Graph-Based Shape Analysis for Heterogeneous Geometric Datasets: Similarity, Retrieval and Substructure Matching

Abstract

Practically all existing shape analysis and processing algorithms have been developed for specific geometric representations of 3D models. However, the product development process always involves a large number of often incompatible geometric representations tailored to specific computational tasks that take place during this process. Consequently, a substantial effort has been expended to develop robust geometric data translation and conversion algorithms, but the existing methods have well known limitations.

The Maximal Disjoint Ball Decomposition (MDBD) was recently defined as a unique and stable geometric construction and used to define universal shape descriptors based on MDBD's associated contact graph. In this paper, we demonstrate that by applying graph analysis tools to MDBD in conjunction with graph convolutional neural networks and graph kernels, one can effectively extract solid geometric features, such as design and manufacturing features, from geometric models regardless of their native geometric representation. We show that our representation-agnostic approach achieves comparable performance with state-of-the-art geometric processing methods on standard yet heterogeneous benchmark datasets while supporting all valid geometric representations.

1. Introduction

1.1. Motivation

An abundance of geometric representations have been defined and tailored to specific engineering applications over the last five decades. For example, NURBS [1] have been the golden standard as a primary boundary representation in all commercial mechanical CAD systems because they are able to represent "free-form" geometry with guaranteed differential properties while providing access to intuitive geometric controls structures to define and edit geometry; triangular meshes [2] are heavily used in graphics applications due to their simplicity and uniformity, which simplify the corresponding processing algorithms, as well as due to their amenability to massive software and hardware accelerations; structural finite element analysis [3] uses solid meshes of the geometry to approximate the solutions of the associated boundary value problems; subdivision surfaces [4] have found their footing in animation and visual effects over the past decades; point clouds are output by depth cameras and can discretely capture the geometry and, in the presence of appropriate processing algorithms, the topology of a physical artifact. Importantly, these and many other specialized representations have been developed for specific computational tasks needed during the product development process, which, in turn, produced a massive demand for the representation conversion services. Unfortunately, converting between existing geometric representations is far from being a trivial or solved problem, and every such conversion generates some information loss that is in general not well understood. Therefore, it is not surprising that current shape analysis systems have limited ability to handle models that natively exist in different representations having different levels of informational content. In fact, the theoretical and practical challenges of model interoperability prompted alternative proposals aimed at defining the queries used between software systems rather than the geometric data formats being exchanged [5].

On the other hand, deep learning methods have reached a high level of performance in 2D image analysis. However, applying deep learning methods to the analysis of 3D shapes is much more challenging, partly because there are significantly fewer sources of 3D models than 2D images available for training, and partly because of the exponential growth in the associated computational cost. For example, smart phones have democratized 2D imaging, while 3D models are still being created by skilled users. Furthermore, there is essentially one common and widely used representation for images, while there are numerous geometric representations used for 3D models. Consequently, it is not surprising that current data-driven methods can only provide limited shape analysis support for 3D shapes, which is why new paradigms for deep learning-based shape analysis approaches need to be developed.

A new geometric concept that can be used as a proxy geometric representation and computed for any other valid geometric representation was introduced in [6]. The concept relies on the concept of distance, which has to be supported by any and all valid geometric representations, to define a maximal disjoint ball decomposition (MDBD) for a given 3D solid domain. Informally, MDBD recursively packs the largest spheres inscribed in the domain, is unique up to isometric transformations, and is stable against small boundary perturbations, which makes it ideally suited to shape similarity, segmentation and processing, and particularly for datasets containing heterogeneous geometric representations.

MDBD captures the geometry and topology of a given domain through the tangency patterns of the spheres in the maximal ball decomposition as well as by their radii. The tangency pattern can be described by a graph whose nodes are the centers of the spheres, and whose edges connect the centers of mutually tangent spheres. Given the maximality constraint applied on the spheres and the hierarchical nature of the decomposition, MDBD captures the geometry and topology of a domain using a sparse weighted graph, which tends to be more compact in terms of the size of the respective data structures than a typical mesh, point cloud, or volumetric representation of the same domain. In this paper, we show that MDBD can be used not only to define representation agnostic shape descriptors, but also to perform various shape analysis tasks by using graph theory and graph analysis tools.

1.2. Shape Descriptors

Shape descriptors can be thought of as mathematical abstractions defined on geometric representations that capture geometric characteristics of shapes and constitute the foundation of most existing shape analysis methods. Measuring the similarity of shapes is one of the most common tasks in shape analysis, which is usually defined in terms of some notion of "distance" between the corresponding shape descriptors.

Many shape descriptors have been defined for this purpose. In [7], by approximating a 3D model with a union of spheres (UoS) hierarchies along with the zero alpha-shape of the UoS [8], feature matching between two models could be reduced to a predefined distance between their corresponding spheres. The Heat Kernel Signature (HKS) [9] and its variants [10] use a pseudo-heat diffusion defined on the model, and capture local shape features of a mesh or a point cloud model [11] up to isometry. Due to its roots in heat diffusion, the HKS is stable against boundary noise. At the same time, HKS relies on an appropriate definition of the Laplacian, which is representation dependent. Even for a given geometric representation, there are infinitely many ways to define discrete Laplacians [12], so it is not surprising that HKS has not been used to measure similarity across different geometric representations. Area Projection Transform (APT) [13] measures the likelihood of points in 3D space to be the center of radial symmetry at selected scales. APT detects salient regions, such as almost spherical and cylindrical surfaces, and the APT histogram across multiple scales can be used as shape descriptors for non-rigid shape retrieval.

Furthermore, machine learning methods can be and have been trained to classify shapes based on a given shape similarity metric. Deep neural networks (DNNs) have been used to learn shape descriptors from volumetric datasets. A large number of convolutional architectures have been designed for binary and probability distribution 3D grids [14, 15, 16, 17, 18]. However, these networks can currently only handle low-resolution models, such as $32 \times 32 \times 32$, due to the limited supply of available data and the limit imposed on the computational resources by the cubic increase of the size of the input data with the increase in the resolution. Another class of approaches classify shapes by using 2D projections coupled with 2D convolutional neural networks operating on the corresponding 2D images [19, 20, 21, 22, 23, 24]. Although these methods achieve a high classification accuracy, they have not been shown to capture 3D topological features or be capable of performing feature recognition. Furthermore, we do not know what directions and how many projections are needed to achieve reasonable classification accuracy. Some DNNs have been shown to perform well when dealing with specific classes of models, such as models of human body [25] and hand [26]. However, these methods do not generalize due to their assumptions used in localizing landmarks.

1.3. Related Work in Graph CNN and Graph Kernels

Graph convolutional neural networks (Graph CNNs), as a generalization of CNNs from regular grids to arbitrary structures, have shown their potential to extract features from structured data. Many graph CNNs have been proposed with various localized graph convolutional filters (convolution operators), which define how the node features aggregate in a neighborhood. SplineCNN [27] applies B-spline bases that are parameterized by a set of trainable control values as convolution kernels, and the neighborhood in the spatial domain is delimited by utilizing the local support of B-splines. Topology Adaptive Graph Convolutional Networks (TAGCN) [28] define the convolution operator as the matrix-vector product between a polynomial of the normalized adjacency matrix of the graph and the node feature vector. TAGCN achieves a better performance than spectral graph CNNs, and has a lower computational complexity. A detailed review of Graph CNNs can be found in [29].

Graph kernels, as a special class of kernel methods, measure the similarity between two graphs and must be symmetric and positive semi-definite. A good recent review of graph kernels appears in [30]. R-convolution kernels are a family of kernels that measure the similarity between objects through their substructures, which include paths, sub-trees, cyclic patterns, and subgraphs. Random Walk kernels [31] and Shortest-Path [32] kernels focus on graph paths. Graphlets [33] characterize a graph by counting the number of specific small sub-graphs. R-convolution kernels generally have very high time complexities. For example, the time complexity of random walk kernel is $O(n^3)$, which can be reduced to $O(n^2)$ with approximate algorithms [34], where n is the number of nodes in a graph.

The Weisfeiler-Lehman Graph Kernels [35] form another family of graph kernels based on the Weisfeiler-Lehman test of graph isomorphism. They have a smaller computation complexity than R-convolution kernels in general, making them a better choice for large graphs. However, most of these graph kernels do not directly support continuous node attributes such as the sphere radii. Togninalli et al. [36] proposed a Weisfeiler–Lehman-inspired embedding scheme for graphs with continuous node attributes and defined the *graph Wasserstein distance* to measure the similarity of graphs.

1.4. Contributions

In this paper, we show that the Maximal Disjoint Ball Decomposition can be effectively used in conjunction with graph analysis tools for shape similarity and geometric processing and analysis, and particularly for datasets containing heterogeneous geometric representations. Since MDBD only requires the ability to compute distance, it can be used with any valid geometric representation and across multiple such representations.

First, we show that MDBD captures global shape features, and we design a DNN based on graph CNNs whose input is precisely the MDBD tangency/contact graph. We test our network with 3 well-known graph convolution methods and show that by exploiting the MDBD tangency graph, the established graph convolution algorithms achieve a comparable classification accuracy with that of other leading classifiers on a widely used benchmark dataset, but with a significantly lower number of parameters. Moreover, by using the MDBD contact graph for global shape features, one can perform shape classification on datasets that use any number of valid geometric representations.

Furthermore, we propose a graph kernel that exploits the hierarchical structure of MDBD and has linear time complexity. We show that this graph kernel can be effectively used to measure shape similarity of models having distinct geometric representations. At the same time, by taking advantage of neighborhood sub-graph matching, the new graph kernel can be used to perform sub-structure matching, which leads to novel, powerful, and *representation-agnostic feature recognition algorithms* that can hardly be built with existing shape descriptors.

2. Preliminaries

2.1. Maximal Disjoint Ball Decomposition

Let Ω be a compact regular semi-analytic subset of the Euclidean 3-space, also known as an *r-set* [37]. Its corresponding

Maximal Disjoint Ball Decomposition (MDBD) is denoted by M_Ω, and we follow the same notation as in [6].

Intuitively, the decomposition process, illustrated in Figure 1, recursively adds the maximal d-dimensional closed ball $b_i \in \Omega$ such that $\mathbf{i}b_i \bigcap \mathbf{i}b_j = \emptyset, \forall i \neq j$ and $\mathbf{i}b_i \bigcap \partial\Omega = \emptyset$, where $\mathbf{i}X$ denotes the interior of a set X, ∂X its boundary, and b_i is the closed maximal ball inserted at step i. Computing M_Ω starts by first computing the Signed Distance Field (SDF) of Ω, which needs to be computed only once. This field is then updated efficiently as maximal balls are being added to the remaining subset of the domain at every step of the hierarchy. The geometric information queries needed during decomposition are distance computations, and the ability to perform point membership classification. Thus, we can construct MDBD for any valid geometric representation and define *universal shape descriptors* that handle distinct geometric representations. We use HAVOC3D[38] to compute the unsigned distance field, and then use PMC to compute the sign and convert the unsigned distance field into an SDF. Note that this implementation must handle the various geometric representations that are being considered. In principle, one can use any of the existing algorithms that compute the SDF [39, 40, 41].

Given a domain Ω, the maximal ball at step i is denoted as $b_i(\mathbf{c}_i, r_i)$, where $\mathbf{c}_i$ is the center of the ball, and r_i its radius. The SDF at step i is defined as

$$\mathcal{D}_{\Omega_i}(\mathbf{p}) = \text{sgn}(\mathbf{p}) \min_{\mathbf{q}\in\partial\Omega_i} \|\mathbf{p}-\mathbf{q}\|_2, \quad \text{and}$$

$$\Omega_i = \Omega -^* (\cup^*_{j\in[0,i-1]} b_j),$$

where $\mathbf{p}$ and $\mathbf{q}$ are position vectors of two points, sgn is the usual sign function with binary values $\{0, 1\}$, and '$*$' implies the regularized versions of the standard Boolean operations [37]. Consequently, $\mathbf{c}_i$ and r_i are calculated as

$$\mathbf{c}_i = \arg\max_{\mathbf{p}\in\Omega_i} \mathcal{D}_{\Omega_i}(\mathbf{p}),$$

$$r_i = \mathcal{D}_{\Omega_i}(\mathbf{c}_i).$$

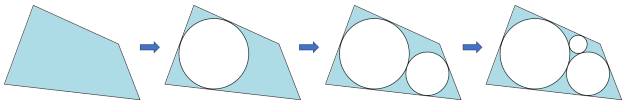

Figure 1: Recursive generation of the Maximal Disjoint Ball Decomposition.

2.2. Contact Graph Definition and Notation

We denote a directed graph by $G = \{V, E\}$, where $V = \{v_1, v_2, ..., v_n\}$ is an ordered set of n vertices and E is the edge set comprised of directed edges. As usual, the function $f : E \rightarrow V \times V$, with $f(e_k) = (v_i, v_j)$ provides the direction of travel for edge e_k from v_i to v_j but not vice versa. Observe that self-connected vertices are not allowed, that is $(v_i, v_i) \notin E$. For an unweighted graph, its adjacency matrix is defined as an $n \times n$ matrix $\tilde{A}$ with $\tilde{A}_{ij} = 1$ if $(v_i, v_j) \in E$ and 0 otherwise. For a weighted graph, $\tilde{A}_{ij} = w_{ji}$ where w_{ji} is the weight assigned to edge $(v_j, v_i) \in E$.

A graph $G' = \{V', E'\}$ is a subgraph of G if $V' \subseteq V$ and $E' \subseteq E$. A walk, or path, in G is a sequence of edges in E that connect a sequence of vertices in V; it is a subgraph of G by definition; and may form a cycle in G. The induced-subgraph on a set of vertices $S \subseteq V$ is a subgraph of G whose vertex set is S, and whose edge set consists of every edge in E whose both endpoints are in S, which is denoted by $G[S]$.

The distance between two vertices v and u of a directed graph G is measured in terms of the smallest number of hops from the source vertex u to the target vertex v. The neighborhood of v in range K is the set of vertices of G having a distance from v equal to or smaller than K, and is denoted by $N(v, K)$. The neighborhood $N(v, K)$ induces a subgraph of G of size K.

By definition, MDBD contains infinitely many spheres, and therefore one operates in practice with truncated decompositions. A truncated MDBD of domain Ω comprised of the first n maximal spheres is denoted as $M_\Omega(n) := \{b_1, b_2, ..., b_n\}$. The tangency pattern of its maximal balls can be described by a graph $G = \{V, E\}$, called the contact graph of MDBD, where the vertex set $V = \{v_1, v_2, ..., v_n\}$ corresponds to the n balls in M_Ω and edge set E captures the mutual tangency of the respective balls. Thus, if ball b_i is tangent to ball b_j, we have $(v_i, v_j) \in E$. For $v_i \in V$, we have the corresponding ball $b_i = \{\mathbf{c}_i, r_i\}$, where $\mathbf{c}_i$ and r_i are the center and radius of the ball, respectively. $V = \{v_1, v_2, ..., v_n\}$ is ordered based on the radii of the corresponding balls, that i,s $r_1 \geq r_2 ... \geq r_n$. Hence, $\mathbf{r}_n = \{r_1, r_2, ..., r_n\}$ is the *radius list* of the truncated decomposition $M_\Omega(n)$.

Weights can be assigned to each node and edge as either discrete labels or continuous variables. MDBD will assume node and edge attributes that capture the specific geometric information of the maximal balls, and can vary depending on the domain of the application.

3. Shape Analysis Applications

In this section we demonstrate that MDBD in conjunction with graph-based methods can be effectively used to perform three specific yet ubiquitous shape analysis tasks: shape classification, articulated shape retrieval, and substructure matching, or feature recognition, among objects using valid geometric representations.

Our method can analyze 3D models that use any valid geometric representation within a uniform computational framework, and with competitive classification performance. At the same time, the MDBD contact graph possesses unique characteristics that make it ideal for shape analysis and classification, such as the geometry and topology-aware hierarchical structure induced by MDBD. As shown in the experiments detailed in this section, this hierarchical structure allows MDBD-based shape analysis methods to focus on the appropriate and meaningful model substructures and do not incorrectly favor the peripheral areas of the models, which is one key limitation of the existing methods. Furthermore, we expect that the attention mechanism [42] developed in the context of natural language processing can further improve the classification accuracy of the MDBD-based shape analysis methods.

3.1. Shape Classification with graph CNNs

As a special case of shape similarity, shape classification is one of the most common shape analysis tasks, and many specialized methods have been proposed to handle classification of models having a single geometric representation. In this section, we show that the MDBD graph can be directly used for shape classification of models using any valid geometric representation in conjunction with graph Convolutional Neural Networks. To this end, we integrate 3 well known graph convolution methods within the same straightforward neural network architecture, as described below, and we compare the classification performance with that of other leading classifiers.

Model. Unlike traditional convolutions on grids integrating the values of a rectangular window around each pixel, the convolutions on graphs combine vertex and edge features from a graph neighborhood within a prescribed range.

For comparison purposes, we implemented several known graph CNNs in Pytorch Geometric [43], and the results of the comparisons are described below. The architecture of our neural network is shown in Figure 3, including the dimensions of input and output channels displayed as numbers in parentheses, which also represent the dimensions of node attribute at each stage.

Data. We evaluate our neural network on the ModelNet10 dataset [14], which consists of 10 categories of 3D mesh models, with 3,991 models for training and 908 models for testing. We eliminate from this dataset the models that are not solid models (closed, regular, semi-analytic sets, or *r-sets* [37]). Thus, the resulting dataset contains 3,957 models that we use for training and 905 models that we use for testing. Some samples of the dataset are shown in Fig 2.

We built the graph of MDBD for each 3D mesh model as follows. The node attribute for vertex v_i is assigned to be $[\mathbf{c}_i, r_i/r_1]$ where r_i/r_1 is the radius of the i^{th} ball normalized by the radius of the largest ball in the decomposition. The edge attribute for edge (v_i, v_j) is assigned to be $1/deg(v_i)$. The number n of vertices of each graph is chosen to be $n = 512$. We use the Adam optimizer and cross-entropy loss for training, and a warm-up applied at the beginning of the training, followed by a cosine annealing learning rate schedule. Batch normalization and dropout are used to avoid over-fitting. We exploited the GPU architecture to accelerate the training-related computations. The training process is implemented in Pytorch.

Details of training performance. The CPU used for the training is Intel(R) Xeon(R) Gold 5218 @ 2.30GHz. The GPU is Quadro RTX 6000. The time cost of one batch is about $0.01s$. The time cost of one epoch is about $9s$. The number of epochs used for convergence is 200, and the memory used during the training process is 93.8 MB.

Results. As shown in Table 1, the three graph CNNs implemented in our simple architecture and processing the MDBD contact graph achieve comparable accuracy with other leading neural networks that operate directly on the 3D models. We note that our neural networks implemented with graph CNNs have significantly fewer parameters than other leading neural networks, and that TAGConv has the lowest number of parameters while achieving a similar classification accuracy with other graph CNNs. Moreover, the existing methods for shape classification assume and are tailored to specific geometric representations and have not been shown to support datasets with multiple geometric representations.

3.2. Articulated Shape Retrieval

The task of shape retrieval is to find the most similar models to a given query model within a model database, and relies on a 3D model similarity metric. There are various established techniques that have been developed based on various shape descriptors, as summarized in section 1.2, and for single and specific geometric representations. However, these methods do not generalize to other geometric representations without requiring a representation conversion, and may not handle articulated models because the metrics may not maintain consistency for models in different poses. This, in turn, suggests that these metrics may fail to determine similarity of 3D engineering models comprised of identical features, but with inconsistent spatial location.

More recent approaches to pose estimation introduce an intermediate representation, such as the Articulation-aware Normalized Coordinate Space Hierarchy (ANCSH) [50], which is a canonical representation for different articulated objects in a given category that captures the normalized poses and scales, orientations, and the states of the kinematic joints. The method in [50] has been specifically designed for point clouds representing articulated objects that contain revolute and prismatic lower kinematic pairs, but its generalization to the analysis of engineering models has not been documented.

On the other hand, the MDBD subgraphs corresponding to individual features are well suited for establishing a generic shape correspondence of articulated and/or feature-based models, and allow us to incorporate high-level shape semantics into the shape analysis pipeline.

3.2.1. A New and Efficient Graph Kernel Based on Hierarchical Random Walks

In this section we show how one can measure the similarity between MDBD graphs of two geometric models by establishing an effective graph kernel based on random walks.

Random walk based kernels [31] measure the similarity of two graphs by counting the same walks shared by the two graphs. These kernels often assume that node attributes of the vertices are discrete labels, which is not appropriate for our task because the ball radii are real numbers. We designed a modified graph kernel based on random walks on a graph that supports continuous node attributes and accommodates the hierarchical structure of MDBD.

Random walks on graphs are special cases of Markov Chains that characterize system transitions from one state to another by probabilistic rules. Here, the system state is the node attribute distribution. For a graph $G = \{V, E\}$ with $|V| = n$ and $|E| = m$, let the column vector $\mathbf{p}^t \in \mathbb{R}^n$ denote the node attribute distribution at time step t.

For an edge $(v_i, v_j) \in E$, we define the corresponding edge weight as $\frac{1}{deg(v_i)}$, where $deg(v_i)$ is the number of vertices directly connected to v_i or, equivalently, the number of edges incident at v_i. Consequently, the probability of a hop happening from v_i to v_j is equally shared by the neighbors of v_i. The transition matrix for the random walk is defined as the weighted adjacency matrix of G, $\tilde{A}$, whose elements are the weights $w_{ij} = \frac{1}{deg(v_i)}, i \neq j$ representing the probability of a hop from v_i to v_j, and $w_{ii} = 0$. The node attribute distribution is iteratively reassigned during the random walk according to

$$\mathbf{p}^t = \tilde{A}\mathbf{p}^{t-1}.$$

Hence, we can construct the feature vector for the graph directly from the results of the random walk process.

As mentioned above, the maximal balls of the MDBD of a domain are ordered according to their radii. Consequently, the larger balls will capture the larger geometric and topological attributes of the domain, while the smaller balls will tend to capture the smaller details of the shape. Thus, we select the first n_r balls as the representatives for the entire group, and their node attributes represent the system state. For a representative node, the weighted average of its node attribute over t time steps is defined as:

$$P_i = \frac{1}{t}\sum_{k=1}^{t} w_t^k p_i^k,$$

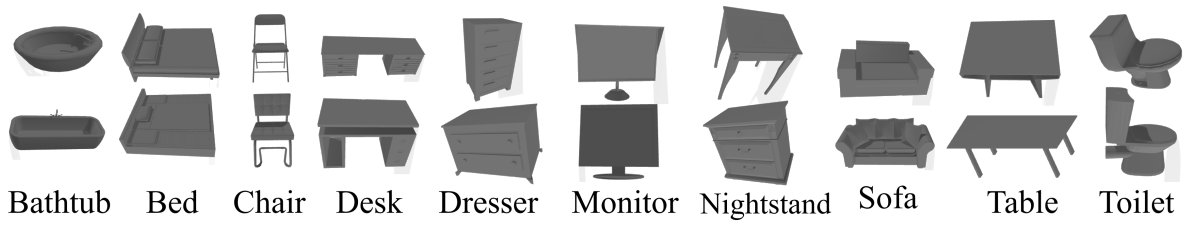

Figure 2: Model samples from ModelNet10 dataset [14], which consists of 10 categories of 3D mesh models.

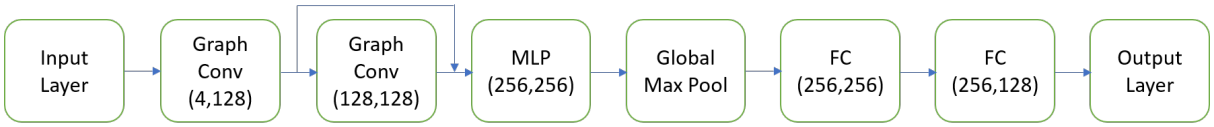

Figure 3: The architecture of our neural network: each vertex initially has four node attributes, namely the center coordinates and the radius of the corresponding ball; the two graph CNN layers extract features directly from the input layer of graph data and have 128 channels on their output; the multilayer perceptron layer concatenates and merges the node attributes from the graph CNNs; a global maximum pooling finds the largest value in each channel over all the nodes of the graph resulting in a shape feature vector with 256 elements; finally two fully-connected layers reduce the dimension of the feature vector and this dimension captures the number of classes.

Network	Accuracy (%)	Data Type	# of Para. (Million)
3D ShapeNet [14]	83.5	Voxel	≈ 38
3D-GAN [44]	91	Voxel	≈ 11
VoxNet [45]	92	Voxel	≈ 0.92
VRN [46]	91.33	Voxel	≈ 18
LP-3DCNN [47]	93.76	Voxel	≈ 2
FusionNet [48]	93.11	Voxel	≈ 118
Our NN with TAGConv	90.90	Graph	≈ 0.24
Our NN with SplineConv	90.68	Graph	≈ 0.64
Our NN with ARMAConv	91.34	Graph	≈ 0.57

Table 1: Comparison of the classification results of our neural network implemented with TAGConv [28], SplineConv [27], and ARMAConv [49], respectively, compared with other leading classification algorithms for models from the ModelNet10 database [14]. The 3 graph CNN algorithms perform the classification directly on the MDBD contact graph.

where w_t is set as 0.9 in our experiments. The feature vector for the graph is defined as the concatenation of the initial node attributes and their weighted averages of the representative nodes:

$$\mathbf{f}(G) = \{p_1^0, p_2^0, ..., p_{n_r}^0, P_1, P_2, ...P_{n_r}\} = \{\mathbf{p}_{n_r}^0(G), \mathbf{P}_{n_r}(G)\}. \quad (1)$$

Given two graphs G_1 and G_2, the distance between the feature vectors is defined as the weighted sum of the difference between their components:

$$d(G_1, G_2) = \sum_{i=1}^{n_r} \lambda_a^i |p_i^0(G1) - p_i^0(G2)| + \sum_{i=1}^{n_r} \lambda_b^i |P_i(G1) - P_i(G2)|, \quad (2)$$

where the parameters λ_a, λ_b are set as $\lambda_a = \lambda_b = 0.75$ in our experiments. Hence, the kernel of the two graph is defined as

$$k(G_1, G_2) = e^{-\lambda d(G_1, G2)}. \quad (3)$$

Computational Complexity. For the feature vector construction, the time complexity is $O(t(n + 2m))$, where t is the total time steps, n is the number of nodes, m is the number of edges. For the distance calculation, the time complexity is $O(n_r)$, where n_r is the number of representative nodes. Therefore, the total time complexity is $O(t(n+2m)+n_r)$, which is much more efficient than the complexity of exiting graph kernels ($O(n^3)$). This attractive computational complexity, in turn, makes our method to be the first practical method for shape analysis of complex models.

3.2.2. Examples: Articulated Shape Retrieval

We evaluate the graph kernel introduced in equation (3) on the SHREC'14 [51] synthetic dataset, which consists of 300 mesh models, including 15 human shapes, each having 20 poses. We focus on the task of measuring the similarity of each model against all the other models and returning a list of models ordered by the similarities with the query model in decreasing sequence. The results are evaluated by various statistical methods, including nearest neighbor (NN), first tier (1-T), second tier (2-T), and discounted cumulative gain (DCG). A discussion of all these methods can be found in [52].

We first compute the MDBD of the models from the dataset, and generate the weighted contact graph. The number of representative vertices is selected to be $n_r = 512$, and the radii of the balls are assigned as the node features of the graphs. Figure 4 shows some examples and their corresponding MDBDs in the dataset. We use different graph kernels to compute the similarities between models, and compare the performance of our designed graph kernel in 3.2.1 with other graph kernels, including Shortest-Path kernel, Weisfeiler-Lehman Subtree Kernel and Pyramid Match kernel implemented in the Python package GraKel [53]. Observe that all these graph kernels only work with discrete node attributes. We round the node attributes to evenly spaced numbers by $round(N_d \times \text{node\_attribute})/N_d$, where $N_d = 25$ in this experiment.

The evaluation results are shown in Table 2. In principle, we

Figure 4: Samples of SHREC'14 [51] (columns 1-5) and their corresponding MDBDs (columns 6-10). Each row shows the same shape of a human model in different poses. The ball distribution patterns of the MDBD for the same model are similar across different poses.

expect the MDBD subgraphs corresponding to individual features of the articulated model to remain the same as the model goes through different poses. However, even though the approximation errors in computing MDBD induce some modifications in the corresponding graphs, our method still possesses an accuracy that is among the best compared with the accuracies obtained with other shape descriptors.

Our experiments also show that our graph kernel achieves better performance than leading algorithms using various graph kernels for the retrieval of articulated shapes, as described below. We hypothesize that this is so because most existing kernels support only discrete node and edge labels of graphs, and do not rely on geometrically meaningful hierarchical structures. On the other hand, our kernel exploits the hierarchical structure of MDBD by assigning weights to the nodes according to their radii.

Moreover, to demonstrate that our method results in consistent results across different geometric representations, we run another experiment on point cloud models. We generate point cloud models from the corresponding mesh models form SHREC'14 by randomly selecting about 20% of the mesh vertices, and compute the MDBD of these point clouds. For the same shape/human model, the average similarity of the MDBD graphs computed from the mesh and point cloud models over the dataset is **0.996**. Table 3 shows the similarity results for articulated shape retrieval on meshes, point clouds, and an even mixture of the two by using the proposed graph kernel defined in equation (3).

Method	NN	1-T	2-T	DCG
HKS-TS [51]	0.467	0.476	0.743	0.729
SIHKS-H [10]	0.427	0.206	0.332	0.562
APT [13]	0.97	0.733	0.927	0.936
Spectral Geom. [54]	0.993	0.832	0.971	0.971
Shortest-Path Kernel [32]	0.323	0.237	0.394	0.580
WL-Subtree Kernel [35]	0.297	0.228	0.373	0.554
Pyramid-Match Kernel [55]	0.290	0.226	0.408	0.564
Our kernel	0.937	0.775	0.953	0.945

Table 2: Evaluation results on SHREC'14 synthetic dataset by nearest neighbor (NN), first tier (1-T), second tier (2-T), and discounted cumulative gain (DCG).

Data Format	NN	1-T	2-T	DCG
Mesh	0.937	0.775	0.953	0.945
Point Cloud	0.913	0.751	0.948	0.932
Mixture	0.927	0.723	0.938	0.926

Table 3: Shape retrieval performance of our graph kernel used for the graphs of MDBD calculated from three kinds of dataset: meshes, point clouds, and even mixture of meshes and point clouds.

3.3. Substructure Matching

The methods developed for shape retrieval and classification extract global shape features from 3D models. However, engineers are often interested in local shape features of 3D models because the functionalities of a mechanical part are usually revealed by substructure features. For example, in engineering design, one might seek to identify high-level machining or design features (e.g. slots, holes and ribs) in a database for design reuse [56, 57], cost estimation [58], manufacturing process reuse [59], manufacturability analysis [60]. In the context of graph-based shape analysis, the identification of engineering relevant features involves graph substructure matching.

Previous shape analysis methods have been developed individually for meshes, point clouds, or volumetric representations, but they do not employ an intrinsic partitioning strategy of the domain that is needed to compare the similarity between graph sub-structures. There are some existing methods that could extract node features, such as Heat Kernel Signature-based methods that can capture surface curvature, but methods that group graph nodes into a feature-relevant substructure of the 3D model have not been documented.

3.3.1. Neighborhood Subgraph Matching

We observe that feature-based object segmentation can be recast into a graph segmentation problem by using the maximal disjoint ball decomposition. The MDBD captures the geometric and topological information of the 3D model via the contact graph and the radii of the associated maximal balls. Moreover, the larger balls capture the larger/global features of the domain, while the smaller balls encapsulate the geometric details, as can be seen for example in Figure 4. From a graph perspective, the structural features reside in the subgraphs. While there are numerous ways to divide a graph into subgraphs, we resort to neighborhood subgraphs 2.2 due to their attractive properties, including the fact that they capture the 'local' geometric information of the protruding features.

Given two domains, Ω_1 and Ω_2, we find their most similar sub-structures by exploring their neighborhood subgraphs, and by finding the subgraph pairs that have the highest value of similarity according to the graph kernel defined in equation (3). The process is summarized in the pseudo-code shown in Algorithm 1, where $G[N(v_i, K)]$ is the subgraph of G induces by neighborhood $N(v_i, K)$

Algorithm 1: Find the most similar neighborhood subgraphs of two MDBD graphs G_1 and G_2

Input: Two graphs $G_1(V_1, E_1)$, $G_2(V_2, E_2)$, range K
Output: The most similar neighborhood subgraph pairs (S_1, S_2)
Initialization
for $i \leftarrow$ *1 to* $|V_1|$ **do**
  for $j \leftarrow$ *1 to* $|V_2|$ **do**
    similar_matrix[i,j]
    $\leftarrow k(G_1[N_1(v_i, K)], G_2[N_2(v_j, K)])$
  end
end
$(i, j) \leftarrow argmax(similar\_matrix)$
$(S_1, S_2) \leftarrow (G_1[N_1(v_i, K)], G_2[N_2(v_j, K)])$

3.3.2. Examples: Substructure Matching

The hierarchical nature of MDBD induces a meaningful and convenient partition strategy of the original domain in terms of subgraphs. In practice, engineering features can be abstracted by MDBD subgraphs and, therefore, their similarity can be measured via graph kernels. The scale of the features that we want to look at can be easily tweaked by changing the neighborhood range K of the neighborhood subgraph. Since substructure matching is an area that has been hardly explored before in this context, there is no benchmark dataset for evaluation and comparison. We employ neighborhood subgraph matching with the graph kernel defined in 3.2.1 to show how to use the MDBD induced subgraph matching to perform the substructure matching for articulated models. In our experiments we chose $K = 2$, and the number of representative nodes $n_r = 10$.

In Figure 5, we show the results of sub-structure matching for a group of articulated human models from SHREC'14 dataset represented as meshes and point clouds. For the human models in Figure 5 (a), we first identify the neighborhood subgraphs that correspond to the head, an arm, and a calf, respectively. We follow that by searching the graphs of the other human models to identify the most similar neighborhood subgraphs of their MDBDs against the query subgraphs. The figure illustrates the effectiveness of our method in performing substructure matching on articulated models. Note the discrepancies for the matched subgraphs for meshes and point clouds, which originates from the reflection invariance of MDBD. In other words, correctly processing the symmetries, such as those arising between the left and right arm, would require additional information - see also [61].

Some DNNs, such as DensePose [25], can partition the domain and recognize individual parts of human bodies. However, these methods rely on the concept of localized shape landmarks, which does not easily generalize to arbitrary shapes. On the other hand, Figure 6 presents the results of performing sub-structure matching with our method on other similar models from COSEG [62] and MCB [63] datasets. It clearly illustrates the fact that our method can detect similar parts among models belonging to the same shape class.

4. Conclusions

In this paper, we showed that powerful shape analysis methods can be developed by exploiting the geometric and topological information stored in the contact graph induced by the maximal disjoint ball decomposition of a domain. Because MDBD only relies on distance computations, which must be supported by any and all valid geometric representations, it can be used to develop a uniform shape analysis framework that can handle any valid 3D geometric models.

First, we showed that the contact graph of the maximal disjoint ball decomposition can be directly used for shape classification by using graph CNNs. By implementing several leading graph convolutions within a simple neural network architecture, we showed that the resulting graph CNNs achieve comparable classification accuracy with that of other neural networks, while using a significantly lower number of parameters. This, in turn, will directly impact the associated computational resources, including a reduction of the number of training cases required and of the demands on the complexity of the required hardware. Consequently, the MDBD graphs could drive a lightweight AI module for 3D shape recognition, such as those that would run on smart devices.

However, shape classification requires only a global measure of similarity and the existing methods cannot be immediately

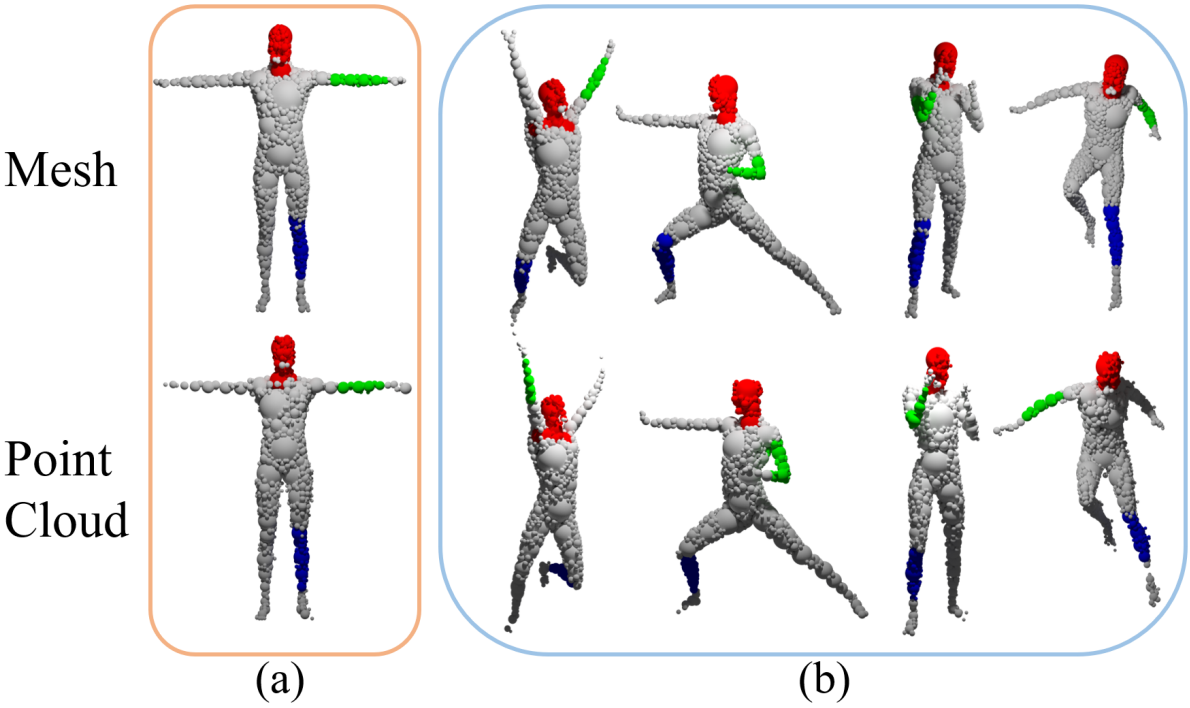

Figure 5: Sub-structure matching for articulated models represented as meshes and point clouds. Models in (a) highlight the substructures corresponding to the head, arm and lower leg. Our method finds the most similar substructures of articulated models in (b) that are matching with the target structures in (a). Observe the consistency of the results for meshes and point clouds.

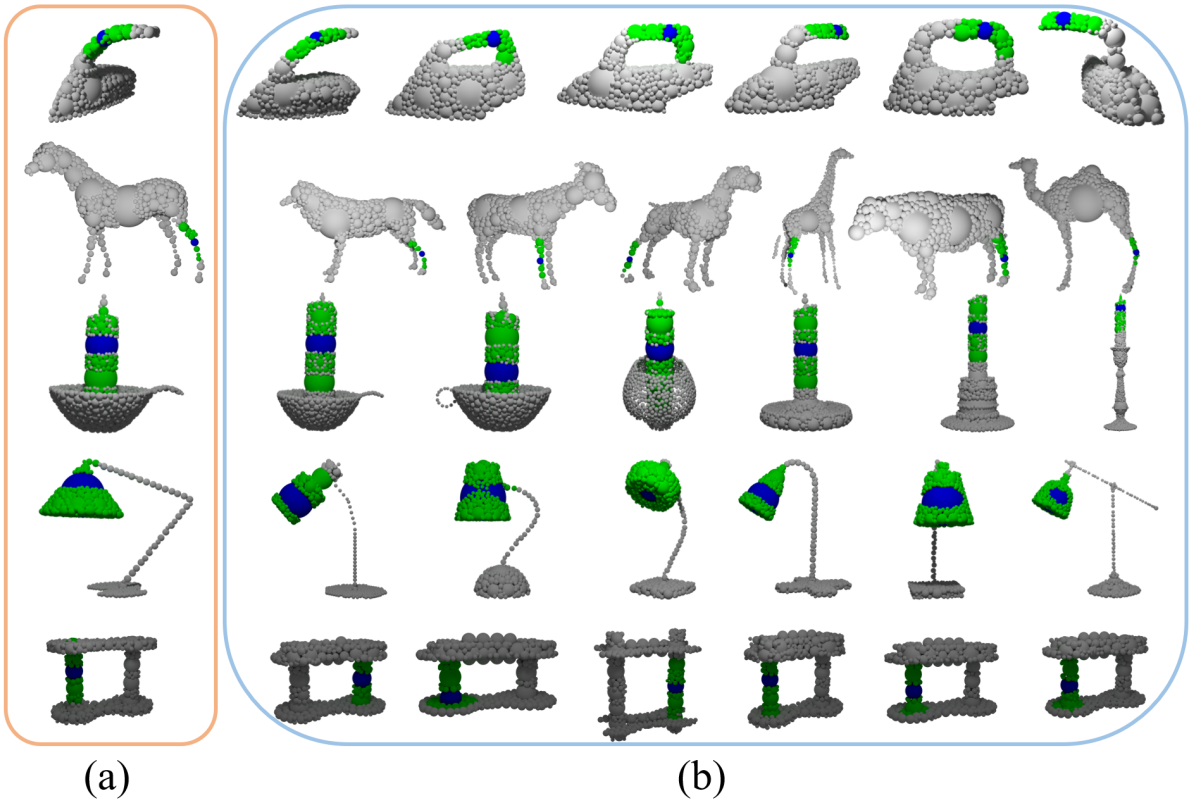

Figure 6: Sub-structure matching among similar models. Models in (a) contain the target substructures; images (b) display the matching substructures for objects in the database. The first four rows of models, namely the irons, four leg animals, candelabra, and lamps, are from COSEG [62]. The last row of models, i.e., the chain drives, are from MCB [63].

used to perform retrieval of articulated shapes or substructure matching/feature recognition that need local similarity measures. On the other hand, the MDBD contact graph encodes the substructure local information in a hierarchical data structure. Thus, geometric features can be detected by neighborhood subgraphs, so their similarity can be measured via graph kernels. To this end, we proposed a new graph kernel based on random walks that exploits the properties of MDBD. The linear time complexity of our kernel makes it much more appealing for shape analysis than existing graph kernels, whose time complexities are usually quadratic or higher. Furthermore, we showed that the versatile form of the MDBD graph provides both global and local features for shape classification, retrieval, and segmentation for any and all valid geometric representations, which is a unique and novel feature among the existing shape descriptors. Specifically, we showed that the proposed graph kernel achieves competitive results for articulated shape retrieval and substructure matching on meshes, point clouds, and an even mixture of the two.

The work described in this paper has not yet taken advantage of the full capabilities of graph analysis tools for substructure matching because we only used neighborhood subgraphs. This type of subgraphs form an efficient yet simple method for graph segmentation, but limits the types of substructures or features that can be recognized. For example, these neighborhood graphs can be used to detect convex features, such as protrusion features, but are not as effective at detecting concave features such as holes. One way to extend the utility of the neighborhood graphs to concave features is to observe that the concave features of a model generate convex features of its complement. This, in turn, would immediately extend the applicability of these neighborhood subgraphs to concave features. However, a detailed treatment of the associated issues and of alternative graph-based methods is beyond the scope of this paper.

References

[1] L. Piegl, On NURBS: a survey, IEEE Computer Graphics and Applications 11 (1) (1991) 55–71.

[2] J. R. Shewchuk, Delaunay refinement algorithms for triangular mesh generation, Computational geometry 22 (1-3) (2002) 21–74.

[3] T. J. Hughes, The finite element method: linear static and dynamic finite element analysis, Courier Corporation, 2012.

[4] T. DeRose, M. Kass, T. Truong, Subdivision surfaces in character animation, in: Proceedings of the 25th annual conference on Computer graphics and interactive techniques, 1998, pp. 85–94.

[5] C. Hoffmann, V. Shapiro, V. Srinivasan, Geometric interoperability via queries, Computer-Aided Design 46 (2014) 148–159.

[6] J. Chen, H. T. Ilieş, Maximal disjoint ball decompositions for shape modeling and analysis, Computer-Aided Design 126 (2020) 102850.

[7] A. Sharf, A. Shamir, Feature-sensitive 3D shape matching, in: Proceedings Computer Graphics International, 2004., IEEE, 2004, pp. 596–599.

[8] H. Edelsbrunner, The union of balls and its dual shape, in: Proceedings of the ninth annual symposium on Computational geometry, 1993, pp. 218–231.

[9] J. Sun, M. Ovsjanikov, L. Guibas, A concise and provably informative multi-scale signature based on heat diffusion, in: Computer graphics forum, Vol. 28, Wiley Online Library, 2009, pp. 1383–1392.

[10] M. M. Bronstein, I. Kokkinos, Scale-invariant heat kernel signatures for non-rigid shape recognition, in: 2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, IEEE, 2010, pp. 1704–1711.

[11] R. M. Williams, H. T. Ilieş, Practical shape analysis and segmentation methods for point cloud models, Computer Aided Geometric Design 67 (2018) 97–120.

[12] M. Wardetzky, S. Mathur, F. Kälberer, E. Grinspun, Discrete Laplace operators: No free lunch, in: Symposium on Geometry processing, Aire-la-Ville, Switzerland, 2007, pp. 33–37.

[13] A. Giachetti, C. Lovato, Radial symmetry detection and shape characterization with the multiscale area projection transform, in: Computer Graphics Forum, Vol. 31, Wiley Online Library, 2012, pp. 1669–1678.

[14] Z. Wu, S. Song, A. Khosla, F. Yu, L. Zhang, X. Tang, J. Xiao, 3D ShapeNets: A deep representation for volumetric shapes, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2015, pp. 1912–1920.

[15] X. Xu, S. Todorovic, Beam search for learning a deep convolutional neural network of 3D shapes, in: 2016 23rd International Conference on Pattern Recognition (ICPR), IEEE, 2016, pp. 3506–3511.

[16] M. Ren, L. Niu, Y. Fang, 3D-A-Nets: 3D deep dense descriptor for volumetric shapes with adversarial networks, arXiv preprint arXiv:1711.10108.

[17] L. Minto, P. Zanuttigh, G. Pagnutti, Deep learning for 3D shape classification based on volumetric density and surface approximation clues., in: VISIGRAPP (5: VISAPP), 2018, pp. 317–324.

[18] C. Ma, Y. Guo, Y. Lei, W. An, Binary volumetric convolutional neural networks for 3-D object recognition, IEEE Transactions on Instrumentation and Measurement 68 (1) (2018) 38–48.

[19] C. Wang, M. Pelillo, K. Siddiqi, Dominant set clustering and pooling for multi-view 3D object recognition, arXiv preprint arXiv:1906.01592.

[20] P. Zanuttigh, L. Minto, Deep learning for 3D shape classification from multiple depth maps, in: 2017 IEEE International Conference on Image Processing (ICIP), IEEE, 2017, pp. 3615–3619.

[21] A. Arsalan Soltani, H. Huang, J. Wu, T. D. Kulkarni, J. B. Tenenbaum, Synthesizing 3D shapes via modeling multi-view depth maps and silhouettes with deep generative networks, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2017, pp. 1511–1519.

[22] A. Kanezaki, Y. Matsushita, Y. Nishida, Rotationnet: Joint object categorization and pose estimation using multiviews from unsupervised viewpoints, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 5010–5019.

[23] H. You, Y. Feng, R. Ji, Y. Gao, Pvnet: A joint convolutional network of point cloud and multi-view for 3D shape recognition, in: Proceedings of the 26th ACM international conference on Multimedia, 2018, pp. 1310–1318.

[24] T. Yu, J. Meng, J. Yuan, Multi-view harmonized bilinear network for 3D object recognition, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 186–194.

[25] A. Bulat, J. Kossaifi, G. Tzimiropoulos, M. Pantic, Toward fast and accurate human pose estimation via soft-gated skip connections, arXiv preprint arXiv:2002.11098.

[26] L. Ge, Z. Ren, Y. Li, Z. Xue, Y. Wang, J. Cai, J. Yuan, 3D hand shape and pose estimation from a single RGB image, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2019, pp. 10833–10842.

[27] M. Fey, J. Eric Lenssen, F. Weichert, H. Müller, SplineCNN: Fast geometric deep learning with continuous B-spline kernels, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 869–877.

[28] J. Du, S. Zhang, G. Wu, J. M. Moura, S. Kar, Topology adaptive graph convolutional networks, arXiv preprint arXiv:1710.10370.

[29] J. Zhou, G. Cui, Z. Zhang, C. Yang, Z. Liu, L. Wang, C. Li, M. Sun, Graph neural networks: A review of methods and applications, arXiv preprint arXiv:1812.08434.

[30] N. M. Kriege, F. D. Johansson, C. Morris, A survey on graph kernels, Applied Network Science 5 (1) (2020) 6. doi:10.1007/s41109-019-0195-3.
URL https://doi.org/10.1007/s41109-019-0195-3

[31] S. V. N. Vishwanathan, N. N. Schraudolph, R. Kondor, K. M. Borgwardt, Graph kernels, Journal of Machine Learning Research 11 (Apr) (2010) 1201–1242.

[32] K. M. Borgwardt, H.-P. Kriegel, Shortest-path kernels on graphs, in: Fifth IEEE International Conference on Data Mining (ICDM'05), IEEE, 2005, pp. 8–pp.

[33] N. Shervashidze, S. Vishwanathan, T. Petri, K. Mehlhorn, K. Borgwardt, Efficient graphlet kernels for large graph comparison, in: Artificial Intelligence and Statistics, 2009, pp. 488–495.

[34] U. Kang, H. Tong, J. Sun, Fast random walk graph kernel, in: Proceedings of the 2012 SIAM international conference on data mining, SIAM, 2012, pp. 828–838.

[35] N. Shervashidze, P. Schweitzer, E. J. Van Leeuwen, K. Mehlhorn, K. M. Borgwardt, Weisfeiler-Lehman graph kernels, Journal of Machine Learning Research 12 (77) (2011) 2539–2561.

[36] M. Togninalli, E. Ghisu, F. Llinares-López, B. Rieck, K. Borgwardt,

Wasserstein Weisfeiler-Lehman graph kernels, in: Advances in Neural Information Processing Systems, 2019, pp. 6436–6446.
[37] A. A. Requicha, Representations of rigid solid objects, in: Computer Aided Design Modelling, Systems Engineering, CAD-Systems, Springer, 1980, pp. 1–78.
[38] K. E. Hoff III, J. Keyser, M. Lin, D. Manocha, T. Culver, Fast computation of generalized Voronoi diagrams using graphics hardware, in: Proceedings of the 26th annual conference on Computer graphics and interactive techniques, ACM Press/Addison-Wesley Publishing Co., 1999, pp. 277–286.
[39] B. Krayer, S. Müller, Generating signed distance fields on the GPU with ray maps, The Visual Computer 35 (6-8) (2019) 961–971.
[40] J. J. Park, P. Florence, J. Straub, R. Newcombe, S. Lovegrove, DeepSDF: Learning continuous signed distance functions for shape representation, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2019, pp. 165–174.
[41] J. Teuber, R. Weller, G. Zachmann, S. Guthe, Fast sphere packings with adaptive grids on the GPU, GI AR/VRWorkshop (Würzburg, Germany, vol. 4.
[42] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, I. Polosukhin, Attention is all you need, arXiv preprint arXiv:1706.03762.
[43] M. Fey, J. E. Lenssen, Fast graph representation learning with PyTorch geometric, arXiv preprint arXiv:1903.02428.
[44] J. Wu, C. Zhang, T. Xue, B. Freeman, J. Tenenbaum, Learning a probabilistic latent space of object shapes via 3D generative-adversarial modeling, in: Advances in neural information processing systems, 2016, pp. 82–90.
[45] D. Maturana, S. Scherer, Voxnet: A 3D convolutional neural network for real-time object recognition, in: 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), IEEE, 2015, pp. 922–928.
[46] A. Brock, T. Lim, J. M. Ritchie, N. Weston, Generative and discriminative voxel modeling with convolutional neural networks, arXiv preprint arXiv:1608.04236.
[47] S. Kumawat, S. Raman, LP-3DCNN: Unveiling local phase in 3D convolutional neural networks, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2019, pp. 4903–4912.
[48] V. Hegde, R. Zadeh, FusionNet: 3D object classification using multiple data representations, arXiv preprint arXiv:1607.05695.
[49] F. M. Bianchi, D. Grattarola, C. Alippi, L. Livi, Graph neural networks with convolutional arma filters, arXiv preprint arXiv:1901.01343.
[50] X. Li, H. Wang, L. Yi, L. J. Guibas, A. L. Abbott, S. Song, Category-level articulated object pose estimation, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2020, pp. 3706–3715.
[51] D. Pickup, X. Sun, P. L. Rosin, R. R. Martin, Z. Cheng, Z. Lian, M. Aono, A. Ben Hamza, A. Bronstein, M. Bronstein, S. Bu, U. Castellani, S. Cheng, V. Garro, A. Giachetti, A. Godil, J. Han, H. Johan, L. Lai, B. Li, C. Li, H. Li, R. Litman, X. Liu, Z. Liu, Y. Lu, A. Tatsuma, J. Ye, SHREC'14 track: Shape retrieval of non-rigid 3D human models, in: Proceedings of the 7th Eurographics workshop on 3D Object Retrieval, EG 3DOR'14, Eurographics Association, 2014.
[52] P. Shilane, P. Min, M. Kazhdan, T. Funkhouser, The Princeton shape benchmark, in: Proceedings Shape Modeling Applications, 2004., IEEE, 2004, pp. 167–178.
[53] G. Siglidis, G. Nikolentzos, S. Limnios, C. Giatsidis, K. Skianis, M. Vazirgiannis, GraKeL: A graph kernel library in Python., Journal of Machine Learning Research 21 (54) (2020) 1–5.
[54] C. Li, Spectral geometric methods for deformable 3D shape retrieval, Ph.D. thesis, Concordia University (2013).
[55] G. Nikolentzos, P. Meladianos, M. Vazirgiannis, Matching node embeddings for graph similarity, in: Thirty-First AAAI Conference on Artificial Intelligence, 2017.
[56] T. G. Gunn, The mechanization of design and manufacturing, Scientific American 247 (3) (1982) 114–131.
[57] M. Li, J. Fuh, Y. Zhang, Z. Qiu, General and partial shape matching approaches on feature-based CAD models to support efficient part retrieval, in: International Design Engineering Technical Conferences and Computers and Information in Engineering Conference, Vol. 43277, 2008, pp. 121–130.
[58] N. Sajadfar, Y. Ma, A hybrid cost estimation framework based on feature-oriented data mining approach, Advanced Engineering Informatics 29 (3) (2015) 633–647.
[59] R. Huang, S. Zhang, X. Bai, C. Xu, B. Huang, An effective subpart retrieval approach of 3D CAD models for manufacturing process reuse, Computers in industry 67 (2015) 38–53.
[60] S. Ghadai, A. Balu, S. Sarkar, A. Krishnamurthy, Learning localized features in 3D CAD models for manufacturability analysis of drilled holes, Computer Aided Geometric Design 62 (2018) 263–275.
[61] N. J. Mitra, L. J. Guibas, M. Pauly, Partial and approximate symmetry detection for 3D geometry, ACM Transactions on Graphics (TOG) 25 (3) (2006) 560–568.
[62] Y. Wang, S. Asafi, O. Van Kaick, H. Zhang, D. Cohen-Or, B. Chen, Active co-analysis of a set of shapes, ACM Transactions on Graphics (TOG) 31 (6) (2012) 1–10.
[63] S. Kim, H.-g. Chi, X. Hu, Q. Huang, K. Ramani, A large-scale annotated mechanical components benchmark for classification and retrieval tasks with deep neural networks, in: Proceedings of 16th European Conference on Computer Vision (ECCV), 2020.