graph-based approach to symmetry detection. In *Symposium on Point-Based Graphics*, pages 1–8, 2008.

[3] A. Berner, M. Bokeloh, M. Wand, A. Schilling, and H.-P. Seidel. Generalized intrinsic symmetry detection. Research Report MPI-I-2009-4-005, Max-Planck-Institut für Informatik, 2009.

[4] A. Berner, M. Wand, N. J. Mitra, D. Mewes, and H.-P. Seidel. Shape analysis with subspace symmetries. *Computer Graphics Forum (Proc. Eurographics)*, 30(2):277–286, 2011.

[5] M. Bokeloh, A. Berner, H. p. Seidel, and A. Schilling. Symmetry detection using feature lines. *Computer Graphics Forum (Proc. Eurographics)*, 28:697–706, 2009.

[6] M. Brédif, B. Vallet, A. Serna, B. Marcotegui, and N. Paparoditis. Terramobilita/iqmulus urban point cloud analysis benchmark. In *Computers & Graphics*, to appear.

[7] F. Cazals and M. Pouget. Estimating differential quantities using polynomial fitting of osculating jets. *Computer Aided Geometric Design*, 22(2):121 – 146, 2005.

[8] Y. Chen and G. Medioni. Object modelling by registration of multiple range images. *Image Vision Comput.*, 10(3):145–155, 1992.

[9] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *CVPR*, pages 886–893, 2005.

[10] R. Gal and D. Cohen-Or. Salient geometric features for partial shape matching and similarity. *ACM Trans. on Graphics*, 25(1):130–150, 2006.

[11] J. Kerber, M. Bokeloh, M. Wand, and H.-P. Seidel. Scalable symmetry detection for urban scenes. *CGF*, 32:3–15, 2013.

[12] V. G. Kim, W. Li, N. J. Mitra, S. DiVerdi, and T. Funkhouser. Exploring collections of 3d models using fuzzy correspondences. *ACM Trans. Graph.*, 31(4):54:1–54:11, 2012.

[13] J. Kolojanov, M. Bokeloh, M. Wand, L. Guibas, P. Slusallek, and H.-P. Seidel. Microtiles: Extracting building blocks from correspondences. In *Symp. Geometry Proc. (SGP)*, 2012.

[14] C. Li and M. Wand. Approximate translational building blocks for image decomposition and synthesis. In *ACM Trans. Graph.*, to appear.

[15] Y. Lipman, X. Chen, I. Daubechies, and T. Funkhouser. Symmetry factored embedding and distance. *ACM Trans. Graph. (Proc. Siggraph)*, 29(4):103:1–103:12, 2010.

[16] J. Liu and Y. Liu. Grasp recurring patterns from a single view. In *CVPR*, pages 2003–2010, 2013.

[17] G. Loy and J.-O. Eklundh. Detecting symmetry and symmetric constellations of features. In *Proc. European Conf. Comp. Vision ECCV*, pages 508–521, 2006.

[18] O. Mattausch, D. Panozzo, C. Mura, O. Sorkine-Hornung, and R. Pajarola. Object detection and classification from large-scale cluttered indoor scans. *Computer Graphics Forum*, 33(2):11–21, 2014.

[19] N. J. Mitra, L. Guibas, and M. Pauly. Partial and approximate symmetry detection for 3d geometry. *Proc. of SIGGRAPH*, 25(3):560–568, 2006.

[20] N. J. Mitra, L. J. Guibas, and M. Pauly. Partial and approximate symmetry detection for 3d geometry. In *ACM Trans. Graph. (Proc. Siggraph)*, volume 25, pages 560–568, 2006.

[21] N. J. Mitra, L. J. Guibas, and M. Pauly. Symmetrization. *Proc. of SIGGRAPH*, 26(3):63:1–63:8, 2007.

[22] N. J. Mitra, M. Pauly, M. Wand, and D. Ceylan. Symmetry in 3d geometry: Extraction and applications. *Computer Graphics Forum*, 32(6):1–23, 2013.

[23] M. Pauly, N. J. Mitra, J. Wallner, H. Pottmann, and L. Guibas. Discovering structural regularity in 3D geometry. *Proc. of SIGGRAPH*, 27(3):43:1–43:11, 2008.

[24] J. Podolak, P. Shilane, A. Golovinskiy, S. Rusinkiewicz, and T. Funkhouser. A planar-reflective symmetry transform for 3D shapes. *Proc. of SIGGRAPH*, 25(3):549–559, 2006.

[25] H. Rahmani, A. Mahmood, D. Q. Huynh, and A. Mian. Hopc: Histogram of oriented principal components of 3d pointclouds for action recognition. In *ECCV*, 2014.

[26] G. A. Seber. *Multivariate Observations*. Wiley, 1984.

[27] P. Simari, E. Kalogerakis, and K. Singh. Folding meshes: Hierarchical mesh segmentation based on planar symmetry. In *Symp. Geometry Proc. (SGP)*, pages 111–119, 2006.

[28] S. Song and J. Xiao. Sliding shapes for 3d object detection in depth images. In *ECCV*, 2014.