Automatic CAD Assemblies Generation by Linkage Graph Overlay for Machine Learning Applications

Lucas Vergez[1], Arnaud Polette[2] and Jean-Philippe Pernot[3]

[1]Arts et Métiers Institute of Technology, LISPEN, HESAM Université, F-13617 Aix-en-Provence, France, lucas.vergez@gadz.org
[2]Arts et Métiers Institute of Technology, LISPEN, HESAM Université, F-13617 Aix-en-Provence, France, arnaud.polette@ensam.eu
[3]Arts et Métiers Institute of Technology, LISPEN, HESAM Université, F-13617 Aix-en-Provence, France, jean-philippe.pernot@ensam.eu

Corresponding author: Lucas Vergez, lucas.vergez@gadz.org

Abstract. This paper introduces an approach to synthetize new CAD assemblies from existing STEP files. The algorithm first generates linkage graph by detecting linkage between components. Then it detects linkages similarities between components of different assemblies while analyzing the associated graphs. Finally, it exchanges the similar components. The similarities in a family of components must be formalized by the user. Then the similar components can be replaced by the other through smart placements. This method allows to automatically generate a wide variety of new consistent assemblies sharing the same semantics, in order to create databases of CAD assemblies ready for machine learning applications. It has been validated on several cases.

Keywords: Shape synthesis, linkage graph, CAD models generation, CAD assembly.
DOI: https://doi.org/10.14733/cadaps.2022.722-732

1 INTRODUCTION

Enlarging 3D model databases by shape synthesis is a large field of research. Indeed, the use of machine learning techniques requires a huge amount of labeled CAD models, and it is therefore crucial to rely on large and varied databases. Most of existing works in shape synthesis focus on everyday life objects generation. However, these methods often do not work on assemblies composed of several CAD models, and it is the aim of this paper to develop a new shape synthesis method to enlarge existing CAD assembly databases.

Today, there exist lots of free databases of non-labeled CAD models (e.g. GrabCAD, 3D Warehouse, Turbosquid) often available as STEP or IGES files. Unfortunately, very few of these databases are labeled. Other databases like PartNet and ShapeNet are currently labeled by crowdsourcing, but they do not contain complex mechanical assemblies.

The method explained in this paper consists in creating new labeled CAD assemblies from existing ones by linkage graph overlay. Here, the STEP file format has been adopted in order to be the most reproductible and to be adaptable. The linkage graphs are automatically created thanks to the identification of the linkages between the components. Indeed, linkages are not included in the STEP files and they need to be computed. These linkage graphs are then analyzed and components with similar linkages are detected. Finally, once the similarities are detected, the corresponding components can be exchanged to created new assemblies for which the labels can be directly inherited from the source assemblies.

The contribution is threefold: (i) a method to create linkage graphs from existing non-labelled CAD assemblies; (ii) a method to recognize basic components using linkage graphs; (iii) a smart overlay method to replace some components while keeping the coherence between all the components of the assembly.

The algorithm has been implemented in Python on FreeCAD and it has been tested on several test cases. The paper first presents a brief state-of-the-art on shape synthesis. Then, it explains the overview of the method, from the graph synthesis to the components overlay, finishing with the replacement of the components. Finally, the results are presented, and a conclusion ends this paper while discussing the next steps.

2 RELATED WORK

This paper addresses different topics. First, it is related to AI-based shape synthesis. GRASS [1] has been the first method to use deep neural network to generate shapes from characteristics learned from families of shapes. The main way is to use auto-encoders to learn and generate varied shapes in a shape family. Then, the auto-encoder can be trained in different ways. For instance, ShapeAssembly [2] sets the training on a domain-specific language (DSL) language. StructureNet [3] uses the auto-encoder combined with linkage graph to improve the results and the coherence of the shapes thus synthetized. Some methods try to manipulate 3D voxels as 2D images (3D shape segmentation) [4] with convolutional network or transcoding across 3D shape representations [5]. They extract features from 2D images, voxel representations and 3D point sets in order to improve the training of a neural network. SascNet [8] has already done an overlay of linkage graph with a trained neural network, but the input database must be labelled. More generally, AI-based methods need labelled database to train a neural network. For instance, PartNet [6] and ShapeNet [7] are two already labelled 3D assembly databases of quite simple shapes and objects. This is however not be the case in this paper where linkages are recognized from raw CAD assemblies.

Our paper describes an assembly of a shape family by means of graphs. Such a representation has already been adopted by meta-representation of shape family [9], assembly-based conceptual 3D modelling [10] and cross-class 3D object synthesis [11]. They all represent shape assemblies as structures which can be defined by graphs or vectors.

Our work is also related to Probabilistic Model for Component-Based Shape Synthesis [12] which suggests a solution to create new assemblies from an existing labelled database. The drawbacks of this method are the labelling of the 3D model databases and the linkage method. Indeed, most of existing methods use slots to link two different components in space. Then, with a least-square minimization, it is easy to assemble two components. The major issue is that it does not work for complex CAD assemblies because all components are placed in a thoughtful way which cannot be approximated.

As a conclusion, all these methods make it possible to create new assemblies thanks to a neural network trained with labelled databases. Thus, the major challenge of shape synthesis has been shifted towards the creation of labelled databases. Moreover, the automatic generation of CAD assemblies often uses already labelled CAD databases of everyday life objects, whereas this paper introduces a solution to enlarge existing databases of mechanical CAD assemblies.

3 OVERALL FRAMEWORK

The proposed method reduces user input in order to be able to apply it for any CAD assembly like pumps, motors, or many more complex CAD assemblies. The method starts by extracting a complete linkage graph from an assembly, and it stores a lot of useful information on its nodes and edges. Then, components sharing the same functional surfaces are recognized thanks to the complete linkage graph, and they are stored in a component family database. Finally, similar components are replaced to define new assemblies, thanks to the information stored into the linkage graphs. The components can be modified before placement to fit with the assembly. This process is illustrated on Figure 1 and detailed in the next sections.

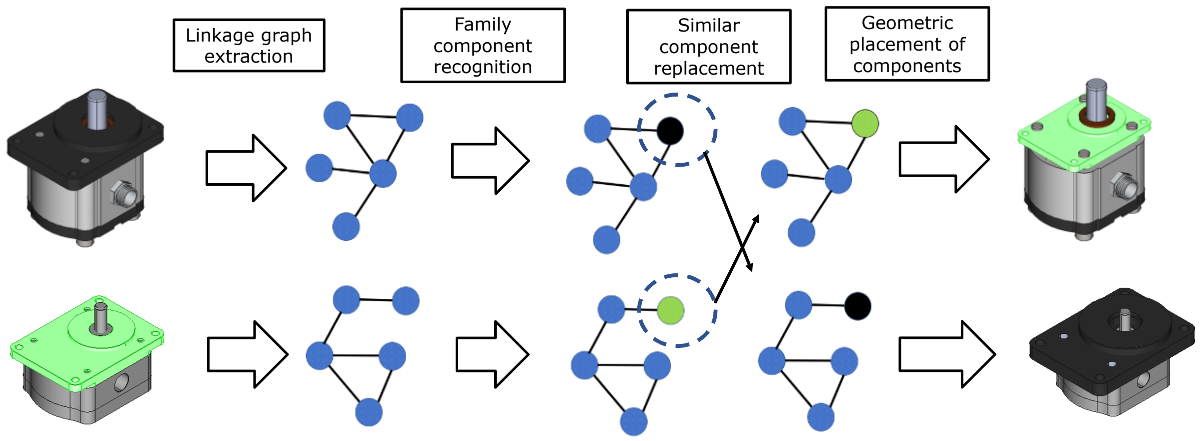
Linkage graph extraction
Family component recognition
Similar component replacement
Geometric placement of components

Figure 1: Overview of the proposed method creating a set of new assemblies (right) from existing ones (left).

4 LINKAGE GRAPH EXTRACTION

This is the very first step of the proposed approach (Figure 1). It aims at creating a linkage graph for each CAD assembly found in the database.

4.1 User Input

To be as generic as possible, and to be applicable on a large variety of CAD models, the method uses the STEP standard as input of our algorithm. Indeed, there are large databases of CAD models in STEP format available for free on the Internet. The 3D models presented in this paper are available from GrabCAD, but 3D models could be downloaded from other platforms such as Turbosquid, 3D Warehouse or ABC Dataset. More precisely, the AP214 protocol has been adopted to allow importing assemblies including different objects, and not through a unique object gathering together the whole assembly. Thus, using this protocol it is possible to work on features extraction of a component and its replacement in a new assembly.

Moreover, by default, parts are labeled with names given by their creators. But our algorithm does not rely on those names, because they may differ depending on their creators, which may be misleading. Here, the goal is to work at the level of the linkages between the components so as to automatically recognize similar configurations without using labels, and thus be able to replace components known to be similar in order to generate new CAD assemblies.

4.2 Graph Representation

The created graphs are composed of nodes and edges. Each node represents a component, and the edge between two components keeps track of the mechanical linkage between them. The edges also support additional information like the mechanical characteristics of the linkage. All this information is described below. An example of enriched linkage graph is shown in Figure 8.

4.3 Linkage Recognition

To create a linkage graph of a CAD assembly, it is mandatory to first recognize linkages between components. In this paper, only two linkage families are studied, but this could be extended with other types of linkages. The considered linkages are the pivot linkage and the flat-surface linkage. These two categories are sufficient to start working on partially-complete graphs, even if some linkages can thus not be detected. In order to recognize all existing linkages, eleven linkages would be necessary: housing, pivot, slide, propeller, sliding pivot, sphere, joint, flat support, linear or circular, and between a sphere and a plane. The graph created is a network of components, with vectors associated to each edge, which completely characterizes the linkage between the two components.

In order to recognize these linkages, the main idea is to browse all faces of the assembly and, for each face, to check the other faces and verify if some conditions are satisfied in order to decide if the components related to those faces are connected or not.

4.3.1 Pivot linkage

In order to recognize a pivot linkage between two components, it is assumed that a pivot linkage is a cylindrical surface contact whose translation is blocked. But even if the translation is not blocked, the linkage will be detected. Here are the conditions checked to test if two surfaces define a pivot linkage:

- The faces belong to two different components
- The two surfaces are cylindrical
- The axes of the cylinders are coaxial
- The radius of the two cylinders is the same
- The two cylinders have not the same orientation
- The two cylinders are nested, i.e. they touch each other

Figure 2 shows some of the hypotheses required to define a pivot linkage: a) the surfaces are touching each other but their orientation is the same, which violates the fifth hypothesis; b) the orientations are the same, but the surfaces are not nested, which contradicts the sixth hypothesis; c) surfaces are touching each other and are nested thus defining a pivot linkage considering only the fifth and sixth conditions.

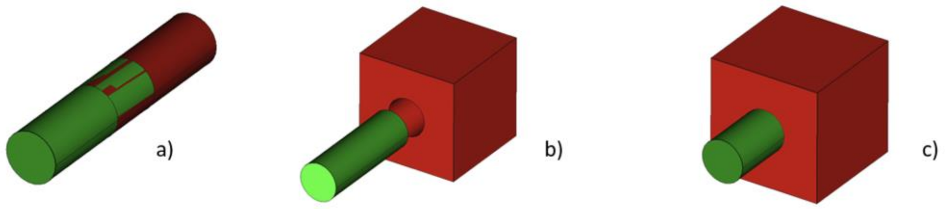

Figure 2: Pivot linkage hypotheses.

When all these conditions are satisfied, a pivot linkage is added to the edge of the graph linking the two related components of the assembly under analysis. A pivot linkage is defined by a vector Vp gathering together several characterizing information: Vp=[{‘Component1': 'Orientation1'}, {'Component2': 'Orientation2'}, Length, Axis, Radius, Center].

The Orientation of a face is defined as 'Forward' or 'Reversed' depending on how the filled side of the face is oriented. For instance, considering the cylinder of Figure 2.c, the cylindrical face of the red cube is 'Reversed' whereas the cylindrical surface of the green shaft is 'Forward'.

This method has false-positive and false-negative. Indeed, a free-form feature will never be detected as a pivot linkage. This may happen if in the STEP file a cylinder is represented by means of a NURBS surface, then the approach will not consider it as a cylinder. This is a new functionnality to be developed in the future. Also, two extruded circles that are not coaxial will be recognized as two distinct pivot linkages between two components, although all together they define a more complex linkage. This is why it is important to post-process the results to clean up duplicated linkages.

4.3.2 Flat-surface linkage

The conditions of a flat-surface linkage are:

- The faces belong to two different components
- The two surfaces are planar
- The orientations of the two surfaces are not the same
- The normal to the two surfaces are colinear
- The distance between the two surfaces is equal to zero.

When all those conditions are met, the flat-surface linkage is added to the edge of the graph linking the two related components of the assembly under analysis. This linkage is characterized by a vector Vs so that Vs=[{'Component1': 'Orientation1'}, {'Component2': 'Orientation2'}, Center, Axis, L1, L2]. For a planar surface, the orientation is directly linked to the normal vector, and it defines the filled size of the surface. The lengths L1 and L2 are the characteristic lengths of the two surfaces. A characteristic length is calculated as the maximum distance between two centers of mass of the surface edges. They are used later in the analysis process to compare two flat-surface linkages.

The false-positives for this linkage recognition are planar surfaces which are touching each other by their edges. Again, these configurations are to be detected and cleaned up during a post-processing step.

4.4 Pseudo-code and Complexity

The proposed recognition method goes through all the faces of the CAD models twice to determine the potential linkages between the components. The pseudo-code of the corresponding algorithm is presented on Figure 3.

```
1  Triaxial geometrical frame of reference Initialization
2  Extract planes:
3        Select all planar surfaces.
4  Extract cylinders:
5        Select all cylinder surfaces.
6  Planar surface linkage graph creation:
7        For each unique pair of surfaces:
8              Test Linkage between the two surfaces (5 tests)
9  Pivot Linkage graph creation:
10       For each unique pair of surfaces:
11             Test Linkage between the two surfaces (4 tests)
```

Figure 3: Pseudo Code.

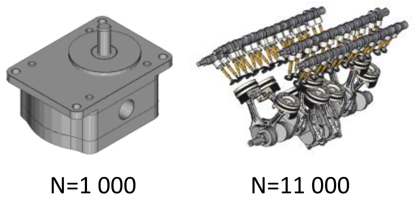

Figure 4: Example of number of surfaces.

The complexity of the graph creation algorithm is O(N²) with N the total number of surfaces in the assembly. This is due to the nested loops visible inside the pseudo-code of Figure 3. Thus, for an assembly with 1000 surfaces, like the small hydraulic pump of Figure 4, the computation time is about 3 minutes with an IntelCore i7 at 2,5 GHz and 8 Go of RAM. But, for a V12 engine with 11 000 surfaces (Figure 4), the algorithm takes 5 hours to analyze all possible pairs of faces. This could be improved in the future, while reducing the computation times, and while further optimizing the number of tests using for instance an octree-based decomposition to discard pairs of faces that evidently cannot be good candidate. Figure 5 shows the result of the linkage graph creation on a pump (1000 faces). The obtained graph is not very visual because it has not yet been labelled with names for the nodes and edges. It will be used as a basis to identify similarities between CAD assemblies in order to be able to replace components and create new assemblies from existing ones.

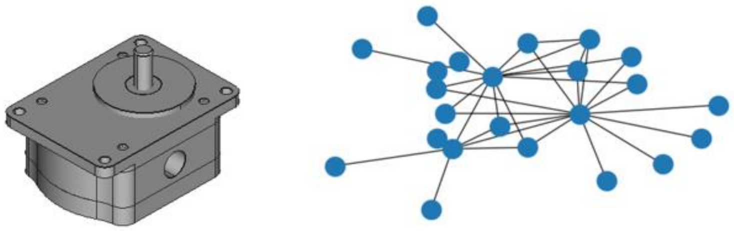

Figure 5: A hydraulic pump and its computed linkage graph.

5 COMPONENTS OVERLAY

5.1 Post-processing of the Linkage Graph

Once the linkage graph created, a post-processing step is applied in order to clean up identical connections, for example to consider only one pivot connection between a screw thread and a bore. For each edge of the graph, the algorithm browses all the related linkages. When two linkages are known to be similar, i.e., when all the characteristics except their placement are the same, the algorithm keeps only one linkage. Once all the edges have been treated, the graph is ready to be used for component recognition.

5.2 Similarity Recognition Between Components

In order to distinguish components that can be replaced from the ones that cannot, the similarities between two components have to be recognized. The hypothesis is that two components are interchangeable if they have the same type of linkage. This notion can differ depending on the type of parts, as parts are linked differently in mechanical assemblies. Thus, in its current version, our algorithm requires a human decision to characterize a family of components by its connections with the others. In the future, this association between the family of components and its types of links is to be automated.

This is illustrated on the example of a screw, which can be easily characterized by its linkages with the parts to which it is linked. Indeed, a screw can be characterized by the following conditions on its linkages (Figure 6):

- A screw has one or more pivot linkages on the same axis with a 'Forward' orientation.
- A screw has a flat-surface linkage with a greater characteristic length L1 than the cylinder radius R.

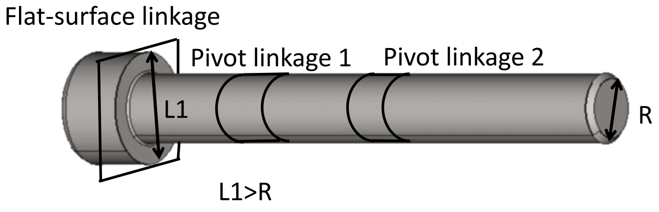

Figure 6: Characteristic linkages of a screw with the components with which it is linked.

Then, the recognition of potential replaceable components is run on multiple assemblies, e.g. on two hydraulic pumps as shown on Figure 7. As illustrated, the proposed method is particularly interesting as it allows identifying similar components with very few hypotheses. As theses components are now known to be similar, they are replaceable and can be exchanged between the considered assemblies.

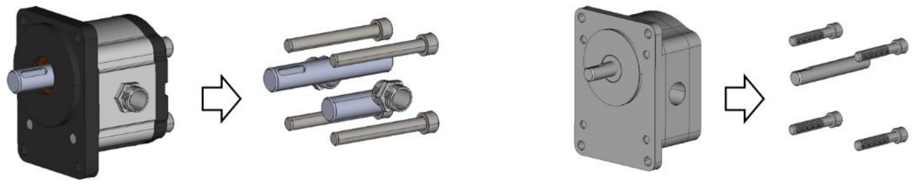

Figure 7: Detection of potential replaceable screws in two different assemblies

5.3 Inter-graph components exchange

Once the list of potential replaceable components has been set up, an enriched graph with colored parts for a better visualization can be displayed (Figure 8).

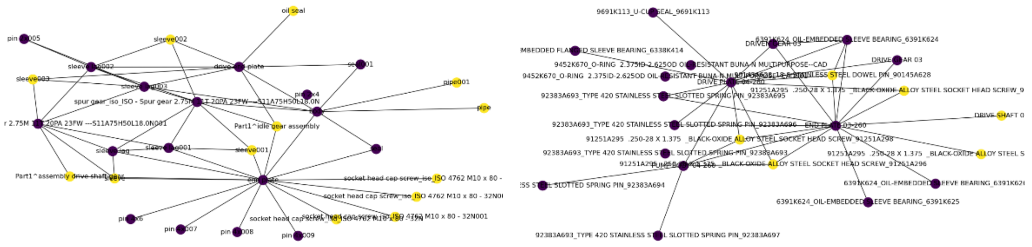

Figure 8: Enriched linkage graphs of two hydraulic pumps.

In these graphs, yellow components are potential replaceable screws for the two considered hydraulic pumps shown in Figure 7. Names used to label the nodes are directly extracted from the original CAD assemblies, but they are not considered during the indentification process. At the end, there is a total of 13 detected components in the first pump, and 6 in the second one.

Our method allows to recognize replaceable components, and it is easy to foresee the number of new assemblies possibly generated. Indeed, if there are 10 assemblies with for each assembly 5 components known to be replaceable, there are 50 replaceable components overall. So, this method can create 50^5 new assemblies, i.e. more than 3 billion assemblies, starting from a database of 10 assemblies only. However, this calculation does not consider configurations wherein components are geometrically similar, like for instance the repetition of a screw in an assembly. Thus, in the current version, the generation has not been fully automated so as to let the user decide which components are to be replaced. Otherwise, this would generate a huge number of assemblies, among which many of them would be exactly the same.

Once the replacements have been decided, the graph of the new assembly can be created. The information associated to the new linkage vector is described Figure 9.

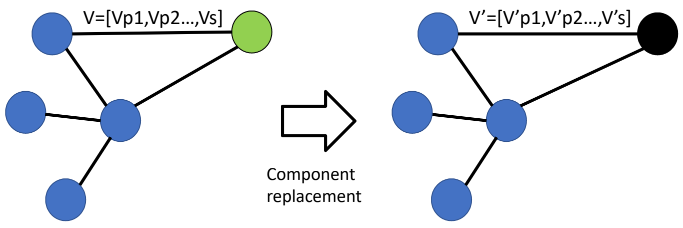

Figure 9: Updated linkage vectors after an exchange of components.

For a pivot linkage, the newly created linkage vector is obtained as follows : V’p=[{‘Component1': 'Orientation1'}, {'Component_replaced': 'Orientation2'}, Length, Axis, Radius, Center]. The underlined component is the new component taken from an other assembly, whereas everything else comes from the original one. A new vector is also created similarly for flat-surface linkages.

These conditions are defined so that it is the new component that has to adapt to fit into the original assembly. Furthermore, each new vector V’ has the complete information of the linkage between the new component and the original assembly.

Finally, the new components have to be copied and pasted but also they need to be transformed so as to fit to their new locations in the new assembly. The placement and the adaptations are described in the next section.

6 COMPONENTS PLACEMENT AND ADAPTATIONS

As the proposed method handles CAD models described in STEP files, when imported in the CAD modeler they are considered as dead models without construction trees. Thus, it is difficult to modify them in order to adjust their shapes to their new positions. Therefore, the modification and placement of the new parts is an important step of our algorithm.

Indeed, the manipulated assemblies often do not have the same scale and shape geometry. The modifications of the geometry depend on the considered family of parts. For example, screws do not have to undergo major geometric modifications, as they just need to adapt to new holes in which they will be placed. For some parts, the required modifications could however be more complex.

This paper focuses on four types of operation: rescale, cropping, fill and drill holes, which already answer many adjustment requirements, like for instance the replacement of a screw by another one at a possibly different location.

6.1 Geometric Modifications

The geometric modifications can create inconsistent configurations. Here, it is assumed that two parts of the same family of parts have approximately the same geometry, i.e., that two components with the same linkages have the same functional surfaces. In this case, the processing is weak, and the part is not very warped. In most cases, a rescaling is sufficient to fit a part correctly in an assembly. But there are cases for which this does not work or for which it is not enough. Problematic cases refer to parts which have more than two linkages in the same direction. Indeed, the rescale tool allows to scale the part in each direction. Then, the two linkages can be replaced in each direction. But if there are more than two linkages, it is not possible to replace all linkages to the right place.

Figure 10 shows a partial list of possible geometric modifications of a part a), with filled holes b), drilled holes c), rescaling d) or cropping e).

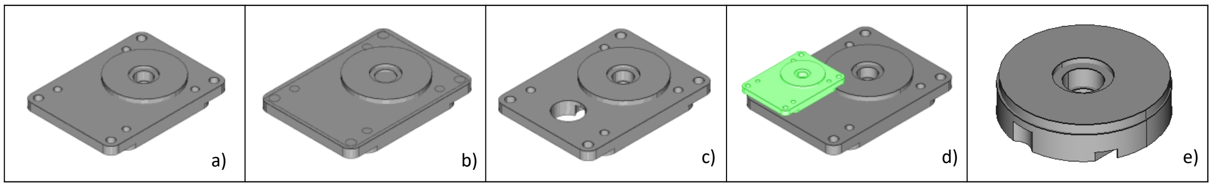

Figure 10: Examples of geometric modifications of a CAD model required to welcome new parts or to fit in a new component.

In case of a family of parts where there are less than two connections per direction, the part is rescaled so that all connections fit with the assembly. For example, if the screw does not fit well with the new assembly, here are the rescaling operations, wherein R2 is the target Radius and R1 the original one, L2 is the target Length and L1 is the original one (Figure 11):

- Rescale of a ratio R2/R1 in X and Y directions,
- Rescale of a ratio L2/L1 in Z direction.

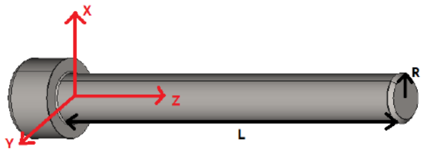

Figure 11: Parameters involved in the geometric adaptation of a screw to prepare its replacement in a new assembly.

Moreover, as screws are well-known parts, in this case, it could be possible to look for replaceable screws in a database like Traceparts. Doing this way, the replaced screws still have standard dimensions available in a catalog.

In case of a family of parts where there are more than two connections per direction, new holes must be drilled at the locations defined in the linkage graph. Reversely, the existing holes must be filled in as they are not anymore hosting parts. There is no technical issue for this operation, but the focus is on the screw replacement in this paper.

6.2 Placement within an Assembly

In order to place the new part within the assembly, a new local reference frame is defined for all parts of the family. This reference frame allows to make transformations with respect to it. Thus, the placement operation will be the same for all parts of the family. For instance, in case of screws, the local frame is defined as described in Figure 11.

Then a rotation and a translation are defined to place it at the right place. All information required to perform those transformations is available in the linkage graph. For example, the target local center of the screw is the so-called 'Center' of the target cylinder projected on the plane of the flat-surface linkage along the axis of the screw.

Once components have been placed, the newly generated assemblies are exported as STEP files. Moreover, this process generates new assemblies from existing ones, if those initial assemblies belong to the same family, then the newly created ones can also be associated to this family. For instance, assemblies generated from the replacements performed on the assemblies of Figure 7 can all be considered as new pump assemblies.

7 RESULTS AND DISCUSSION

The method has been tested on several assemblies, and it is illustrated here for the generation of new hydraulic pumps from the replacement of screws. Figure 12.a shows two hydraulic pumps, where the body and the drive plate are hidden, and used as inputs of the assembly generation algorithm. The associated linkage graphs are automatically created (Figure 12.b), and the similar components are detected (Figure 12.c). All those components can replace the two components of the second pump (Figure 12.d). Then, $5^2=25$ CAD assemblies can be created with automatically adapted and placed components as shown in Figure 12.e.

Here, the geometrically similar components are cleaned to avoid duplicate assemblies. For example, only one screw of each pump is taken. In addition, a few created assemblies are not meaningful like the one with the green pipe used as the drive shaft. However, our method does not yet verify collisions within the created assemblies, and some operations like verification of semantic interoperability could be improved. Indeed, the components are replaced here by geometrically similar ones but may slightly differ semantically. This can result in assemblies that make no sense semantically. It would be interesting to test this method by mixing semantically different assemblies to be able to check the coherence of the created assemblies. This is part of our future works.

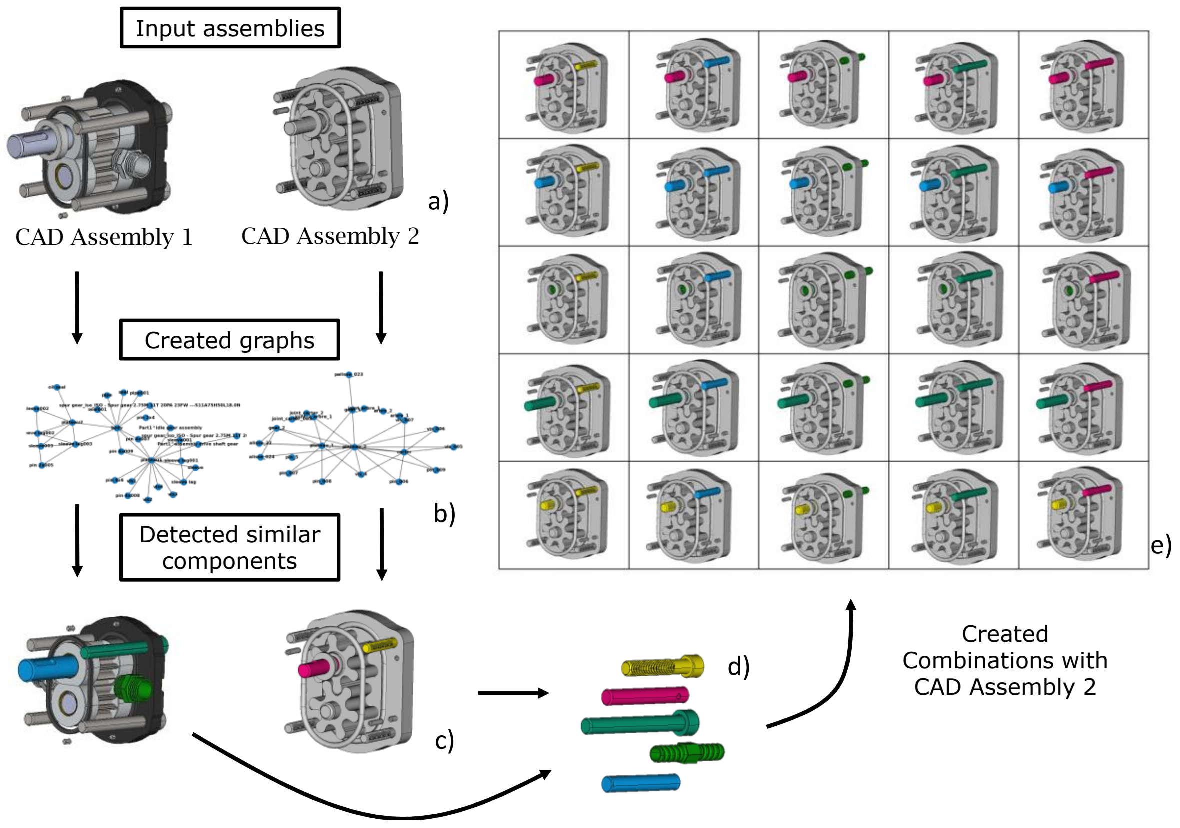

Figure 12: Creation of 25 new assemblies while automatically replacing screws between two input assemblies.

8 CONCLUSION

This paper introduces a method to create new assemblies from diverse existing assemblies of a same family. It allows to generate linkage graph from a non labelled assembly, and it analyzes these graphs to find similar components. Finally, it replaces similar components in an assembly to create multiple new consistent CAD assemblies. For the moment, this method requires a human interaction during a pre-processing step, in order to define the characteristics of linkages for a part family. Therefore, the application of the method is only shown for the case of screws. Our method works well with part assemblies consisting of cylinders and flat surfaces. But it is still limited for free form features or for assemblies with connections that are not cylindrical or flat. However, even if this is a limitation as there exists more types of linkage, in practice, these two types of linkage cover already a wide range of concrete configurations found in industrial models. Furthermore, the method could be improved by changing the structure of the graph before creating new assemblies. Indeed, a semantic graph could improve the diversity of the assemblies during the creation process. Finally, the proposed approach creates many CAD assemblies, which can be stored in a database and used for Machine Learning applications.

REFERENCES

[1] Li, J.; Xu, K.; Chaudhuri, S.; Yumer, E.; Zhang, H.; Guibas, L.: GRASS: generative recursive autoencoders for shape structures, 2017, ACM Trans. Graph. 36, 1–14. https://doi.org/10.1145/3072959.3073637

[2] Jones, R.K.; Barton, T.; Xu, X.; Wang, K.; Jiang, E.; Guerrero, P.; Mitra, N.J.; Ritchie, D.: ShapeAssembly: Learning to Generate Programs for 3D Shape Structure Synthesis, ACM Trans. Graph., 2020. 39, 1–20. https://doi.org/10.1145/3414685.3417812

[3] Mo, K.; Guerrero, P.; Yi, L.; Su, H.; Wonka, P.; Mitra, N.; Guibas, L.J.: StructureNet: Hierarchical Graph Networks for 3D Shape Generation, 2019, arXiv:1908.00575 [cs].

[4] Kalogerakis, E.; Averkiou, M.; Maji, S.; Chaudhuri, S.: 3D Shape Segmentation with Projective Convolutional Networks, in: 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). Presented at the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), IEEE, Honolulu, HI, 2017, 6630–6639. https://doi.org/10.1109/CVPR.2017.702

[5] Furuya, T.; Ohbuchi, R.: Transcoding across 3D shape representations for unsupervised learning of 3D shape feature, Pattern Recognition Letters, 2020, 138, 146–154. https://doi.org/10.1016/j.patrec.2020.07.012

[6] Mo, K.; Zhu, S.; Chang, A.X.; Yi, L.; Tripathi, S.; Guibas, L.J.; Su, H.: PartNet: A Large-Scale Benchmark for Fine-Grained and Hierarchical Part-Level 3D Object Understanding, in: 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Presented at the 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), IEEE, Long Beach, CA, USA, 2019, 909–918. https://doi.org/10.1109/CVPR.2019.00100

[7] Chang, A. X.; Funkhouser, T.; Guibas, L.; Hanrahan, P.; Huang, Q. ; Li, Z.; Savarese, S.; Savva, M.; Song, S.; Su, H.; Xiao, J.; Yi, L.; Yu, F.: ShapeNet: An Information-Rich 3D Model Repository, 2015, arXiv:1512.03012 [cs].

[8] Lang, X.; Sun, Z.: Structure-aware shape correspondence network for 3D shape synthesis, Computer Aided Geometric Design, 79, 2020, 101857. https://doi.org/10.1016/j.cagd.2020.101857

[9] Fish, N.; Averkiou, M.; van Kaick, O.; Sorkine-Hornung, O.; Cohen-Or, D.; Mitra, N.J.: Meta-representation of shape families, ACM Trans. Graph., 2014, 33, 1–11. https://doi.org/10.1145/2601097.2601185

[10] Jaiswal, P.; Huang, J.; Rai, R.; Assembly-based conceptual 3D modeling with unlabeled components using probabilistic factor graph, Computer-Aided Design, 74, 2016, 45–54. https://doi.org/10.1016/j.cad.2015.10.002

[11] Su, X.; Chen, X.; Fu, Q.; Fu, H.: Cross-class 3D object synthesis guided by reference examples, Computers & Graphics, 54, 2016, 145–153. https://doi.org/10.1016/j.cag.2015.06.009

[12] Kalogerakis, E.; Chaudhuri, S.; Koller, D.; Koltun, V.: A probabilistic model for component-based shape synthesis, ACM Trans. Graph. 31, 2012, 1–11. https://doi.org/10.1145/2185520.2185551