A THREE-LEVEL SIGNATURE BY GRAPH FOR REVERSE ENGINEERING OF MECHANICAL ASSEMBLIES

Marina Bruneau
Roberval Laboratory, UMR 7337
Sorbonne Universités
Université de technologie de Compiègne
marina.bruneau@utc.fr

Alexandre Durupt
Roberval Laboratory, UMR 7337
Sorbonne Universités
Université de technologie de Compiègne
alexandre.durupt@utc.fr

Laurent Vallet
Lionel Roucoules
Jean-Philippe Pernot
Arts et Métiers Paris Tech, Aix-en-Provence
CNRS, LSIS
{laurent.vallet, lionel.roucoules, jean-philippe.pernot}@ensam.eu

ABSTRACT

Several approaches exist to provide Reverse Engineering solutions on mechanical parts. Mechanical assemblies and the expertise information retrieved at the same time with the model geometry are not really taken into account in the literature. Thus, the main challenge of this contribution is to propose a methodology to retrieve the Digital Mock-Up of a mechanical assembly from its meshed data (from digitalization). The output DMU consists of expertise information and parameterized CAD models. The methodology proposed relies on a signature by a three-level graph. It enables to provide an adequate level of details by identifying the corresponding functional surfaces in meshed data. The first-level graph is a connectivity graph; the intermediate level is the same as the first with the geometric type of face added to each node (plane, cylinder and sphere) and the deepest level corresponds to a precedence graph. This one provides information such as functional surfaces and position between them (perpendicularity, coaxiality etc.). The solutions developed and the results are presented in this paper.
The methodology is illustrated thanks to an industrial use-case with a scan of an assembly with a connecting rod and a piston. The conclusion and perspectives will complete this paper.

KEYWORDS

Reverse Engineering, CAD, signature, shape descriptor, graph comparison

1. INTRODUCTION

Reverse Engineering (RE) is defined as a process starting from a physical product and ending to digital representation in two or three dimensions. It can be used at several steps in the product lifecycle and in a routine way. Vinesh and Fernandes [1] present in Figure 1 the link between CAD and CARE which is Computer-Aided Reverse Engineering. Most of CARE solutions propose to rebuild the CAD model with geometry without any possibility to change

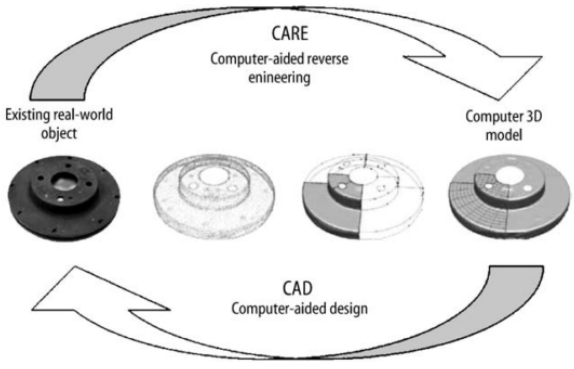


Figure 1 Reverse Engineering process from [1]

parameters. Retrieving the digital model thanks to product information expertise is a current key issue.

Scanning an assembly with several components is time consuming, a fortiori when it is necessary to scan separately each component. Then, processing the points cloud is tedious especially when we wish to get back a CAD model which can be reused in the engineering process. Thus heterogeneous data integration could save time by enriching information useful to digital mock-up reconstruction. These heterogeneous data belong to the product and can be design drafts, points cloud, CAD models, pictures etc. By scanning assemblies without disassembly, data are not complete. So, scanned components from mechanical assemblies need to be identified automatically in spite of their wholeness. The use of a Knowledge DataBase (KDB) could be helpful in order to identify incomplete data. The presence of CAD templates in the database could also enable to get back a parametrized model of each component. Several use-cases and applications of our RE activity have been addressed in previous paper [2].

A methodology called HDI-RE (Heterogeneous Data Integration for Reverse Engineering) has been developed. It is divided into three steps: data segmentation, data signature and signature comparison (as described in Figure 2). This paper focuses only on the data signature and signature comparison steps which enable to retrieve a parametrized CAD model in output of the RE process. The data signature aims at disposing a set of characteristics (extracted from segmentation step) in order to describe appropriately the data. The shape descriptor studied is a graph-based method. Then the comparison step enables to identify the different mechanical components inside the graph-based signature thanks to a knowledge database composed of graph-based signatures.

First, we will present a state-of-the-art in the shape matching domain for graph-based signature. Thanks to related works in the Knowledge-Based Engineering (KBE) domain, we will have an overview of solutions using knowledge database and how these solutions can inspire our RE methodology. In second time, we will introduce our graph-based signature process. In the fourth part, we will expose an applicative scenario with a mechanical assembly. To finish, we will conclude and present our future work.

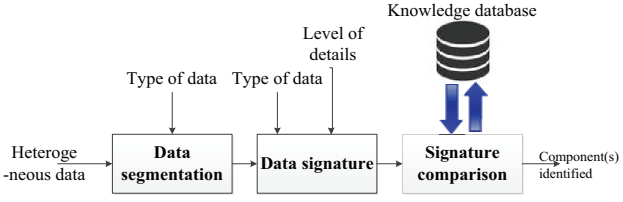


Figure 2 HDI-RE, the global process.

2. RELATED WORK

In this section, we will start by presenting the different shape descriptors and in particular graph-based methods. Then we will examine the related works in the KBE domain which supports routine design. In a previous work [2], a state-of-the-art about Reverse Engineering and segmentation techniques for 2D and 3D data has been performed. We consider that we start this signature step with a segmented data (2D or 3D).

2.1. Graph-based shape descriptors

Tangelder and Veltkamp [3] have performed a classification of all shape retrieving methods. For graph-based techniques, they consider three categories: model-graph, skeleton and Reeb graph. Model-graphs are mostly used to describe 3D data. It concerns non-oriented graphs which enable to describe a 3D model such as Boundary Representation (B-Rep). The graph's nodes and edges correspond respectively to the B-Rep faces and to the links between its faces. El-Mehalawi and Miller [4] call it attributed graph. They associate to graph edges as much information as possible concerning object's surfaces and edges. Graph represents component topology beside the graph attributes depict component geometry. In the literature, this type of graph is also known as Face Adjacency Graph (FAG). Ma et al.[5] use it to compare CAD models structures. In traditional FAG, each node contains geometric information about each face. They propose to add other information such as geometric information about the neighbor faces of the current face. It reduces time comparison as nodes are more diversified comparing to traditional methods.

The second type of graph-based method is skeleton. It is used for both 2D and 3D data. Sundar et al.[6] define it as the "central-spine" of an object. For 2D data, this skeleton refers to the medial-axis of the picture for example. For 3D objects, a medial surface is computed. Skeleton are simplified representation comparing to graph-model such as they have a lower number of elements. Amenta et al.[7] propose a

method by approximation of the Medial Axis Transform (MAT) of the 3D object. An inverse transform is used to retrieve the surface representation from the MAT. Approaches based on shock-graph are similar such as in Sebastian et al.[8]. This shape descriptor is used in order to recognize 2D shape outlines.

A third type of graph-based method is Reeb Graph (RG). The difference with skeleton lies in the way of building the graph and a ponderation criterion is used. RG principle is to cut the shape (2D or 3D data) by parallel level lines. For each section obtained, center of mass is computed, which corresponds to one node of the graph. Once all the nodes are linked, the graph is created. Instead of computing the center of mass, other criteria have been used in the literature such as [9][10]. Tierny et al.[11] propose a new method which is invariant to 3D shape position and mesh resolution. Biasotti et al.[12] compare sub-part correspondence. Their method is able to provide a global measure of the similarity between two parts but also to indicate where the similarities occur in the shapes. It is performed thank to RG which enable to couple geometrical and structural information about the shape.

Model-graphs and in particular attributed graph seem to be an adequate solution for our signature process. Indeed this representation provides a higher level of details in the signature such as topological and geometrical information, which can be interesting if the user wants to retrieve a parametrized CAD model of each component of the mechanical assembly.

2.2. Signature matching and KBE application

Tangelder and Veltkamp [3] consider the matching step as a process enabling to determinate similarity between two shapes, by computing their “distance” between each other. As graph-based signatures are studied in this paper, only graph comparison will be treated. To compare the signatures, our system disposes of a knowledge database composed of signatures. Each signature is linked to a component which belongs to an mechanical assembly, a function etc. Chapman and Pinfold [13] define Knowledge-Based Engineering (KBE) as a method combining oriented computer programming, artificial intelligence techniques and CAD tools. KBE tools enable to capture information linked to the product, in order to reuse it in engineering activities. These systems aim to capitalize best design practices and also information expertizes and they store them in a knowledge database. Durupt et al.[14] propose a methodology called KBRE which combines RE and KBE activities. The given solution uses a functional and structural skeleton from a knowledge database. It leads their RE process by instantiating the skeleton thanks to dimensions extracted from points cloud. The skeleton is composed of two types of characteristics: design intents and manufacturing characteristics. They contain information expertizes in order to rebuild a parametrized CAD model. Recently Ali [15] came up with a methodology called REFM (Reverse Engineering For Manufacturing). She associates information expertizes link to the manufacturing process with the RE process. A knowledge database enables to store this information whose aim is first to bring a structured method to the user for the purpose of remanufacturing the component. Then this knowledge, stored under precedence graph formalism, can be reused for RE of similar components. These KBE solutions could be useful in our context. Let us now have a look on the related works dealing with graph comparison.

Through the comparison of graphs, we seek similarities between the two components. The result of comparison is used to filter (restrict) the set of nodes of each of the graphs. Then this improves the precision of the signature and it increases the percentage of matched components. Thus, we are confronted with problem of finding Maximum Common Subgraph (MCS) of two graphs. This problem is NP-hard optimization problem: there is no known polynomial solution to this problem. Several libraries and software allow comparison graphs: Boost Graph Library (BGL)[16], Vflib2[17], Simpack[18], Nauty[19], etc. Unfortunately some seem to only support directed graphs (Vflib2 and Simpack for example). In our case, the graphs are not oriented: nodes are linked by edges whereas in oriented graph, they are linked by arrows. Others solutions do not fulfill to the problem (i.e. MCS). So we opted for the BGL library in C ++. It provides the algorithm « Mc Gregor common subgraphs ». Thus, it perfectly meets the issue: it resolves MCS problem, it supports undirected graphs, it allows customizing the output display of the results (via user callback) - which can be operated easily -, and finally, the vertices and edges may receive values, which allow them to compare themselves (via vertices equivalent and edges equivalent predicates).

Once mappings between the two graphs were obtained, it remains to apply a criterion of similarity to mean how the two graphs are similar.

Thanks to [14][15], we can measure the advantages of KBE systems because they can help considerably the user in the RE process and accelerate the identification of the components. The combination of graph and knowledge database could permit to identify also uncompleted components inside the data.

3. HDI-RE : HETEROGENEOUS DATA INTEGRATION FOR REVERSE ENGINNERING

3.1. The whole methodology

HDI-RE is a global methodology which answers the main issue presented in introduction of this paper. In this section, we will explain the ins and the outs of each step (see Figure 2).

The first step concerns the data segmentation. According to its type (2D or 3D), a technique of segmentation is applied to retrieve segments. For this paper, we consider only points cloud data (others heterogeneous data are presented in [2]). The aim is to retrieve segments which can be identified topologically as planes, cylinders and sphere. The others types such as torus or cones are not studied in this work since the number of parameters of canonical surface to approximate is larger. The method used relies on hierarchical mesh segmentation. The entire methodology is described in Attene et al.[20].

A graph is built (second step of our methodology) and it corresponds to the signature of the data. Each cluster (segment) is associated to a node and the edges refer to the connectivity between each cluster of the segmented data. If the final aim of the RE process is to rebuild a digital mock-up with parametrized CAD models, the signature of the data (the graph) may contain an appropriate level of details such as topological information or dimensions of the main characteristics. It will be explained in the next section.

The last step concerns the signature comparison. The graph signature of the data is compared with other signatures existing in the KDB. We consider that a component signature can't be recognized if there is no existing signature of the related component in the database. A capitalization of the enterprise proper knowledge is performed at each RE process.

3.2. A three-levels geometrical signature

In this section, we present one of the signatures developed in the HDI-RE process which is dedicated to points cloud data. Another one has been presented in a previous work [2].

The Figure 3 presents a global representation of our three-level signature. Depending of the level of details required to (re)build the DMU, five scenarios can be possible and they are described in [21]. One of them will be presented in section 4.1. From a global point of view, a segmented mesh[1] is signed as follows:

- First-level signature: each node represents a cluster (also called segment) of the mesh. The edges refer to the connectivity links between the clusters. Let us consider G_1this first-level graph; N_1 the set of nodes of $G_1$and E_1 the connected edges.
- Second-level signature: the first-level signature is reused. Each cluster (graph's node) is classed as plane, cylinder, sphere or other. This

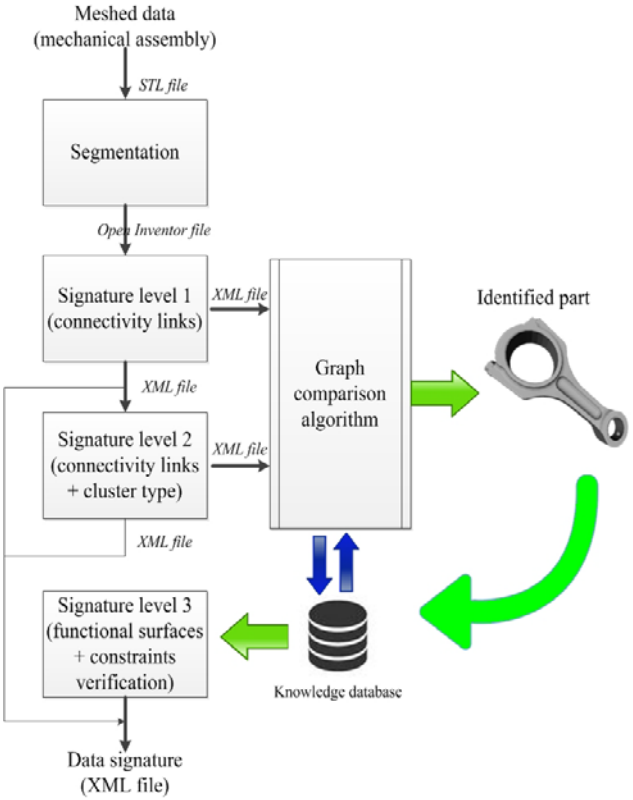


Figure 3 A three-level signature for meshed data.

[1] We consider points cloud is meshed previously by the user in dedicated software.

information is added to each corresponding node as attribute. The second signature corresponds to a second graph G_2 with N_2 its nodes and E_2 the connected edges. We can notice that $E_1 = E_2$.

- Third-level signature: this one is dependent of the second level. Let us consider G_3this third-level graph; N_3 the set of nodes of G_3 and E_3 the connected edges. It is built thanks to the intersection between a mapping (cited later) and its second-level signature. It is called precedence graph: an example is given in Figure 5.

It is a formal representation of design intents used in engineering process. It is presented in details in Ali [15]. This type of graph is useful for the part instantiation (digital mock-up reconstruction). Its

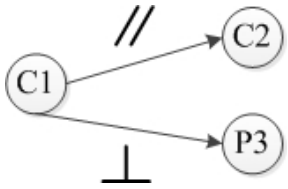


Figure 4 Example of a precedence graph.

edges have labels which correspond to geometrical constraints and its nodes have the same attributes as the second-level graph. The main advantage is the limited number of nodes of this graph: only the functional surfaces of the component have a corresponding node in the graph (the number of nodes in a precedence graph is smaller than the number of faces). Thus the amount of nodes to compare is also reduced. We consider that the knowledge database contains precedence graphs previously. Moreover compared to the two previous graphs, this one is oriented. Thus the graph in Figure 5 is interpreted as follows: the node C2 (cylindrical surface) is positioned thanks to C1 by a constraint of parallelism. The plane P3 is perpendicular to C1 cylinder's axis. There are many types of constraints existing so, for our work, we consider only these ones: perpendicularity, coaxiality and parallelism.

3.3. Signature comparison

We consider that all the signatures (levels 1, 2 and 3) of the components in KDB have been created upstream to the RE activity: they have been segmented thanks to Efpisoft and then they have been signed such as in the process described in the previous section. These signatures are used as reference to identify components in our input data.

After the first-level signature, the first-level graph of our input data is compared with the first-level graphs in KDB. Let us consider SG_1, one of the maximum common subgraphs and which relies on one of the mapping. This last is a list of correspondences between N_1 and N'_1 the set of nodes of G'_1, one graph of database which matched. Thus $SG_1 = N_1 \cap N'_1$. Figure 6 (step A) illustrates the comparison between G_1, the input data signature and G'_1, one graph of the database. Then the mapping results are shown (step B): these are all the combinations of correspondences between the two graphs (SG_1 is equivalent to one combination). We consider that a component (reference) is identified in our input data if its similarity score with the input data is superior to zero percent. Many reference components can match with a low similarity score. Thus we will keep the highest or one of the highest similarity score (e.g. 80 %) and we will consider that our input data is 80% similar, for example, with the reference component (e.g. a connecting rod). At this stage, our input data is considered as similar to others components thanks to connectivity information.

For the second level, the graph comparison is done between G_2 and the second-level signatures of the KDB. SG_2 is one of the maximum common subgraphs belonging to the mapping between G_2, one of the graph from KDB (N'_2 its nodes). We consider that G'_2 got the highest matching score comparing to the others graphs of KDB. At this stage, the component is identified topologically and geometrically and it is possible to determine the similarity between the meshed data and one component identified (our reference). In our example, a connecting rod has been identified in our meshed data (cf. Figure 3).

For the third level, the comparison can be processed only if there is at least one maximum common subgraph (SG_2) existing after the second-level comparison. Let us consider G'_3, the precedence graph (PG) of the connecting rod from the KDB

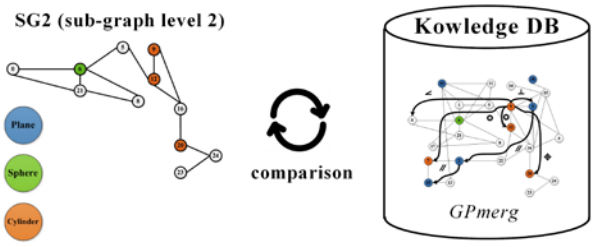


Figure 5 Set of data used for 3rd level comparison.

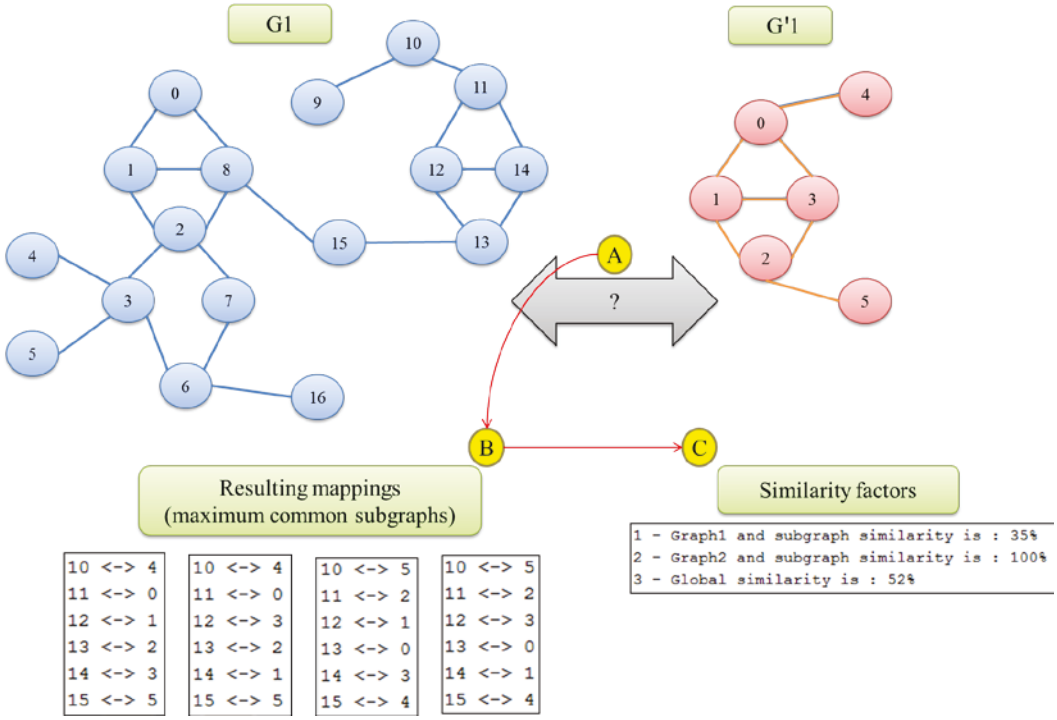


Figure 6 Example of comparison of two graphs.

(reference component). The existing knowledge in the database gives us GP_{merg} which is the merging between G'_2 and G'_3 the corresponding second and third levels signatures of the component identified (i.e. the connecting rod). Figure 4 represents the different data used for this level of comparison. The blue nodes are planes, the green ones are spheres and the orange ones are cylinders (information given thanks to second-level signature).

The third-level comparison is performed thanks to three steps:

- to identify GP_{merg}'s nodes which exist in SG_2;
- to check if GP_{merg}'s pairs of nodes exist in SG_2;
- in potential pairs, to check if geometrical constraints are valid (label on graph connections).

The algorithm 1 describes this three-step procedure. Let us consider two tables as inputs. There are $Tab\_GP_{merg}(i,3)$ which contains i pairs of nodes from GP_{merg} and 3 columns (id node 1, id node 2, geometrical constraint) and $Tab\_SG_2(j,1)$ which includes j nodes from SG_2. The output is $Tab\_GraphLev3(k,3)$ which is the respective table to our input-data's third-level graph with k, the number of pair of nodes and there are 3 columns (id node 1, id node 2, geometrical constraint between 1 and 2).

Algorithm 1

```
%Initialization
count = 1 %counter
%1rst colunm of Tab_GPmerg
FOR i=1 to lenght(Tab_GPmerg)
 FOR j=1 to lenght(Tab_SG2)
  IF Tab_GPmerg(i,1)==Tab_SG2(j)
  Potential_pair(j,1) = Tab_GPmerg(i,1).value
   %2nd colunm of Tab_GPmerg
   FOR k=1 to lenght(Tab_GPmerg)
    FOR l=1 to lenght(Tab_SG2)
     IF Tab_GPmerg(k,2)==Tab_SG2(l)
      Potential_pair(k,2) = Tab_GPmerg(k,2).value
       %Geometrical constraint checking
       IF Check_Geometrical_Constrainst() is TRUE
        Tab_GraphNiv3(i,1)= Tab_GPmerg(i,1).value
        Tab_GraphNiv3(k,2)= Tab_GPmerg(k,2).value
        Tab_GraphNiv3(count,3)= Tab_GPmerg(i,3).value
        count = count + 1
       END IF
      END IF
      l=l+1
     END FOR %EnfFor l
     k=k+1
    END FOR %EndFor k
   END IF
   j=j+1
 END FOR %EndFor j
 i=i+1
END FOR %EndFor i
Generate_XML(Tab_GraphNiv3)
%EndAlgorithm
```

The set of matched nodes (from $Tab\_GraphLev3$) is called n_3 and it corresponds to the functional surfaces inside the meshed data. As described previously, the last step of the comparison procedure consists in verification. It is performed by computing local characteristics of each node of n_3 (cluster of the meshed data). A threshold is defined by the user. For each constraint between two segments of the mesh data computed, the relative error should be inferior to this threshold. If constraints are validated (e.g. perpendicularity between two clusters), thus each corresponding edge of the graph is generated (it corresponds to the third column of $Tab\_GraphLev3$). Geometrical information of N_3(e.g. diameter and length of a cylindrical segment) could be used (in future works) to instantiate a CAD template whose functional surfaces are piloted by these parameters. To finish, G_3 is stored in the database for knowledge capitalization (G_1 and G_2 are also stored).

3.4. Factor of similarity between graphs

We used the "node count" as the type of similarity (step C - Figure 6). Three similarities emerge: the similarity to the first graph S_{G_1}, the similarity to the second graph $S_{G'_1}$, and the overall similarity $S_{G_1G'_1}$ (of how much the two graphs are similar). They are displayed as a percentage.
For each of the similarities, the nodes of the largest subgraph (resulting from comparison of graphs) and the nodes of the two initial graphs are used.
Let us consider: $N_{G_1}$the number of nodes in the first graph, $N_{G'_1}$ the number of nodes in the second graph and $N_{G_{Sub}}$ the number of nodes in the subgraph.
Then the similarities are defined by:

$$S_{G_1} = \frac{N_{G_{Sub}} * 100}{N_{G_1}}$$

$$S_{G'_1} = \frac{N_{G_{Sub}} * 100}{N_{G'_1}}$$

$$S_{G_1G'_1} = \frac{2 * N_{G_{Sub}} * 100}{N_{G_1} + N_{G'_1}}$$

4. APPLICATION

The use case presented here is a scanned assembly of a piston and a connecting rod (green cell of the Table 1). The set of data provided in the KDB is composed of three signatures (for each level) for each data. The aim is to identify the components in the input data and extract from them topological and geometrical information in order to instantiate in the future a CAD template.

The number of segments (nodes) has an influence on the graph comparison. It is not possible to predict the best number of nodes for the input data (operation realized manually by the user). So, in order to maximize our ability to match with a signature (with a defined number of nodes) in KDB, we generated a large set of signatures (in KDB) with a varying number of clusters.

4.1. Graph-based signature and comparison

Let us consider the steps presented in Figure 2. Our scenario starts when the scanned assembly (mesh) is imported. We consider that the points cloud has already been meshed by the user outside of our process.

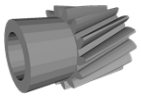
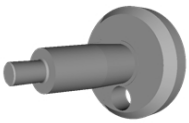
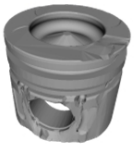
Scanned data 	Helicoidal gear 	Spur gear 
Connecting rod 	Assemble connecting rod 	Gear 
Crankshaft 	Piston 	Scanned pulley 

Table 1 Scanned assembly and data available in database

The segmentation is performed in Efpisoft software developed by Attene et al.[20]. The input file is the STL file of the scanned assembly (ASCII format). The user defines manually the number of segments (ideally this number corresponds to the sum of all the components assembly's surfaces). This can be adjusted thanks to the software's GUI. An illustration of the segmentation is given in Figure 7. The output file is saved in Open Inventor Scene format thanks to Efpisoft software. Another file is generated thanks to separate algorithm. This file is a XML file which contains the Open-Inventor-Scene's file information in a standard format. It will be used for the second-level signature. It gives information about each

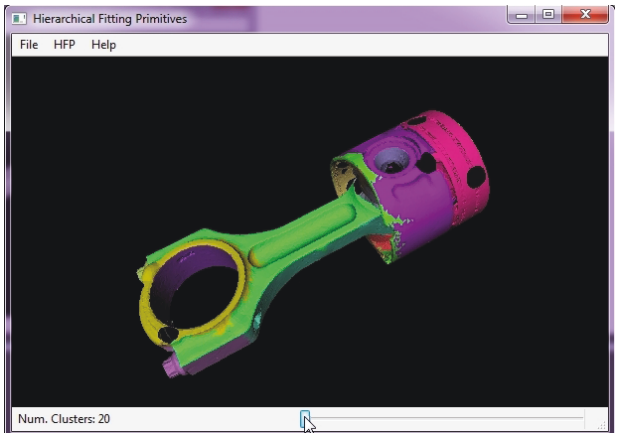


Figure 8 Segmentation in Efpisoft with 20 clusters.

cluster: the ID of the points which compose each cluster (segment) and the coordinates (x, z, z) of each point.

Then we propose to sign the data with the scenario as defined in Figure 8. This is an application of the global method presented in Figure 3.

The segmented mesh is signed (step 2 of Figure 8) thanks to an internal algorithm. The input file is the Open Inventor Scene. The output file (the signature) is a XML. It gives information about each node (cluster ID, center of mass coordinates, mesh area) and the connectivity between nodes (e.g. node 2 linked to node 3) which corresponds to the first-level graph.

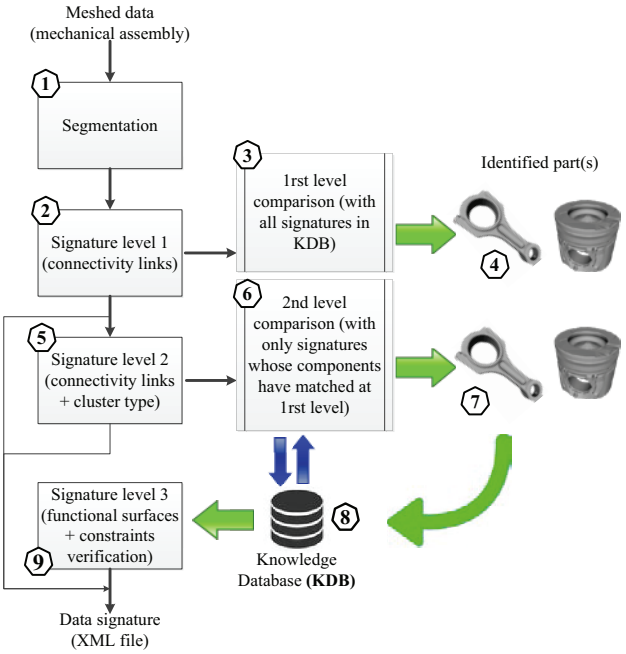


Figure 7 Three-level signature (scenario n°5 from [21]).

Then, the first-level comparison (step 3) is realized. In inputs, there are the XML file corresponding to the first-level signature and all the first-level graphs of components of the database (components of Table 1). Comparison of graphs is performed using McGregor's algorithm of the C++ Boost Graph Library[22]. This algorithm finds all common subgraphs between two graphs, and it returns several mapping between the two graphs. Only the largest, discovered subgraphs are kept. As output, a text file is generated automatically. It gives the similarity score obtained between our input data and each first-level graph of the database. The text in the black frame (Figure 6 – step B) is an extract of this mapping information.

At step 4, the user retrieves a list of similarity values (percentage) which enables him to select the components of the database whose score is higher than the other ones. In this example, a piston and a connecting rod are selected. This step enables to reduce the number of graphs compared in the second level comparison. After step 4, the results of this comparison only give us similarity from a "connectivity" point of view. In our case, we want to retrieve geometrical and topological information (to rebuild parametrized CAD model). So it is necessary to continue the process in order to match graph nodes according to their topological characteristics. The second level will enable to identify and match the type of nodes into the input data and the third level will point out the clusters that correspond to functional surfaces. In another RE context, the scenario could stop after step 4. Hence the user has a quick result about similarity with the others signatures in KDB.

For the second-level signature (step 5), an executable algorithm developed in MatLab software is launched in order to determine the type of each cluster. The clusters are fitted to canonical surfaces which can be: plane, sphere, cylinder or other (for free forms, torus, cone…). The inputs are: the XML file from Efpisoft (conversion of Open Inventor Scene) and the first-level signature. The output corresponds to the XML of the first-level signature which is enriched. The type of cluster is added to each node.

For the second-level comparison (step 6), an algorithm is launched based on the previous one. The information about cluster type is considered. As explained previously, our second-level signature is compared to a limited set of signatures from KDB. These last ones correspond to the component(s) who

matched at step 3 (a piston and a connecting rod). A text file which corresponds to the mapping information (similar to the first-level results) is generated automatically. Moreover, the similarity score is displayed for each comparison performed with a signature form KDB. Information given (thanks to mapping) enables to identify topologically our input data.

At step 7, the user needs to select the best matching results depending of two criteria: similarity score, and number of fitted[2] nodes inside the mapping. Indeed, the maximum common subgraph between our input data and the signature of the KDB needs to have at least one fitted node (defined as plane, sphere or cylinder). Without this condition, it is impossible to continue with the third-level signature because this graph relies only on fitted nodes.

Step 8 is performed upstream to our process: a Graphical User Interface (GUI) helps the RE user to generate GP_{merg} in KDB. The precedence graph is linked with the second-level graph of the corresponding component. This GUI is presented in [21].

For the third-level signature (step 9), the mapping results (subgraph information) of one of the signatures from step 6 is used. Contrary to the two previous levels of signature, this one is performed as many times as there are components selected by the user at step 7. In our example, that means that the signature is first performed in order to match with the precedence graph information of the connecting rod ($GP_{merg\_ConnectingRod}$) and then, it is performed again with the information from the piston (GP_{merg_Piston}). Thus for the connecting rod, the inputs files for the third level are:

- XML file of signature of the third-level ($GP_{merg\_ConnectingRod}$) for the connecting rod selected in step 7. These XML file comes from the knowledge database.
- A text file containing the mapping results from the second-level comparison : it is a list of pair of matched nodes;
- A text file which contains geometrical information about each segment (e.g. normal coordinates for a plane, direction coordinates of a cylinder, cylinder diameter etc.). This information is extracted during the type identification in the second-level signature.

As explained previously, this process of comparison is different. Algorithm 1 is applied: it is mainly a geometrical comparison. Then the algorithm is also applied for the piston. The output of this step is the third-level signature of the scanned data. It represents the functional relationships between the segments (clusters) for the two components that compose the mechanical assembly. This signature (XML file) is saved in the database.

Thanks to the third-level signature (XML file), geometrical information extracted from the functional surfaces in the input data will be used to value the CAD templates parameters (for the digital mock-up reconstruction).

4.2. Results

In this section, we will present the results obtained for each comparison level according to the set of data available in the KDB.

The input data is segmented in Efpisoft with 20 clusters (Figure 7). This quantity has been determinate according to our capacity of computation. The experiments were conducted on a 2-way Intel Xeon machine operating at 2.40 GHz with 8.00 Go of RAM. The comparison was not possible with graphs with more than 20 nodes, in particular for the first-level comparison.

The number of nodes of the graphs (signatures) in the KDB is also limited. Indeed, it was not possible to compare graphs with more than 15 nodes in particular for the first-level comparison (infinite computing time). The results are displayed in the Table 2.

Each value corresponds to the factor of similarity $S_{G'_1}$ (presented in section 3.4) between the graph of mechanical assembly (data to identify) and the others graphs of the KDB. This similarity factor has been chosen as a mechanical assembly component is looked up in the KDB. The highest scores (with

Scoring for first-level comparison (in %)			
Helicoidal gear (15cl.)	73	Gear (15 clusters)	60
Spur gear (15 clusters)	80	Cranskshaft (15 clusters)	73
Connecting rod (15 cl.)	60	Assemble connecting rod (5 cl.)	**100**
Piston (5 clusters)	**100**	Assemble connecting rod (10 cl.)	80
Piston (10 clusters)	90	Assemble connecting rod (15 cl.)	N/A
Piston (15 clusters)	80	Scanned pulley (15 clusters)	80

Table 2 Results for the first-level comparison.

[2] A fitted node is a node (cluster) which has been fitted to a canonical surface during the second-level signature.

100%) correspond to the piston (5 clusters) and the assemble connecting rod (5 clusters). We can notice that the comparison with the assemble connecting rod with 15 clusters was not possible.

For the second level (step 6 from Figure 8), only the two second-level signatures of the components identified are considered. Thus the number of comparisons is reduced to these two components: piston (5 clusters) and assemble connecting rod (5 clusters). It follows from the user selection (step 4) among the different results.

Then the results are presented in Table 3 and correspond to the factor of similarity $S_{G'_1}$.

The piston matched with 80% and the assemble connecting rod matched with 100%.

For the third-level, each component is processed separately. However we notice that the common subgraphs obtained for the piston and the connecting rod don't have any fitted node. It follows from the weak number of nodes of the signatures who

Scoring for second-level comparison (in %)			
Piston (5 clusters)	80	Assemble connecting rod (5 cl.)	100

Table 3 Results for the second-level comparison.

matched at step 3. As this condition is not satisfied, the third-level can be processed (any fitted node to identify). Thus, we propose to abort this scenario and we decided to skip over the first-level comparison.

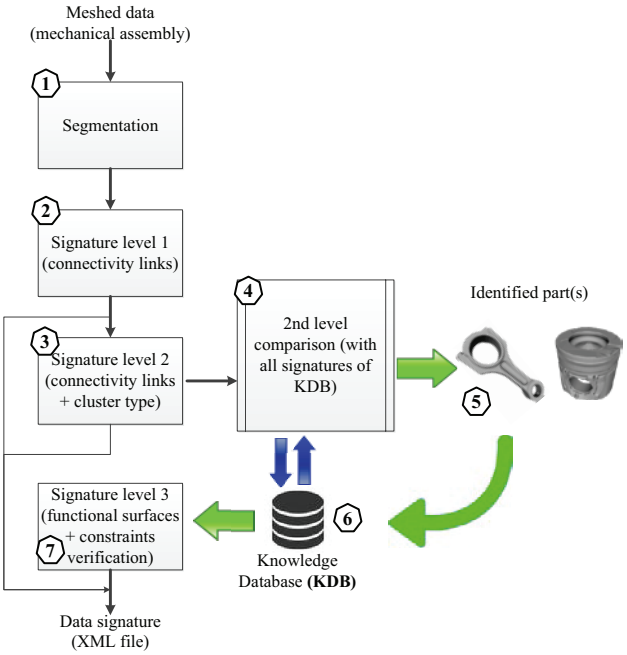


Figure 9 Second scenario without first-level comparison.

Scoring for second-level comparison (in %)			
Helicoidal gear (15clusters)	0	Cranskshaft (15clusters)	40
Gear (15clusters)	0	Piston (25clusters)	44
Gear (20clusters)	0	Scanned pulley (25clusters)	44
Helicoidal gear (25clusters)	0	Assemble connecting rod (25clusters)	48
Gear (25clusters)	0	Scanned pulley (20clusters)	50
Piston (60clusters)	8	Spur gear (15clusters)	53
Piston (50clusters)	10	Assemble connecting rod (20clusters)	55
Piston (55clusters)	14	Piston (20clusters)	55
Piston (45clusters)	15	Assemble connecting rod (10clusters)	60
Cranskshaft (25clusters)	20	Assemble connecting rod (15clusters)	60
Piston (40clusters)	25	Piston (15clusters)	60
Spur gear (20clusters)	25	Connecting rod (15clusters)	60
Cranskshaft (20clusters)	25	Connecting rod (20clusters)	60
Spur gear (25clusters)	32	Scanned pulley (15clusters)	66
Piston (30clusters)	36	Assemble connecting rod (5clusters)	80
Piston (10clusters)	40	Piston (5clusters)	100

Table 4 Results for the second-level comparison (2nd scenario).

This new scenario is presented in Figure 9.

Table 4 presents the results obtained at step 4. For this second-level comparison, all the second-level signatures from KBD are considered (also with a varying number of clusters). We can notice that the possible number of clusters for the signature from KDB has been raised and the number of signatures compared has increased also. Indeed the second-level comparison is less time consuming.

At step 5 (Figure 9), the user has to choose the mapping results that correspond to signatures with a maximal number of fitted nodes. For now, this selection is manual: one of the common subgraphs chosen between our input data and the assemble connecting rod (25 clusters) has 12 nodes whose one cylinder; one of the common subgraphs chosen between the input data and the piston (50 clusters) has 5 nodes whose one cylinder.

After the previous selection, the third-level signature (step 6 & 7) can start. The two third-level signatures $GP_{merg\_ConnectingRod\_25clusters}$ for the connecting rod and $GP_{merg\_Piston\_50clusters}$, for the piston are used. As there is only one node available to compare for each component, the algorithm 1 doesn't need to be applied. The only operation consists in checking if the two fitted nodes from each subgraph exist in $GP_{merg\_ConnectingRod\_25clusters}$ for the connecting rod and in $GP_{merg\_Piston\_50clusters}$ for the piston. However any constraint can be checked such as there is only one node that can be identified for each graph. Finally, the third-level signature can be generated because of the absence of constraint to be verified.

4.3. Discussions

The number of clusters chosen by the user during the segmentation influences all the process and especially the results obtained in the comparison steps. The graphs comparison algorithm is limited by the computing capacity. For now, it is not possible to generate graphs with as clusters as component's faces. Thus in order to get some results, the number of clusters has been decreased: 20 nodes as a maximum for the input meshed data, 60 nodes for the graphs in database for the second-level comparison and until 15 nodes for the first-level comparison.

Several tests have been performed to measure efficiency of the first-level graph (time computing beside quality of results). The results demonstrate that the use of the first-level graph is disputable in our RE methodology. Indeed, the majority of graphs of the database which have matched have a low number of nodes (between 5 and 10). The comparison with others graphs (more than 15 nodes) exceeds computing capacity. Moreover the signatures matched in first level (with only 5 nodes) don't provide enough information to perform third-level comparison. Indeed, these five-node signatures have clusters which are faces agglomerates, thus the node's attribute given is "other". An illustration is given in Figure 10 with a connecting rod with 10 clusters. We can notice that several clusters contain plans and cylinders in the same cluster. The second-level signature relative to this segmentation is presented below. Each node attribute is represented by a letter before the node's number: 'P' for plan, 'S' for sphere and 'C' for cylinder. We notice that only 3 nodes are fitted with a canonical surface. It can be deduced that only 3/10 nodes can be matched during the second-level comparison. For this reason the first scenario has been aborted in order to start again our process signature and by skipping the first-level signature. Even with this improvement, the results obtained don't enable to validate our third-level signature. It is mainly due to the low number of nodes of the input data. Ideally this number should be equal to the sum of all the components assembly's surfaces (for the corresponding CAD models).

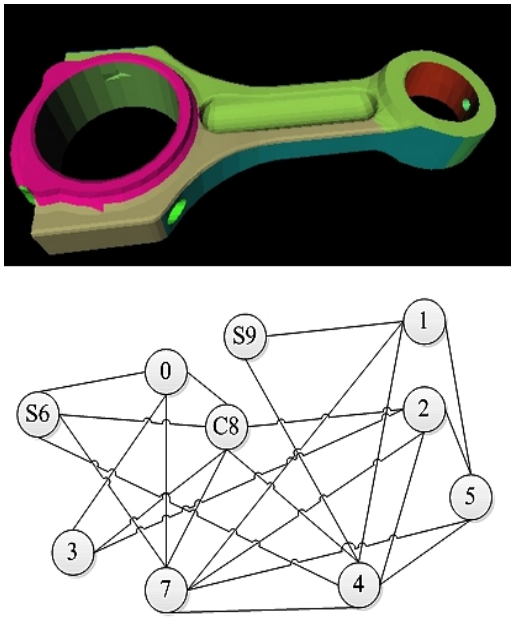


Figure 10 A 10 clusters segmentation and second-level signature.

The data set used in this study involves only components with surfaces which fit only with planes, cylinders or spheres. Free form, conical and toric surfaces haven't been processed such as the second level signature algorithm requires more complex algorithms. It will be carried out in future works.

To finish, the selection of the signature(s) after each comparison needs to be automatized. A solution could be a weighted similarity score in function of the number of fitted nodes in the maximum common subgraph. In addition, uncomplete data (in input) and noise can affect the comparison results and it needs to be also studied.

5. CONCLUSIONS AND PERSPECTIVES

HDI-RE is a methodology which enables to retrieve a digital mock-up from a set of heterogeneous data. It relies on three steps: segmentation data, signature data and comparison of the signature with the knowledge database.

This paper deals with the signature and comparison steps. One of the signatures developed in our project research and its comparison mechanism are presented. It is a three-level signature by graph. Depending of the level, graph provides different information: connectivity between faces or segments (for the input data), surface type such as plane, cylinder or sphere and functional relationships between faces for the third level.

An applicative example is given with a scanned assembly as input data. Two components are identified: a piston and a connecting rod. The third-level signature enabled to extract some geometric information that could help to instantiate a CAD template (available in the KDB) in order to rebuild the final digital mock-up.

The development work related to the third-level

signature is not complete. The last comparison is run manually. The perspectives are to finish it and to perform several tests on it. The number of clusters of the segmented data has an influence on the results, thus this work will be continued in order to improve the final results (influence of the number of clusters, noise etc.). The aim is to identify as much functional surfaces as possible.

To finish, components instantiation (digital mock-up reconstruction step) will be coded in the future thanks to CAD templates also available in the knowledge database.

ACKNOWLEDGMENTS

These works have been supported by the ANR (National Agency of Research) through the Numerical Model program (Project METIS – 12-MONU-004).

Authors want to thank Attene et al.[18] for letting us using the software Efpisoft in our research works.

Industrial data for the application section have been provided by a French company called IFPEN (Institut Français du Pétrole et des Energies Nouvelles).They are part of the consortium project whose these works are relying on.

REFERENCES

[1] R. Vinesh et K. J. Fernandes, *Reverse Engineering*. London: Springer London, 2008.

[2] M. Bruneau, A. Durupt, L. Roucoules, J.-P. Pernot, et H. Rowson, « A methodology of Reverse Engineering for large mechanical assemblies products from heterogeneous data », in *Proceedings of TMCE 2014*, 2014, n° 2, p. 1307‑1318.

[3] J. W. H. Tangelder et R. C. Veltkamp, « A survey of content based 3D shape retrieval methods », *Multimed. Tools Appl.*, vol. 39, n° 3, p. 441‑471, déc. 2007.

[4] M. El-Mehalawi et R. A. Miller, « A database system of mechanical components based on geometric and topological similarity. Part I: Representation », *CAD Comput. Aided Des.*, vol. 35, n° 1, p. 83‑94, 2003.

[5] L. Ma, Z. Huang, et Y. Wang, « Automatic discovery of common design structures in CAD models », *Comput. Graph.*, vol. 34, n° 5, p. 545‑555, 2010.

[6] H. Sundar, D. Silver, N. Gagvani, et S. Dickinson, « Skeleton based shape matching and retrieval », in *Shape Modeling International 2003*, 2003, p. 130‑139.

[7] N. Amenta, S. Choi, et R. K. Kolluri, « The power crust », *Proc. sixth ACM Symp. Solid Model. Appl. - SMA '01*, p. 249‑266, 2001.

[8] T. B. Sebastian, P. N. Klein, et B. B. Kimia, « Recognition of shapes by editing shock graphs », in *IEEE International Conference on Computer Vision*, 2001, p. 755‑762.

[9] M. Hilaga, Y. Shinagawa, T. Kohmura, et T. L. Kunii, « Topology Matching for Fully Automatic Similarity Estimation of 3D Shapes for Topology Matching », in *ACM SIGGRAPH 2001*, 2001, p. 203‑212.

[10] D. Bespalov, W. C. Regli, et A. Shokoufandeh, « Reeb Graph Based Shape Retrieval for CAD », in *ASME 2003 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, 2001.

[11] J. Tierny, J. Vandeborre, et M. Daoudi, « Graphes de Reeb de Haut Niveau de Maillages Polygonaux 3D », in *COmpression et REprésentation des Signaux Audiovisuels (CORE-SA'06)*, 2006.

[12] S. Biasotti, S. Marini, M. Spagnuolo, et B. Falcidieno, « Sub-part correspondence by structural descriptors of 3D shapes », *Comput. Des.*, vol. 38, n° 9, p. 1002‑1019, sept. 2006.

[13] C. B. Chapman et M. Pinfold, « Design engineering - a need to rethink the solution using knowledge based engineering », *Knowledge-Based Syst.*, vol. 12, p. 257‑267, 1999.

[14] A. Durupt, S. Remy, G. Ducellier, et E. Guyot, « A new reverse engineering process, the combination between the knowledge extraction and the geometrical recognition techniques », *2009 Int. Conf. Comput. Ind. Eng.*, p. 1367‑1372, 2009.

[15] S. Ali, « La rétro-conception de composants mécaniques par une approche "Concevoir pour fabriquer" », Université de Technologie de Troyes, France, 2015.

[16] J. G. Siek, L.-Q. Lee, et A. Lumsdaine, *The Boost Graph Library: User Guide and Reference Manual*. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc., 2002.

[17] L. P. Cordella, P. Foggia, C. Sansone, et M. Vento, « A (sub)graph isomorphism algorithm for matching large graphs. », *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 26, n° 10, p. 1367‑72, oct. 2004.

[18] A. Bernstein, E. Kaufmann, C. Kiefer, et C. Bürki, « SimPack: A Generic Java Library for Similarity Measures in Ontologies », *Univ. Zurich*, p. 20,

2005.

[19] B. D. McKay et A. Piperno, « Practical graph isomorphism,II », *J. Symb. Comput.*, vol. 60, p. 94‑112, 2014.

[20] M. Attene, B. Falcidieno, et M. Spagnuolo, « Hierarchical mesh segmentation based on fitting primitives », *Vis. Comput.*, vol. 22, n° 3, p. 181‑193, 2006.

[21] M. Bruneau, « Une méthodologie de Reverse Engineering à partir de données hétérogènes pour les pièces et assemblages mécaniques », Université de Technologie de Compiègne (France), 2016.

[22] D. G. Vossler, T. Stonecipher, et M. D. Millen, « Femoral artery ischemia during spinal scoliosis surgery detected by posterior tibial nerve somatosensory-evoked potential monitoring. », *Spine (Phila. Pa. 1976).*, vol. 25, n° 11, p. 1457‑9, juin 2000.