# CaS: Collection-aware Segmentation

Raquel Costa   Manuel J. Fonseca   Alfredo Ferreira
INESC-ID/IST/Technical University of Lisbon
Lisboa
rgcc@ist.utl.pt {mjf,alfredo.ferreira}@inesc-id.pt

**Resumo**
Ao longo dos tempos, a segmentação tem provado ser um desafio devido à sua subjectividade. A segmentação depende não apenas do domínio em causa mas acima de tudo da interpretação que os humanos fazem do objecto. Para cada contexto, diversas soluções específicas foram propostas com diferentes objectivos, limitações e vantagens. Neste trabalho propomos ultrapassar algumas dessas limitações usando o algoritmo de segmentação Collection-aware Segmentation (CaS). Este algoritmo identifica segmentos de objectos em colecções baseados na sua individualidade nessa colecção. Para esse efeito realizámos um conjunto de testes para compreender como as pessoas segmentam objectos numa colecção. A partir dos resultados destes testes desenvolvemos os algoritmos Adaped-CaS e Geons-augmented CaS. Avaliações experimentais com utilizadores mostraram que a abordagem proposta produz segmentações com significado para os humanos.

**Abstract**
Segmentation has always proven to be a challenge because of its subjectiveness. It depends not only of the application domain but also most on the human interpretation. To each context, several specific solutions were proposed with different goals, limitations and advantages. With this work we propose to overcome some of those limitations by improving the Collection-aware Segmentation algorithm (CaS). This algorithm identifies segments of objects in collections based on their individuality among the collection in which the objects belong. To that end we performed a set of tests to understand how humans segment a collection of objects. From the results of these tests we developed the Adaped-CaS and the Geons-augmented CaS algorithms. Experimental evaluation with users revealed that our approach produces a segmentation that is meaningful for humans.

**Keywords**
3D Object Segmentation, 3D Object Collections, Automatic Segmentation, Similarity Estimation

## 1. INTRODUCTION

Each human being interprets the environment from his own point of view. This generates an huge range of possible interpretations of the world, and his components. Thus, segmentation of objects may vary from individual to individual. Indeed, it and has been subject of studies in different areas, from mathematics to philosophy.

Over the past decades object segmentation has been also tackled within the computer graphics domain, as a result of the growing number of 3D objects in digital format and the widespread of applications that use them. Many computer graphics applications, as animation, collision detection, object indexing and retrieval use the segmentation approaches as a stage of the core process. However, most of the existing object segmentation techniques are domain or context dependent. These perform well in the domain and context for which they were designed for, but not so good in other domains or contexts.

To overcome this limitation we adopt a different approach that extends the Collection Aware Segmentation (CaS) method. This was originally proposed in [Ferreira09] embedded in a solution for indexing and retrieval of 3D objects. In this paper we extend CaS to make it a stand-alone segmentation technique that produces meaningful results to the users, independently of the object domain.

With the present work we isolated the CaS approach of the indexing and retrieval application, thus creating a segmentation algorithm that is application independent. As the original CaS approach, the Adapted CaS is based on the Hierarchical Fitting Primitives [Falcidieno06] algorithm and uses Spherical Harmonics shape descriptor [Kazhdan03]. By adding geon analysis [Biederman87], we improved the algorithm, achieving better segmentation, closer to human perception.

To evaluate the proposed approach, we conducted an experimental evaluation where several tests were performed. The first test focusing on understanding how humans segment 3D objects. From the results of this test we devel-

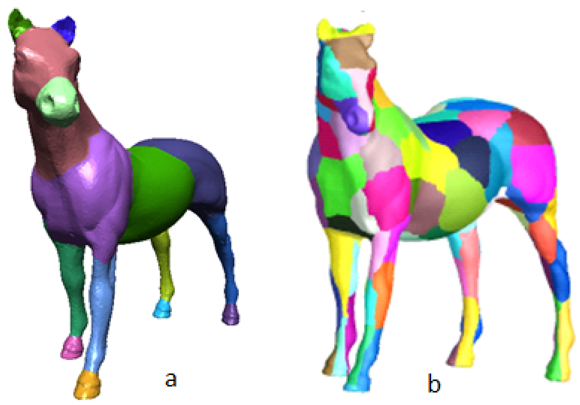


**Figure 1: Two different types of segmentation: a) Part Based Segmentation; b) Surface Based Segmentation. (Figure taken from [Shamir08])**

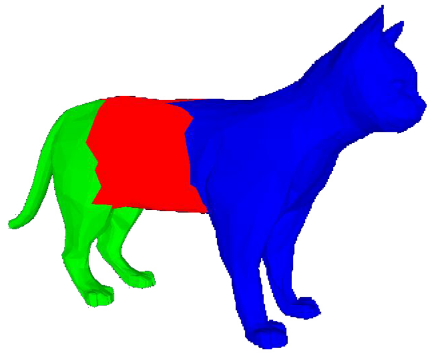

**Figure 2: Hierarchical decomposition with fuzzy area represented in red. (Figure taken from [Katz03])**

oped and refined our approach. Then, to evaluate the efficacy and effectiveness of the approach we executed performance tests to study execution time and memory requirements. Finally, to validate the quality of the segmentation, we organized a test where users where asked to compare the results of manual segmentation made by humans with results produced automatically by segmentation algorithms.

In the remaining of this document we start with a brief presentation of related work on three-dimensional object segmentation. Then we describe the original Collection Aware Segmentation algorithm, followed by its evolution and the explanation on detail of how these work. On section 4 we describe the evaluation tests and discuss the corresponding results. In the last section, we present the conclusions of this work and reflect on future research paths on this topic.

## 2 RELATED WORK

### 2.1 Categorization of segmentation approaches

Some authors [Agathos07, Shamir08] classify the segmentation techniques in two categories depending on the kind of segmentation accomplished, represented in Figure 1. The part based segmentation is closer to user perception and divides the object into sub-components, while the surface based segmentation are accomplished by analysing the surface shape features.

Among these two types, the 3D object segmentation has been widely studied and different solutions that uses distinct approaches have been proposed. Some of the existing and more relevant are presented next.

### 2.2 Region Growing and Clustering

The region growing methodology follows a exploratory approach that starts by visiting a seed (a face of the mesh), then agglomerates the adjacent faces by transversing the mesh and it stops when it reaches a stop condition. Such condition can be the segment convexity. Then the process is restarted using a non visited face as a new seed. The approach proposed by Zuckerberger on [Zuckerberger02], uses a depth first or breadth first search to transverse a graph that represents the mesh and the seed where it starts is a node of this graph. From this transverse is creates a segment and then, it restarts transversing on an unvisited node of the graph forming a new segment.

In a similar way, Attene et al. [Falcidieno06] approach also agglomerates faces, but instead of creating patch by patch, it creates all patches simultaneously. These are organized as an hierarchical tree in which the leafs are the mesh triangles and the root is the entire object. This tree is built bottom to top and clusters adjacent faces with the minimum merging cost that can be calculated on different ways. Attene et al. uses primitives by fitting them to the resultant cluster.

Previously, Katz and Tal [Katz03] had proposed to generate, in an iterative clustering, various results of segmentation given a number of clusters, and then chooses which is the best segmentation. It starts by creating a representative group of clusters and then each adjacent face is clustered until it reaches a ray $r$ between the seed and the limit of the patch. This is used to agglomerate the faces that have more probability of belonging to a given segment, thus creating fuzzy areas, illustrated Figure 2. Lastly it is needed to transform the fuzzy decomposition in a final decomposition by refining the limits.

### 2.3 Skeleton Based

This approach uses the skeleton of the object to determine the segmentation. The most used method to extract the skeleton is the Reeb Graph. For instance, Tierny et al. [Tierny07] extract an enhanced topological skeleton by finding feature points located on object extremities using the geodesic distances. These are used to create a function that indicates the distance from a given point of the mesh to each feature point of the mesh and from there is built the Reeb graph. After having the skeleton, the object is segmented in the areas where each skeleton node corresponds to a segment of the object.

### 2.4 Geometry and Structure-Based

The Taylor and Plumber algorithms, by Mortara et.al. [Mortara03] uses the object shape to perform segmentation based on geometry and structure. These algorithms detects tubular features by blowing bubbles that starts on a seed that is predefined and stops blowing when it finds an abrupt change on the object shape, such as a bifurcation. Later, Mortara et al. [Mortara06] used the Plumber approach to find the tubular zones in objects that represents the human body.

### 2.5 Feature Points and Core Extraction

In an attempt to overcome the pose invariant limitation a new approach arises called Feature Points and Core Extraction that was proposed by Katz et al. in [Katz05]. This initially creates a pose invariant representation of the object, then extracts the feature points and the core. After finding the core it is necessary to extract the rest of the segments which is done by matching each part to the feature points. In the end, it reverse the initial process so the object can come back to the same shape.

### 2.6 SDF

A different approach was proposed by Shapira et al. in [Shapira08]. They use the shape diameter function (SDF) for segmenting objects. In short, this gives the diameter of an object in a neighbour of a point and is used to merge points that have the same or close diameter values using a histogram.

### 2.7 Automatic Segmentation of 3D Collections

The above referred approaches, as most existing approaches, segment 3D objects individually. The segmentation is performed object by object individually, instead of segmenting various objects simultaneously. Indeed, this is a relatively recent concept: to accomplish automatic simultaneous segmentation of sets of objects. The approaches that use the automatic segmentation of 3D collections use the information of similar objects to improve the results. The group of Thomas Funkhouser in Priceton is one of the groups that is already studying this subject. They presented an algorithm [Golovinskiy09] that builds a graph whose nodes represents the mesh faces and the edges represents the edges of the mesh that connects adjacent faces of the same object. They also represent the correspondence between the faces of different meshes. In the next step the algorithm executes a hierarchical clustering of the graph, were the adjacent faces of the same model, and the correspondent faces of different models are going to probably belong to the same segment.

### 2.8 Discussion

Several segmentation approaches have emerged as decomposition of 3D objects as is used in many different applications in distinct domains that require different segmentations. This also makes the task of evaluating the approaches hard to perform, due due the large number of approaches. Nevertheless, Funckhouser and his team defined a benchmark for 3D mesh segmentation [Chen09], compared seven mesh segmentation algorithms and draw some interesting conclusions.

However, a common limitation is the need of inputs provided by the users in order to decompose the objects. Some need to predefine the seeds like in Region Growing and Iterative Clustering Approaches. Others to select the level of the hierarchy generated by the segmentation approach, these are all the approaches that produce a hierarchical segmentation. The result of the segmentation has more segments that it should have producing over segmented result. Some approaches try to overcome this limitation by using post-processing stage that removes the extra segments by merging them to others or by initially predefining a maximum number of segments.

The individual segmentation of objects can be considered a limitation if we consider segmenting a collection of objects. If the segmentation decomposes object by object as it is in the major approaches, then it can take more time to decompose the entire collection than if segmentation was performed simultaneously.

In order to overcome the limitations highlighted above, we propose a different solution based on the CaS algorithm, presented in the next section. This is a segmentation approach that will be application independent.

## 3 COLLECTION-AWARE SEGMENTATION

In this section we present a distinct approach for segmenting 3D objects whose main goal is to overcome some limitations found on existing approaches. We extracted the Collection-aware Segmentation (CaS) algorithm from the indexing and retrieval context where it was embedded [Ferreira09], as a step of a complete solution, and extended it to a fully fledged 3D object segmentation algorithm. This led to the original CaS and to the evolution for Geons-Augmented CaS.

### 3.1 Original CaS for Retrieval

The original CaS, integrated in a retrieval solution, did not indeed produced any object segmentation *per se*. Instead, it decomposes all objects in a collection and stores their subparts in a shape pool, used then for indexing the collection.

This approach, depicted in Figure 3 has two main stages: the initialization stage and the iteration stage. In the first stage the foundations of the segmentation are computed and loaded into memory, while in the following stage the objects of the collection are iteratively segmented into subparts.

The initialization starts by generating, for each object on the collection the respective hierarchical segmented mesh (HSM) using the Hierarchical Fitting Primitives (HFP) as proposed by Attene et al. [Falcidieno06]. This approach consists on merging clusters and then fitting them to primitives to create the final clusters. Next it saves the resulting HSM on the HSM Set, producing a set of seg-

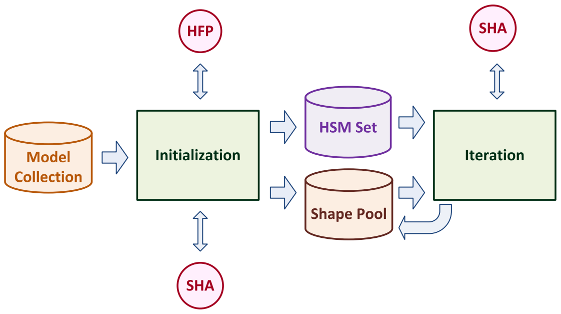


**Figure 3: Overall architecture of the original CaS, with the two main stages: initialization and iteration.**

mented meshes. Then is computed for each mesh, the object signature using the spherical harmonics shape descriptor (SHA) introduced by Kazhdan et al. [Kazhdan03]. The resulting signatures are stored in the shape pool for further processing.

In the iteration stage, each signature on the shape pool is verified for its uniqueness, thus being flagged for decomposition. This means that if the number of similar elements is below a pre-defined threshold, it is considered as unique, thus decomposable. This similarity is computed using the differences between object signatures.

If the object is decomposable, then it is decomposed by exploring the corresponding HSM on the HSM Set and getting its child nodes, computing their signatures and adding them to the Shape Pool. If the object is not decomposable, it passes to the next element on the shape pool. The iteration stage finishes when there are no more decomposable segments on the shape pool and the algorithm ends by returning the entire shape pool. Indeed, it does not produce segmented versions of the objects in the model collection.

### 3.2 Adapted CaS for Decomposition

To produce segmentations for objects in a collection, we adapted the original CaS for decomposition. As in the original, the Adapted CaS consists on two main stages: the initialization and the iteration. These stages have similar purpose to the those on the original CaS, but are slightly different and comprise new data structures, as depicted in Figure 4.

The Adapted CaS decomposer receives as input the entire collection of objects on the initialization stage. On this stage, for each object, it computes the corresponding HSM using HFP. This generates a binary tree that is built bottom to top. It starts on the leafs that represents the triangles of the mesh, then for each pair of neighbours it calculates the merging cost. This cost is calculated by fitting a primitive (cylinder, cone, sphere and plane) to the resulting cluster, then it compares the values and clusters the pairs that have less cost. This new cluster generated from the clustering are saved as a parent node of these clusters, in the tree.

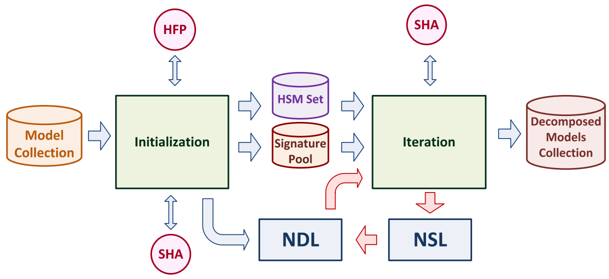


**Figure 4: Overall architecture of Adapted CaS, with the new NDL and NSL structures.**

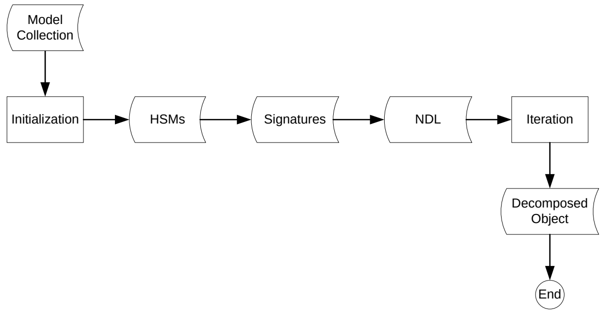


**Figure 5: High level fluxogram of Adapted CaS.**

It repeats the process until it reaches the top of the three where it is saved the entire object. These trees, stored in the HSM set, will be transversed during the iteration stage to produce the decomposed objects. Figure 5 presents an high level fluxogram of the decomposition process.

Since the decomposition algorithm is based on the singularity of an object, the second step of the initialization stage is to compute the SHA signatures of each object. These are shape descriptors, that is, a numerical representation of the object in a multidimensional space. Using these representations makes the comparison between segments easier and this comparison is used to label a segment as decomposable or not. Thus, each computed signature is stored on the signature pool for later use.

The main goal of this Adapted CaS technique is to present the collection of objects decomposed. Thus the shape poll is no longer necessary to and was replaced by the Signature Pool. It is only necessary to visit this set to label a segment as decomposable or not. Without the shape pool, no longer exists a second HSM build during the iteration stage. Instead two lists are used: the Non Segmented List (NDL), that contains the segments to be processed and the segments that are not labeled as decomposable and the Non Segmented List (NSL) that contains the segments that are going to be processed on the next iteration. So, at the end of the initialization stage, the NDL contains all the objects in the collection.

During its execution, the iteration stage traverses the

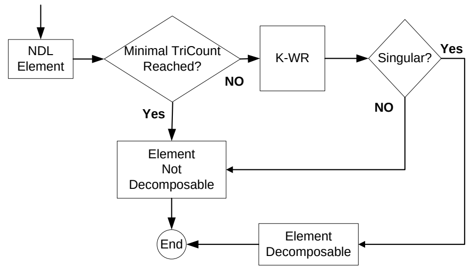


**Figure 6: Decomposition fluxogram of Adapted CaS.**

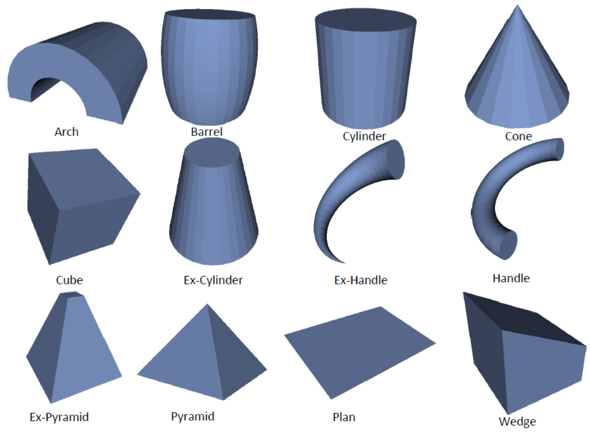


**Figure 7: Group of geons used on the Geons-augmented CaS.**

HSMs, level by level, that is, each new iteration corresponds to a level on the hierarchy. In order to support the segmentation, the iteration stage uses the NDL. In practice, each iteration of the stage is an iteration to the NDL. So, this stage starts on the first element of the NDL and verifies if is decomposable or not.

To segment the objects based on the collection where they belong, it is necessary to compare each sub-part with all other sub-parts. Figure 6 illustrates the steps for labelling a segment as decomposable or not. It first verifies if has reached the minimal triangle count by calculating the number of triangles between both child nodes and compares this with a previously defined value - the minimum triangle count. If the triangle count is below this threshold, the segment is flagged as non-decomposable. This step prevents from reaching the triangles of the mesh.

In case the minimum triangle count have not been reached, the element passes to the next step in the decomposition process. This step consists on the execution of a K-Within Range (K-WR) search. This algorithm returns the first K-elements whose signatures are within a predefined range and is used to verify if a segment is singular. For that end, two thresholds were previously defined: the similarity and similar count thresholds. The first threshold is used to verify if two objects are similar or not, to be similar the distance of two objects on the multidimensional space has to be less them the similar threshold, this is computed using the segments signatures and the distance between them. Then, the second threshold is used when comparing a segment with all the segments that are on the signature pool, it cannot have more similar segments than the similar count threshold. With this result it is verified if the segment is singular, to be singular the number of similar segments has to be above the similar count, if is that the case, then the segment is labelled as decomposable. If not, it labels as not decomposable.

If the result is the segment being decomposable, then it is going to decompose it, for that, it goes to the respective HSM and get his child nodes. After having the child nodes, it calculates their respective signatures, adds them to the signature pool and insert both nodes on the Non Segmented List (NSL), this list contains the segments that are going to be processed on the next iteration and removes the processed element from the NDL passing to the next element on the NDL that also will be processed. Also in the case of the result being not the decomposable it passes to the next element on the NDL.

An iteration ends when it reaches the end of the NDL but the iteration stage may ended or not. This depends on the NSL, if the NSL is not empty it means that are new segments to be processed. So, it appends all the elements on the NDL, remove them from the NSL and restarts a new iteration by vising again the first element on the NDL.

The iteration stage and the entire algorithm ends when it has reached the end of the NDL and the NSL is empty, meaning that there are no more segments to be decomposed. So, the approach ends by returning the entire collection of decomposed objects.

### 3.3 Geons-augmented CaS Decomposer

During the execution of the manual segmentation test using HFP, one of the complains of the users was that some objects were over segment. This happens because it was used the HFP approach. After executing the adapted CaS approach with different similarity and similar count thresholds we notice that this problem still happens on this approach, so, to overcome this limitation it was introduced a new feature. The introduction of the geons to verify if an object is decomposable or not.

The geons are simple 3D objects, like cylinders, cones, cubes. The theory proposed by Biederman on [Biederman87] called "Recognition by Components" states that, like English words that are constituted by a number of phonetics, complex objects can also be composed by these simpler 3D objects, that is, by segmenting complex objects we get these simpler objects. Thus we added a set of geons, partially depicted in Figure 7, to improve decomposition results according to human perception.

Since the CaS approach uses the HFP that uses primitives to calculate the cost of merging clusters and there are some

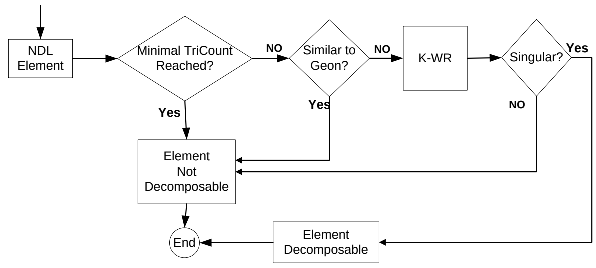


**Figure 8: Decomposition fluxogram for Geons-Augmented CaS.**

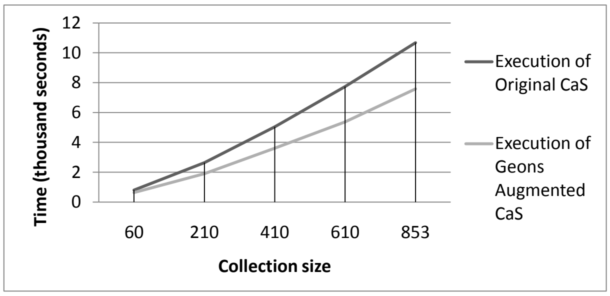


**Figure 9: Execution times for both approaches regarding collection size**

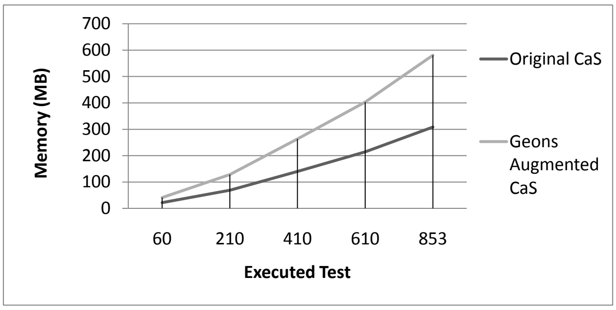


**Figure 10: Memory requirements for both approaches with respect to collection size.**

of the geons that are the same as these primitives, using these objects can helps to stop the segmentation when a geon is found. Figure 8 represents the decomposition fluxogram for the Geons-augmented CaS. The process is similar to the one described in previous section and illustrated in Figure 6. The main difference lays after the first comparison. If the minimal triangle count of the child nodes is above the minimum triangle count predefined, then segments are compared with the set of geons. This is done by comparing the signature of the segment with each precomputed geon signature. Then, if it is not similar, it proceeds as the previous description, that is, it executes the K-WR search. In the case of being similar to any of the geons, then the segment is labelled as not decomposable.

## 4 EXPERIMENTAL EVALUATION

To validate and evaluate the proposed approach implementation, we verified the efficiency and effectiveness of the algorithm. Thus, several tests were organized, from performance tests to tests with users which involved a sample group of twenty people.

### 4.1 Manual Segmentation Tests

This test was used to understand the human interpretation of an object, more precisely, how humans segment the objects. To that end we used a small set of randomly chosen objects of the Engineering Shape Benchmark (ESB) [S.Jayanti06]. A preliminary conclusion we made from this test was that familiar objects are easier to decompose than those people see for the first time. Other conclusion was that users segment objects consistently. But draw such conclusions was not the primary goal of the tests and further studies should be made to validate such observations.

Based on the results obtained in these tests it was possible to refine the approach and also make it automatic by defining the similarity and similar count thresholds. In order to accomplish that, the results of the manual segmentation were compared to the results produced by the Geons-augmented CaS using different thresholds by comparing the number of segments and assigning a classification to the results.

### 4.2 Execution time

The time of execution was obtained by executing the Geons-augmented CaS and the original CaS prototypes that were developed by implementing both approaches. It was used collections of objects with different amounts of objects. It is possible to conclude by observing Figure 9 that the time has a linear growing and comparatively, the geon-augmented CaS presents better results than the original CaS. It was also possible to observe that the signatures computation is time consuming and is where most of the time is spent.

### 4.3 Memory requirements

A similar study was performed to the memory requirements, as shown in Figure 10 it is used more memory on the geons augmented CaS than on the original CaS. This happens because it is used more lists in order to reduce the execution time. The memory is nowadays a cheap resource and has been increased over the years. So, having to spend more memory but as a result we get a approach faster and with better results to the users is a good trade off between these two important measures.

### 4.4 Evaluation with users

To prove the results quality, it was performed a test with users where they had to compare pairs of results.

The users were asked first to compare the results of the geons augmented with the ones produced by the original

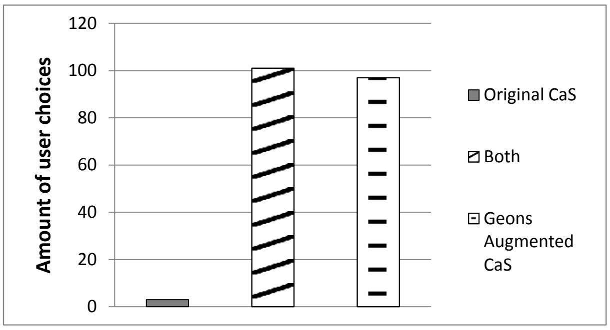


**Figure 11: Users choice of best segmentation between original CaS and Geons-augmented CaS.**

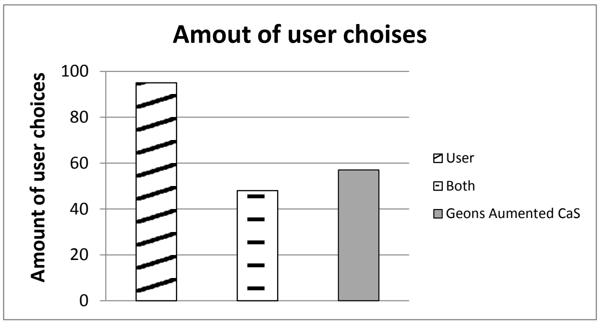


**Figure 13: Users choice of best segmentation between men-made segmentation and Geons-augmented CaS.**

**Figure 12: Segmentation of an object with the original CaS (left) and the geons-augmented CaS (right).**

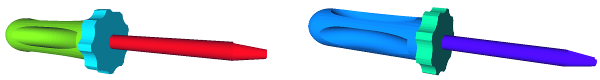

**Figure 14: Segmentation of an object made by humans (left) and through geons-augmented CaS (right).**

CaS. On this test the results produced by Geons-augmented CaS approach proved to be more meaningful to the user as is shown on Figure 11 where most of user chosen both results or the geons augmented approaches but rarely only the original CaS. As shown in Figure 12 the segmentation result of an object produced by the original CaS approach has three more segments than the segmentation result using Geons-augmented CaS approach. These three more segments produce an over-segmentation according to user's perception.

From these results we conclude that the Geons-augmented CaS approach, according to the users, produces better results than the original CaS approach.

Additionally, we asked users to compare the results produced by the geons-augmented CaS with objects manually segmented by humans in a previous test. As shown in Figure 13, half of the users have chosen their results while the other half have chosen both results or the geons-augmented CaS. Indeed, as shown in Figure 14 the segmentation of an object produced by the the manual segmentation has very similar results when performed with geons-augmented CaS approach.

This analysis allow us to conclude that besides having some objects where the segmentation is not the best for the user, the majority of the collection objects results have been chosen by the user as the same or better than the one he chose. We can then conclude that manual segmentation prevails over the automatic segmentation algorithm, but often results are quite similar.

## 5 CONCLUSIONS and FUTURE WORK

We presented an extension of the CaS algorithm that segments a collection of 3D objects simultaneously in an automatic manner. The algorithm produces a meaningful segmentation regarding human perception, according to the results of the experimental evaluation. We also improved time complexity comparatively to the original CaS. Additionally, the proposed algorithm avoids over-segmentation by using geons as primitives.

Different tests were performed in order to test the efficacy and effectiveness of the approach and the quality of the produced results. From these tests we have concluded that the geons-augmented is faster than the original CaS but spends more memory. It was also clear that the presented solution produces results that are meaningful for humans. However, we consider that using HFP as a basis for the decomposition might be a limitation. Thus, it is necessary on future work to study the possibility of using other hierarchical based segmentation approaches.

It seems also promising to experiment other signature besides the rotation invariant spherical harmonics shape descriptor. Also, as one of the characteristics of these descriptors is that are not scale invariant, it is necessary to perform a deeper study on the geons to avoid having to use multiple geons on different scales.

In the future we believe that the proposed approach might be improved to develop into a robust, stable and scalable solution for the automatic decomposition of 3D object collections.

## 6. ACKNOWLEDGEMENTS

The work described in this paper was partially supported by the Portuguese Foundation for Science and Technology (FCT) through the project 3DORuS, reference PTDC/EIA-EIA/102930/2008 and by the INESC-ID multiannual funding, through the PIDDAC Program funds.

## 7. REFERENCES

[Agathos07] A. Agathos, I. Pratikakis, S. Perantonis, N. Sapidis, and P. Azariadis. 3d mesh segmentation methodologies for cad applications. *Computer-Aided Design and Applications*, 4(6):827–841, 2007.

[Biederman87] I. Biederman. Recognition-by-components: A theory of human image understanding. *Psychological Review*, 94:115–147, 1987.

[Chen09] Xiaobai Chen, Aleksey Golovinskiy, and Thomas Funkhouser. A benchmark for 3D mesh segmentation. *ACM Transactions on Graphics (Proc. SIGGRAPH)*, 28(3), August 2009.

[Falcidieno06] M. Atteneand B. Falcidieno and M. Spagnuolo. Hierarchical mesh segmentation based on fitting primitives. *Visual Computer*, 22(3):181–193, 2006.

[Ferreira09] A. Ferreira, S. Marini, M. Fonseca M. Attene, M. Spagnuolo, J. Jorge, and B. Falcidieno. Thesaurus-based 3d object retrieval with part-in-whole matching. *International Journal of Computer Vision*, 2009.

[Golovinskiy09] A. Golovinskiy and T. Funkhouser. Consistent segmentation of 3d models. *Computers & Graphics*, 33(3):262–269, 2009.

[Katz03] S. Katz and A. Tal. Hierarchical mesh decomposition using fuzzy clustering and cuts. In *ACM SIGGRAPH 2003 papers*, pages 954–961, NY, USA, 2003. ACM.

[Katz05] S. Katz, G. Leifman, and A. Tal. Mesh segmentation using feature point and core extraction. *The Visual Computer*, 21(8–10):649–658, 2005.

[Kazhdan03] Michael Kazhdan, Thomas Funkhouser, and Szymon Rusinkiewicz. Rotation invariant spherical harmonic representation of 3d shape descriptors. In *Proceedings of the 2003 Eurographics/ACM SIGGRAPH symposium on Geometry processing*, SGP '03, pages 156–164, Aire-la-Ville, Switzerland, Switzerland, 2003. Eurographics Association.

[Mortara03] M. Mortara, G. Patane, M. Spagnuolo, B. Falcidieno, and J. Rossignac. Blowing bubbles for multi-scale analysis and decomposition of triangle meshes. *Algorithmica*, 38(1):227–248, 2003.

[Mortara06] M. Mortara, G. Patane, and M. Spagnuolo. From geometric to semantic human body models. *Computers & Graphics*, 30(2):185–196, 2006.

[Shamir08] A. Shamir. A survey on mesh segmentation techniques. *Computer Graphics Forum*, 27(6):1539–1556, 2008.

[Shapira08] L. Shapira, A. Shamir, and D. Cohen-Or. Consistent mesh partitioning and skeletonization using the shape diameter function. *The Visual Computer*, 24(4):249–259, 2008.

[S.Jayanti06] S.Jayanti, Y. Kalyanaraman, N. Iyer, and K. Ramani. Developing an engineering shape benchmark for cad models. *Computer-Aided Design*, 38(9):939–953, 2006.

[Tierny07] J. Tierny, J.-P. Vandeborre, and M. Daoudi. Topology driven 3d mesh hierarchical segmentation. In *International Conference on Shape Modeling and Applications*, pages 215–220, 2007.

[Zuckerberger02] E. Zuckerberger. Polyhedral surface decomposition with applications. *Computers and Graphics*, 26(5):733–743, 2002.