# Large scale 3D shape retrieval by exploiting multi-core and GPU

**M. BENJELLOUN[1], E.W. DADI[2], and E.M. DAOUDI[2]**
[1]University of Mons, Faculty of Engineering. Dept of Computer Science
Mons, BELGIUM
[2]University of Mohammed First, Faculty of Sciences, LaRi Laboratory,
Oujda, Morocco

**Abstract -** *this paper addresses the problem of 3D shape retrieval in large databases of 3D objects (large retrieval). While this problem is emerging and interesting as the size of 3D object databases grows rapidly, the main two issues the community has to focus on are: computational efficiency of 3D object retrieval and the quality of retrieved results. In this work we deal with the first consideration, namely the computational efficiency of 3D object retrieval by exploiting new implementations based on parallel computing by exploiting multi-core and GPU architectures. Experimental results, show that the large scale retrieval can be achieved using the multi-core environment.*

**Keywords:** 3D Content-based Shape Retrieval, Large Scale Retrieval, GPU, multi-core architecture, CMBOF method, OpenMP

## 1 Introduction

Currently, there are an increasing number of 3D objects on the web, leading to large databases, thanks to recent digitizing and modeling technologies. The need of efficient methods for 3D shape-content based retrieval, in order to ease navigation into related large databases, and also to structure, organize and manage this new multimedia type of data, has become an active topic in various research communities such as computer vision, computer graphics, mechanical CAD, and pattern recognition. The 3D shape retrieval is the processing of retrieving visual similar objects to given 3D object query.

Various 3D shape retrieval methods have been proposed in the literature [3,4,5,6,7]. All recent methods are based on the shape descriptors; that consists in designing an efficient canonical characterization of the objects. This characterization is referred as a descriptor or a signature. Since the descriptor serves as a key in the search process, it is a critical kernel with a strong influence on the searching performances (i.e. computational efficiency and relevance of the results). Designing an efficient canonical characterization of the objects was become a major challenge in 3D objects indexation. A good 3D shape retrieval method must satisfy at least the two following conditions simultaneously [3]:

- The relevance and the quality of retrieval results: the first 3D objects returned by the method must be the most similar to the query.
- Computational efficiency of 3D object retrieval: the retrieval results should be fast.

Most existing methods do not satisfy the above conditions simultaneously. Moreover, for the large database, the retrieval process needs more computational time which does not permit the large scale retrieval. In order to achieve faster retrieval of 3D object, we propose in this work, new implementations based on parallel computing by exploiting multi-core and GPU architectures. For the tests, we use the CMBOF method proposed by Lian et al. [8], since it gives the best result comparing to many other methods in particular the view based methods [9,10,11,12].

Generally, the retrieval process is performed into two essentials stages: indexation and shape matching. Our contribution in this paper is to study the problem of retrieval in large database in particular the stage of matching, since this stage needs more computational times regarding to the size of 3D object databases which grows rapidly.

The rest of the paper is organized as follows. In section 2 we give a brief description of the CMBOF method. The parallelization on multi-core is presented in section 3. We conclude the paper in section 4.

## 2 Description of the CMBOF method

The CM-BOF (Clock Matching Bag-Of-Features) is a 3D retrieval method, proposed by Lian et al [8], this method gives the best result comparing to many other methods in particular the view based methods [9,10,11,12].

The two essential stages of the 3D shape retrieval are: indexation and shape matching. In the following we give a brief description of the two stages of the CMBOF method.

### 2.1 Shape indexing

The indexation is the process for computing the descriptor of a given 3D object. It is based on the following steps:

- *Normalization and alignment*: 3D objects are given in arbitrary position, orientation and scale. So, a step of normalization and alignment of the 3D shape is necessary in order to assure the invariance of affine transformations (scaling, translation rotation and reflection). To compare two 3D objects using multi-view methods, these objects must have the same length, orientation and position. Actually there is no efficient method of normalization and alignment which satisfies at the same time the following constraints: rotation invariance, best alignment and computational time.
- *Multi-view rendering*: in this step, a set of depth-buffer views (2D images) are uniformly captured around the 3D object. CMBOF method uses vertices of a given unit geodesic sphere which is obtained by subdividing the unit regular octahedron. The number of views to be captured is 6, 18, 66 or 258. Lian et al [8] showed that 66 is the best number of views.
- *SIFT feature extraction:* a 3D object can be approximately represented by a set of depth-buffer views. In the CMBOF method, each view is described as a word Histogram (descriptor of the view) using SIFT (Scale Invariant Feature Transform) algorithm [13]. This algorithm is used to extract salient local features from each 2D view. In this case, a 3D object is characterized by several descriptors depending on the number of the views captured around this object instead one descriptor as proposed in other multi-view methods [9,10,11,12].
- *Vector quantization and histogram generation:* Each SIFT feature extracted from 2D view is quantized as a vector or visual word (descriptor) by using a global visual codebook. The quantized local features are accumulated into a histogram which becomes the feature vector (descriptor) of the corresponding 2D view.

### 2.2 Shape matching

To retrieve visual similar 3D objects to a given 3D object-query, the descriptor of the query is compared with the descriptors of objects in the database. Instead of completely solving the problem of normalization and alignment, Lian et al [8] are proposed the Clock Matching approach. The basic idea of this approach is to consider 24 matching pairs when comparing two 3D Objects, by placing one object in the original orientation while the second one may appears in 24 different poses. The dissimilarity between the two 3D objects is measured by the minimum distance of their all (24) possible matching pairs. After the query is compared with each object in the database, the obtained distances are sorted according to query; the top k results returned should be the most similar to the query.

## 3 Parallelization of the retrieval process

Current machines offer microprocessors composed of multiple cores (processors) and Graphical Processing Units (GPU). The major challenge is to exploit efficiently the potential of these architectures at their maximum performance. The aim of this work is to propose new implementations based on parallel computing to achieve faster retrieval of 3D object and therefore allows the large scale retrieval (the retrieval in large databases) by exploiting the potential of GPU and multi-core architectures. Recall that the retrieval process, using the CMBOF method, is performed into the following two essential phases that are performed online:

- Indexation of the 3D query-object: computing the descriptor of the query shape
- Shape matching: comparing the descriptor of the query-object with the descriptor of each 3D objects of the database.

Note that a few works have been proposed in the literature to implement the 3D shape retrieval under HPC environments. These works are partial, since they only concern indexation phase (particularly the SIFT Quantization [1,2]) and not the shape matching phase. On the other hand, several works have been proposed in the literature based on sequential solutions to allow the large scale of 3D shape retrieval [14,15].

Experimental are performed as follow:

- Tests on GPU (Graphics Card Units) are performed on GeForce GT610, 2048 MB of global memory, L2 Cache size 6,6 MB.
- For the GPU programming we have used CUDA5.0.
- Tests on multi-core are performed on a Dell PowerEdge R910 Server Intel® Xeon® Processor E7-4850 2GHz of 10 cores, 32GB RAM.
- For multi-core programming we have used OpenMP
- For the 3D-object data base, we have used Princeton 3D Shape Benchmark database [17] composed of 1800 - 3D Objects. In order to make tests in large database, we have increased the size of the data base by duplication of the 3D-objects.
- To capture 66 views around a 3D object, we have used the executable provided by Lian [8].

### 3.1 Indexing phase

To compute the descriptor of a given 3D object, the CMBOF method [8] proposes to describe the 3D-object by several word histograms (descriptors) where each descriptor corresponds to a 2D view (2D image) captured around the shape of this object. To compute the descriptor of a given 2D view, the following steps are performed:

- Extraction of the SIFT salient local features from this 2D view.
- Vector quantization and histogram generation.

For the first step, the extraction of SIFT salient local features using SIFT algorithm executed on CPU is very time consuming; especially if it is necessary to extract SIFT

features from several 2D views as in case of CM-BOF method.

In this work, we extracted the local features of each 2D view using the SIFT-GPU version proposed by Changchang Wu [18], since it gives a higher computing performance compared to the CPU version [16]. For 66 views, the SIFT is performed in 1,8s on GPU compared to 10,8s on CPU.

## 3.2 Shape matching phase

In this phase, the descriptor of the query is compared with descriptors of each 3D object belonging to the database. Note that the descriptors of objects in the database are computed offline.

Sequentially shape matching in a large database is time consuming. The advantage of using multi-core is to compare simultaneously the query-object with p objects; where p is the number of cores. After the comparison of the query with each 3D object in the database is completed, the obtained distances will be sorted.

Assume that the database is composed of m 3D objects. To exploit the potential of the multi-core architecture at their maximum performance and improve the load balancing between different cores we use a dynamic data distribution as follows:

- Each core deals with an initial workload (a number k of objects with $k \leq m/p$). The remaining block of objects will be shared between all cores.
- All cores execute simultaneously the comparison process between the query object with their own objects.
- As soon as a processor (core) completes its work, it takes one objects from the shared block (the remaining objects). This process is repeated as long as it remains untreated objects.

Figure 1 presents the evolution of the execution time of the shape matching process by various cores using the clause "dynamic schedule" of OpenMP. Tests are performed on 3D-objects of a database composed of 10800 3D-objects. Thus the Figure 1 states that, the execution time is significantly reduced in proportion to the number of cores.

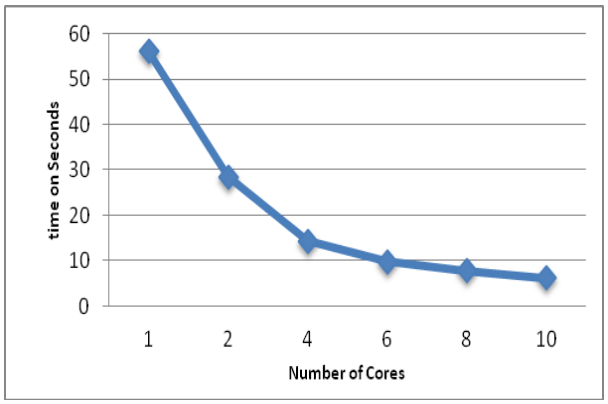


**Figure 1 .** Execution time for the shape matching process

In Figure 2, we report the results of the measured speed up (sequential_time/parallel_time) of the shape matching process compared to the ideal one. The obtained speed (up to 90% ) shows that the proposed implementation is scalable, which means that; if we increase the number of cores, the parallel time remains close to the sequential time divided by the number of cores.

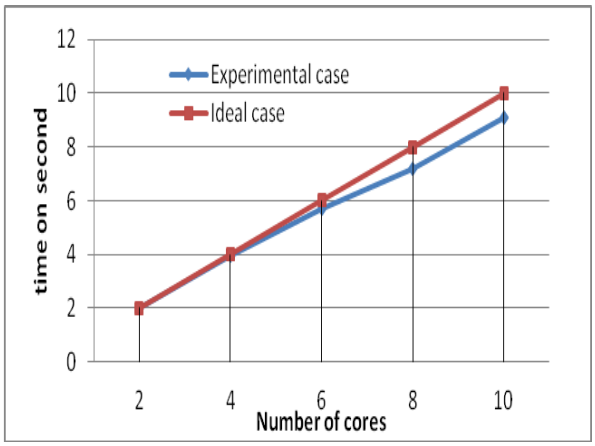


**Figure 2 .** Mesured speed up compared to the ideal one

In Figure 3, we measure the efficiency (the percentage of the using processor performance when increasing the number of cores)

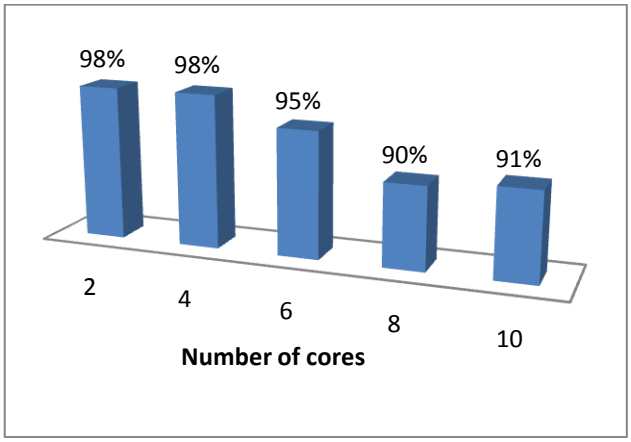


**Figure 3 .** Efficiency

In Figure 4, we compare the execution time of the matching process for different sizes of databases. In this test we report the evolution of the parallel time on 10 cores versus sequential time. The results show that the size of the database does not affect the speed up. If we use large databases on p cores, the parallel time remains close to the sequential time divided by p. We conclude that the large scale can be achieved by using a great number of cores.

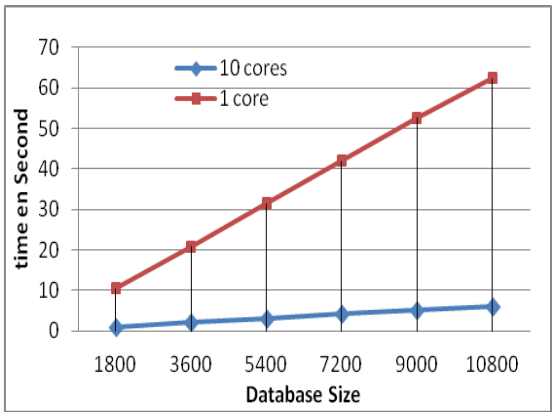


**Figure 4 .** Execution time of the retrieval process on databases with different sizes

# 4 Conclusion

In this paper we are interested by the computational efficiency of 3D objects retrieval. We have proposed new implementations of the retrieval process based on parallel computing by exploiting the multi-core environment and GPU accelerators. The proposed implementations, for the matching process, are independent of the 3D object algorithm. So that, for our tests, we have used the CMBOF method proposed by Lian et al. [8], since it gives the best results compared to many other methods, proposed in the literature, in particular the view based methods.

First, we have compared the SIFT algorithm both on CPU and GPU. For the version on GPU, we have used the algorithm proposed in [18].

Then, we have proposed parallel implementations on multi-core environment. The experimental results show that the execution time is significantly reduced as the number of cores grows. On the other hand, for fixed number of cores, the execution time grows linearly (almost linear) as the size of the database grows. We conclude that the large scale can be achieved using parallel computing.

# 5 Acknowledgment

We thank TNAC team "Théorie des Nombres et Applications en Cryptographie" for the use of their machine Dell PowerEdge R910 Server

# 6 References

[1]. Ryutarou Ohbuchi, Takahiko Furuya, "Accelerating Bag-of-Features SIFT Algorithm for 3D Model Retrieval", SAMT09

[2].Quansheng Kuang, and Lei Zhao, "A Practical GPU Based KNN Algorithm", Proceedings of the Second Symposium International Computer Science and Computational Technology(ISCSCT '09) , Huangshan, P. R. China, , pp. 151-155, 26-28,Dec. 2009

[3].J.W.H. Tangelder and R.C. Veltkamp, "A survey of content based 3D shape retrieval methods," Multimedia Tools and Applications, vol. 39, no. 3, pp. 441–471, Sept. 2008.

[4].Shilane P, Kazhdan M, Min P, Funkhouser T, "The princeton shape benchmark". In: Proc. shape modeling international 2004, pp 157–166

[5].T. Zaharia and F. Preteux, "3D versus 2D/3D shape descriptors: A comparative study," in SPIE Conf. on Image Processing: Algorithms and Systems III - IS & T/ SPIE Symposium on Electronic Imaging, Science and Technology '03, San Jose, CA, Jan. 2004, vol. 5298

[6].B. Bustos, D. A. Keim, T. Schreck, and D. Vranic, "An experimental comparison of feature-based 3D retrieval methods," in 2nd Int. Symp. on 3D Data Processing, Visualization, and Transmission (3DPVT'04), Thessaloniki, Greece, Sept. 2004.

[7].A. Del Bimbo and P. Pala, "Content-based retrieval of 3D models," ACM Trans. Multimedia Com

[8].Zhouhui Lian, AfzalGodil, Xianfang Sun: "Visual Similarity based 3D Shape Retrieval", IEEE International Conference on Shape Modeling and Applications (SMI) 2010

[9].Ding-Yun Chen, Xiao-Pei Tian, Yu-TeShen et Ming Ouhyoung :"On visual similarity based 3d model retrieval". In EUROGRAPHICS, Granada, Spain, sep 2003.

[10]. R. Ohbuchi, K. Osada, T. Furuya, T. Banno, "Salient Local Visual Features for Shape Based 3D Model Retrieval", Proc. IEEE International Conference on Shape Modeling and Applications (SMI'08), Stony Brook University, June 4 - 6, 2008.

[11]. P. Daras and A. Axenopoulos, "A 3D shape retrieval framework supporting multimodal queries", Int'l Journal of Computer Vision (IJCV).

[12]. M. Chaouch and A. Verroust-Blondet, "A new descriptor for 2D depth image indexing and 3D model retrieval," in Proc. ICIP'07,vol. 6, 2007, pp. 373–376.

[13]. D.G. Lowe, "Distinctive Image Features from Scale-Invariant Keypoints", Int'l Journal of Computer Vision, 60(2), November 2004.

[14]. El Wardani Dadi , El Mostafa Daoudi, Claude Tadonki, "Fast 3D shape retrieval method for classified databases",IEEE International Conference on Complex Systems (ICCS) 2012

[15]. Remco C. Veltkamp, Geert-Jan Giezeman, Hannah Bast, Thomas Baumbach, Takahiko Furuya, Joachim Giesen, AfzalGodil, Zhouhui Lian, RyutarouOhbuchi, WaqarSaleem, SHREC'10 Track: Large Scale Retrieval, Eurographics Workshop on 3D Object Retrieval (2010)

[16]. A. Vedaldi, SIFT++ A lightweight C++ implementation of SIFT, http://vision.ucla.edu/~vedaldi/code/siftpp/siftpp.html

[17]. P. Shilane, P. Min, M. Kazhdan, and T. Funkhouser, "The Princeton shape benchmark," in Shape Modeling and Applications Conference, SMI'2004, Genova, Italy, June 2004, IEEE, pp. 167–178

[18]. A GPU Implementation of Scale Invariant Feature Transform (SIFT) http://cs.unc.edu/~ccwu/siftgpu