A 3D Shape Descriptor based on Depth Complexity and Thickness Histograms

Wagner Schmitt*, Jose L. Sotomayor*, Alexandru Telea†, Cláudio T. Silva‡, João L. D. Comba*
*Instituto de Informática, PPGC, UFRGS, Brazil
†University of Groningen, Netherlands
‡New York University, United States

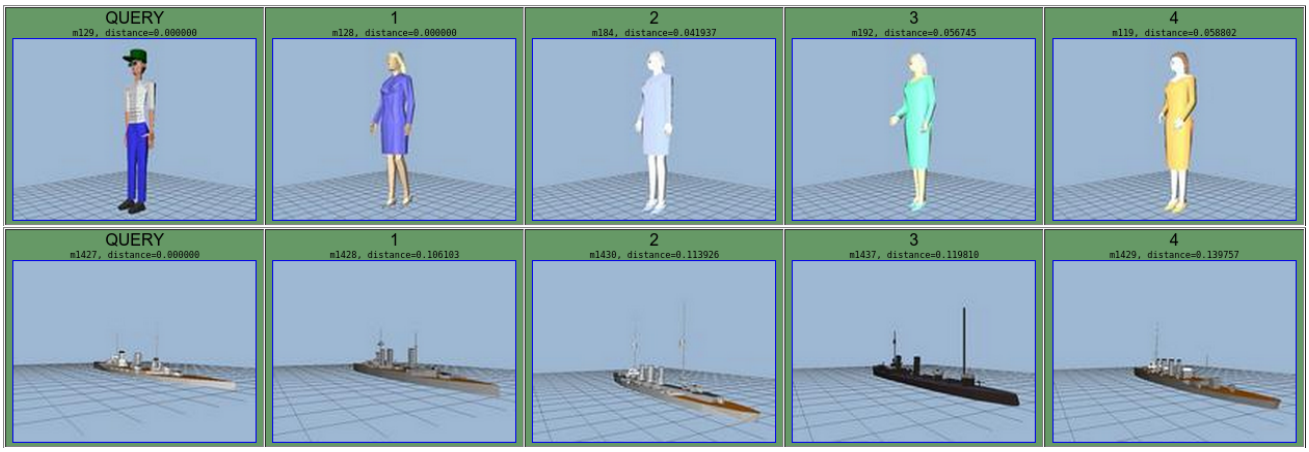


Fig. 1. Searching for similar 3D models. A query model is given as input. A DCTH descriptor is computed for this shape, which corresponds to a 2D histogram containing depth complexity and thickness information. Using The query database is searched comparing pre-computed descriptors of each model against the query descriptor, which produces a ordered ranking of closest shapes, from left to right. We illustrate above results for a human and ship shapes.

***Abstract*—Geometric models play a vital role in several fields, from the entertainment industry to scientific applications. To reduce the high cost of model creation, reusing existing models is the solution of choice. Model reuse is supported by content-based shape retrieval (CBR) techniques that help finding the desired models in massive repositories, many publicly available on the Internet. Key to efficient and effective CBR techniques are shape descriptors that accurately capture the characteristics of a shape and can discriminate between different shapes. We present a descriptor based on the distribution of two global features measured in a 3D shape, depth complexity and thickness, which respectively capture aspects of the geometry and topology of 3D shapes. The final descriptor, called DCTH (depth complexity and thickness histogram), is a 2D histogram that is invariant to the translation, rotation and scale of geometric shapes. We efficiently implement the DCTH on the GPU, allowing its use in real-time queries of large model databases. We validate the DCTH with the Princeton and Toyohashi Shape Benchmarks, containing 1815 and 10000 models respectively. Results show that DCTH can discriminate meaningful classes of these benchmarks and is fast to compute and robust against shape transformations and different levels of subdivision and smoothness.**

***Keywords*-Shape matching, Shape analysis, Depth complexity, Thickness, Histograms, Content-based retrieval.**

I. Introduction

Advances in 3D modeling and scanning techniques marked the last few years. Powerful modeling tools have emerged along affordable laser scanners that can rapidly generate 3D mesh models of real-world objects. The increase in mesh details and the ability of Graphics Processing Units (GPUs) to render high-resolution 3D models has increased the demand for 3D content in several fields, from medical and engineering applications to games and movies. As a result, an explosion in the number of 3D models available publicly occurred.

Today, there are on-line databases with thousands of free 3D models Retrieving 3D models from such databases increases the reuse and interchange of 3D models, avoiding the expense of creating 3D content from scratch, which is usually a laborious task that requires skill, time, and dedicated tools. Moreover, models in many such databases have high mesh quality, sampling resolution and are free of meshing defects, which only increases their reusability appeal.

Content-Based Retrieval (CBR) tools play a key role in supporting such model reuse. Central to CBR is the usage of *shape descriptors* to capture the characteristics of 3D models. Given a descriptor, 'searching by example' accounts to comparing the example descriptor to descriptors of all models in the database (Fig. 2). Many such descriptors have been proposed in the last decade [1], [2], [3], [4]. However, no such proposal could present a definitive result to become a standard. One main reason for this is CBR is much harder for 3D shapes than for text or image retrieval [1], due in

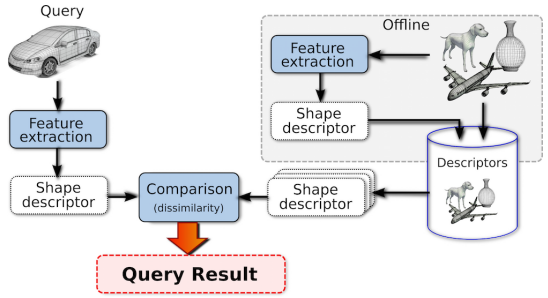


Fig. 2. Conceptual diagram for a content-based shape retrieval system

turn to the large dimensionality and variability of 3D shapes. For instance, 3D shapes are rarely easy to parameterize and can have arbitrary topologies. Such inherent 3D shape aspects prevent methods used to analyze other data types, *e.g.* Fourier analysis, to be generalized to create 3D shape descriptors. Separately, the scalability and quality of many of the CBR techniques for 3D shapes have not been tested with a massive number of models. This is due to the lack of shape databases containing classification information to be used as ground truth for computing CBG precision and recall [5]. Until recent years, benchmarks contained less than 1000 shapes, such as the Princeton shape database [6], created in 2004, which contains 907 models. Only more recently, in 2010, the large Toyohashi benchmark, containing 10000 models, was published [7].

In this work we propose a 3D shape descriptor called the *depth complexity and thickness histogram* (DCTH). DCTH uses a 2D histogram of depth complexity and thickness measured from a 3D shape, which captures both topological (depth-complexity) and geometrical (thickness) shape characteristics. DCTH is rotation, translation, and scale invariant, and can be quickly computed on the GPU. We validate DCTH using both the Princeton and Toyohashi benchmarks.

A. Related work

Much effort has been put in designing content-based retrieval (CBR) systems. Early work focused mainly on text-based document retrieval systems [8], [9]. More recently, CBR systems also have been developed for other types of media, such as images [10], audio [11], video [12] and 3D models [1], [13], [14], [2]. Since then, many approaches defining signatures to use in 3D shape matching emerged in the literature [15]. An extensive list of feature-based, graph-based and other types descriptors is given in [5].

Osada *et al.* [1] used a distribution of global features to describe a 3D object. Distributions were evaluated using a pseudo-metric to obtain a dissimilarity measure. They also proposed five simple descriptors computed by measuring characteristics such as angle, distance, area and volume using a random uniform distribution of points over the 3D shape surface. Their D2 distance measure presented the best results among the studied descriptors. Kazhdan *et al.* [2] proposed the spherical harmonics descriptor. They used a rotationally-invariant set of spherical functions based on the original spherical harmonics descriptor in [16]. Rotational invariance is obtained by decomposing the functions into their spherical harmonics and summing the harmonics for each frequency component. Results show that a rotationally-invariant descriptor is better than one that requires pose normalization obtained using principal component analysis (PCA). However, this descriptor does not explicitly capture topology information. Unlike [1], we propose using *two* distributions to capture both geometric and topological features. Hence, our work can be considered a feature-based technique that uses feature distribution to find measure shape dissimilarity.

The DB-VLAT descriptor produced very good precision-recall plots testing with models from the Toyohashi Shape Benchmark. Their state-of-the-art technique outperformed established descriptors proposed in the literature, such as the D2 [1] and the Spherical Harmonics [2]. For the six classes tested by Tatsuma *et al.*, the only one that did not perform well was a vehicle-like 'tank' class. The authors stated that the geometry of this vehicle was harder to discriminate due to its oblong cannon. No information was given about the computational cost and scalability of their descriptor. The descriptors listed above focus on shape matching. Other descriptors also exist, *e.g.* for partial matching [17], local description [18], object poses [4] or deformable 3D shapes [19].

II. The DCTH Shape Descriptor

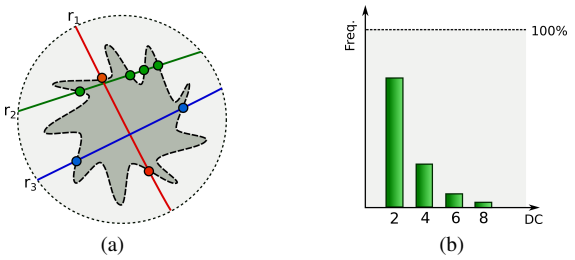
A shape descriptor is a compact way to represent features of 2D or 3D shapes. One use of a shape descriptor is to compare shapes using a dissimilarity function. Besides computation speed, good shape descriptors should satisfy additional application-specific features. For shape retrieval, the following features are considered relevant [20], [5], [21], [22]:

- **discriminative** accuracy, or the ability to capture subtle geometric and topological differences between shapes;
- **invariance** to translation, scaling, and rotation, collectively known as pose normalization;
- **robustness** against small-scale shape perturbations or noise and sampling resolution;
- **scalability** in terms of computational speed and memory.

Guided by these desirable features, we propose DCTH, a statistical descriptor based on two measures: a depth complexity (DC) distribution and thickness (T) distribution, explained in Secs. II-A and II-B respectively. The two measures attempt to capture the geometric and topological features of a shape respectively, thereby increasing the descriptor's discriminative power. Both measures are robust in the presence of transformations and can be efficiently implemented on the GPU.

A. Depth Complexity Distribution Descriptor

The first component of the DCTH is a histogram of *depth complexity* (DC) information. To explain the DC, consider a ray $\mathbf{r}$ in 3D space. The depth complexity of a 3D model with respect to $\mathbf{r}$ is defined as the number of intersections of $\mathbf{r}$ with the model. Now, consider a bounding sphere around our



(a) (b)

Fig. 3. DC function sampled for rays $\mathbf{r}_1$, $\mathbf{r}_2$ and $\mathbf{r}_3$ for a simple shape. (a) For each ray, the number of colored shape-ray intersection dots gives the DC value. (b) Corresponding DC distribution

given shape. Each ray that intersects the shape also has two intersection points with the sphere, described by their polar coordinates θ_1, γ_1 and θ_2, γ_2. Hence, the depth complexity for a given ray $\mathbf{r} = (\theta_1, \gamma_1, \theta_2, \gamma_2)$ can be encoded as a function $DC(\theta_1, \gamma_1, \theta_2, \gamma_2) \rightarrow \mathbb{N}$ of four variables. The 4D shape function DC, defined over the space of oriented rays, thus captures the depth complexity of the entire shape. Fig. 3 (a) shows the DC function values along three rays $\mathbf{r}_1$, $\mathbf{r}_2$ and $\mathbf{r}_3$ for a simple shape, sketched on 2D for illustration purposes.

We compute a depth-complexity distribution by using a uniform random sapling of the DC function over its ray space. To obtain an optimal balance between sampling density and computational speed, we computed distributions for varying sample counts on several models and using them further in performing CBR of 3D shapes. From our experiments, $500K$ ray samples provided sufficient resolution while being fast enough to compute (see Sec. III). Fig. 3(a) shows such a distribution for the shape in Fig. 3(b). Note that geometrically closed (watertight) shapes mostly have even values in their distributions, since intersections with the shape occur in pairs for all rays except those touching the shape at vertices.

The DC distribution has several desirable properties. First, it is invariant to translation, rotation and scaling. Secondly, its computation does not impose any constraints on the mesh type used to represent the input shape (*e.g.* triangle, general-polygon, polygon soup) or mesh quality (clean structure *vs* having duplicate or T-vertices or very thin triangles). This allows its direct application on all models of shape databases without requiring expensive clean-up operations. In contrast, shape descriptors that perform geometric computations on the mesh *surface*, such as [1], [2], [23], may require preprocessing to enforce such mesh constraints.

B. Thickness Distribution Descriptor

The second component of the DCTH descriptor is the thickness (T) distribution. Unlike the DC descriptor, thickness discriminates shapes according to their geometrical properties. As a starting point, we use the definition of shape thickness along a given ray $\mathbf{r}$ given in [24]: A ray $\mathbf{r}$ having $2n$ intersections $\mathbf{x}_i$, $1 \leq i \leq 2n$, with a shape generates n thickness values $t_i = \|\mathbf{x}_{2i+1} - \mathbf{x}_{2i}\|$, $1 \leq i \leq n$ as being the distances between consecutive intersection points on the ray. We capture thickness only using t_0, *i.e.* using the distance between the first and second ray-shape intersections. This arguably does not decrease the description quality, since deeper intersections are (a) either captures by t_0 values computed for other rays, or (b) pertain to 'deep' shape details which are hardly visible (or not visible at all) from outside the shape from any direction. Moreover, this allows an efficient and simple GPU-based implementation for the thickness descriptor (see next Sec. III).

Liu *et al.* proposed a related technique called Directional Histogram Model (DHM), where they define thickness as the distance between the first and last ray-shape intersection points [25]. DHM can be easily computed in a two-pass OpenGL-based rendering process – front-facing polygons are rendered first, followed by back-facing polygons (for convex shapes). This can be easily implemented in the fixed OpenGL pipeline, by placing the camera at the desired position (sphere sample point) and orienting it towards the center of the bounding sphere. However, as [24] argues, the main disadvantage of DHM is the loss of internal geometrical shape properties, which occurs for complex models having concavities, such as buildings or CAD models. A similar GPU-based technique to the DHM was used to compute the medial axis, or curve skeleton, of 3D shapes [26], with similar limitations for concave shapes. Separately, we note that more accurate and geometrically-motivated methods to estimate the local thickness of a shape exist, such as using the distance between a surface point and its closest medial surface point [27]. However, such methods require the computation of the 3D medial surface, a process that is slow and delicate.

Similarly to the DC descriptor, we next convert the thickness (T) values to a distribution, using the same number and position of sample rays. Thickness is obtained during the same rendering pass that computes depth complexity, thus adding only a small overhead to the total computation (see Sec. III). However, in contrast to the DC measurement, a scaling normalization must be used for the T measurement. For this, we proceed as follows. First, we define the distance between z_{near} and z_{far} of the OpenGL near and far clipping planes as being the diameter of the bounding sphere around the shape. This can be done by casting a ray along the viewing direction and retrieving the intersections against the bounding sphere. The two intersection points are then used to define the near and far clipping planes. This way, the z (depth) values, retrieved from the OpenGL Z-buffer, are all in the same range for different scales of the same shape. A second aspect to consider is the fact that the Z-buffer uses a nonlinear scale, which gives more precision objects near the near clipping plane. Hence, we find thickness values by converting 'raw' z values from the Z-buffer to linear depth values z_{linear} by

$$z_{linear} = \frac{(2 * z_{near})}{(z_{far} + z_{near} - z * (z_{far} - z_{near}))}. \quad (1)$$

C. DCTH descriptor

The DCTH shape descriptor is a 2D histogram of depth complexity and thickness obtained by joining the 1D DC and T histograms. DCTH correlates frequency values of both above histograms to obtain a better discrimination than using each histogram separately. Fig. 6 (bottom row) shows the

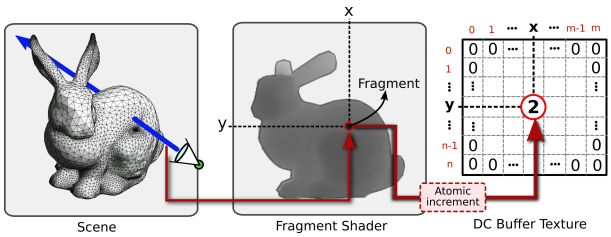


Fig. 4. GPU computation of DC for a given view direction. The shape is rendered for the each viewing direction. Using a fragment shader, the number of fragments for each pixel position (x, y) is counted and saved into a texture. The texture is bound to an OpenGL image unit that allows to perform atomic increments inside the shader program.

2D DCTH histograms for three different 3D models, with thickness mapped to the $x-$axis and depth complexity mapped to the y-axis respectively, and frequency (counts) mapped to color, using a blue-to-red logarithmic color map.

D. Dissimilarity Measures

Once the DCTH descriptor is computed for a set of 3D shapes, dissimilarities between shape-pairs can be computed using various distance metrics based on the shapes' descriptors. Common distance metrics include the Minkowski L_N norms, match distances [28], the earth mover's distance (EMD) [29], and the Hellinger distance [30], which is closely related to the Bhattacharyya distance [31]. Given two DCTH histograms H_1 and H_2 and a distance metric d, we assume that distance values are normalized, *i.e.* $d(H_1, H_2) \in [0, 1]$, with 0 implying perfect similarity and 1 maximal dissimilarity.

We tested the L_1, Hellinger, Chi-square, correlation, and EMD distance metrics, aiming to optimize the trade-off between computational performance and retrieval precision. The best results were given by the Hellinger distance.

III. Efficient GPU Implementation

We next describe the efficient GPU-based computation of the DC and T distributions. The proposed algorithm allows the DCTH descriptor to be scalable for large databases containing thousands of complex 3D shapes. Our approach has three steps: shape sampling (Sec. III-A), data collection (Sec. III-B), and normalized histogram construction (Sec. III-C).

A. Shape sampling

Points are uniformly sampled over the bounding sphere around the model. At each sample point, an OpenGL camera is pointed towards the sphere center. For each camera position, the 3D object is rendered, and the DC and T values are computed. Camera parameters are configured as follows:

Projection: We use orthogonal rather than perspective projection to avoid precision loss due to perspective distortions that occur in distant parts of the scene, *e.g.*, triangles becoming very small. As such, each rendering pass samples rays that are parallel to the viewing direction.

Frustum size: We use a view frustum that best fits the shape to prevent losing depth precision during rendering. For this, we use a 3D bounding box of the input shape to define the camera coordinate system and also set the frustum size.

Resolution: The viewport resolution offers a trade-off between precision and speed. A resolution of 256^2 pixels gave good results for all tested models. Larger resolutions are better for high-precision queries for a database having many complex objects with small detail polygons.

Clipping planes: As mentioned in Sec. II-B, thickness values must be normalized for different shape scales. To obtain depth values within the same fixed range, we set the near and far clipping planes to (a) encompass the entire object and (b) have fixed values for all viewpoints. For this, we set the distance between the near and far planes to the diameter of the bounding sphere around the shape.
Finally, we note that the view up vector, which controls the camera rotation around the viewing direction, is irrelevant since both depth- complexity and thickness values for a 3D shape depend only on the viewing direction and distance to the object, and not the camera rotation around this direction.

B. Data collection

In each rendering pass for a different viewpoint, OpenGL fragment shaders are used to compute the DC and T values from that viewpoint. Our shader uses two textures. The first texture has an integer channel to store the DC values. The second one has two floating-point channels to store the z values of the first and second intersections. We found this solution to be significantly faster than using an array of 2D textures inside the shader. The two textures are bound to OpenGL image units, allowing us to perform safe atomic writes into the textures in the shader Fig. 4 shows the procedure used to find the DC values for each ray by counting the number of fragments rendered at each pixel of the viewport. Similarly, for the thickness T, the depth value of each fragment is used to compute the distance between consecutive fragment-pairs.

C. Normalized histogram construction

In the last step, data collected from all sampled viewpoints is normalized by the total number of drawn rays, creating the DCTH histogram, containing both T and DC values. These steps are performed after each sampled viewpoint is rendered to avoid creating many separate textures. This step is performed on the CPU. Apart from this step, the DCTH is entirely computed on the GPU. Since the texture resolution is relatively small (see Sec. III-A), computing the final histogram normalization on the CPU does not incur significant penalties.

IV. Evaluation

The evaluation of the DCTH descriptor is an important aspect to validate the proposal. Given a query model M and a shape database D, we call *relevant* retrievals all models in D to which M should match. More specifically, for a given M, a perfect descriptor should return all shapes in D

marked as being in the same class, or of the same type, as M. The definition of a relevant retrieval is subject to user choices or application scope. For example, in some cases a wide retrieval may lead to many matches of M in D, whereas narrow matches might be preferred in other cases.

A. Precision-Recall Curves

In information retrieval, precision P and recall R are defined in terms of a set of retrieved instances S and a set of relevant instances S_R, both belonging to a given database D. Precision is the fraction of retrieved instances that are relevant to the query while recall is the fraction of relevant instances retrieved by the query. Precision and recall are computed as

$$P = \frac{|S_R \bigcap S|}{|S|}, \qquad R = \frac{|S_R \bigcap S|}{|S_R|}. \tag{2}$$

The *order* in which objects are returned to the user in a query is also important. Ideally, more relevant objects (from the query perspective) should appear earlier in the query result S. The Average Precision (AP) metric is used for this purpose. Precision and recall are computed for each position of the ranked sequence S. The precision $P(R)$ is defined as a function of recall R, for all sequence positions. Typically, the function $P(R)$ varies over $R \in [0, 1]$. When the recall R is small, precision $P(R)$ is often large. When the recall R is large, precision $P(R)$ tends to be small. The AP of the query corresponds to the average of $P(R)$ over the entire range of R for that query.

The Mean Average Precision (MAP) evaluates the effectiveness of a set of Q queries, rather than a single query. MAP is equal to the mean of the AP for each query:

$$MAP = \frac{1}{Q} \sum_{q=1}^{Q} AP_q \tag{3}$$

where AP_q is the AP of the q-th query in the sequence of Q queries. MAP is a standard metric for ranked results in text retrieval [32] and retrieval evaluation benchmarks [33].

V. Experimental Results

Several experiments were designed to evaluate the robustness, shape discrimination and efficiency of the DCTH shape descriptor. The implementation was written in C++, using g++ version 4.7. Results were measured on an Intel Core i7 PC, with 12 GB RAM, running Linux Ubuntu 12.04.

Accepted input formats for 3D models include .obj or .off, which are common formats for publicly available 3D shape databases. For our tests, we used the Princeton and Toyohashi benchmarks. Princeton has 907 models for testing and 907 models for training for typical shape retrieval algorithms. Since the DCTH algorithm does not need a training step, all 1814 models were used in our tests. These models are further divided into 80 different shape-classes. Toyohashi has 10000 models, organized in 352 classes. During testing, only classes containing at least 20 models were used to avoid a small number of samples in the MAP plots. Typical classes include humans, plants, airplanes, quadrupeds, chess pieces,

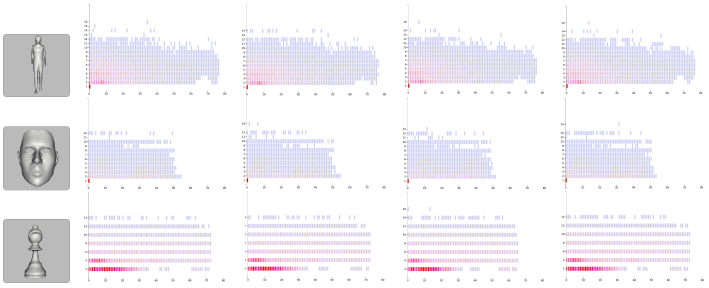


Fig. 5. Each row displays the model and the DCTH histograms after applying rotation, anisotropic scaling, mirroring, and shape subdivision-smoothing. In each row the histograms are similar before and after transformations.

among others. The list of distance metrics used to compare DCTH descriptors is given in Tab. III). All metrics, except the simple L_1 norm, were computed using the optimized implementations provided in the OpenCV library. The evaluation of the descriptors uses three criteria: robustness (Sec. V-A), performance (Sec. V-B), and efficiency (Sec. V-C).

A. Robustness

The robustness of the DCTH descriptor was measured by testing its invariance to affine transformations and mesh resolution issues. Three generic shapes from the Princeton database were selected and subjected to one of the following transformations:

- rotations of 45 degrees around the z, y and z axes respectively;
- anisotropic scaling by factors of 2.5 in the x-axis and 0.25 in the z-axis;
- mirroring against the $x-y$, $y-z$, and $x-z$ planes;
- surface subdivision using the Catmull-Clark algorithm [34] with two iteration levels.

The 2D DCTH histograms were computed after applying each transformation. As shown in Fig. 5, these histograms remain almost unchanged after each of the tested transformations, thus demonstrating the robustness of DCTH.

B. Retrieval Performance

Measuring the retrieval performance of DCTH requires querying the database and evaluating precision-recall plots. We implemented a shape retrieval tool to generate the average precision-recall plots. As a reference, we used the shape-class descriptions provided in the Princeton Shape Benchmark. Descriptors are compared using the Hellinger distance, which gave the best trade-off between quality and speed.

Plots of the separated components of the DCTH descriptor can give insight on how the DCTH changes according to a given shape. Fig. 6 shows the DC and thickness (T) 1D histograms, as well as the combined 2D DCTH histogram for three different models. The DC histogram is similar for the panther and the chess piece, as both shapes have few concavities and an average local thickness. The DC histogram for the tree is quite different, as this shape has a very different topology. In this example, the DC histogram is not enough

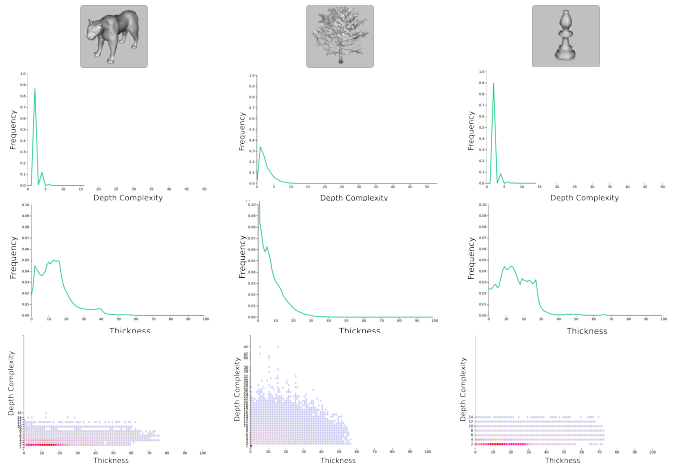


Fig. 6. Comparison of DC, T, and DCTH histograms (bottom three rows) for three objects of typical classes in the Princeton Shape Benchmark (shown in the top row). The DCTH histogram combines the discriminative power of both DC and T histograms (Sec. V-B).

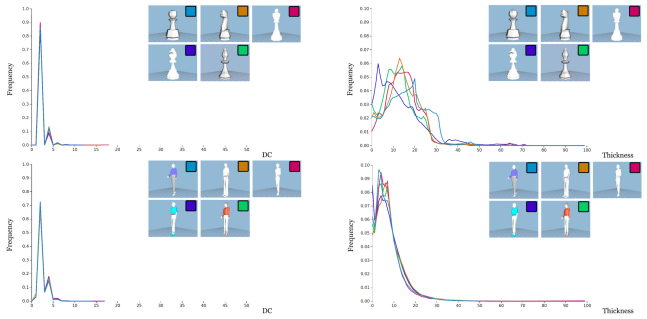


Fig. 7. DC and T histograms for five shapes in the classes *humans* and *chess pieces*. Same-class objects have similar histograms (Sec. V-B).

to discriminate the three shapes. In contrast, the T histogram is more discriminative, revealing the fact that the geometric (local thickness) properties of these objects are quite different. The DCTH histogram for the tree looks quite different from the DCTH histograms of the other two shapes, which reflects its combined discriminatory power.

Fig. 7 reveals insights about the discriminatory power of DC and T histograms. Five objects belonging to the *humans* and *chess pieces* were tested. As can be seen, the same-class DC and T histograms are quite similar. Differences exist between the DC histograms and T histograms of different-class objects. However, if we combine the DC and T histograms in the DCTH descriptor, an increased discriminatory power is obtained. Fig. 8 shows how the DCTH descriptor separates three objects in the *humans* and *chess pieces* classes.

Fig. 9 shows the precision-recall plots for queries pertaining to objects located in six different shape classes. The best result was achieved for the *faces* class. One explanation for this result is that face shapes are obtained from 3D scanners. As such, shapes have a simple topology consisting of an open surface, and a DC histogram having a peak around 1 (most rays have one intersection with the shape). The DC histograms

TABLE I
DCTH DESCRIPTOR COMPUTATION TIME FOR PRINCETON DATASET.

Princeton shape benchmark			**1815 models**
Model	Vertices	Faces	Time (ms)
Largest model	160940	316498	32884
Smallest model	31	35	206
Typical model	7691	15294	652
Average time per model			**0.968 sec**
Total running time			**1757 sec**

for other shapes have quite different forms since such shapes are typically closed (watertight). The DCTH descriptor is successful in separating faces from other shapes.

Similar tests were performed on the Toyohashi shape benchmark, which contains 10000 shapes organized in 352 classes. Fig. 10 shows the average precision-recall plots for the same shape-classes as in Fig. 9 (Princeton benchmark). Precision-recall results are worse for the Toyohashi benchmark than for the Princeton Benchmark. Upon closer analysis, it was observed that the Toyohashi benchmark contains degenerated models, and many models were disassembled or had many missing or duplicated triangles. Such models lead to spurious or missing ray-shape intersections, thereby introducing noise in the DC and T histograms.

We also run our descriptor for the same classes tested by Tatsuma *et al.* [7] to evaluate their DB-VLAT descriptor. In the specific classes discussed in their work DB-VLAT presented better precision-recall results than DCTH. The main reason is the lack of robustness of our technique when 3D models present missing or duplicated triangles, which happens in the majority of the models of these classes. More conclusive comparisons could not be performed since we were not able to obtain an implementation of their descriptor. In addition, no information was given about the computational cost and scalability of the DB-VLAT descriptor. Although, retrieval quality is very important, factors such as speed and ease of implementation are also essential for a scalable approach.

C. Computational Efficiency

A CBR system following the pipeline of Fig. 2 was used to measure the efficiency of the DCTH descriptor using the Princeton and Toyohashi databases. Descriptors were generated for each database model and stored in its respective database. As a query, we used either a shape of the database or a third-party shape. For each query shape, we computed the DCTH descriptor and compared it against all stored descriptors. Matches were returned in increasing order of dissimilarity. Clearly, this query process is suboptimal in terms of query speed as its complexity is linear in the number of database models. Hierarchical structures can speed up the search by avoiding unnecessary descriptor comparisons. Since our goal was the design of an efficient and computationally effective *descriptor*, rather than a computationally effective database *search structure*, we did not implement such structures.

Tables I and II show DCTH computation time for the

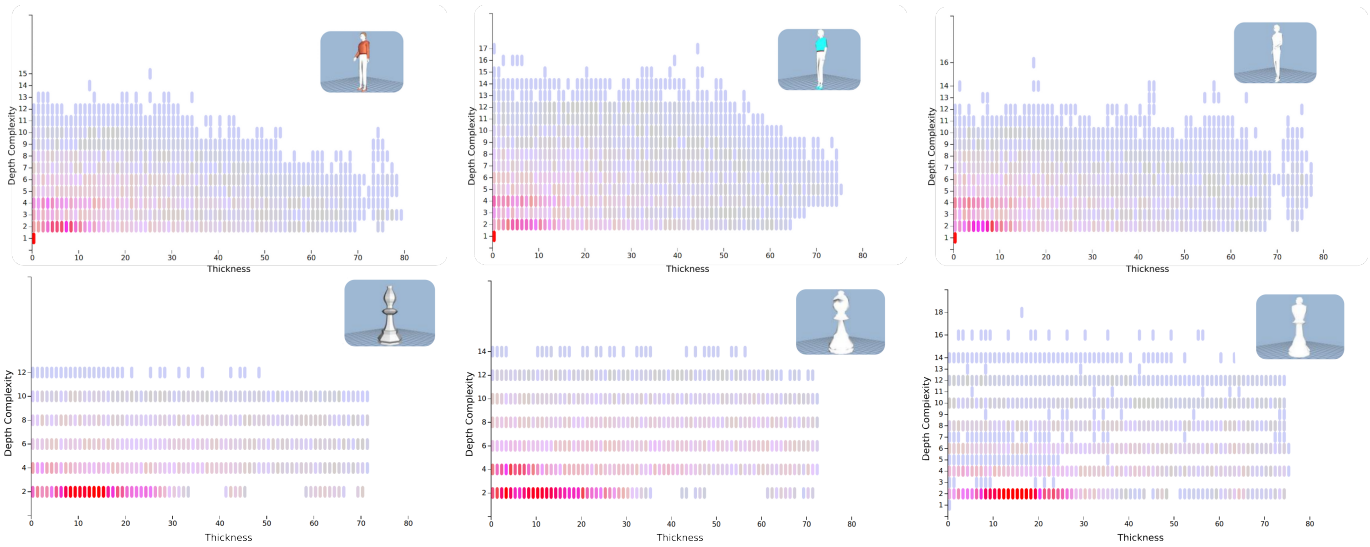
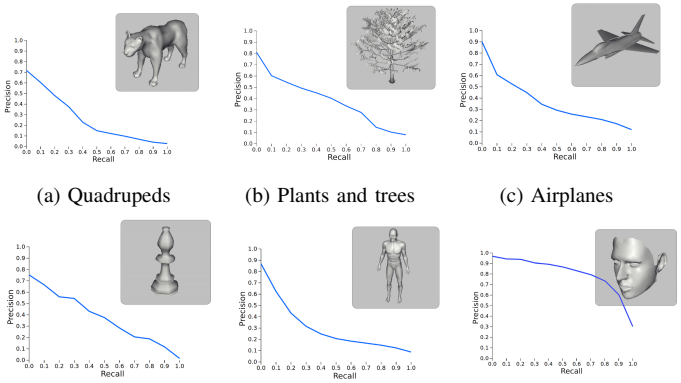


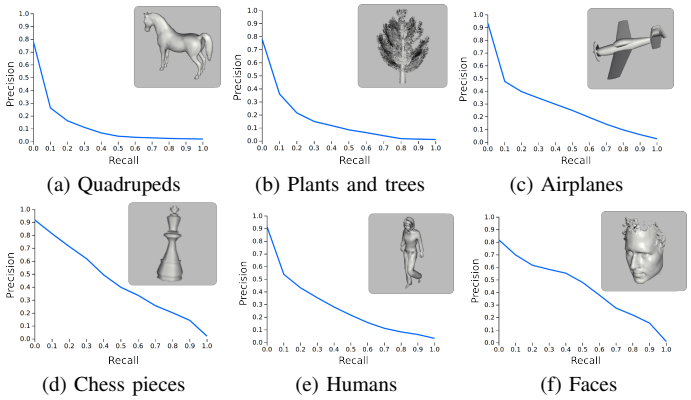
Fig. 8. DCTH histograms for objects of the *human* and *chess pieces* classes in the Princeton Shape Benchmark. The histograms clearly discriminate between the two classes. Although shapes in both classes have many rays with $DC = 2$, the chess pieces have a higher thickness T (Sec. V-B).



(a) Quadrupeds (b) Plants and trees (c) Airplanes

(d) Chess pieces (e) Humans (f) Faces

Fig. 9. Precision-recall plots of six shape-classes in the Princeton Shape Benchmark using the DCTH descriptor and Hellinger distance.



(a) Quadrupeds (b) Plants and trees (c) Airplanes

(d) Chess pieces (e) Humans (f) Faces

Fig. 10. Precision-recall plots of six shape-classes in the Toyohashi Shape Benchmark using the DCTH descriptor and Hellinger distance.

Princeton and Toyohashi benchmarks. The parameters used were the same as those for the results in this paper, *i.e.*

TABLE II
DCTH DESCRIPTOR COMPUTATION TIME FOR TOYOHASI DATASET.

Toyohashi shape benchmark			**10000 models**
Model	Vertices	Faces	Time (ms)
Biggest model	181912	362935	25377
Smallest model	6	8	122
Typical model	12858	24685	785
Average running time per model			**0.601 sec**
Total running time			**6013 sec**

TABLE III
TIME TO COMPARE A QUERY DESCRIPTOR WITH ALL DESCRIPTORS

	Princeton dataset	**Toyohashi dataset**
	Total time (ms)	Total time (ms)
L_1	165	722
Hellinger	**44**	**216**
Chi-square	36	196
Correlation	35	188
EMD	882090	4410000

$RS = 500K$ ray samples and $W = 256^2$ screen resolution. Our implementation took 1757 seconds to generate DCTH descriptors for the Princeton database and 6013 seconds for the Toyohashi database. As both databases have similar models with respect to the numbers of vertices and faces, a linear performance in the model count was observed.

Table III shows the time required to compare a query descriptor with all descriptors stored in the two databases using 5 different distance metrics. The relative low speed of the L_1 metric is explained by the fact that no optimized data structures to store 2D sparse histograms was used. In contrast, the Hellinger, Chi-square and correlation metrics used such optimizations, provided by OpenCV, and showed a better (and comparatively similar) performance. Finally, the EMD

distance was considerably slower than all other metrics, given its inefficiency when dealing with sparse histograms.

VI. Conclusions

In this paper, we presented DCTH, a new 3D shape descriptor based on the distribution of depth complexity and thickness information. The result is a 2D depth-complexity-and-thickness (DCTH) histogram, aimed at generic shape retrieval under translation, rotation and scale invariance. The descriptor has proven to be computationally efficient and robust. The DCTH descriptor exhibits a promising retrieving power, in terms of capturing shape geometry and topology, as tested on the well-known Princeton and Toyohashi benchmarks. DCTH can easily be implemented on the GPU and delivers performance rates compatible with online and interactive retrieval rates, even when using a linear search algorithm.

One avenue for future work is to extend DCTH to measure additional global shape features, such as the geodesic distance between ray-shape intersection points. This can be computed using GPU methods [35] and skeleton properties.

Acknowledgments

This work was supported by a Google and IBM Faculty Awards, the Moore-Sloan Data Science Environment at NYU, the NYU School of Engineering, the NYU Center for Urban Science and Progress, AT&T, NSF award CNS-1229185, and CNPq Processes 476685/2012-5 and 309483/2011-5.

References

[1] R. Osada, T. Funkhouser, B. Chazelle, and D. Dobkin, "Shape distributions," *ACM Trans. Graph.*, vol. 21, no. 4, pp. 807–832, Oct. 2002.

[2] M. Kazhdan, T. Funkhouser, and S. Rusinkiewicz, "Rotation invariant spherical harmonic representation of 3d shape descriptors," in *Proceedings of the 2003 Eurographics/ACM SIGGRAPH symposium on Geometry processing*, ser. SGP '03. Aire-la-Ville, Switzerland, Switzerland: Eurographics Association, 2003, pp. 156–164.

[3] ——, "Symmetry descriptors and 3d shape matching," in *Proceedings of Eurographics/ACM SIGGRAPH Symposium on Geometry Processing*, ser. SGP '04. ACM, 2004, pp. 115–123.

[4] C. Chen, Y. Zhuang, and J. Xiao, "Silhouette representation and matching for 3d pose discrimination a comparative study," *Image and Vision Computing*, vol. 28, no. 4, pp. 654 – 667, 2010.

[5] J. W. Tangelder and R. C. Veltkamp, "A survey of content based 3d shape retrieval methods," *Multimedia Tools Appl.*, vol. 39, no. 3, pp. 441–471, Sep. 2008.

[6] P. Shilane, P. Min, M. Kazhdan, and T. Funkhouser, "The Princeton shape benchmark," in *Shape Modeling International*, Jun. 2004.

[7] A. Tatsuma, H. Koyanagi, and M. Aono, "A large-scale shape benchmark for 3d object retrieval: Toyohashi shape benchmark," in *Signal Information Processing Association Annual Summit and Conference (APSIPA ASC), 2012 Asia-Pacific*, 2012, pp. 1–10.

[8] M. Mitra and B. Chaudhuri, "Information retrieval from documents: A survey," *Information Retrieval*, vol. 2, no. 2-3, pp. 141–163, 2000.

[9] C. D. Manning, P. Raghavan, and H. Schütze, *Introduction to information retrieval*. Cambridge university press Cambridge, 2008, vol. 1.

[10] A. W. Smeulders, M. Worring, S. Santini, A. Gupta, and R. Jain, "Content-based image retrieval at the end of the early years," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 22, no. 12, pp. 1349–1380, 2000.

[11] P. Cano, M. Kaltenbrunner, F. Gouyon, and E. Batlle, "On the use of fastmap for audio retrieval and browsing." in *ISMIR*, 2002.

[12] M. Flickner, H. Sawhney, W. Niblack, J. Ashley, Q. Huang, B. Dom, M. Gorkani, J. Hafner, D. Lee, D. Petkovic *et al.*, "Query by image and video content: The qbic system," *Computer*, vol. 28, no. 9, pp. 23–32, 1995.

[13] D.-Y. Chen, X.-P. Tian, Y.-T. Shen, and M. Ouhyoung, "On visual similarity based 3d model retrieval," *Computer Graphics Forum*, vol. 22, no. 3, pp. 223–232, 2003.

[14] T. Funkhouser, P. Min, M. Kazhdan, J. Chen, A. Halderman, D. Dobkin, and D. Jacobs, "A search engine for 3d models," *ACM Trans. Graph.*, vol. 22, no. 1, pp. 83–105, Jan. 2003.

[15] D. Zhang and G. Lu, "Review of shape representation and description techniques," *Pattern Recognition*, vol. 37, pp. 1–19, 2004.

[16] D. Saupe and D. Vrani, "3d model retrieval with spherical harmonics and moments," in *Pattern Recognition*, ser. Lecture Notes in Computer Science, B. Radig and S. Florczyk, Eds. Springer Berlin Heidelberg, 2001, vol. 2191, pp. 392–397.

[17] Z.-B. Liu, S.-H. Bu, K. Zhou, S.-M. Gao, J.-W. Han, and J. Wu, "A survey on partial retrieval of 3d shapes," *Journal of Computer Science and Technology*, vol. 28, no. 5, pp. 836–851, 2013.

[18] P. Heider1, A. Pierre-Pierre, R. Li, and C. Grimm, "Local shape descriptors, a survey and evaluation," in *Proc. EG 3DOR*. Eurographics, 2011, pp. 49–56.

[19] C. Li and A. BenHamza, "A multiresolution descriptor for deformable 3d shape retrieval," *The Visual Computer*, vol. 29, no. 6-8, pp. 513–524, 2013.

[20] I. Kazmi, L. You, and J. J. Zhang, "A survey of 2d and 3d shape descriptors," in *Computer Graphics, Imaging and Visualization (CGIV), 2013 10th International Conference*, Aug 2013, pp. 1–10.

[21] Y. Mingqiang, K. K. Idiyo, and R. Joseph, "A Survey of Shape Feature Extraction Techniques," *Pattern Recognition, Peng-Yeng Yin (Ed.) (2008) 43-90*, pp. 43–90, Nov 2008.

[22] N. Iyer, S. Jayanti, K. Lou, Y. Kalyanaraman, and K. Ramani, "Three-dimensional shape searching: state-of-the-art review and future trends," *Computer-Aided Design*, vol. 37, no. 5, pp. 509 – 530, 2005, ¡ce:title¿Geometric Modeling and Processing 2004¡/ce:title¿.

[23] M. Hilaga, Y. Shinagawa, T. Kohmura, and T. L. Kunii, "Topology matching for fully automatic similarity estimation of 3d shapes," in *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, ser. SIGGRAPH '01. New York, NY, USA: ACM, 2001, pp. 203–212.

[24] Y. Liu, J. Pu, H. Zha, W. Liu, and Y. Uehara, "Thickness histogram and statistical harmonic representation for 3d model retrieval," in *3D Data Processing, Visualization and Transmission, 2004. 3DPVT 2004. Proceedings. 2nd International Symposium on*, 2004, pp. 896–903.

[25] X. Liu, R. Su, S. B. Kang, and S. Heung-Yeung, "Directional histogram model for three-dimensional shape similarity," in *Computer Vision and Pattern Recognition, 2003. Proceedings. 2003 IEEE Computer Society Conference on*, vol. 1. IEEE, 2003, pp. I–813.

[26] J. Kustra, A. Jalba, and A. Telea, "Probabilistic view-based 3D curve skeleton computation on the GPU," in *Proc. VISAPP*, 2013, pp. 137–145.

[27] A. Telea and A. Jalba, "Voxel-based assessment of printability of 3D shapes," in *Proc. ISMM*. Springer LNCS 6671, 2011, pp. 393–404.

[28] H. C. Shen and A. K. Wong, "Generalized texture representation and metric," *Computer Vision, Graphics, and Image Processing*, vol. 23, no. 2, pp. 187 – 206, 1983.

[29] Y. Rubner, C. Tomasi, and L. Guibas, "A metric for distributions with applications to image databases," in *Computer Vision, 1998. Sixth International Conference on*, Jan 1998, pp. 59–66.

[30] E. Hellinger, *Neue Begründung der Theorie quadratischer Formen von unendlichvielen Veränderlichen*, 1909.

[31] A. Bhattacharyya, "On a measure of divergence between two statistical populations defined by their probability distributions," *Bulletin of the Calcutta Mathematical Society*, vol. 35, pp. 99–109, 1943.

[32] TREC, "Text retrieval conference site," http://trec.nist.gov, 2014.

[33] CLEF, "The CLEF initiative – conference and labs of the evaluation forum," http://www.clef-initiative.eu, 2014.

[34] E. Catmull and J. Clark, "Seminal graphics." New York, NY, USA: ACM, 1998, ch. Recursively Generated B-spline Surfaces on Arbitrary Topological Meshes, pp. 183–188. [Online]. Available: http://doi.acm.org/10.1145/280811.280992

[35] A. Jalba, J. Kustra, and A. Telea, "Surface and curve skeletonization of large 3D models on the GPU," *IEEE TPAMI*, vol. 35, no. 6, pp. 1495–1508, 2013.