EUROGRAPHICS 2018 / D. Gutierrez and A. Sheffer
(Guest Editors)

Volume 37 (2018), Number 2

Feature Curve Co-Completion in Noisy Data

Anne Gehre Isaak Lim Leif Kobbelt

Visual Computing Institute, RWTH Aachen University

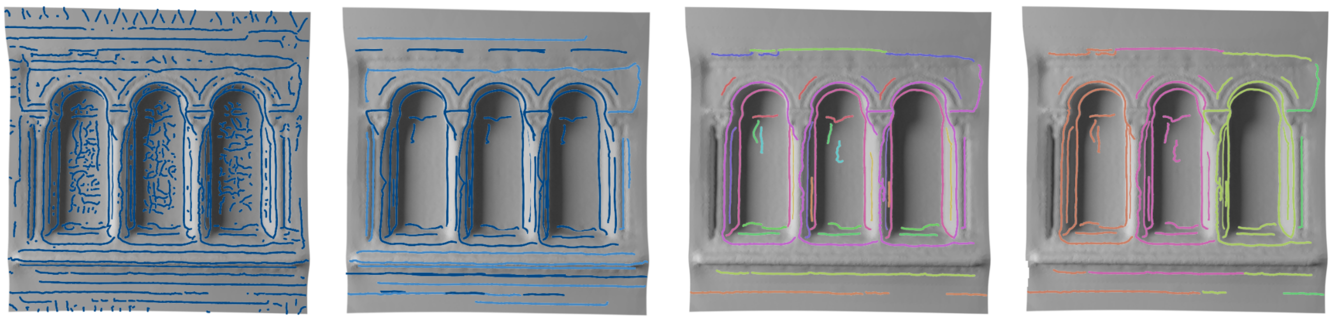


Figure 1: *Our method takes a noisy feature curve network (FCN) as input and produces a completed and denoised output FCN. This is done by performing co-occurrence analysis on the feature curves with which we identify corresponding curves and omit noise. Reoccurring groups are merged into a template, which is used for co-completion of the network. In the completed network, dark blue curves indicate the reoccurring templates, while strong (non-reoccurring) curves are visualized in light blue. The example shown above is computed on an excerpt of a scan of a gothic cathedral.*

Abstract

Feature curves on 3D shapes provide important hints about significant parts of the geometry and reveal their underlying structure. However, when we process real world data, automatically detected feature curves are affected by measurement uncertainty, missing data, and sampling resolution, leading to noisy, fragmented, and incomplete feature curve networks. These artifacts make further processing unreliable. In this paper we analyze the global co-occurrence information in noisy feature curve networks to fill in missing data and suppress weakly supported feature curves. For this we propose an unsupervised approach to find meaningful structure within the incomplete data by detecting multiple occurrences of feature curve configurations (co-occurrence analysis). We cluster and merge these into feature curve templates, which we leverage to identify strongly supported feature curve segments as well as to complete missing data in the feature curve network. In the presence of significant noise, previous approaches had to resort to user input, while our method performs fully automatic feature curve co-completion. Finding feature reoccurrences however, is challenging since naïve feature curve comparison fails in this setting due to fragmentation and partial overlaps of curve segments. To tackle this problem we propose a robust method for partial curve matching. This provides us with the means to apply symmetry detection methods to identify co-occurring configurations. Finally, Bayesian model selection enables us to detect and group re-occurrences that describe the data well and with low redundancy.

CCS Concepts

•***Computing methodologies*** $\rightarrow$ *Shape Analysis;*

1. Introduction

The detection of meaningful feature curve networks (FCNs, cf. Section 4) on surfaces has a wide range of applications. They can be used to guide low level geometry processing tasks like remeshing as well as facilitate high level abstractions. For man-made shapes, which consist mostly of smooth and planar regions bounded by discontinuities, a meaningful FCN can be found by simply tracing out these discontinuities and high curvature regions.

The development and availability of 3D scanners allows capturing of real world data, which is under the influence of noise and measurement uncertainty. Furthermore, it can occur that parts of the geometry are not visible from a given scan position, leading to partially missing data. FCNs containing meaningful feature curves would greatly benefit the processing and analysis of geometric data acquired from the real world. However, the automatic generation of FCNs commonly only relies on local information. Hence the resulting feature curves can be prone to noise, incomplete, and fragmented. A common (local) denoising technique is to smooth the surface or threshold the feature curves according to their curvature. This also removes weak feature curves, which can be essential to describe the surface. Furthermore, this method does not complete fragmented or missing segments. In this paper we use the global co-occurrence information of feature curve groups to suppress noisy weak features and complete missing data in the FCN.

Feature curve configurations that reappear multiple times have a high probability of being meaningful, and are less likely due to noise. By detecting co-occurring groups of curves this global feature information can be used to identify relevant curves and complete the FCN. This task is especially challenging due to various sources of measurement and data uncertainty. Due to this uncertainty previous work in the field [SJW*11, BWM*11] rely on user input (e.g. user specified curve templates, or point correspondences) in the presence of noise.

In this paper we present a fully automatic approach to feature curve co-completion in noisy data. By globally identifying reappearing feature curve configurations we are able to consolidate these co-occurrences to a completed feature curve template. A template which describes multiple reoccurrences of the same feature curves provides us with the means to complete the feature curve information as well as neglect sporadic short and weak features which are not observed multiple times. Furthermore, we obtain high level shape information to describe and abstract the geometry.

1.1. Contributions

Our contributions can be summarized as follows:

- We present a novel unsupervised approach to extract templates of reoccurring feature curve configurations on given surfaces. (Together with their generating transformations these templates abstract the given shape.)
- To compare feature curves we have developed a new method to match feature curve sub-segments robustly.
- We use the feature curve templates to complete FCNs under the influence of noise. These networks can be used for further processing tasks.

The rest of this paper is organized as follows: In Section 2 we summarize related work in the field. Section 3 describes an overview over the template generation process, while Section 4 and 5 give the mathematical and implementational details. In Section 6 we show results of the co-occurrence analysis and the FCN completion.

2. Related Work

Feature Curve Detection There exist various possibilities for the generation of FCNs. Several view dependent contour detection approaches were proposed that extract aesthetic curves on a shape based on a certain viewing position (e.g. [ZHX*11, KST09, KST08, DFRS03]). However, to utilize the curves in downstream applications a view independent feature detection method is usually preferable, if the geometry is not fixed to one viewing position. E.g. feature curves can be extracted as patch boundaries from a segmentation (for example [NSP10, WG09, CSAD04]). These patch layouts usually do not trace out weak features and might not result in a detailed FCN. In this case a different line of research, which traces ridges and ravines of the surface can be used (e.g. [HPW05, YBS05, CP05, OBS04, MF10, WB01, BPK98]). The traced curves can capture up to very small details, but can also identify high frequency noise as false positives or stop the tracing of a feature curve if its principal direction is unclear. Hence, using these approaches in the presence of noise and data uncertainty leads to inaccurate and fragmented FCNs, which are not useful to further process the geometry. The shortcomings of both patch-based methods and feature curve tracing approaches are summarized in [CIE*16]. However, especially for noisy data further processing of the geometry is relevant. Since the ridge/valley approaches detect weak as well as strong features we use these as input to our system to identify meaningful curves in the network by co-occurrence analysis and complete the fragmented networks with detailed curve templates. A clean FCN is especially useful for applications that rely on feature curves as an input for, e.g, segmentation [ZDCJ17], simplification [GLK16], shape matching [GBK16], or remeshing [ECBK14, LHJ*14].

Symmetry Detection Several approaches to symmetry detection have been proposed over the last decade. It is out of scope of this paper to present all of these and we refer the interested reader to [MPWC13]. Here, we only describe the most related approaches.

Symmetry detection based on transformation space clustering was proposed in [MGP06, PMW*08, SAD*16]. In these approaches the co-occurring parts are detected in a three step procedure. First, similar point pairs (according to a point signature) are detected and a transformation between each pair of points is computed. In a second step these transformations are cast into a voting space, where clusters of transformations are sought. If several points are transformed by the same transformation it is likely that this transformation describes the re-occurrence of the same geometric entity. Some methods detect whether the extracted transformations align on a grid (cf. [PMW*08, SAD*16]) in a third step, which allows them to find orbits of multiple occurrences. These approaches strongly rely on the quality of the initial point matches. In the presence of noise local point signatures tend to be inexact, hence the search space can become very large and the false-positive rate in the set of point correspondences increases. This makes it hard to find structure in such a transformation space. Feature curve segments trace out significant ridges/ravines of the geometry and thus provide information on an intermediate (more global) level. We propose a robust partial curve matching technique (cf Section 4.1.1), which avoids the computation of unreliable point signatures and is thus more robust to noise. Robust feature curve matching allows to decrease the false positive rate of potential matches and simplifies the identification of reoccurring instances of the same geometry. Furthermore, considering only the points that lie on the feature curves of a shape greatly re-

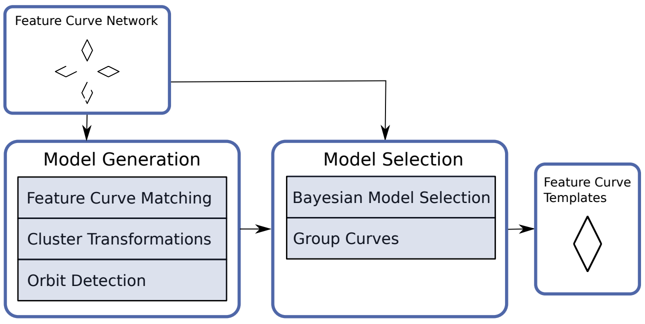


Figure 2: *System overview. We generate templates of feature curves in a two step procedure. In the model generation step we detect co-occurring groups of feature curves. In the model selection step, we select the set of transformations that generates the geometry and group the curves based on these transformations.*

duces the complexity of the search space. Similar to Pauly et al. we propose a three-step procedure to analyze the transformation space of feature curves to identify co-occurring curve configurations.

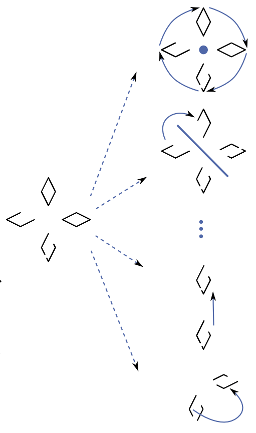
Feature Curve Based Symmetry Detection Feature line based symmetry detection has been investigated in [BBW*09]. Bokeloh et al. use a subsampling of a set of strong feature curves, in a RANSAC-based pattern matching approach. However, their approach requires a preprocessing of the feature curves by neglecting weak features and generating longer connected segments. Initial matches are found by greedily grouping close feature curves which poses assumptions the curve distribution on the surface. In a setting with noise it is hard to separate groups curves by distance.
Berner et al. [BWM*11] detect subspace symmetries by finding linearly correlated parts in graphs of surface features and extract correspondences in these that occur multiple times. Their method relies on high curvature regions and discards matches that do not have a matching graph topology. In the presence of noise a user has to set the correspondences of the re-occurring feature points.
In our approach we postpone important decisions which feature curves of the entire input curve set to keep/complete until we have collected all the evidence. In contrast to pure symmetry detection as it is done by [BBW*09, BWM*11] our objective is to complete the FCN. For this it is crucial to identify reoccurring feature configurations among weak and strong feature curves.
Ceylan et al. [Ceylan2016] use feature curve templates as input to detect reoccurrences of variations of these in raw data. They point out that it is very challenging to obtain curve templates automatically from real world data. In our work we compute such templates fully automatically on scanned noisy input, which can be used as building blocks for such applications.

Feature Completion A semi-supervised approach for feature completion has been presented in [SJW*11]. Here a user draws feature curve templates onto instances of the reoccurring geometry. The system automatically finds multiple reoccurrences of this curve. In our work we automatically find groups of reoccurring feature curves, which allows us to compute feature curve templates without any user input.

Li et al. have formulated an expectation maximization approach to feature *point* completion in the context of co-occurrence analysis [LWWS15]. In contrast, we generate templates for feature *curves*, which can be used for further processing such as remeshing. This is not possible using feature points directly.

3. System Overview

Our goal is to extract *feature curve templates* from a fragmented FCN computed on noisy data. We construct such templates by identifying reoccuring features curves, which can then be grouped together. Due to the more global information encoded in a template, we are able to complete noisy FCNs. In the following we give a system overview of our approach. The *co-occurrence analysis*, which is necessary for the template construction, consists of two main steps (cf. Figure 2):



Model Generation: In the *model generation* phase sets of reappearing feature curves are identified, with the objective to find rigid transformations that generate reoccurring parts of the observed data. E.g. the image to the right shows the feature curves of a reoccurring rhomb. Several transformations (rightmost) can be used to explain parts of the data. We define a sub-set of such transformations as a model. However, real world data (e.g. acquired from laser scans) underlies multiple forms of data uncertainty and inaccuracy:

1. *Geometric inaccuracy:* The measured (real world) geometry is not equal at each perceived re-occurrence.
2. *Measurement uncertainty:* The measurements can suffer from noise.
3. *Missing data/coverage:* Depending on the scan position not all parts of geometry are scanned equally.
4. *Feature curve uncertainty:* In a local setting it is hard to evaluate whether a feature curve is merely noise or whether it represents a meaningful part of an object.

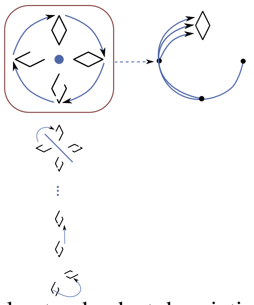
These forms of uncertainty imply the following requirements:

1. *Avoid over-reliance on local decisions:* Local point descriptors are unreliable in the presence of noise, which makes it hard to find structure based on these (e.g. in the transformation space). We find correspondences on a more global level by taking the geometry of the feature curve into account.
2. *Soft feature curve identification:* We only leverage feature curves with strong evidence (which are long or which have high average curvature) for the model generation process. Note that the *entire* noisy FCN is considered in the *model selection* step.
3. *Partial feature curve matching:* Correspondences between feature curve pairs need to be described as partial matches (Section 4.1.1), since feature curves are traced locally and can thus be fragmented or overlap at different reoccurrences.

4. *Transformations clustering:* To identify transformations that appear multiple times a cluster analysis is performed rather than testing for strict equality.

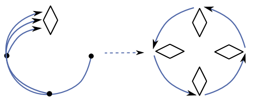
As a final step of the model generation phase we detect orbits in the transformation space (i.e. identifying whether subsets of the reoccurring transformations can be described by a grid), as in [PMW*08, SAD*16].

Model Selection: In the second phase, the *model selection*, a descriptive yet concise subset of these transformations needs to be selected. Since feature curves can be generated by multiple transformations, the goal is to extract a sparse subset of transformations which describes the set of reoccurring feature curves well. E.g. in the example to the right the topmost 90° rotation around the blue rotation center is selected, since it provides the least redundant description of the data. The selected generating transformation can be used to group the reoccurrences into one template (rightmost).



If we had perfect knowledge of the reoccurrences of feature curves we would be able to evaluate measures as the minimum description length [Ris83] for model selection. However, we have to identify which transformations describe the entire FCN containing weak and noisy feature curves best. Therefore, we cannot simply minimize the description length of a set of models. Instead we take the inherent uncertainty in the data into account and assign soft likelihood values to each feature curve, which describe how likely it is that the feature curve has been generated by a given set of transformations (model). With Bayesian model selection (Section 4.2.1) we are able to identify a concise and descriptive set of models.

We use the generated templates to *complete* the feature curve information, by applying the inverse transformation and inserting the feature template at every location where a reoccurrence was observed.



4. Cooccurence Analysis

Given a shape $\mathscr{S}$, a FCN $\mathscr{F}$ of $\mathscr{S}$ consists of a set of curves C which lie on $\mathscr{S}$. In a discrete setting these curves $c \in C$ are described by sequences of consecutive vertices $v \in \mathbb{R}^3$, i.e. $c = (v_0, \ldots, v_k)$.

A rigid transformation $T \in \mathbb{R}^{4\times4}$ can be applied to a feature curve $c \in C$ by either transforming each vertex or a subset of the vertices of the feature curve. A set of transformations that allows the generation of the feature curve data from a sparse subset of the initial FCN provides a good abstraction of the data. Hence our goal is to find such a model $M = \{T_0, \ldots, T_m\}$ (i.e. a set of transformations) that is descriptive yet non-redundant.

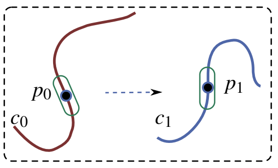
4.1. Model Generation

The goal of the model generation step is to identify groups of similar feature curves that reoccur under the same rigid transformations $T_0, \ldots, T_n$. For this we first need to identify similar feature curves, which will be described in Section 4.1.1. In a second step similar transformations are grouped into clusters (cf. Section 4.1.2).

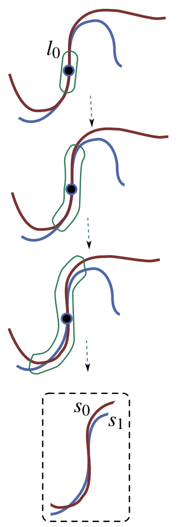
4.1.1. Partial Feature Curve Matching

Due to measurement and data uncertainty it is not possible to find unique correspondences between entire feature curves. Since cooccurring feature segments can overlap it is crucial to find partial matches between curves. This is a challenging task, since we would have to match all pairs of partial curves. To solve this problem we present a novel "sub-string matching" technique to identify partial matches between feature curve segments. Given a pair of feature curves $c_0 \in C$ and $c_1 \in C$ this two-step procedure first computes point correspondences between vertices on c_0 and c_1. In a second step the partial matches are obtained from the point matches by voting for the most likely transformation between c_0 and c_1.

Feature BLAST In the presence of noise local descriptors and respective point matches can become unreliable. Therefore we exploit the more global information provided by the feature curve geometry to establish robust point- and partial curve matches. Hence, we avoid the computation of unreliable descriptors in the presence of noise. To obtain point correspondences between a point p_0 on c_0 and p_1 on c_1 (i.e. points lying on feature curves, see image below) we have developed a method which is based on a similar idea as the Basic Local Alignment Search Tool (BLAST) [AGM*90]. This method is used in genetics for DNA sequencing. Instead of matching DNA strands we compute partial matches between feature curves. Algorithm 1 describes the steps of *Feature BLAST*.



As an input we provide the minimal match length $l_0 \in \mathbb{R}$ (which is encoded by the green ellipsoid in the image on the right) of two feature curve sequences as well as two hysteresis thresholds $s \in \mathbb{R}$ and $t \in \mathbb{R}$, with $s < t$. The lower threshold s evaluates whether an initial match was found (line 4 of Algorithm 1) by measuring the distance of two feature line segments s_0 and s_1 that are traced from p_0 and p_1 with the minimal match length l_0. The distance function is evaluated by translating p_0 onto p_1 and computing the best rotation between the two point sets. A match is only found if s is greater than the computed error. While the distance value d is less than the greater threshold t the length of the match is extended by Δl (third image from the top) until either d exceeds t (fourth image from the top) or the feature curve ends. In our experiments we set the parameters l_0 to $20-30$ times the average edge length of the underlying triangle mesh, s to the average edge length, and $t = 2 \cdot s$. Δl (cf. Algorithm 1) is set to 5 times the average edge length. Note that we only use curves with strong evidence in this step, which are longer than l_0 times the average



Algorithm 1 Feature BLAST

```
 1: function FEATUREBLAST($l_0$, $s$, $t$, $p_0$, $p_1$)
 2:     $l \leftarrow l_0$
 3:     distance ← DISTANCE($l_0$, $p_0$, $p_1$)
 4:     if distance $< s$ then
 5:         while distance $< t$ do
 6:             $l \leftarrow l + \Delta l$
 7:             distance ← DISTANCE($l$, $p_0$, $p_1$)
 8:     return $l \cdot \exp(-\text{distance}/(0.5 \cdot t))$
 9: function DISTANCE(length, $p_0$, $p_1$)
10:     $s_0 \leftarrow$ TRACECURVE($p_0$, $c_0$, length)
11:     $s_1 \leftarrow$ TRACECURVE($p_1$, $c_1$, length)
12:     $s_0^t \leftarrow s_0 - p_0 + p_1$
13:     $R \leftarrow$ ICP($s_0^t$, $s_1$)
14:     return AVERAGEEUCLIDEANDISTANCE($R(s_0^t)$, $s_1$)
```

edge length or have an average curvature along the curve which is higher than the $0.5 \cdot C$, where C denotes the average maximal absolute curvature, averaged only along the feature curves.

As a result we receive point correspondences between the vertices of the two feature curves with the respective match length as well as an optimal rigid transformation and an affinity value (both are used to vote for partial feature curve matches in the next step):

$$\text{FEATUREBLAST}(p_0, p_1) = l \cdot e^{-\frac{d}{0.5 \cdot t}}$$

between the two points p_0 and p_1 associated with the curve segments s_0 and s_1. In our experiments we compute this value for all pairs of points lying on the curves c_0 and c_1 and leverage these to vote for an optimal transformation between c_0 and c_1.

Voting for Transformations To find an optimal transformation for a pair of feature curves we exploit transformations between the point correspondences found in the previous step by letting the point correspondences vote for the best transformation. The 7-dimensional voting space (3 dimensions for translations and 4 for rotations in quaternion representation) is binned and the affinity values of the point matches whose transformations fall in to the same bin are summed up. In our experiments we use bin sizes of 2 times the average edge length of the underlying geometry for the translations and 1.5° for rotations in unit quaternion representation. We extract the maxima from the voting space as best transformations between two feature curves. Note that this second step is only applied if a correspondence is detected in the previous step.

4.1.2. Cluster Transformations

Finally, to find reoccurrences of similar transformations we need to cluster the transformations that derive from the previous step (global voting) over all pairs of feature curves. We use a hierarchical clustering approach where we stop clustering when the distance of the next element is larger than a predefined maximal error of the transformations (we use 2 times the average edge length of the underlying geometry for the translations and an angle of 1.5° between the unit quaternion representation of the rotations in our examples).

If these reoccurring transformations can be described by a regular grid we can find multiple reoccurrences of the same geometric entity, which are described by a generating transformation. In order to detect these orbits we apply the method presented in [PMW*08].

4.2. Model Selection

In the previous step we have extracted a set of reoccurring transformations $\mathcal{T} = \{T_0, \ldots, T_n\}$ from the data. The model we select in this step is represented by a subset of $\mathcal{T}$. Due to the different sources of uncertainty in the data (the feature curves) we can mere assign soft liklihood values based on the observed evidence. In this probablistic setting Bayesian model selection allows to identify the model that maximizes the support for the data. In Section 4.2.1 we will explain how Bayesian model selection can be used to choose a descriptive model. However, the amount of possible models that can be used to abstract the data is given by the powerset of $\mathcal{T}$. It is infeasible to compute and compare all of these models. We will present an effective heuristic to select the best model in Section 4.2.2. In Section 5.2 we describe how feature curves are grouped to a single feature curve template per generating transformation.

4.2.1. Bayesian Model Selection

Given a set of models $\mathcal{M} = \{M_0, \ldots M_k\}$ that describe the data $C = \{c_0, \ldots, c_m\}$ (the entire noisy FCN), we are able to choose a model that describes the data well while providing low redundancy based on Bayesian model selection. The output of this model selection step is a model $M_i = \{T_0^i, \ldots T_q^i\}$ that assigns a transformation to each reoccurring feature curve and neglects the curves that are not supported by M_i. According to Bayes' rule the probability of a model given the data can be computed as

$$P(M_i|C) = \frac{P(C|M_i) \cdot P(M_i)}{P(C)}.$$

Since we can assume that the data is measured independently we can regard each data observation independently and compute $P(M_i|C)$ as

$$P(M_i|c_0, \ldots, c_m) = \frac{\prod_{k=0}^m P(c_k|M_i) \cdot P(M_i)}{\prod_{k=0}^m P(c_k)}.$$

The Bayes ratio can be used to compare two models [KR93]. It is especially useful since the term in the denominator is constant and thus factored out. The Bayes ratio is computed by

$$K = \frac{P(M_i|C)}{P(M_j|C)} = \frac{P(C|M_i) \cdot P(M_i)}{P(C|M_j) \cdot P(M_j)}.$$

A value of $K > 1$ means that the model M_i has stronger support for the data than M_j. According to [Jef98, KR95] K gives insights on the weight of evidence to select one model over the other. Following their scheme a value of $K > 10$ strongly supports the model M_i over M_j, while a value close to 1 means that the models have a similar descriptive power.

To evaluate the Bayes ratio we assume a uniform distribution as prior $P(M)$ (all models are equally likely). We describe the likelihood values of the feature curves given a model M_i, i.e. $P(c_k|M_i)$ by testing whether the feature curve c_k was generated from other feature curves in C by applying one of the transformations $T_j^i \in M_i$. Therefor we apply the inverse transformation $(T_j^i)^{-1}$ for all $T_j^i \in M_i$ to the points p_{c_k} that compose the curve c_k

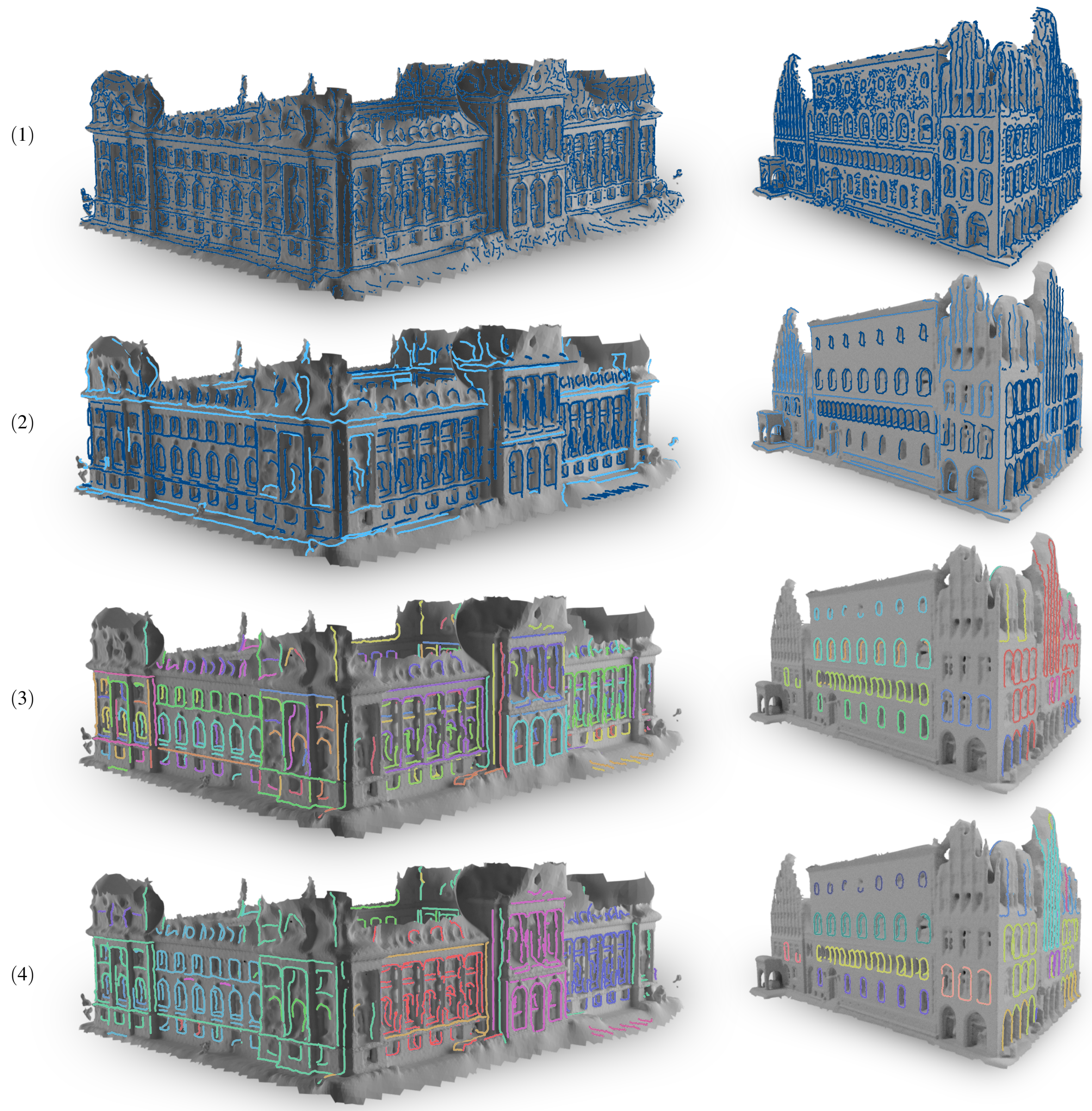


Figure 3: *Scan of a museum (left) and a townhall (right), with (1) the initial dense feature network generated with [YBS05]. (2) shows the completed FCN. (3) depicts the connected components. (4) shows the feature curve groups that fall into the same template. Dark blue curves indicate the reoccurring templates, while strong (non-reoccurring) curves are visualized in light blue.*

$(p_{c_k}^{\mathrm{Tr}} = (T_j^i)^{-1} \cdot p_{c_k})$. Then we compute the average Feature BLAST affinity value $a(T_j^i, c_k)$ over the vertices of the feature curve (as described in Section 4.1.1 only that the transformation is given by $(T_j^i)^{-1}$). The corresponding feature curve for the BLAST matching is found as the feature curve on which the nearest neighbor $p_{c_k}^{\mathrm{nn}}$ of $p_{c_k}^{\mathrm{Tr}}$ lies. In case the transformation describes a grid we sum up the average affinity values for all reoccurrences.

We add T_{null} to each model, which encodes that a feature curve was not generated by any of the given transformations, and set

$$a(T_{null}, c_k) = l_0 \cdot \exp\left(-\frac{t}{0.5 \cdot t}\right) = l_0 \cdot e^{-2},$$

which is the smallest possible affinity value if the minimum match length l_0 is met. The likelihood of feature curve c_k given the model M_i is then computed as

$$P(c_k|M_i) = \max_{T_j^i \in M_i} a(T_j^i, c_k). \tag{1}$$

For numerical stability all computations are performed in log-space.

Note that after model selection point correspondences on the reoccurring curve instances are automatically given by the association of p_{c_k} and $p_{c_k}^{\mathrm{nn}}$. In case the curves on which p_{c_k} and the reoccurrences $p_{c_k}^{\mathrm{nn}}$ lie are assigned to the same transformation and the corresponding BLAST score exceeds the matching threshold s the points $p_{c_k}^{\mathrm{nn}}$ are considered as reoccurrences of p_{c_k}.

4.2.2. Heuristic Model Selection

Since it is infeasible to compare all possible models ($2^{|\mathscr{T}|}$ many), we propose a heuristic to neglect redundant transformations. We start with the model that holds the entire set of transformations, i.e. $M^0 = \mathscr{T}$ and recursively remove transformations from this set until the Bayes ratio between the models M^i and M^{i+1} shows strong support for the model M^i (i.e. $K > 10$). We greedily remove the transformation with least support for the data (i.e. with minimal $P(C|M) \cdot P(M)$). Note that T_{null} is excluded from this removal. If $K > 10$ after removing a transformation we know that the model M^i was of significance. Hence, we can find a descriptive model (w.r.t. the definition of [Jef98, KR95]) with M^i after at most $|\mathscr{T}|$ iterations (since one transformation is removed in each iteration), i.e. after at most $|\mathscr{T}|^2$ computations of the Bayes ratio.

5. Feature Curve Co-Completion

In the final FCN feature curve classification noise is discarded (Section 5.1) and reoccurring groups of feature curves are grouped into templates, which are completed based on their co-occurrence information (Section 5.2). The Bayesian model selection results in a set of transformations $M_i = \{T_0^i, \ldots, T_q^i, T_{null}\}$ with an assignment of each feature curve c to one of the transformations T_c in M_i. T_c is selected as *arg max* for equation (1). The mapping $B_{T_c \in M_i} : C \to C$ is then defined between pairs of feature curves under the selected transformation.

5.1. Feature Curve Classification Noise Removal

Curves that are asigned to the model T_{null} do not reoccur with respect to one of the transformations in the selected model M_i. Hence, we assume that weak non-reoccurring curves were classified as false postives by the feature detection method. These are removed from the network. Only siginficant curves are retained. For these non-reoccurring curves we set a high threshold so that no classification noise is added (curves that are longer than $3 \cdot l_0$ times the average edge length or have an average curvature higher than C and are longer than l_0 times the average edge length). In our results we indicate these curves in light blue.

5.2. Template Generation

The remaining curves correspond with respect to one of the transformations $T_0^i, \ldots, T_q^i$. These are grouped into templates, which are extracted for each of the transformations $T_0^i, \ldots, T_q^i$ separately.

Connected Component Computation First, all feature curves that belong to the same component are identified. We define a graph $\mathscr{G} = \{C, E\}$, where C represents the set of feature curves (the nodes) and $E = \{(c, B_{T_c}(c)) | c \in C\}$ the set of undirected edges. In this graph nodes are connected by an edge if and only if their corresponding curves are mapped onto each other by the transformation T, i.e. if point-correspondences exist on the respective curves (cf. Section 4.2.1). Hence, the connected components of this graph represent reoccurrences of the same feature curve. We show the connected components as third image of our results in Figures 1, 3-6, and 8. Note, that these (exact) feature curve correspondences with point-to-point reoccurrence information are essential for the FCN completion step (otherwise it would be hard to separate and complete close curves individually).

Connected Component Completion To compute the templates we overlay the reoccurrences of each connected component by applying the inverse transformation. To consolidate the curves (i.e. select one alternative of the multiple reoccurrences) we implement a shortest path search on the possible (reoccurring) paths through the FCN. Note that for this step the identification of corresponding feature curve segments is crucial (i.e. previous symmetry detection methods do not provide this information). Only those segments where point-correspondences on other instances exist are considered in the graph. To identify alternative paths we project the end-points of the same connected components onto their nearest neighbors of the re-occurrences in an ε-radius. The new start- and end-points of this network are all vertices that are projected to end-points only (not onto an intermediate point of the feature curve). In this network we iteratively find the longest *shortest path* from one start-point to an end-point and remove all redundant paths.

Grid Origin Computation Different connected components that are assigned to the same transformation and are seeded at the same origin should be grouped into one template. The transformations alone do not provide information on where detected grids are seeded. In the following we describe a method to locate the origins by using the curve information only.

In case we have detected a grid in the orbit detection a seed point for the grid needs to be found. To find this point we project the feature curves belonging to the same connected component onto their generators. In case the grid is spanned by two translation vectors g_0 and g_1 and these are perfectly orthogonal we can simply project onto g_0 and g_1. Otherwise, the we need to account for the shear and compute two projection directions $\bar{g_0}$ and $\bar{g_1}$:

$$\begin{aligned} n &= g_{0_n} \times g_{1_n} \\ g_0^{\perp} &= g_{1_n} \times n \\ \bar{g_0} &= \langle g_0, g_0^{\perp} \rangle \cdot g_0^{\perp} \end{aligned}$$

where g_{i_n} are the normalized generators, $\times$ denotes the cross product and $\langle \cdot, \cdot \rangle$ the dot product between the respective vectors. The same computation can be performed for $\bar{g_1}$. In case the generator is a rotation angle we project the feature lines onto a circle around the respective center of rotation. The projection directions (or circle) are binned and the amount of projected curves is stored in these bins. The optimal offset for the grid with generator g is found by computing the minimal value for the offset index z:

$$\min_z \sum_{i=0}^{k} \text{binentry}(z + i \cdot \parallel g \parallel)$$

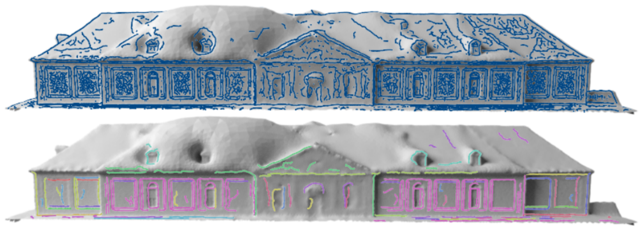


Figure 4: *Scan of a manor house, with the initial dense FCN generated with [YBS05] (top). The bottom row shows the connected components in the same color. Note, that we are able to detect corresponding feature curves even though these are very jaggy given the high amount of noise in the data.*

Noise	distance	co-occurring curves (ours)	all curves	no curves
0.05%	Hausdorff	$\mathbf{7.96 \cdot 10^{-3}}$	$1.83 \cdot 10^{-2}$	$1.64 \cdot 10^{-2}$
0.05%	average	$\mathbf{4.92 \cdot 10^{-4}}$	$7.79 \cdot 10^{-4}$	$8.31 \cdot 10^{-4}$
0.1%	Hausdorff	$\mathbf{1.49 \cdot 10^{-2}}$	$1.78 \cdot 10^{-2}$	$1.50 \cdot 10^{-2}$
0.1%	average	$\mathbf{5.47 \cdot 10^{-4}}$	$8.03 \cdot 10^{-4}$	$1.05 \cdot 10^{-3}$

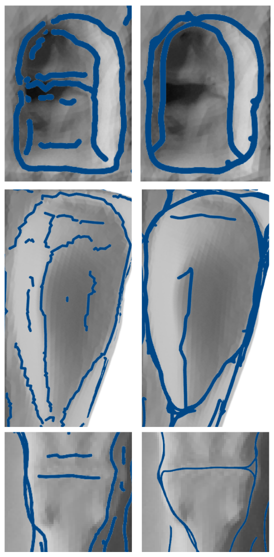
Table 1: *Hausdorff- and average distance values for smoothing experiment. We apply feature preserving C^1 smoothing (100 iterations) to the meshes in Figure 6, by retaining points on the curves detected by our method (3rd column) and the initial curves (4th column) and compare these to unconstrained smoothing (5th column). The bounding box diagonal has length* 1.

Connected components that fall into the same grid are then grouped together into one template. Furthermore, this allows us to identify gaps in the grids, which are detected if at least two neighboring grid positions contain no evidence of the curve template. At these points we split the grid for the template extraction. In the fourth images of Figures 1 and 5 we encode the curves that were assigned to the same template with the same color. The final templates are obtained by transforming the completed connected components that are assigned to the same template into a common grid cell with respect to the defined grid origin.

For the resulting FCN we transform the computed templates to each position where we have measured an occurrence of the respective template. In addition strong non-reoccurring curves are added to the network as described in Section 5.1.

6. Results

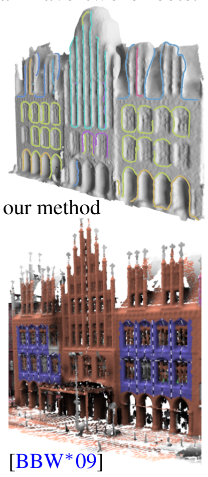
In the following we will show results of the co-occurrence analysis and the feature curve completion. Figure 4 depicts the output of the co-occurrence analysis on a scanned object with high amount of noise. The colors encode the connected components of the feature curves. Note, that although the input feature curves are very dense and noisy we are able to identify reoccurrences of the same curve configurations.



Feature Curve Co-Completion Figures 1, 3, 5, and 8 (1) show the dense input feature curves. Note that all FCNs are generated on noisy point cloud data. The initial curves are generated automatically with the method presented in [YBS05]. (2) visualizes the completed FCNs. To complete the network we insert the extracted templates at each detected reoccurrence (dark blue) as well as strong features, which were not detected as part of the reoccurring configurations (light blue) into the network. (3) shows the connected components (described in Section 5.2). Figures 1 and 5 (4) depict the groups which where assigned to the same grid position. In Figures 3 and 8 (4) we encode the feature curves that are grouped into the same template in the same color.

With our method we are able to identify co-occurring configurations of feature curves and use this information to complete the FCNs. E.g. in the images above we show close-ups of the original noisy and fragmented curves (left) of the models. Our approach is able to complete significant curves and remove the noisy features (right).

Noise For a synthetic example we show the effect of noise on the feature curves and the resulting co-completion in Figure 6. Gaussian noise with a standard deviation of 0.05% (middle) and 0.1% (right) of the length of the bounding box diagonal is added. The initial feature curves are more jagged and fragmented compared to the curves computed on the synthetic data. Nonetheless, we are able to detect most of the reoccurrences for these noise levels (bottom row) and complete the network based on these (middle row). However if the noise level increases the completed curve networks can also include jagged curves. To measure the effectiveness of our method we remove the artificial noise by applying 100 iterations of feature preserving C^1 smoothing by retaining points on the completed FCN. Hausdorff- and average distances to the orginal data are given in Table 1. For comparison we show the results using all input curves and unconstrained smoothing. Using all input curves can have two effects. First, the distance values are higher since more displacement noise is preserved. Second, in case of incomplete curve information the geometry is smoothed, increasing the distance to the original surface.



Comparison Co-occurrence Analysis We compare our results to the feature line based symmetry detection from [BBW*09] in the image on the right. Although we do not find all reoccurrenes of the yellow windows we are able to separate the components more distinctly compared to [BBW*09], which is crutial for the templated extraction and the completion of FCNs. Furthermore, we do not perform any preprocessing and use the input curves that are computed fully automatically with [YBS05].

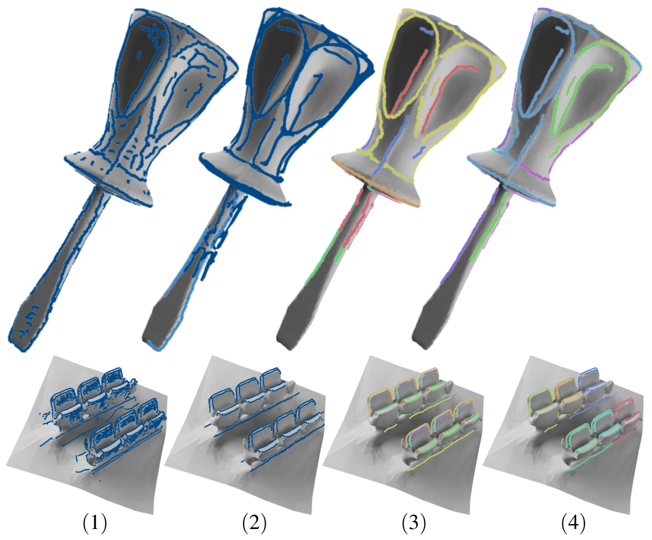


Figure 5: *Screwdriver (top) and scan of theater seats (bottom), with the initial dense feature network generated with [YBS05] (1), the FCN which was completed with our method (2), the connected components (3), and the feature curve groups that fall into the same orbit (4). Dark blue curves indicate the reoccurring templates, while strong (non-reoccurring) curves are visualized in light blue.*

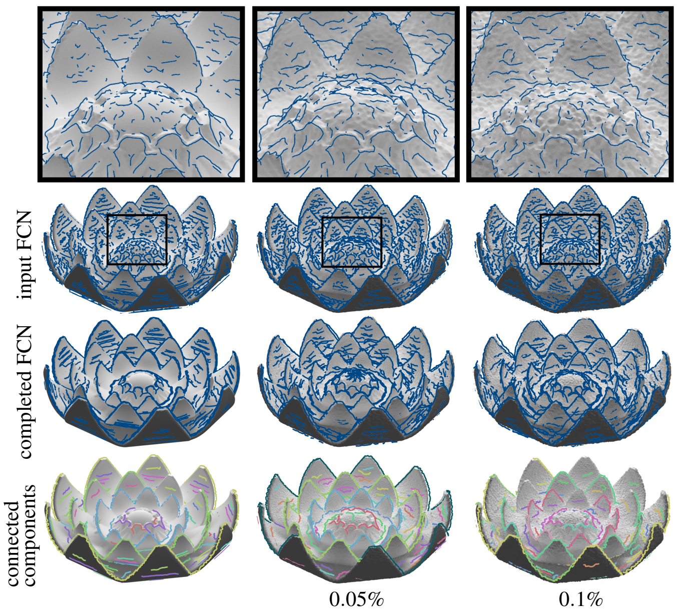
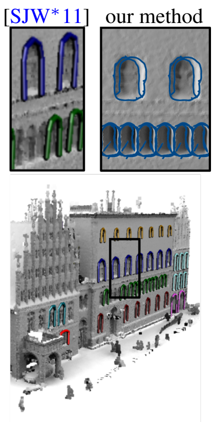


Figure 6: *Noise test. Different levels of Gaussian noise with a standard deviation of* 0.05% *(middle) and* 0.1% *(right) of the bounding box diagonal were added to the synthetic lotus-flower mesh (left). The completed FCNs are depicted in the intermediate row. The bottom row shows the connected components. Note that most reoccurences are detected even with a high amount of noise.*

We compare our method to symmetry detection with [MGP06] in Figure 7 (right). With [MGP06] we find two translational (left to right) reoccurences of the chairs (green and red). The heat kernel signature [SOG09] is used as underlying descriptor for point matching. Our method benefits from the connectivity of the feature curves which allows to find connected components in each reoccurrence. Hence, our method detects all instances of the chairs and is robust to noise. In contrast, local descriptors can become unreliable under the influence of noise, which leads to a scattered transformation space in which it is hard to find meaningfull cluster centers. For [MGP06] we measured a computation time of 1511.31s for the point-signatures and 174.21s for the symmetry detection. With our method it took 74.59s (cf. Table 2).

Furthermore, we compare our results to [BWM*11] in Figure 8. Berner et al. are able find reoccurring feature curve groups of the windows of the church (purple lines) from the input FCN (yellow lines). However, their input FCN only contains high curvature feature lines. This is sufficient to detect the co-occurences, but to complete the FCN the identification of corresponding weaker curves is crutial. The detection method relies on a similar topology of the reoccurrences. In the presence of noise this cannot be ensured. E.g. the topology of the pillars next to the windows varies (cf. Figure 8 (bottom row)), hence they are not included into the mapping. In these cases a user has to define corresponding points in the reoccurring instances. The feature curve input of our method can be very dense, jagged, and noisy. However we analyze the entire FCN, since this is relvant to include weak and strong feature curves into the templates for completion. Nonetheless, we are able to find the reoccurrences and extract templates fully automatically for the the completion of the FCN.



Feature-Completion In the image on the right we compare with the supervised feature line learning approach presented in [SJW*11]. The feature completion approach by Sunkel et al. requires a user to draw the curve templates onto instances of the geometry. In contrast our method is unsupervised and finds the reoccurring templates automatically. See Figure on the right (right) for comparison.

Curvature Threshold A local method to denoise feature curves is to threshold these, e.g. based on their curvature. An example is given in Figure 9, which shows an excerpt of the museum depicted in Figure 3. By increasing the curvature threshold, the number of curves are reduced. Note that even in the example with the highest curvature threshold (bottom left) not all noise is removed, while relevant curves are missing. I.e. usually it is not possible to set one threshold for the entire geometry. With our method (bottom right) relevant features are detected by co-occurrence. Additionally, reoccurring groups are completed (e.g. the missing part of the window on the bottom right).

Feature Curve Input In Figure 10 we demonstrate that the method is flexible with respect to the FCN input. We compute three different kinds of FCNs (for which the software is available online): [YBS05], CGALs [CP05] implementation, and [OBS04] with

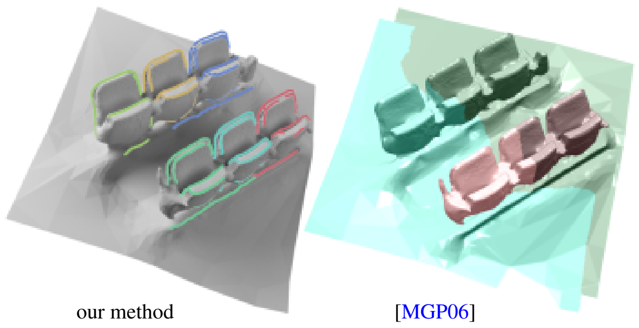


Figure 7: *Comparison to symmetry detection with [MGP06]. Two translational reoccurences (one in the front row and one in the back row, both left to right, red and green colored geometry indicate the two different translations) of the chairs have been detected with the approach of [MGP06]. Our method benefits from the detection of feature curve correspondences, which avoids the computation of unreliable local point descriptors (the underlying descriptor used for [MGP06] is the heat kernel signature [SOG09]). Our method (left) detects and divides the six instances of the cooccurring chairs exactly (each instance is indicated in a different color). This is crucial for the definition of a feature curve template.*

Model	BLAST	Clustering	Orbits	Model Selection
Windows Cathedral	83.40	0.01	0.14	3.72
Seats Theater	66.81	0.05	1.44	5.29
Screwdriver	40.75	0.02	2.08	9.84
Town Hall	7475.47	490.85	48.81	733.46
Manor House	18056.78	228.19	3.68	741.70
Museum	1924.36	1.73	2.23	682.56
Church	6126.55	8.65	2.62	5438.34

Table 2: *Timings of the different steps of our algorithm in seconds. Note that BLAST and Model Selection can be parallelized easily.*

the curvature computation [Rus04]. In each example the reoccuring window configurations are completed similarly.

Table 2 shows the runtimes of our algorithm. The results were computed on a commodity notebook (Intel Core i7, 16 GB RAM). Note that the Bayesian model selection as well as the computation of the Feature BLAST are easily parallelized.

Limitations In cases with extreme noise levels where only very small fragmented feature segments are traced our method will not be able to detect structure, since we will have no evidence for the model generation. I.e. BLAST will not find reliable matches such that a transformation space clustering will not lead to reasonable results. Furthermore, it can occur that a detected reoccurrence falls into a subgrid. E.g. the handle of the screwdriver in Figure 5 has repeated feature curves in a 60° degree angle, while the shaft repeats in a 180° angle. The 180° grid falls into the 60° grid. So when the feature curves are transformed back for completion they are repeated at every grid location. Another limitation derives from very jaggy curves. In this case the feature curve consolidation can fail and multiple arcs representing the same feature occur in the template. In future work we plan to extend the template generation by selecting the features by extracting part based models.

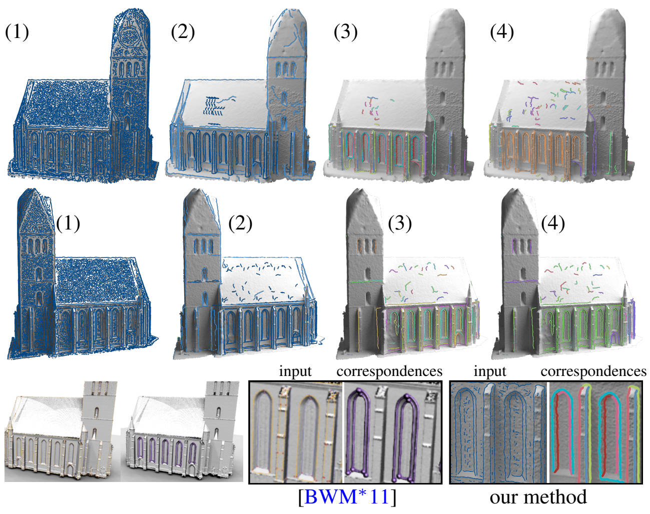


Figure 8: *Comparison to [BWM* 11]. The top and middle row show results of our feature curve co-completion with (1) the input FCN, (2) the co-completed FCN, (3) the connected components, and (4) the feature curve groups that fall into the same template. The bottom row depicts the results of [BWM* 11]. Note that our method does not rely on topological correspondence of the curves and does not require a user to select reoccurences in the presence of noise.*

7. Conclusion and Future Work

In this paper we have presented a fully automatic method to complete FCNs in the presence of noise. Previous methods rely on user input for the completion of a FCN. For this we have developed a novel robust method for partial feature curve matching, which is the basis for the co-occurrence analysis. With Bayesian model selection we are able to group feature curves to their generating transformations, and identify feature curves that are not supported by the given model. We show structure detection in noisy feature curve networks, which were generated fully automatically on scanned real world data and compute completed networks that contain meaningful feature curves.

These networks can be applied in various geometry processing tasks, e.g. to apply feature preserving remeshing or smoothing of the noisy geometry. The structure information can be used to complete the geometry and to support various shape analysis tasks such as retrieval, procedural modeling, or segmentation, which are more challenging on real world data. Our method provides valuable input to these approaches in the presence of noise.

Acknowledgements

The research leading to these results has received funding from the European Research Council under the European Union's Seventh Framework Programme (FP7/2007-2013)/ERC grant agreement n° [340884]. Several point clouds were provided by the

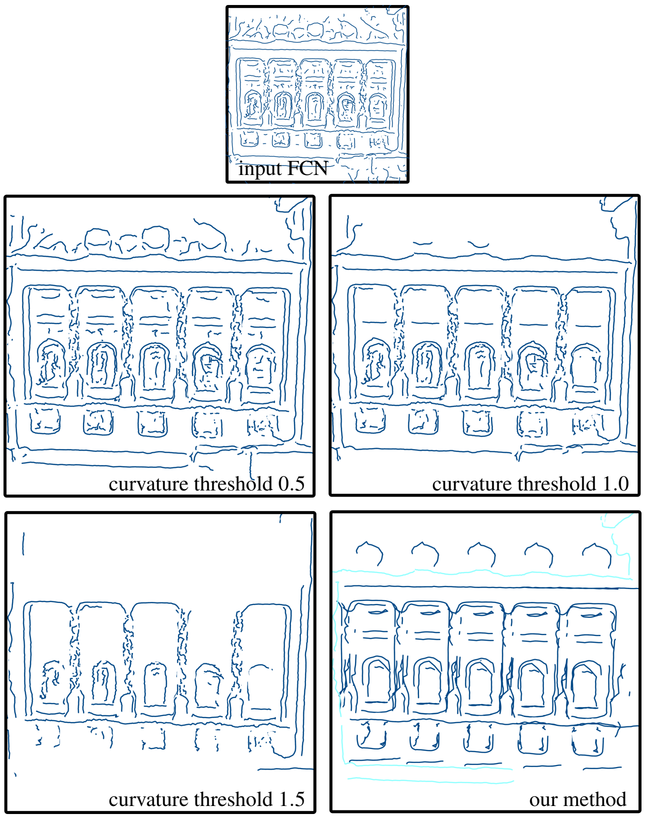


Figure 9: *Comparison of local feature curve denoising with three different curvature thresholds to our method. By discarding the input feature curves (top) based on their average curvature, weak but relevant features are suppressed before all the noise is removed. The co-occurrence analysis in our method (bottom right) allows to identify reoccurring curve configurations and complete feature curves that are not in the input feature set as well as neglect noisy features that do not reoccur.*

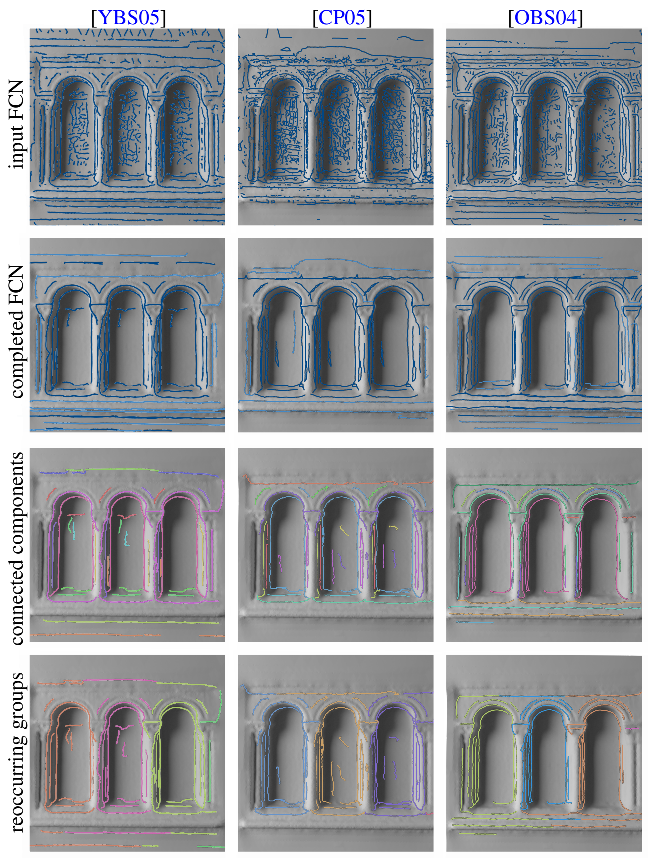


Figure 10: *Different feature detection methods. Initial feature curves are computed with [YBS05], [CP05], and [OBS04]. Although the input curves are different similar connected components and reoccuring groups are detected.*

Institute for Cartography and Geoinformatics of the Leibniz University Hannover. We would like to thank Jan Möbius for creating and maintaining the geometry processing framework OpenFlipper as well as the reviewers for their insightful comments.

References

[AGM*90] Altschul S. F., Gish W., Miller W., Myers E. W., Lipman D. J.: Basic local alignment search tool. *Journal of molecular biology 215*, 3 (1990), 403–410. 4

[BBW*09] Bokeloh M., Berner A., Wand M., Seidel H.-P., Schilling A.: Symmetry detection using feature lines. In *Computer Graphics Forum* (2009), vol. 28, pp. 697–706. 3, 8

[BPK98] Belyaev A. G., Pasko A. A., Kunii T. L.: Ridges and ravines on implicit surfaces. In *Computer Graphics International, 1998.* (1998), pp. 530–535. 2

[BWM*11] Berner A., Wand M., Mitra N. J., Mewes D., Seidel H.-P.: Shape analysis with subspace symmetries. *Computer Graphics Forum 30*, 2 (2011). 2, 3, 9, 10

[CIE*16] Campen M., Ibing M., Ebke H.-C., Zorin D., Kobbelt L.: Scale-invariant directional alignment of surface parametrizations. *Computer Graphics Forum 35*, 5 (2016). 2

[CP05] Cazals F., Pouget M.: *Topology driven algorithms for ridge extraction on meshes*. PhD thesis, INRIA, 2005. 2, 9, 11

[CSAD04] Cohen-Steiner D., Alliez P., Desbrun M.: Variational shape approximation. In *ACM SIGGRAPH 2004 Papers* (New York, NY, USA, 2004), SIGGRAPH '04, ACM, pp. 905–914. 2

[DFRS03] DeCarlo D., Finkelstein A., Rusinkiewicz S., Santella A.: Suggestive contours for conveying shape. *ACM Transactions on Graphics (Proc. SIGGRAPH) 22*, 3 (July 2003), 848–855. 2

[ECBK14] Ebke H.-C., Campen M., Bommes D., Kobbelt L.: Level-of-detail quad meshing. *ACM Trans. Graph. 33*, 6 (Nov. 2014), 184:1–184:11. 2

[GBK16] Gehre A., Bommes D., Kobbelt L.: Geodesic iso-curve signature. In *Proceedings of the Conference on Vision, Modeling and Visualization* (2016), Eurographics Association, pp. 17–28. 2

[GLK16] Gehre A., Lim I., Kobbelt L.: Adapting feature curve networks to a prescribed scale. In *Computer Graphics Forum* (2016), vol. 35, pp. 319–330. 2

[HPW05] Hildebrandt K., Polthier K., Wardetzky M.: Smooth feature lines on surface meshes. In *Symposium on geometry processing* (2005), pp. 85–90. 2

[Jef98] JEFFREYS H.: *The theory of probability*. OUP Oxford, 1998. 5, 7

[KR93] KASS R. E., RAFTERY A. E.: *Bayes Factors and Model Uncertainty*. Tech. Rep. 571, Carnegie Mellon University Dept of Statistics, Pittsburgh, PA 15213, 1993. 5

[KR95] KASS R. E., RAFTERY A. E.: Bayes factors. *Journal of the american statistical association 90*, 430 (1995), 773–795. 5, 7

[KST08] KOLOMENKIN M., SHIMSHONI I., TAL A.: Demarcating curves for shape illustration. *ACM Trans. Graph. 27*, 5 (Dec. 2008), 157:1–157:9. 2

[KST09] KOLOMENKIN M., SHIMSHONI I., TAL A.: On edge detection on surfaces. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on* (2009), IEEE, pp. 2767–2774. 2

[LHJ*14] LING R., HUANG J., JÜTTLER B., SUN F., BAO H., WANG W.: Spectral quadrangulation with feature curve alignment and element size control. *ACM Transactions on Graphics (TOG) 34*, 1 (2014), 11. 2

[LWWS15] LI C., WAND M., WU X., SEIDEL H.-P.: Approximate 3d partial symmetry detection using co-occurrence analysis. In *3D Vision (3DV), 2015 International Conference on* (2015), IEEE, pp. 425–433. 3

[MF10] MIAO Y., FENG J.: Perceptual-saliency extremum lines for 3d shape illustration. *The Visual Computer 26*, 6 (2010), 433–443. 2

[MGP06] MITRA N. J., GUIBAS L. J., PAULY M.: Partial and approximate symmetry detection for 3d geometry. In *ACM Transactions on Graphics (TOG)* (2006), vol. 25, ACM, pp. 560–568. 2, 9, 10

[MPWC13] MITRA N. J., PAULY M., WAND M., CEYLAN D.: Symmetry in 3d geometry: Extraction and applications. *Comput. Graph. Forum 32*, 6 (2013), 1–23. 2

[NSP10] NIESER M., SCHULZ C., POLTHIER K.: Patch layout from feature graphs. *Comput. Aided Des. 42*, 3 (Mar. 2010), 213–220. 2

[OBS04] OHTAKE Y., BELYAEV A., SEIDEL H.-P.: Ridge-valley lines on meshes via implicit surface fitting. *ACM Trans. Graph. 23*, 3 (Aug. 2004), 609–612. 2, 9, 11

[PMW*08] PAULY M., MITRA N. J., WALLNER J., POTTMANN H., GUIBAS L. J.: Discovering structural regularity in 3d geometry. In *ACM transactions on graphics (TOG)* (2008), vol. 27, ACM, p. 43. 2, 4, 5

[Ris83] RISSANEN J.: A universal prior for integers and estimation by minimum description length. *Ann. Statist. 11*, 2 (06 1983), 416–431. 4

[Rus04] RUSINKIEWICZ S.: Estimating curvatures and their derivatives on triangle meshes. In *Symposium on 3D Data Processing, Visualization, and Transmission* (Sept. 2004). 10

[SAD*16] SHI Z., ALLIEZ P., DESBRUN M., BAO H., HUANG J.: Symmetry and orbit detection via lie-algebra voting. In *Computer Graphics Forum* (2016), vol. 35, pp. 217–227. 2, 4

[SJW*11] SUNKEL M., JANSEN S., WAND M., EISEMANN E., SEIDEL H.-P.: Learning line features in 3d geometry. In *Computer Graphics Forum* (2011), vol. 30, pp. 267–276. 2, 3, 9

[SOG09] SUN J., OVSJANIKOV M., GUIBAS L.: A concise and provably informative multi-scale signature based on heat diffusion. In *Proceedings of the Symposium on Geometry Processing* (Aire-la-Ville, Switzerland, Switzerland, 2009), SGP '09, Eurographics Association, pp. 1383–1392. 9, 10

[WB01] WATANABE K., BELYAEV A. G.: Detection of salient curvature features on polygonal surfaces. In *Computer Graphics Forum* (2001), vol. 20, pp. 385–392. 2

[WG09] WEINKAUF T., GÜNTHER D.: Separatrix persistence: Extraction of salient edges on surfaces using topological methods. In *Proceedings of the Symposium on Geometry Processing* (Aire-la-Ville, Switzerland, Switzerland, 2009), SGP '09, Eurographics Association, pp. 1519–1528. 2

[YBS05] YOSHIZAWA S., BELYAEV A., SEIDEL H.-P.: Fast and robust detection of crest lines on meshes. In *Proceedings of the 2005 ACM Symposium on Solid and Physical Modeling* (New York, NY, USA, 2005), SPM '05, ACM, pp. 227–232. 2, 6, 8, 9, 11

[ZDCJ17] ZHUANG Y., DOU H., CARR N., JU T.: Feature-aligned segmentation using correlation clustering. *Computational Visual Media* (2017), 1–14. 2

[ZHX*11] ZHANG L., HE Y., XIA J., XIE X., CHEN W.: Real-time shape illustration using laplacian lines. *IEEE Transactions on Visualization and Computer Graphics 17*, 7 (2011), 993–1006. 2