Int. J. Simul. Multidisci. Des. Optim. **13**, 26 (2022)

https://doi.org/10.1051/smdo/2022011

IJSMDO
Available online at:
https://www.ijsmdo.org

Computation Challenges for engineering problems

RESEARCH ARTICLE OPEN ACCESS

Feature recognition and machine learning in finite element models through a clustering algorithm

Sunil Premkumar*, Davidson Jebaseelan, and Krishnamoorthy Annamalai

School of Mechanical Engineering, VIT University, Chennai 600127, Tamilnadu, India

Received: 30 January 2021 / Accepted: 10 September 2022

Abstract. The feature identification of the CAD model is a significant task in any CAD algorithm. Enormous computational time, huge memory allocation, lack of understanding in computational geometry, etc., are some of the complications faced while implementing the feature recognition algorithms. This paper represents a clustering algorithm procedure in finite element models, which is the predominant component in analysis methodology. This study performs the clustering of data groups through density-based clustering algorithms such as mean shift clustering and K-means clustering algorithm. In addition to that, experimental evaluation based on the structured algorithm procedure for identifying the features of CAD geometries is investigated. Finally, the study evaluates the performance of the proposed structured algorithm and its efficiency in terms of both computational time and computational memory.

Keywords: Mean-shift clustering / k-mean clustering / clustering algorithm / centroid / index-based sorting

1 Introduction

Identification of features based on the clustering of data sets is a reliable method in CAD geometry feature recognition. For any CAD geometry, meshing or segmenting into meaningful small parts or features is fundamental for identifying its 3D shape [1]. The research undertaken by [2] represents a convolutional neural network (CNN)-inspired neural network trained with a custom dataset. The network demonstrates the robustness of the approach and its potential to adapt to augmented datasets in the future. Mostly, in dataset matching, researchers are interested in reducing the data points using mathematical concepts such as single value decomposition (SVD) and Principal Component Analysis (PCA) [3]. Point set registration is one of the methods which uses a rigid and non-rigid transformation to match the features exactly with the database [4,5]. Several algorithms in the concept of point set registration have been evolved with different concepts of implementation such as least square method [6], iterative closest point, and robust point matching [7], etc. [8]. Another method is matching through the connectivity of surfaces of the CAD geometry through graph matching algorithms, and it uses adjacency matrices explicitly. Some concepts have been evolved in subgraph isomorphisms such as GADDI, GraphQL, Spath, QuickSI, Ullmann, and VF2, which are used to find specific pattern matching in corresponding CAD geometry [9].

There are several partitioning and hierarchical algorithms for finding out the clusters. Mean shift clustering is one of the effective methods for clustering data from noise signals. Tang et al. [10] performed the experiment with the help of a mean shift clustering algorithm with weighted euclidean distance. The partitioning algorithm includes two basic steps. The first one is the determination of k representatives, minimization of an objective function, and cluster assignment, which is nearest to the cluster points [11,12]. In order to avoid the number of iterations, the computational complexity density-based algorithms for discovering clusters were used for sorting them. But these concepts are time-consuming and need more computational memory [13]. Implementation of machine learning in a database needs a better algorithm to overcome the problem of data storing, pattern recognition, and implementation of stored instructions with proper priority [14]. In this study, the clustered data points of meshed geometry are extracted and stored in a database which is compared with the imported model consisting of finite element software. The issues with pattern matching or feature matching have a degree of difficulty in storing data points and maintaining efficiency in sorting matched patterns from databases having an enormous number of data sets.

The algorithm should possess the ability to accurately model the transformation that requires the alignment of stored point sets with a traceable computational complexity. It should have the ability to handle high dimensional point sets and robustness to deal with degradations such as noise outliers and missing points

* e-mail: sunilpremkumar7@gmail.com

that occur due to imperfect image acquisition and feature extraction. This work makes an effort to develop a sequence of algorithm procedures to handle the extraction of data effectively even though the extracted image pattern involves considerable noise.

2 Methodology

2.1 Point seeding algorithm

In CAD software, the representation of geometry can be B-Rep or voxel or point cloud data, and the approximate structure of the model is plotted by the pixel intensity. A point represents the modification in the continuity of outline in any structure. So the points are seeded in the geometry lines based on their characteristics, such as free or fixed or change in curvature. For instance, a point is considered at the location where the tangency of the line changes or the line terminates. This behavior is similar to a wireframe model, which exists in almost all modeling and analysis software, and will give a clear representation of a model with only points. These points are not geometrical point sets that are already presented in the CAD geometry, and they are temporary points created based on the geometric features.

2.2 Centroid based clustering algorithm

The data is organized and aligned in their principal Eigen vectors, which represent the direction of point distribution. Based on the distribution, the model is partitioned by principal directions (i.e., Eigen values). The principal components are evaluated by using the principal component analysis approach. Orienting a geometric model based on its principal axis helps to avoid the dependency of local coordinates. The scattered data points which represent the whole model are clustered by a centroid-based clustering algorithm which is a mean-shift clustering method. Each data point is clustered based on the radius (p), which is a function of the model size and the number of data points. Initially, the clustering process starts from a distributed direction. In each iteration, the mean is calculated for the cluster and shifted to the newly calculated location. The iteration will continue until the centroid point is fixed. The coordinates of each fixed point are stored, and the whole iteration is repeated in different directions. Another set of fixed points are collected and stored in a database.

A cluster (k) includes the randomly selected point, which is considered as a center point, and the points which are in the range of the cluster radius from the center point. The cluster will propagate until there are no data points found across the spherical volume of specified radius p. The whole algorithm iterates throughout all the data points and merges with any one of the clusters. The second step involves the calculation of the centroid (C) of each cluster, which is found out by calculating the mean of all the nodal coordinates in a cluster set of data points (k). From this step, each cluster in a cluster set (K) is represented by the mean centroid. The centroid point represents the feature of the model. The procedure of the clustering process is shown in Figure 1.

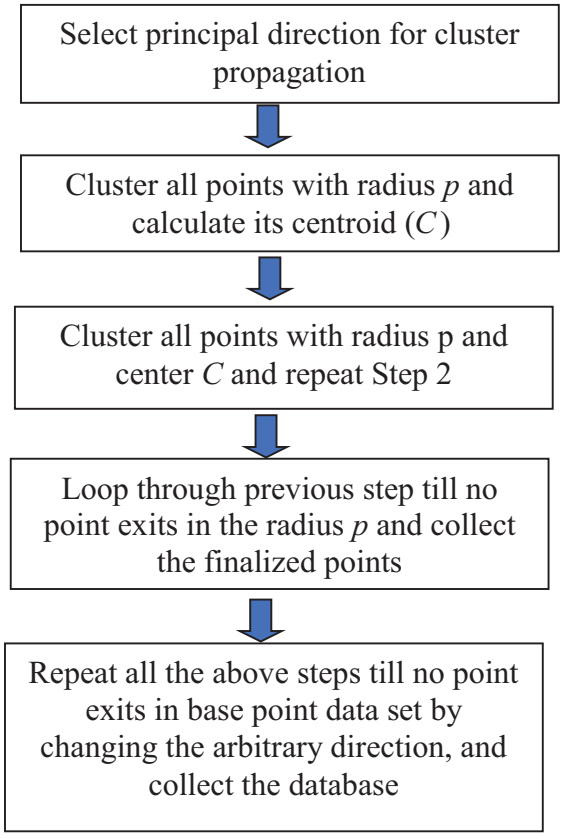

Fig. 1. Clustering Procedure.

2.3 Data sorting algorithm – index-based sorting

For querying the existence of the loaded CAD data in the library database, the sorting procedure needs to be more simple and reliable. This is because this section requires enormous computational time for the iteration of each model in the library database. The Index-based sorting algorithm is chosen to sort out the required geometry. The clusters of data points are sorted by the numerical index, which represents the mean cluster dataset of models. In the data library, several models may have the same index because of the coincidence of cluster numbers. But, the index of clusters inside each cluster set is rarely similar. To simplify the searching based on the sorting through each cluster index in the data, the indexing algorithm sorts out clusters with an enormous number of data points which represents the complex feature of the model. This method gives a way of identifying the model based on some specialized features in the model. The algorithm of querying and sorting is explained in Figure 2.

2.4 Clustering of data points and node seeding

Based on the geometrical lines, the data points are seeded in the CAD geometry. The list of data sets is arranged in a random order in the data set (D). For instance (as shown in Fig. 3), the 730 data points are used to represent the whole feature of the geometric model, which is already meshed into finite elements (Fig. 4). Dealing with more data points

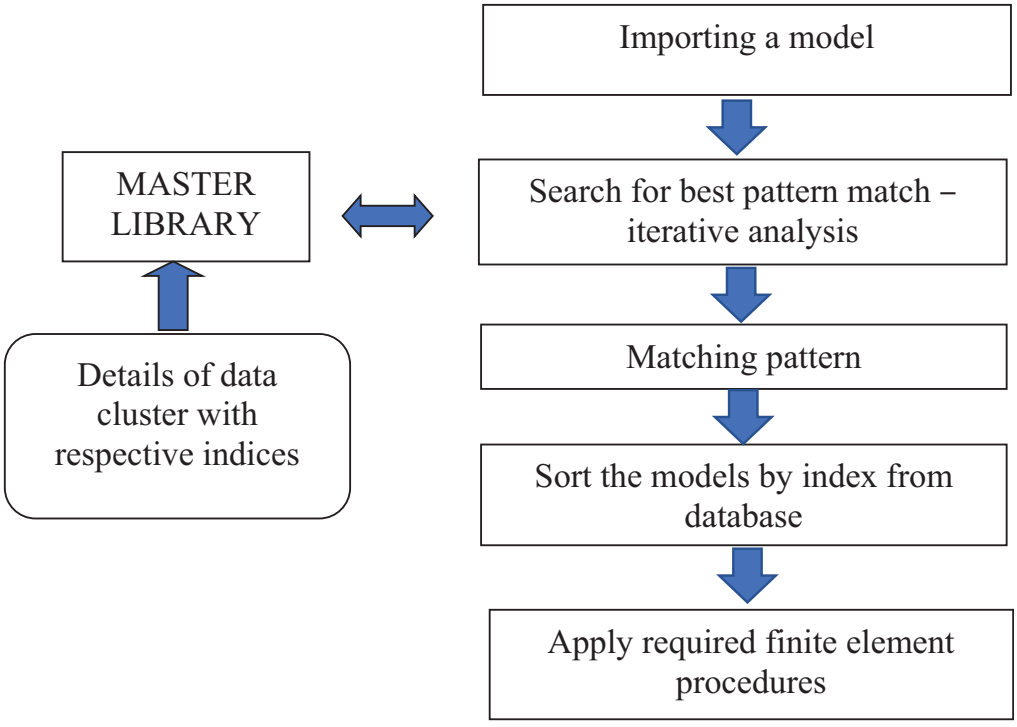

Fig. 2. Sorting Procedure.

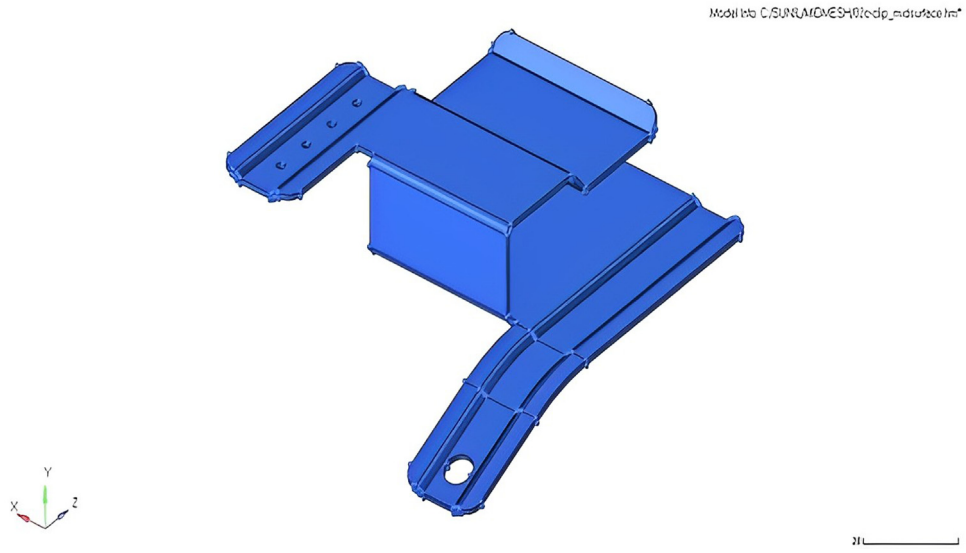

Fig. 3. Model for Consideration of size 470 kb.

may cause excessive computational time. The clustering of data points is performed to avoid complexity. The clustering of data points (Fig. 5) is done by the regressive propagation of a spherical volume with a radius (p) which is fully dependent on the size of the model, as mentioned in Figure 1.

In the second step, the clustered data sets (K) are reduced to a mean centroid data-point by calculating the mean of all the coordinates of the data points in a single cluster set. This results in the mean centroid coordinates of the cluster points (Fig. 6).

Each model in the database is reduced to data points and then to clusters with a respective index which forms a data library.

2.5 Iteration of data points

The model required to be compared is imported and followed by the implementation of a clustering algorithm to evaluate the cluster sets. Then the cluster's data is queried in the library database by means of its indices of data sets followed by a critical cluster index. The critical cluster index has an

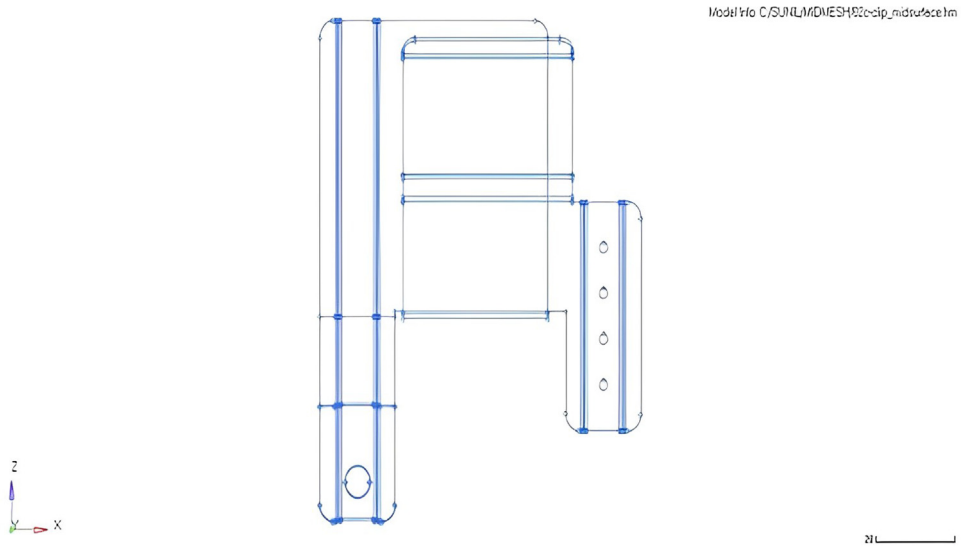

Fig. 4. Node seeding from geometry line.

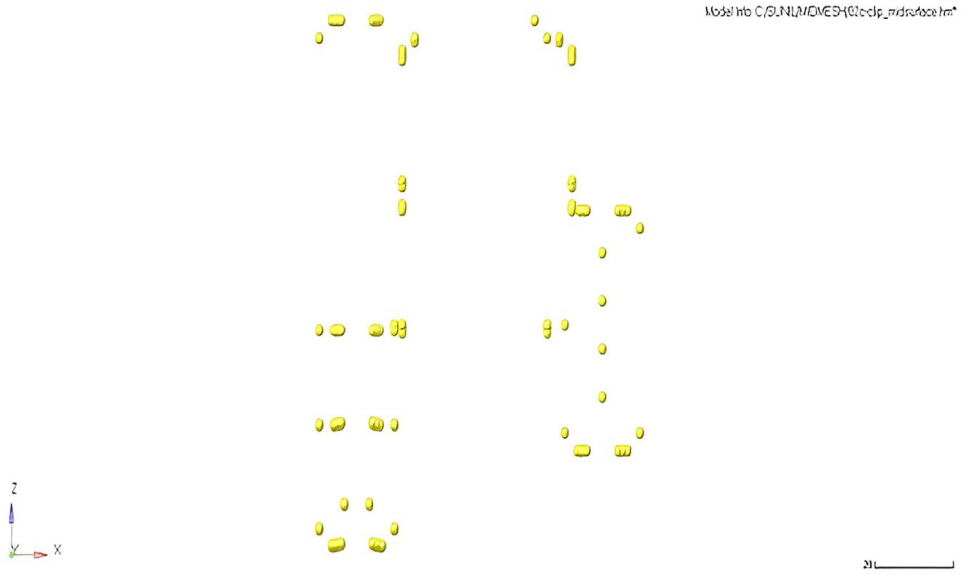

Fig. 5. Clustering of data points.

enormous number of data points in the set. It covers the critical features of the model, and hence it reduces the computational sorting out time because of the critical matching features.

2.6 Pattern matching and assigning parameters

Finally, all the calculated cluster mean points are compared with a tolerance of spherical volume radius (p) with one specific model data set from each database. If all the criteria are matched, the model file is identified with a corresponding model from the database.

2.7 Storing data by voronoi principle

Storing involves a procedure of steps or instructions which plays a significant role in creating an artificial intelligence system. In this section, two techniques are used to avoid computational complexity and reduce computational time. The first approach involves retrieving the small part where the local modification is performed in the CAD geometry. This method requires a model splitting algorithm that efficiently splits the model into an equally difficult level of partitions. In this study, the principle of the Voronoi diagram is used, which gives the partition of the plane based on the sets of points of the CAD geometry, as shown in Figure 7.

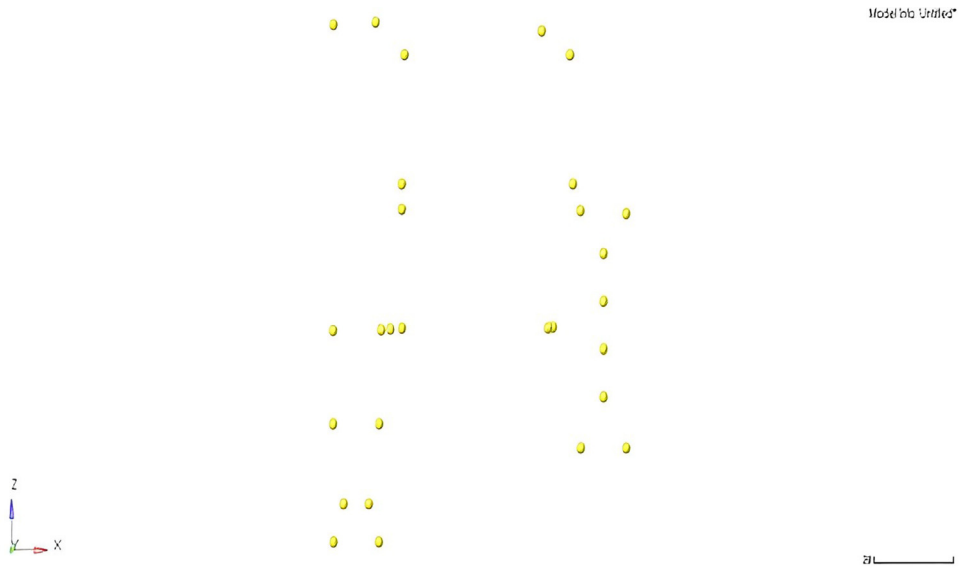

Fig. 6. Centroid nodes/points of clusters sets.

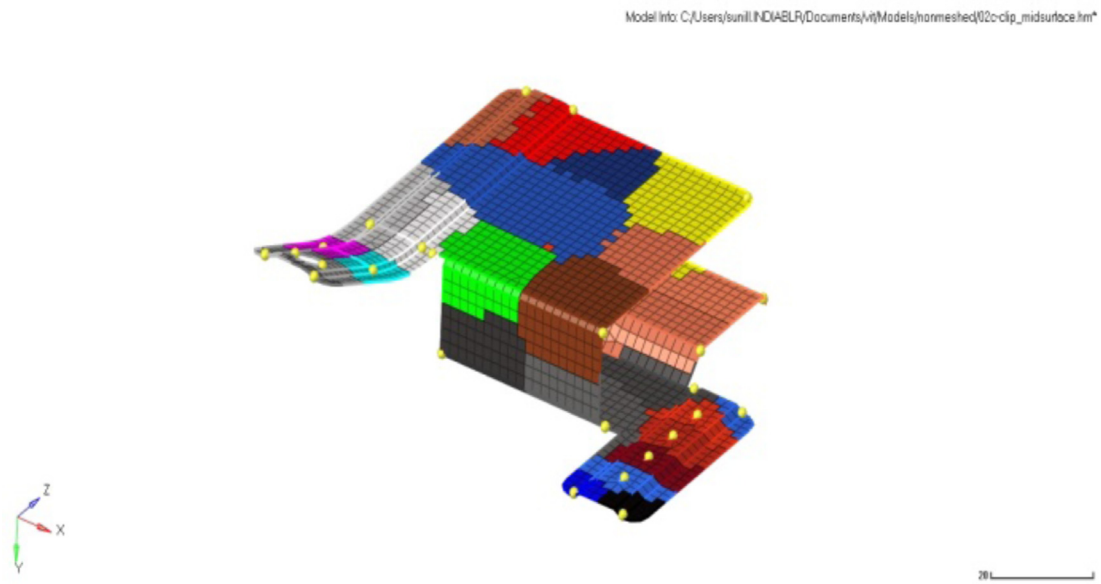

Fig. 7. Partition of geometry by Voronoi Principle.

The complexity of geometric features can be identified by the proximity of mean clustering points. If the proximity is less, the geometric complexity is high. If the model size is large and has fewer clustering mean points, the model is indicated as large in shape and less complex in profile. To implement the Voronoi principle in finite element partitions, the mean clustering points are base points, and each mean point is supposed to grab the elements surrounding it. All the elements in the model come under one of the mean clustering points in the set. For example, a mean point $p1$ will capture the elements around it with a spherical radius limit and then finds another mean point $p2$ which is continued. In the next iteration, the spherical radius is increased by a small increment value which increases the spherical volume.

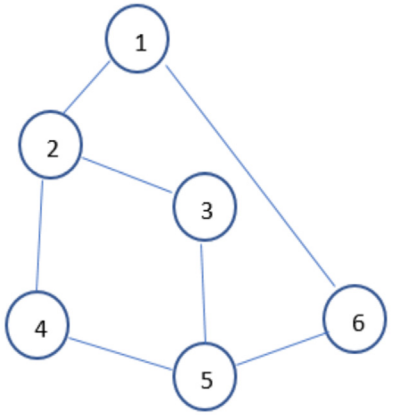

Fig. 8. Graphical representation of finite element partitions.

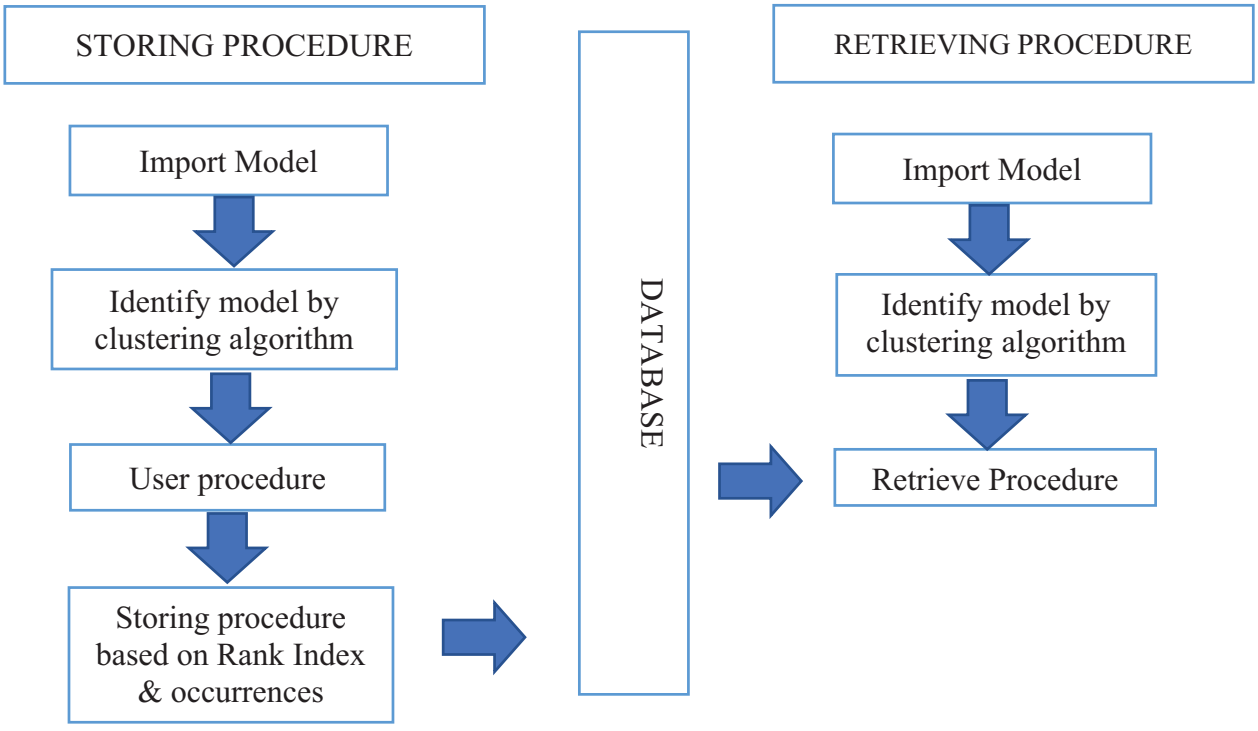

Fig. 9. Real-time Data storage.

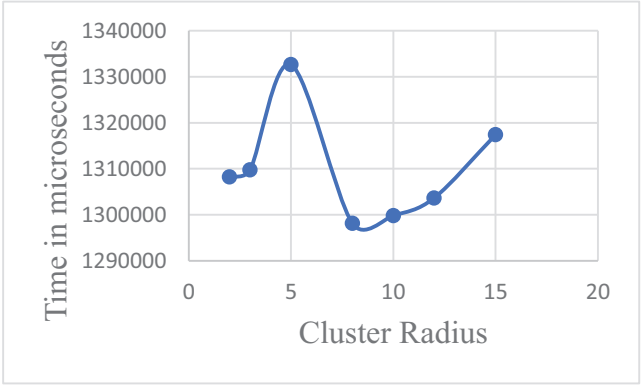

Fig. 10. Time variation of Datapoint collection for different cluster radius.

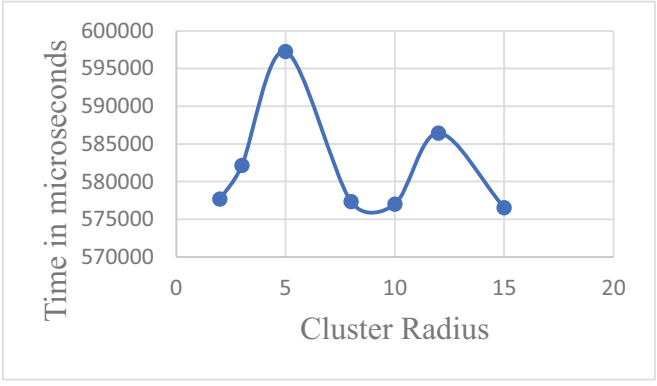

Fig. 11. Time variation of clustering procedure for different cluster radius.

For understanding the exact principle of the Voronoi diagram, the points should expand their limit simultaneously. But for the simplicity of programming, the cluster points expand their limits one by one with a reduced incremental value. The incremental value is directly proportional to the finite element separation. Each partition can be stored as a graphical representation. For example, the model which is subjected to local modification can be stored as a local partition that is separated by the Voronoi principle.

The graph, as shown in Figure 8, indicates the numerical representation of the partitioned elements. The modification may be done within one partition or it may be sets of partitions like {1, 2, 3}, {1, 2, 4, 5}, {3, 5, 6} etc. The details of modification in a single partition or a set of partitions can be stored in a database and used for retrieval. On some scenarios, the retrieving procedure involves a set of instructions such as changes that happened in {1, 2, 3} followed by {3, 5, 6} in the graph. In this case, the latter set needs the input of the previous set. Hence, in this method of storing procedures, the steps of procedures are stored as steps for the ease of retrieving.

2.8 Real-time procedure storing principle

Figure 9 indicates the real-time storing methodology in which the modifications are stored. When the operation is performed, this leads to less computational memory

Fig. 12. Time variation of sorting procedure for different cluster radius.

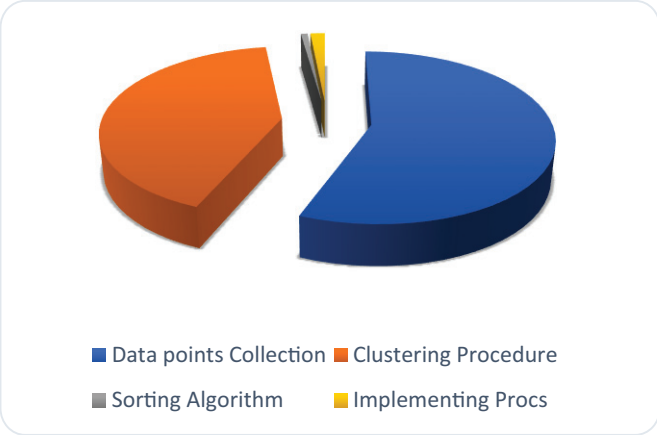

Fig. 13. Time allocation for different algorithm in feature recognition process.

compared to the former. Once the required data is queried from the database, the real-time storage allows the user to store the steps of procedures in separate files for the particular geometry with the respective procedure name, which is user-defined at the time of operation. These steps result in the user enhancement to select the appropriate procedure to retrieve from the database.

3 Results and discussion

The performance evaluation of different steps of the structured algorithm procedure is plotted below.

Generally, from the observation of different data sets, the approximate percentage of time allotted for carrying out different steps of an algorithm is mentioned below.

From Figure 10, it is revealed that the point collection algorithm is independent of the cluster radius and the performance of the data points seeding algorithm depends on the geometry complexity.

Figure 11 reveals that the general performance is inversely proportional to the cluster radius, which highlights its significance in the algorithm's performance. Moreover, the spherical volume of the algorithm can produce a fewer number of cluster sets, which affects the accuracy in capturing the feature of the CAD geometries. For the sorting algorithm, the performance is inversely proportional to the cluster radius. The reason behind this is also the same as the previous case, because of the reduced number of cluster sets. The allocation of time for all the four algorithms in the structured procedure is mentioned in Figure 13.

So finally, the performance of the structured procedure algorithm depends mostly on the point collection algorithm and the clustering algorithm. The point seeding algorithm may take more time when it faces geometrical complexity.

The analysis results of creating the data points for ten random models with varied sizes ranging from 500 kb to 25 Mb are shown in Figure 14. It clearly indicates that the seeding data point is dependent on the size of the model.

The performance evaluation of various model sizes for the structured algorithm procedure is shown in Figure 15.

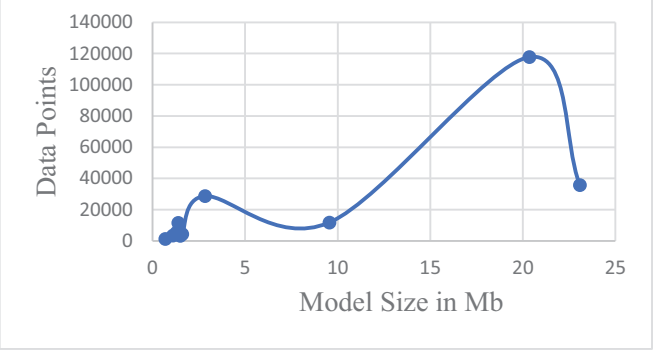

Fig. 14. Comparison of model size with data points count.

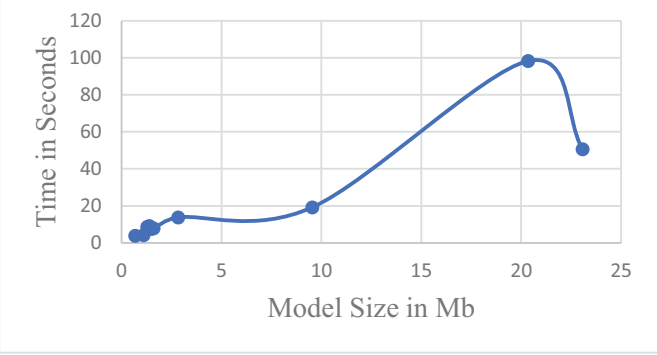

Fig. 15. Comparison of model size with computation time.

It indicates that the computational time depends on the number of data points which are not based on the model size. Figures 13, 14 and 15 show similarities to each other.

The performance evaluation of the structured algorithm procedure based on the mean cluster data points is shown in Figures 16. It depicts that the computational time

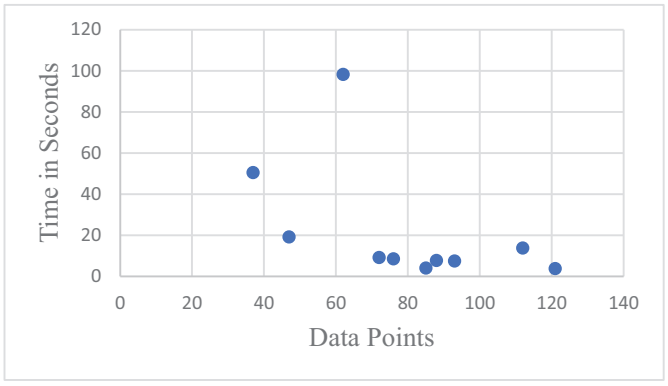

Fig. 16. Comparison of cluster mean data points with the total time for the algorithm.

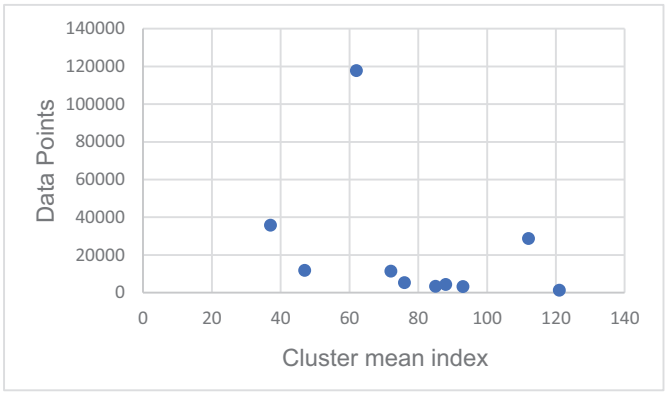

Fig. 17. Comparison of cluster mean index with data points.

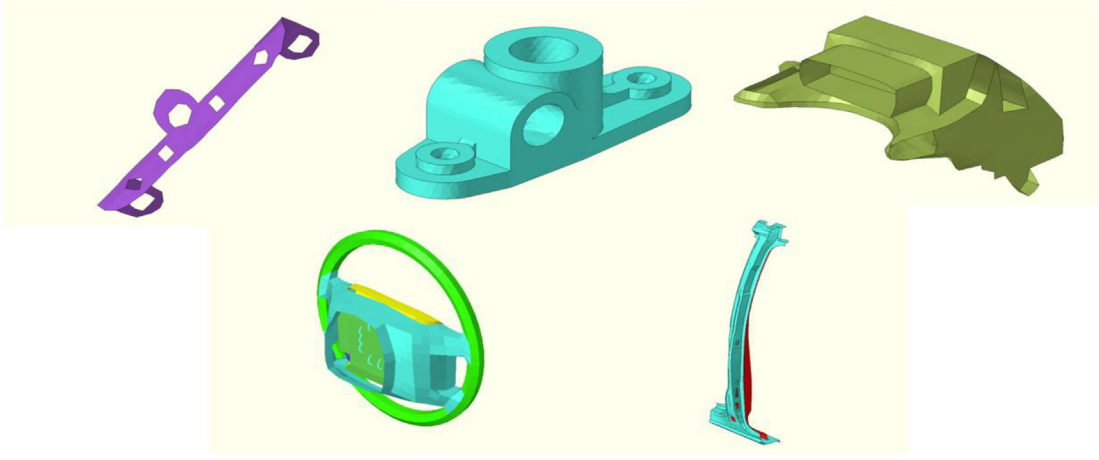

Fig. 18. Geometries used for evaluation (Model 1 to Model 5 from left).

Table 1. Performance of algorithms for various models and their computational time with different steps of the algorithm.

Model Name	SIZE (MB)	Point Seeding (Sec)	Clustering (Sec)	Sorting (Sec)	Total Time (Sec)
Model 1	1.602	4.491	3.149	0.185	7.835
Model 2	2.829	7.012	6.531	0.245	13.79
Model 3	9.544	12.655	6.482	0.0977	19.243
Model 4	20.355	43.879	54.09	0.319	98.308
Model 5	23.077	28.337	22.066	0.113	50.533

depends not only on the number of data points but also on the distance and orientation of the data points. This clearly defines the time consumption for the algorithm based on the geometrical complexity.

The performance analysis of the clustering algorithm is depicted in Figure 17 by comparing the cluster mean index with a number of data points of various geometries. The similarity of Figures 16 and 17 clearly indicates that the variation of time is because of the clustering algorithm. The performance of algorithms of various geometries Figure 18 and their computational time with different algorithms are given in Table 1.

4 Conclusion

In this study, the structured algorithm procedures for identifying the feature of CAD geometries are explained, and their performance evaluations are carried out. The experimental evaluation indicates that the significance of the index-based sorting algorithm procedure is effective in terms of both computational memory and computational time.

From a machine learning point of view, the partitioning of models and real-time procedure storing in a database for retrieving the previous operations in the specific model reduces the complexity, which makes it simpler for finite element feature recognition. The study makes significant inroads in finite element automation in heavy scale industries where repetitive models/processes can be identified with minimum iterations. It also deals with the finite element model partitioning, storing, and retrieving of the processes/operations.

References

1. E. Kalogerakis, A. Hertzmann, K. Singh, Learning 3D Mesh segmentation and labelling, ACM Trans. Graph. **29**, 3 (2010)
2. A. Van Biesbroeck, F. Shang, D. Bassir, CAD model segmentation via deep learning, Int. J. Comput. Methods **18**, 4–7 (2021)
3. S.M. Holland, Principal components analysis (PCA) (Departmentof Geology, University of Georgia, Athens, GA, 30602–2501, 2008)
4. P. Wang, P. Wang, Z. Qu, Y. Gao, Z. Shen, A refined coherent point drift (CPD) algorithm for point set registration, Sci. China Inf. Sci. **54**, 2639–2646 (2011)
5. A. Myronenko, X. Song, Point set registration: Coherent point drift, IEEE Trans. Pattern Anal. Mach. Intell. **32**, 2262–2275 (2010)
6. J.E. Mottershead, C. Mares, S. James, M.I. Friswell, Stochastic model updating: part 2—application to a set of physical structures, Mech. Syst. Signal Process. **20**, 2171–2185 (2006)
7. Myronenko, X. Song, M.A. Carreira-Perpinán, Non-rigid point set registration: Coherent point drift, In *Advances in Neural Information Processing Systems* (2007), pp. 1009–1016
8. Z. Li, J. Wang, Least squares image matching: a comparison of the performance of robust estimators, ISPRS Ann. Photogram. Remote Sens. Spat. Inf. Sci. **2**, 37 (2014)
9. J. Lee et al., An in-depth comparison of subgraph isomorphism algorithms in graph databases, in *Proceedings of the VLDB Endowment* (VLDB Endowment, 2012), vol. 6, pp. 133–144
10. D. Tang, J. Man, L. Tang, Y. Feng, Q. Yang, WEDMS: An advanced mean shift clustering algorithm for LDoS attacks detection, Ad Hoc Networks **102**, 2–3 (2020)
11. J.A. Hartigan, M.A. Wong, Algorithm AS 136: A k-means clustering algorithm, J. Royal Stat. Soc. Ser. C **28**, 100–108 (1979)
12. T. Kanungo, D.M. Mount, N.S. Netanyahu, C.D. Piatko, R. Silverman, A.Y. Wu, An efficient k-means clustering algorithm: analysis and implementation, IEEE Trans. Pattern Anal. Mach. Intell. **24**, 881–892 (2002)
13. M. Ester, H.P. Kriegel, J. Sander, X. Xu, A density-based algorithm for discovering clusters in large spatial databases with noise, Kdd **96**, 226–231 (1996)
14. G. Kumar, P.K. Bhatia, A detailed review of feature extraction in image processing systems, in Advanced Computing & Communication Technologies F (ACCT), 2014 Fourth International Conference on (IEEE, 2014), pp. 5–12

Cite this article as: Sunil Premkumar, Davidson Jebaseelan, Krishnamoorthy Annamalai, Feature recognition and machine learning in finite element models through a clustering algorithm, Int. J. Simul. Multidisci. Des. Optim. **13**, 26 (2022)