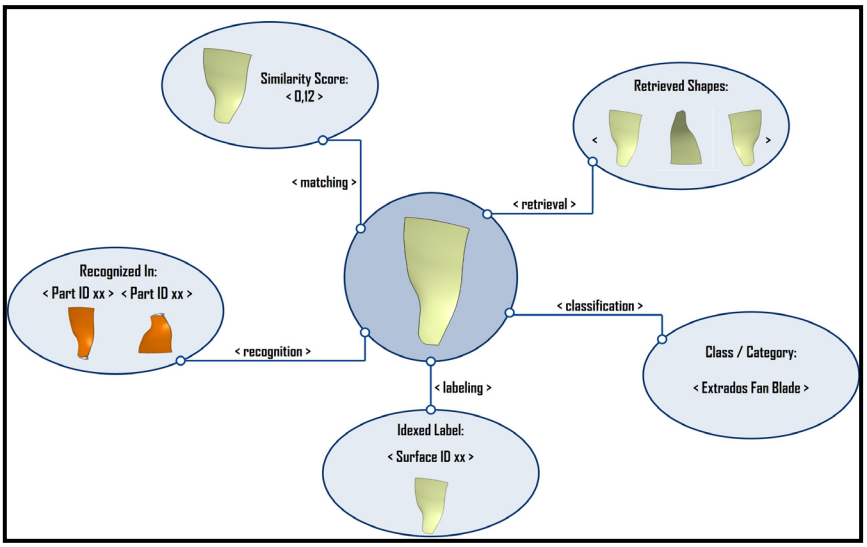
Analyse de forme pour la Rétro-Ingénierie de modèles 3D : application à l'industrie aéronautique

Philippe Williatte

Par **Philippe WILLIATTE**

Analyse de forme pour la Rétro-Ingénierie de modèles 3D : application à l'industrie aéronautique

Thèse présentée
pour l'obtention du grade
de Docteur de l'UTC



Soutenue le 6 février 2024
Spécialité : Génie Industriel : Unité de recherche en Mécanique - Laboratoire Roberval (FRE UTC - CNRS 2012)
D2796

Thèse présentée à

L'UNIVERSITE DE TECHNOLOGIE DE COMPIEGNE

Pour l'obtention du
GRADE DE DOCTEUR DE L'UNIVERSITE DE TECHNOLOGIE
DE COMPIEGNE

Analyse de forme pour la Rétro-Ingénierie de modèles 3D : application à l'industrie aéronautique

Par **Philippe WILLIATTE**

Spécialité : Génie Industriel

Préparée au sein de la société
SAFRAN AIRCRAFT ENGINES, GROUPE SAFRAN
Et du laboratoire de recherche ROBERVAL

Régie par la
CONVENTION INDUSTRIELLE DE FORMATION PAR LA RECHERCHE

Thèse soutenue le 6 février 2024

Encadrement **:**

Sébastien Remy, sebastien.remy@utt.fr, Spécialité ingénierie mécanique, Laboratoire LASMIS, UTT – 12 Rue Marie Curie, 10300 Troyes – Directeur de thèse

Alexandre Durupt, alexandre.durupt@utc.fr, Spécialité Ingénierie mécanique, Laboratoire Roberval, UTC – Directeur de thèse

Matthieu Bricogne, matthieu.bricogne@utc.fr, Enseignant-chercheur, Spécialité ingénierie industrielle, Laboratoire Roberval UTC – Encadrant

Stive Sinna, stive.sinna@safrangroup.com, Pilote Projets Méthodes & Outils, Département Conception collaborative et Management configuration - Méthodes, Process & Outils SAFRAN Aircraft Engines – Encadrant

Jury

Mme Lilia Gzara, professeure des universités, examinatrice, INSA Lyon, département génie industriel, Villeurbanne

Mme Claire Lartigue, professeure des universités, examinatrice, université Paris-Saclay, laboratoire LURPA, Gif-sur-Yvette

M. Jean-Philippe Pernot, professeur des universités, rapporteur, Arts & métiers ParisTech, laboratoire LISPEN, Aix-en-Provence

M. Bertrand Rose, professeur des universités, rapporteur, université de Strasbourg, laboratoire Icube, Illkirch

M. Stive Sinna, pilote projets méthodes & outils, Safran Aircraft, département YOX, Moissy-Cramayel

M. Matthieu Bricogne, enseignant chercheur, examinateur, université de technologie de Compiègne, laboratoire Roberval

M. Alexandre Durupt, maître de conférences, directeur de thèse, université de technologie de Compiègne, laboratoire Roberval

M. Sébastien Rémy, enseignant chercheur, directeur de thèse, université de technologie de Troyes, laboratoire LASMIS

M. Benoît Eynard, enseignant chercheur, examinateur, université de technologie de Compiègne, laboratoire Roberval

Résumé

Analyse de forme pour la Rétro-Ingénierie de modèles 3D : application à l'industrie aéronautique

Philippe WILLIATTE
Laboratoire Roberval, Université de Technologie de Compiègne
Safran Aircraft Engines

Contexte scientifique des travaux de recherche

Au cours des deux dernières décennies, l'émergence de l'outil informatique a révolutionné les pratiques de l'ingénierie. La transformation des supports traditionnels de développement et de conception a engendré une augmentation massive des volumes de données numériques, confrontant les entreprises à de nouveaux défis en matière de gestion des informations associées à leurs produits et systèmes.

Dans les industries manufacturières, la maquette numérique joue un rôle central dans les activités d'ingénierie. Plus qu'une simple représentation tridimensionnelle (3D), les modèles de Conception Assistée par Ordinateur (CAO) qui composent la maquette numérique intègrent une richesse sémantique issue des connaissances expertes exploitées pour leur développement. De manière générale, la gestion des modèles CAO sur l'ensemble des étapes du cycle de vie des produits est assurée par les Systèmes d'Information (SI) de l'entreprise, notamment ceux de *Product Data Management* (*PDM*).

Cependant, de nombreuses entreprises continuent de rencontrer des difficultés avec des données parfois incomplètes ou obsolètes. On parle alors de problématiques de continuité numérique, que l'on définit comme la capacité d'une entreprise à disposer de l'ensemble des informations numériques d'un produit tout au long de son cycle de vie. En plus de l'amélioration des processus d'ingénierie et leur meilleure intégration dans une logique de continuité numérique, il est parfois nécessaire de « réparer » ou « recréer » ce qui existe en l'état, en ayant recours à des activités de Rétro-Ingénierie afin de garantir l'accès à des informations complètes, disponibles et exploitables.

La Rétro-Ingénierie (*RE* pour *Reverse Engineering*) est le processus d'analyse et de compréhension d'un produit ou d'un système existant dans le but de recréer et retrouver des informations et connaissances relatives à sa définition. En particulier, le *RE* s'intéresse aux techniques avancées d'apprentissage automatique (ML pour *Machine Learning*) afin d'aider l'humain à mieux raisonner sur ses données et lui offrir un accès unifié et un traitement haute performance de l'information. Les progrès des méthodes et outils d'analyse de forme (*SA* pour *Shape Analysis*) permettent d'améliorer l'exploration de bases de modèles CAO pour les comparer, les contextualiser et créer du lien entre ces données. Ces techniques permettent l'extraction de diverses informations et la création de connaissances utiles aux activités de *RE*.

Problématique de recherche et proposition scientifique

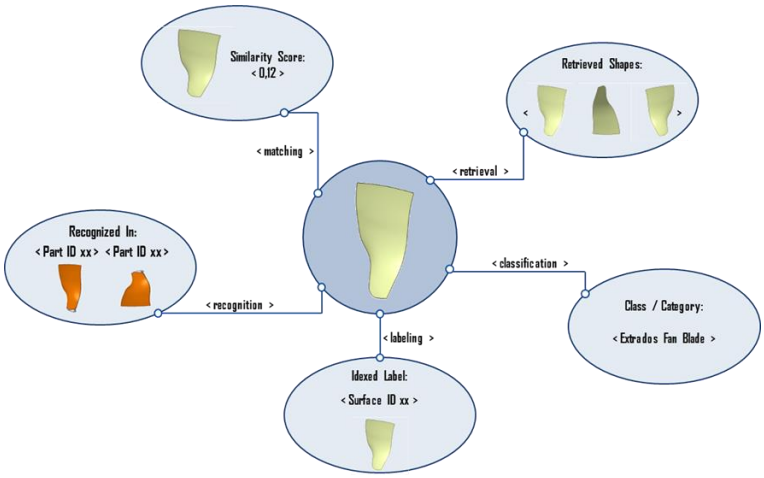
L'industrie aéronautique est confrontée à des défis de continuité numérique de plus en plus complexes. En cause, les évolutions des logiciels de CAO et des SI au cours de la longue durée de vie des produits aéronautiques sont susceptibles de rendre certains formats 3D non éditables ou de bas niveau sémantique. Par ailleurs, la grande complexité des produits aéronautiques nécessite la coopération et l'interopérabilité entre les nombreux acteurs et expertises métiers des différentes disciplines d'ingénierie impliquées. Cela se manifeste généralement par une diversité des formats de données utilisés ainsi que le cloisonnement et la dispersion des informations dans divers éléments du SI. Une perte sémantique des modèles CAO qui se traduit par leur incomplétude, leur obsolescence, et parfois même leur absence est donc constatée.

Face aux problématiques de continuité numérique des entreprises industrielles, ces travaux de recherche s'intéressent à l'exploitation des activités de *SA* dans le cadre de processus de *RE* pour le maintien de l'intégrité des modèles CAO en environnement *PDM*.

Dans cette perspective, un nouveau type de descripteurs de forme 3D est proposé pour s'adapter aux spécificités géométriques des modèles aéronautiques, et plus particulièrement pour capturer et représenter efficacement les caractéristiques des surfaces aérodynamiques complexes. Sur la base de techniques avancées de géométrie computationnelle, d'analyse statistique et d'apprentissage automatique, ce nouveau descripteur s'intègre aux méthodes courantes de *SA* tels que le *matching*, le *retrieval*, la *classification* ou encore la *recognition.* De plus, le *labeling* de formes est proposé comme une nouvelle activité de recherche et d'identification précise de modèles 3D complexes en vue d'améliorer la compréhension des modèles CAO et contribuer à l'avancement des capacités de recherche et d'exploration au sein des SI industriels.

Pour une meilleure exploitation des activités de *SA* dans les processus de *RE*, un modèle de données est proposé pour la gestion des informations et des connaissances relatives aux modèles CAO. Le *Knowledge-Based Reverse Engineering Model* (*KBRE Model*) intègre des fonctionnalités avancées de traitement et d'analyse des modèles CAO ainsi que des bases de données qui les contiennent. Le *KBRE Model* facilite ainsi le développement et la validation de techniques avancées de SA et leur intégration aux processus de RE.

Cette thèse apporte une contribution dans le domaine de l'analyse de formes 3D et de la Rétro-Ingénierie et de modèles CAO dans un contexte industriel.



Remerciements

Je tiens à remercier mes directeurs Alexandre Durupt et Sébastien Remy pour l'encadrement et le support scientifique qu'ils m'ont apportés tout au long de ces années de thèse. Pour les mêmes raisons, merci à Matthieu Bricogne, sans qui je ne me serais pas orienté vers le doctorat à la sortie de mes études d'ingénieurs.

Mes remerciements à mes managers de safran Aircraft Engines, Delphine Grohens et Paul-Emiland Marques qui m'ont offert l'opportunité de réaliser ce projet de recherche dans un cadre industriel passionnant, avec la confiance et le support nécessaires.

Il serait long de tous les citer, mais merci à l'ensemble de mes collègues de safran Aircraft Engines pour mon intégration dans l'équipe. Leur sympathie, l'aide apportée et l'ambiance conviviale ont largement participé à ma satisfaction professionnelle.

Tout particulièrement, merci à mon encadrant industriel Stive Sinna pour sa bienveillance et ses conseils afin de mieux intégrer mes recherches au contexte industriel. Merci aussi à Cyrille Le-Lann d'avoir permis le lancement du projet malgré un contexte incertain de pandémie mondiale.

Un grand merci à l'ensemble de ma famille ainsi que mes amis pour tout ce qu'ils m'apportent.

Enfin, merci à mes parents sans qui rien de tout cela n'aurait été possible.

Table des matières

Index des notations

3D : tridimensionnel

AFR : Automatic Feature Recognition

AISM : Algebraic Implicit Surface Model

AM : Adjacency Matrix

ANN : Artificial Neural Network

BOM : Bill of Material

BRep : Boundary Representation

CAE : Computer Aided Engineering

CAO : Conception Assistée par Ordinateur (CAD pour Computer Aided Design)

CNN : Convolutionnal Neural Network

CSG : Constructive Solid Geometry

DL : Deep Learning

Freeform : forme libre, non primitive

KB : Knowledge Base

KBE : Knowledge-Based Engineering

KBRE : Knowledge-Based Reverse Engineering

KM : Knowledge Management

ML : Machine Learning

MLP-C : Multi-Layer Perceptron Classifier

PDM : Product Data Management

PLM : Product Lifecycle Management

POO : Programmation Orientée Objet

RE : Reverse Engineering

SA : Shape Analysis

SI : Système d'Information

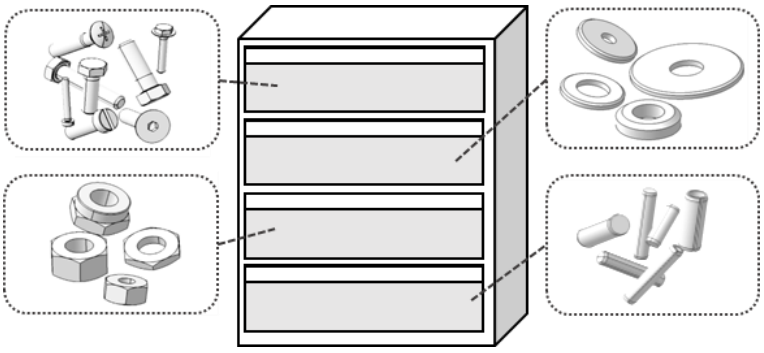
UML : Unified Modeling Language

Table des figures

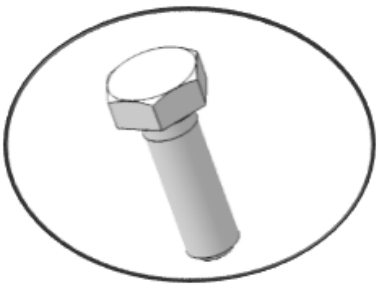
Avant-propos

Connaissez-vous Bob ?

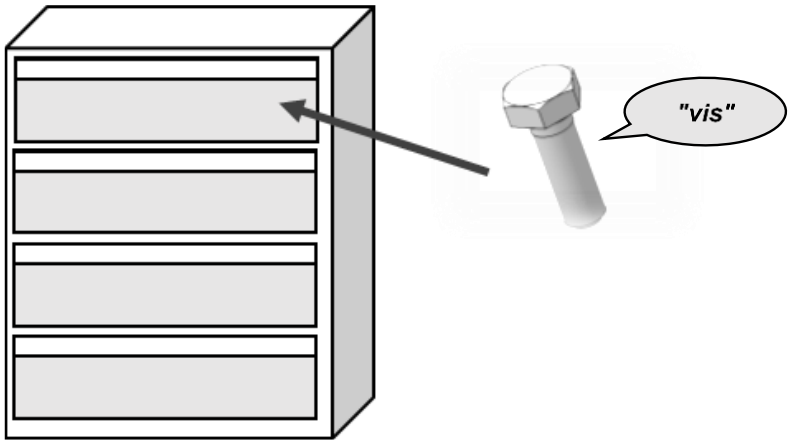
Dans son garage, Bob a une armoire dans laquelle il range les éléments de visserie qu'il utilise pour ses projets personnels.
Étant un tantinet méticuleux, ce dernier range les objets selon leur catégorie : un tiroir pour les vis, un pour les écrous, un pour les rondelles, et le dernier pour les goupilles.



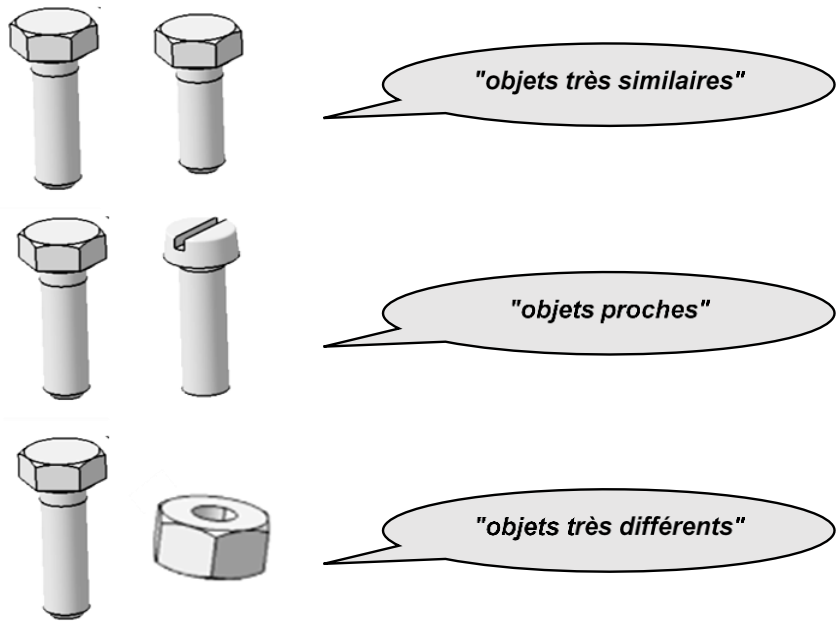
Bob trouve un objet sur le sol de son garage. sans plus d'informations à son sujet, ce n'est qu'en l'analysant qu'il peut mieux le connaitre.



Tout d'abord, Bob doit le catégoriser, c'est-à-dire déterminer de quel « type » d'objet il s'agit.
S'il peut dire qu'il s'agit d'une vis, il sait alors classifier l'objet (*object classification*). Bob peut maintenant le ranger dans le tiroir des vis.

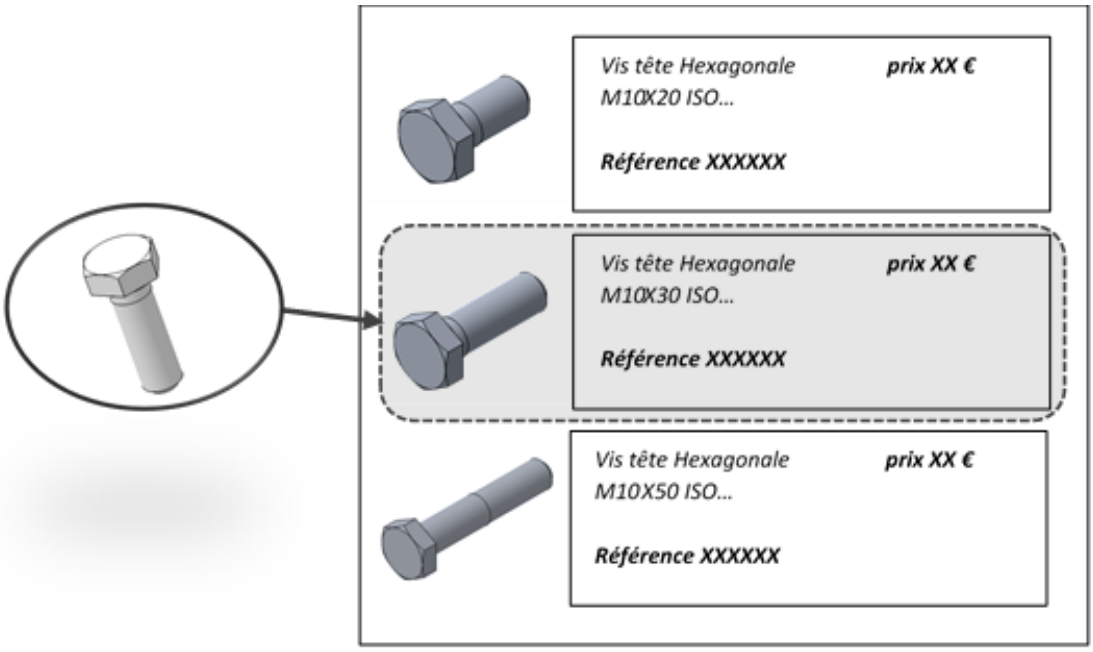


Pour son nouveau projet de réparation, Bob a besoin de six vis semblables. Il recherche dans son tiroir les cinq vis qui ressemblent le plus à celle dont il dispose déjà. Pour cela, il la compare à chaque objet du tiroir des vis et estime leur similarité (*object matching*).



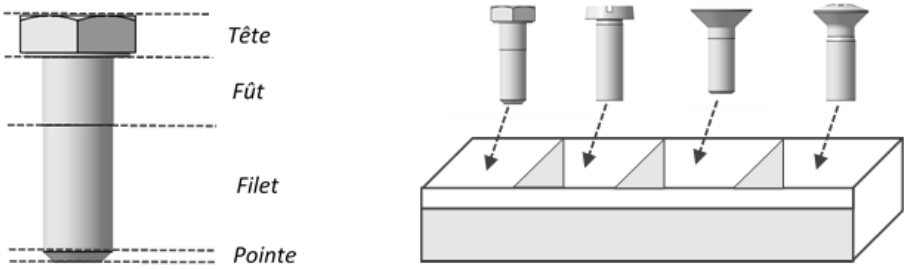
En sélectionnant les cinq objets qu'il considère comme les plus similaires à sa vis, Bob a retrouvé des objets proches (*object retrieval*).

Seulement, les vis retrouvées ne conviennent pas toutes. Pour en acheter de nouvelles, Bob consulte un catalogue fournisseur et cherche à identifier le bon produit.
En déterminant la bonne référence avec certitude, Bob vient de labéliser un objet (*object labeling*).



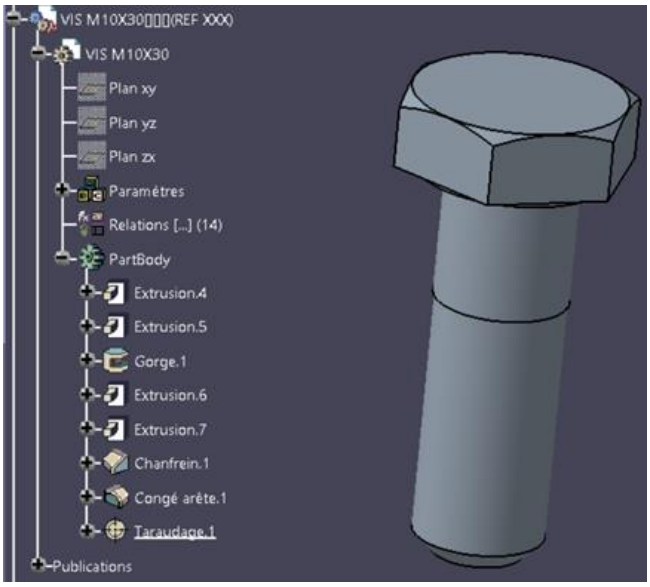
Sa méthode de rangement n'étant pas optimale, Bob décide de réorganiser son tiroir à vis en le divisant en quatre compartiments. Les objets seront classés en fonction du type de leur tête, indépendamment de leur taille ou de leur diamètre.

Pour cela, Bob doit segmenter les vis (*object segmentation*) pour n'en analyser qu'une partie, la tête. Il peut alors reconnaître une composante des objets (*object recognition*).



Enfin, Bob souhaite disposer d'un modèle numérique 3D de cette vis pour l'intégrer à ses projets de conception personnels. En utilisant un logiciel de modélisation 3D, il reconstruit la géométrie de la vis (*3D model reconstruction*).

Dans cette donnée numérique, Bob peut intégrer différentes informations relatives à l'objet, telles que son nom, sa référence, sa matière, etc. Une fois stocké dans un Système d'Information, ce modèle 3D de Conception Assistée par Ordinateur représente le support de toute la connaissance que Bob a su recréer et retrouver au sujet de l'objet analysé.



En analysant une donnée, la vis, Bob a su la contextualiser et la catégoriser, la transformant ainsi en information exploitable. Par ailleurs, ses analyses lui ont également permis de déduire diverses connaissances à son sujet comme sa référence dans un catalogue fournisseur ou encore sa ressemblance avec d'autres informations disponibles.

Supposons maintenant que Bob est au travail. Il est projeteur dans une entreprise industrielle du secteur aéronautique, et rencontre des besoins similaires à ceux rencontrés lorsqu'il a retrouvé une vis isolée. Seulement, les activités d'analyse ne portent plus sur un petit ensemble de pièces de quincaillerie, mais sur de larges volumes de données numériques comprenant les modèles CAO ainsi que des données diverses et variées relatives à la définition de composants aéronautiques complexes.

Lorsque l'objet d'étude est un produit ou un système mécanique issu d'un processus d'ingénierie, on qualifie de **Rétro-Ingénierie** ou ***Reverse Engineering*** l'ensemble des processus visant à recréer et retrouver des informations et connaissances qui lui sont associées.
Grâce aux activités **d'analyse de forme**, un humain peut étudier et comprendre la géométrie d'un modèle 3D pour en déduire des connaissances, ainsi que faciliter la recherche d'informations pertinentes au sein des très importants volumes de données de l'entreprise.

L'analyse de forme ou *Shape Analysis* regroupe un ensemble d'activités, qui, intégrées aux processus de ***Reverse Engineering***, participe avec l'ensemble du **Systèmes d'Information** au processus de création, d'organisation, d'identification, de stockage, de diffusion et enfin d'utilisation des informations au sein d'une entreprise, appelé la **Gestion des Connais**sa**nces,** ou ***Knowledge Management***.

Introduction

Les travaux de recherche présentés dans ce manuscrit ont été menés dans le cadre d'une thèse de doctorat en partenariat avec le laboratoire de recherche Roberval de l'Université de Technologie de Compiègne et le département Conception collaborative et Management de la configuration de la société Safran Aircraft Engines.

Contexte de recherche

Au cours des deux dernières décennies, l'émergence de l'outil informatique a révolutionné les pratiques d'Ingénierie. La transformation des supports traditionnels de développement et de conception a engendré une augmentation massive des volumes de données numériques, confrontant les entreprises du secteur industriel à de nouveaux défis en matière de gestion des informations associées à leurs produits et systèmes (Chandrasegaran et al., 2013)

Au cœur des activités d'ingénierie pour la conception et de développement de produits, les modèles de Conception Assistés par Ordinateurs (CAO) et les maquettes numériques (DMU pour *Digital Mock Up*) qu'ils composent jouent un rôle essentiel. Plus qu'une simple représentation tridimensionnelle des objets, les modèles CAO intègrent diverses informations géométriques et fonctionnelles issues des connaissances expertes mises en œuvre par les activités d'ingénierie. Le maintien de l'intégrité des modèles CAO s'impose donc comme une nécessité contre la perte d'informations.

Tout au long des étapes du cycle de vie des produits, la gestion des données CAO, de même que celle de l'ensemble de ses données de définitions, est généralement assurée par un élément du Système d'Information (SI) de l'entreprise appelé le système de *Product Data Management* (Lupinetti et al., 2019). Le *PDM* est ainsi un élément clé pour la sauvegarde du capital d'une entreprise et le maintien de sa compétitivité.

Dans ce contexte de transformation numérique pour l'industrie 4.0, l'industrie aéronautique est confrontée à diverses problématiques de continuité numérique, que l'on définit comme la capacité d'une entreprise à disposer de l'ensemble des informations numériques d'un produit tout au long de son cycle de vie. Des causes inhérentes aux outils et systèmes informatiques ou à la structure organisationnelle complexe de l'entreprise compliquent l'accès à l'ensemble des informations relatives aux produits. Tout d'abord, les évolutions des logiciels de CAO et du SI au cours de la longue durée de vie des produits sont susceptibles de rendre certains formats 3D non éditables ou de bas niveau sémantique. Par ailleurs, la grande complexité des produits aéronautiques nécessite de multiples représentations pour la coopération et l'interopérabilité entre les nombreux acteurs et expertises métiers des différentes disciplines d'ingénierie impliquées. Cela se manifeste généralement par une diversité des formats de données utilisés ainsi que le cloisonnement et la dispersion des informations dans divers éléments du SI. Une perte sémantique des modèles CAO qui se traduit par leur incomplétude, leur obsolescence, et parfois même leur absence est donc constatée.

L'hypothèse sous-jacente à nos travaux de recherche consiste à supposer que les informations manquantes à la richesse des modèles CAO restent néanmoins présentes quelque part dans le SI. En plus de l'amélioration des processus d'ingénierie et leur meilleure intégration dans une logique de continuité numérique, il est parfois nécessaire de « réparer » ou de

« recréer » ce qui existe en l'état. Le défi qui se pose consiste alors à établir le lien entre des données provenant de sources diverses et de formats fortement hétérogènes afin de garantir l'accès à des informations complètes, disponibles et exploitables. Pour cela, et face à la complexité et le nombre croissant des modèles CAO dans le SI, il est essentiel de mettre en place des techniques avancées d'analyse des modèles CAO et d'exploration des bases de données pour relever le défi de la continuité numérique.

La Rétro-Ingénierie (*RE* pour *Reverse Engineering*) qualifie le processus d'analyse et de compréhension d'un produit ou d'un système existant dans le but de recréer et retrouver des informations et connaissances relatives à sa définition. Traditionnellement menée sur des composants sans supports CAO, les finalités industrielles du *RE* sont le contrôle des résultats de production, la réindustrialisation par la reconstruction de modèles CAO, ou encore l'identification de données relatives à l'objet d'étude dans les SI d'une entreprise (Durupt, 2020). Aujourd'hui, le *RE* s'intéresse particulièrement aux techniques avancées d'Intelligence Artificielle (IA) comme l'apprentissage automatique (ML pour *Machine Learning*) afin d'aider l'humain à mieux raisonner sur ses données et lui offrir un accès unifié et un traitement haute performance de l'information.

En partant du principe que la forme d'un modèle CAO est une caractéristique qui lui est intrinsèque, constante, indépendante de son format, sa méthode de modélisation géométrique ou encore de ses métadonnées, l'analyse de forme (SA pour *Shape Analysis*) est une discipline qui se concentre sur l'extraction et l'interprétation des caractéristiques et des structures des objets tridimensionnels pour en déduire des informations significatives (Laga et al., 2018). Les activités de *SA* visent à simplifier l'accès aux bases de modèles CAO pour les comparer, les contextualiser et créer du lien entre ces données.

Proposition scientifique

Face aux problématiques de continuité numérique des entreprises industrielles, les travaux de recherche présentés dans ce manuscrit s'intéressent à l'exploitation des activités de *SA* dans le cadre de processus de *RE* pour le maintien de l'intégrité des modèles CAO en environnement *PDM*. Deux axes d'études ont structuré les travaux afin d'aboutir à une proposition scientifique et à son évaluation.

Le premier axe concerne les moyens techniques pour l'analyse des modèles CAO. Pour s'adapter aux spécificités géométriques des modèles aéronautiques, nous proposons un nouveau type de descripteurs de forme 3D. Ces descripteurs sont spécialement conçus pour capturer et représenter efficacement les caractéristiques des surfaces aérodynamiques complexes. Sur la base de techniques avancées de géométrie computationnelle, d'analyse statistique et d'apprentissage automatique, ce nouveau descripteur s'intègre aux méthodes courantes de *SA* pour diverses applications : catégoriser des modèles 3D (*classification*) ; les comparer deux à deux par le calcul d'un score de dissimilarité (*matching*) ; rechercher des modèles similaires en base de données (*retrieval*), ou encore reconnaitre certaines composantes locales d'un modèle (*recognition*). En complément des applications « traditionnelles » de SA, nous introduisons une nouvelle activité de recherche et d'identification précise de formes identiques. Cette activité vise à améliorer la compréhension des modèles CAO et à renforcer les capacités de recherche et d'exploration au sein des systèmes d'information industriels. Le *labeling* des formes a pour enjeu de dépasser la notion de « similarité » entre modèles 3D, en proposant un critère de comparaison suffisamment précis pour différencier des formes géométriquement très proches. Le *labeling* est particulièrement pertinent dans le domaine aéronautique, où la volumétrie des modèles 3D et la complexité des surfaces qui les composent présentent un réel défi aux méthodes de SA. De faibles écarts géométriques peuvent en effet entraîner des conséquences significatives sur les performances mécaniques et aérodynamiques des composants. Ainsi, la capacité à distinguer précisément de faibles variations géométriques est essentielle pour garantir la fiabilité des informations issues des activités de SA.

Le second axe de notre étude se concentre sur le déploiement des activités de SA autours des données en environnements *PDM*, afin d'en exploiter le potentiel dans le cadre de processus de *RE.* Le *Knowledge-Based Reverse Engineering Model* (*KBRE model*) est proposé comme un outil technique pour la manipulation de modèles 3D, leur analyse à différentes échelles, et la gestion des informations et connaissances qui leurs sont relatives.

Du prétraitement de divers formats de modèles 3D à l'inférence de résultats d'analyse, en passant par l'apprentissage machine et la capitalisation d'informations sur les bases de données, le KBRE Model représente une plateforme d'expérimentation pour le développement et la validation de techniques avancées de SA, ainsi que leur intégration à divers scénarii de RE.

Cette thèse marque ainsi une contribution dans le domaine de l'analyse de forme 3D et de la Rétro-Ingénierie de modèles CAO dans un contexte industriel.

Méthodologie de recherche et structure du manuscrit

La méthodologie de recherche suivie est représentée par la structure des quatre chapitres de ce manuscrit :

Le Chapitre 1 détaille le positionnement du sujet de recherche. Les contextes scientifique et industriel y sont présentés, ainsi que l'identification des besoins liés aux applications de *RE*. Une fois l'orientation des recherches justifiée, la problématique et les verrous scientifiques sont exposés.

Le Chapitre 2 explore divers concepts clés du *RE* dans un état de l'art mettant en avant l'importance de l'analyse de forme 3D pour la compréhension des modèles CAO et la recherche d'informations. Nous y examinons les solutions existantes ainsi que les manques, en soulignant l'importance croissante des techniques d'apprentissage automatisé.

Le Chapitre 3 concerne la proposition scientifique de nos travaux de recherche. Dans un premier temps, des méthodes et outils sont proposés pour les activités de *SA* de composants aéronautiques. Dans un second temps, le modèle de données proposé pour le déploiement des activités de SA est détaillé.

Le Chapitre 4 comprend une série d'expérimentations menées dans le cadre de la phase de validation des éléments de la proposition. Le plan d'expérience suivi est détaillé avant la critique des résultats d'expérimentation.

La conclusion de ce projet de thèse mettra en lumière les connaissances scientifiques nouvellement acquises dans le cadre de nos travaux de recherche et les apports techniques associés, avant de terminer sur les perspectives relatives à l'analyse de forme 3D et la Rétro-Ingénierie de modèles CAO.

Chapitre 1 : Contexte et problématique de recherche

Ce premier chapitre pose le cadre des travaux de recherche exposés dans ce manuscrit.

Pour commencer, nous clarifions les concepts généraux associés aux activités d'ingénierie dans un contexte industriel. Les principales problématiques associées à la gestion des informations et connaissances relatives aux produits d'une entreprise sont identifiées, et les solutions apportées par le Reverse Engineering sont mises en avant.

Ensuite, nous apportons des précisions sur le contexte industriel particulier dans lequel se sont déroulées nos recherches, en collaboration avec l'unité de recherche en mécanique du laboratoire Roberval de l'Université de Technologie de Compiègne et safran Aircraft Engines. Des exemples d'applications de Reverse Engineering sur les modèles CAO d'un Système d'Information industriel sont présentés afin de justifier le positionnement scientifique et l'orientation de nos recherches.

Enfin, au terme de ce chapitre, la problématique ainsi que la question de recherche associée sont énoncées, puis les objectifs relatifs à la proposition de nos travaux sont définis.

I. Contexte scientifique

L'histoire industrielle est marquée par trois révolutions basées sur le progrès technique et organisationnel qui l'ont fondamentalement transformée, conduisant ainsi à une réduction des efforts physiques en faveur de l'utilisation de machines. La compétitivité d'une entreprise était désormais définie par sa capacité à augmenter les volumes produits tout en en réduisant les cycles et les coûts de développement et de production (Guérineau et al., 2022).

En un peu plus de 20 ans, l'émergence de l'outil informatique a radicalement transformé les techniques d'Ingénierie par la numérisation des supports de développement et de conception traditionnels. Des outils informatiques et Systèmes d'Information (SI) ont été déployés pour gérer des données numériques associées aux activités de développement des produits et systèmes. Les éléments constituant le SI d'une entreprise s'appuient tous sur des bases de données qui constituent « l'espace de stockage » des données numériques de l'entreprise. Une fois contextualisées et structurées, ces données sont considérées comme des connaissances relatives aux produits et systèmes qu'elles définissent.

*Un **système** est une construction ou une collection de différents artefacts techniques qui sont artificiels, concrets, principalement dynamiques et constitués d'éléments ordonnés, qui, interdépendants, produisent des résultats qui ne peuvent être obtenus uniquement par les artefacts. La valeur ajoutée par le système dans son ensemble et son comportement, auquel les parties contribuent indépendamment, est principalement créée par la relation entre les parties ; c'est-à-dire comment ils sont interconnectés. (Vosgien, 2015)*

*Un **système d'information** (SI) est un ensemble organisé de ressources qui permet de collecter, stocker, traiter et distribuer de l'information. (De Courcy, 1992)*

*Un **produit** peut être défini comme le résultat d'un ensemble d'activités corrélées ou interactives qui transforme des éléments d'entrée en éléments de sortie (ISO, 2008)*

Ces dix dernières années ont vu l'émergence et l'intégration de technologies numériques de plus en plus complexes dans le secteur industriel, telles que l'Intelligence Artificielle (IA), le *Big Data*, le *cloud computing*, *l'Internet of things* ou la réalité augmentée (X. Xu et al., 2021). Ces technologies sont à la base de la 4e révolution industrielle, souvent appelée *Industry 4.0*, ou plus récemment *Industry 5.0* qui souhaite repositionner l'humain et les limites planétaires au centre des systèmes. Ces nouvelles révolutions sont étroitement liées aux technologies des systèmes cyber-physiques (i.e. *Cyber Physical Systems*), qui sont des systèmes de collaboration entre des entités du monde physique et les technologies de l'information (Lu, 2017). On qualifie alors de « smart » les systèmes automatiques physiques communiquant avec des technologies informatiques capables de traiter et d'analyser les données en temps réel.

Qu'ils soient cyber-physiques ou non, les produits et les systèmes reposent sur des données numériques, dont la quantité augmente de manière exponentielle (Dekhtiar et al., 2018). Les entreprises doivent aujourd'hui faire face à une augmentation massive des quantités de données numériques associées à leurs produits, soulevant certaines insuffisances des solutions *d'Engineering Data Management* (EDM) en termes de gestion des informations et surtout des connaissances (Lupinetti et al., 2019).

*L'**Engineering Data Management** (EDM) est un processus d'organisation, de stockage et de suivi des informations de conception créées par les activités d'ingénierie et de développement (Feng et al., 2009)*

La subtilité entre données et connaissances réside dans l'interprétation et l'utilisation qu'un être humain peut en faire pour accomplir des tâches complexes. Une connaissance est donc une information riche et hautement structurée (Milton, 2008). Contrairement à la gestion des données, qui vise principalement le stockage de l'information, le *Knowledge Management* (*KM*) concerne l'utilisation de techniques et outils afin de mieux utiliser les actifs intellectuels d'une entreprise. Le *KM* peut donc être défini comme le processus de création, d'organisation, d'identification, de stockage, de diffusion et enfin d'utilisation des connaissances au sein du SI d'une entreprise.

Parmi les éléments du SI, les *Product Data Management* (*PDM*) sont des systèmes de gestion et de stockage des données relatives aux produits, ainsi que toute information liée à l'entièreté de son cycle de vie (Eynard et al., 2006). Le cycle de vie d'un produit représente son évolution dans le temps, depuis l'élaboration du concept jusqu'à sa fin de vie. Les fonctions orientées utilisateur d'un système *PDM* sont la gestion de documents et de coffre-fort pour l'accès et le partage des données d'ingénierie selon les droits de l'utilisateur, la gestion des *workflows*, la gestion de la structure du produit, la catégorisation des données et la gestion de projets. Les fonctions orientées système concernent la transmission, la visualisation et la conversion de données, ainsi que la communication, la notification utilisateur, et l'administration du système (Feng et al., 2009; Stark, 2022). Un système *PDM* est ainsi l'implémentation d'une solution spécifique répondant aux critères de la stratégie globale du *Product Lifecycle Management* (PLM).

*Le **Product Lifecycle Management** (PLM) est une approche stratégique regroupant un ensemble de concepts, de méthodes et d'outils logiciels permettant de créer et d'entretenir les produits industriels tout au long de leur cycle de vie (DS, 2010; Stark, 2022)*

Les données d'ingénierie d'un *PDM* comprennent entre autres la définition du produit, sa configuration, sa nomenclature (*BOM* pour *Bill of Material*), des documents de projet et de gestion des processus, des gammes d'assemblages et de fabrication, des outils d'analyse et de requête, et enfin la catégorisation de pièces et de documents. Ces informations sont structurées par des articles qui représentent les différents niveaux de représentation des produits et des systèmes, et permettent la gestion des multi-vues des produits destinés à fournir uniquement une information pertinente selon les métiers et expertises des utilisateurs.

*Les **données d'ingénierie** correspondent à tous les types de données contenant une expertise utilisateur et possédant une valeur ajoutée pour l'entreprise. Ces données peuvent être considérées comme connaissance et être utilisées par d'autres experts (Durupt, 2020)*

Au cœur des articles du *PDM*, les maquettes numériques (*DMU* pour *Digital Mock Up*) sont généralement considérées comme des représentations centrales contenant les informations et connaissances relatives à la définition des produits (Lupinetti et al., 2019). La *DMU* est un assemblage de modèles CAO positionnés dans un espace 3D, et permet notamment la visualisation du produit assemblé, la conception collaborative et le *design* en contexte. Les modèles CAO jouent un rôle essentiel dans le processus de conception et de fabrication de produits dans de nombreux domaines industriels. Ils permettent de représenter de manière précise et détaillée la géométrie tridimensionnelle des objets. Les solutions logicielles de CAO intègrent avant tout des capacités de modélisation, soit la création et la modification de formes géométriques. Différentes méthodes de modélisation géométrique sont détaillées dans l'Annexe 1.

*La **maquette numérique** (Digital Mock Up - DMU) est une représentation numérique étendue du produit, utilisée comme plateforme de développement produit/processus, de communication et de validation durant toutes les phases de la vie du produit (Eynard et al., 2023) – d'après le consortium du projet Européen AIT-DMU BP: Advanced Information Technology in Design and Manufacturing –Digital Mock-Up Buisiness Process*

*La **conception assistée par ordinateur** (CAO, ou CAD pour computer aided design) consiste en l'utilisation de systèmes informatiques pour assister les activités de création, de modification et d'optimisation des conceptions (sarcar et al., 2008)*

On parle de ***feature-based design***, (i.e. modélisation implicite selon (Borrmann & Berkhahn, 2018)) pour qualifier les méthodes de modélisation basées sur une conception paramétrique à partir d'esquisses 2D et d'opérations de construction. Ces modèles intègrent des paramètres pilotant la géométrie ainsi que des règles et contraintes qui représentent les informations fonctionnelles et les intentions du concepteur. D'autres fonctionnalités sont disponibles comme la conception d'assemblages de différents modèles CAO, la cinématique des modèles et l'application de matériaux par exemple.

Le tableau ci-dessous résume l'information possiblement contenue par un modèle CAO moderne (Durupt, 2020):

Information géométrique	Information fonctionnelle
▪ Définition mathématique des surfaces ▪ Topologie du modèle (i.e. relations de connexité entre les surfaces) ▪ Représentation solide (assemblage de surfaces délimitant un volume fermé) ▪ Représentation simplifiée (maillage 3D)	▪ Métadonnées (nom, référence) ▪ Matériaux ▪ Assemblages (positionnement et contraintes entre les solides) ▪ Arbre de modélisation (esquisses, features, paramètres) ▪ Tolérancement 3D (GD&T–*Geometric Dimensioning and Tolerancing*) ▪ PMI –*Product and manufacturing information* ▪ Annotations (ajouts de textes pour la conception et l'industrialisation)

Tableau 1 : informations géométriques et fonctionnelles des modèles CAO

Un modèle CAO ne contenant que de l'information géométrique est qualifié de « sémantiquement pauvre ». Inversement, si le modèle contient aussi des informations fonctionnelles, il sera alors qualifié de « sémantiquement riche ». Notons que la richesse d'un modèle CAO n'est pas une notion binaire et que son appréciation dépend d'un point de vue métier et des besoins associés.

Ainsi, les modèles CAO sont aujourd'hui au centre de la définition des produits et des systèmes. Ils intègrent l'ensemble des connaissances expertes ayant participé au développement des produits et représentent des clés essentielles pour accéder à leurs multiples représentations, et par extension aux connaissances qui y sont associées.

Ce sont ces données et connaissances qui assurent la durabilité des produits et le succès des entreprises. Leur intégrité doit ainsi être maintenue à tout prix par des processus de création, de partage et de valorisation des connaissances (Nzetchou et al., 2021).

Au long de la vie d'un produit, diverses raisons organisationnelles ou éléments liés aux technologies des *model*eurs CAO et des SI peuvent perturber l'accès aux données et connaissances qu'ils contiennent.

Tout d'abord, les applications de CAO font parties d'un ensemble d'applications appelées *Computer Aided Engineering* (*CAE*), parmi lesquelles les outils d'analyse et de simulation ou encore de *Computer Aided Manufacturing* (*CAM*) manipulent divers formats et contiennent diverses informations. On parle alors de multi-représentation d'un produit, chacune relative à un point de vue métier particulier. Or, les formats natifs de la majorité des logiciels CAO ne sont pas compatibles avec les autres solutions du marché. En effet, les « kernel » de modélisation géométrique spécifiques à chaque éditeur déterminent la manière dont le modèle CAO est représenté mathématiquement, sémantiquement, ainsi que la précision interne des définitions géométriques (Nzetchou et al., 2019). Le passage d'un logiciel à un autre n'est généralement pas possible avec un format natif, du moins si les logiciels en question ne sont pas du même éditeur. De même, du fait des évolutions des outils CAO, il se peut que certains formats 3D ne soient plus éditables sur le long terme.

Pour pallier ces problèmes de « *legacy* », des formats interopérables et neutres ont été développés. Cependant, la plupart de ces formats neutres n'intègrent que les informations géométriques, et la majorité des informations fonctionnelles (cf. Tableau 1) sont perdues lors de la traduction. Malgré cette perte sémantique, de nombreuses applications d'ingénierie utilisent ces formats neutres. Par exemple, pour pérenniser la capacité de lecture des modèles sur le long terme, l'archivage est parfois réalisé dans un format neutre. De même, pour contrôler et limiter l'information transmise lors d'échanges entre partenaires ou pour de simples besoins de visualisations de larges assemblages, les CAO au format natif peuvent être convertis en représentations 3D simplifiées sémantiquement pauvres. Enfin, certaines applications de simulation et de calcul transforment les modèles natifs en une représentation à base d'éléments géométriques discrets (cf. Annexe 1). Les informations créées par ces activités de *CAE* sont donc propres à une vue spécifique du produit et non forcément reliées aux autres vues.

D'autres limites d'accès aux informations des produits sont directement liées aux liens relationnels entre les données dans le SI. Les liens de dépendances d'une *DMU* concernent les modèles 3D qui y sont assemblés, les liens de dépendances d'une CAO concernent des éléments géométriques (interfaces, solides, références géométriques) issus d'autres modèles CAO, et les liens de dépendances des articles concernent leur structure et les données qui y sont rattachées. Ainsi, isoler une *DMU* ou une CAO de son *PDM*, ou encore changer leur format de données et métadonnées, peut engendrer une perte de ces liens ayant pour conséquence une perte d'informations fonctionnelles. Par ailleurs, différents éléments du SI sont parfois utilisés pour chaque point de vue. Par exemple, lorsqu'un *PDM* est utilisé pour les activités de modélisation, un autre élément, appelé *SDM* pour *Simulation Data Management*, pour les activités de calcul et un troisième pour les métiers d'industrialisation, on parle alors de « silos » d'informations. Les données sont dispersées dans différents systèmes de gestion, ce qui rend complexe l'accès à l'ensemble des informations contextualisées d'un produit.

Les exemples précédents sont représentatifs d'un manque de continuité numérique des données, et plus particulièrement des modèles CAO dans les processus d'une entreprise.

*La **continuité numérique** est la capacité à disposer de l'ensemble des informations numériques d'un produit, système ou infrastructure de manière pérenne, pendant toute la durée du cycle de vie. C'est-à-dire que les informations sont complètes, disponibles et donc utilisables. (MacLean & Davis, 1998)*

Il est possible de vulgariser la continuité numérique comme le fait d'avoir toujours les bonnes informations, au bon moment, pour prendre de meilleures décisions. La solution idéale serait alors un unique SI commun à toutes les activités d'ingénierie, offrant une connaissance exhaustive des produits sur toutes les phases de leurs cycles de vie via un accès rapide et intelligent aux données et informations contextualisées. Dans les faits, le défi de la continuité numérique consiste à relier des données provenant de sources diverses et de formats fortement hétérogènes.

Plusieurs propositions visent à améliorer les capacités de continuité numérique des modèles 3D d'une entreprise. On trouve, entre autres, le développement d'une stratégie, d'un format de données, ainsi qu'un projet collaboratif.

Le *Model-based definition* (*MBD*) est une stratégie liée à celle du *PLM*, basée sur la transition des modèles CAO de simples collecteurs de données géométriques vers des sources complètes d'informations pour le cycle de vie global du produit. Avec le *MBD*, la plupart des données relatives à un produit sont structurées dans des modèles CAO natifs, au lieu d'être dispersées sous différentes formes dans les bases de données. Les objectifs de *MBD* sont la suppression des documents et des dessins redondants, une meilleure cohérence des données, une meilleure virtualisation des produits/processus et un meilleur support pour toutes les tâches des technologies assistées par ordinateur dans les disciplines d'ingénierie et de fabrication (Alemanni et al., 2011).

Cependant, si le *MBD* permet de limiter le nombre de données et de regrouper l'information, l'interopérabilité des formats n'est pas solutionnée. Pour que l'interopérabilité soit possible dans une stratégie *PLM*, les formats de données neutres doivent contenir l'ensemble des informations d'un modèle natif après traduction. Sous la norme ISO 10303, une variété de modèles au format neutre *STEP* a été développée dans le but de faciliter les échanges entre solutions de *CAE* (Pratt & others, 2001). Organisés en *Application Protocols* (*AP*), ces derniers servent de schéma pour l'échange de données de différentes catégories couvrant une variété d'usages. Le récent développement de *STEP AP 242 « Model-Based 3D Engineering »* définit un modèle standardisé et interopérable contenant de riches informations fonctionnelles (Venkiteswaran et al., 2016). Selon les travaux de (Nzetchou et al., 2019, 2021), le format *STEP AP 242* semble être aujourd'hui le format neutre le plus riche s'il intègre toutes les fonctionnalités couvertes par les *AP* les plus couramment implémentés et utilisés. On note cependant que l'utilisation du format *STEP AP 242* dépend principalement de son intégration complète et effective par les éditeurs de solutions CAE.

Le projet *LOTAR -Long Term Archiving and Retrival-* est un projet international ayant pour objectif principal de développer, tester, publier et maintenir une série de standards pour l'archivage et la récupération des données de produit (données CAO et données *PDM*). Basé sur la norme ISO 14721 - *Open Archival information System* (Bahloul et al., 2008), le projet est globalement porté par les industries de l'aérospatiale et de la défense. Le projet n'a pas pour finalité de développer de solutions applicatives ou matérielles à l'exception de prototypes et de preuves de concepts. *LOTAR* a notamment pour but d'aider à la définition et l'intégration par les éditeurs des *APs* de la norme ISO 10303 et de permettre l'accréditation du format.

Ainsi, plusieurs initiatives visent à améliorer la continuité numérique des systèmes de développement et de production des entreprises en facilitant le maintien de l'intégrité des modèles et maquettes numériques. Ces éléments visent à répondre à la question de **« comment mieux faire ? »**. Outre l'amélioration des processus et leur intégration dans un environnement numérique respectant la logique de continuité numérique, il est aussi nécessaire de s'intéresser aux données en l'état, et de fournir aux activités d'ingénierie des solutions pour « réparer » l'existant.

La Rétro-Ingénierie (*RE* pour *Reverse Engineering*) en génie mécanique et génie industriel tente de répondre à la question de **« comment refaire ? »** et contribue ainsi à identifier des approches pour solutionner le défi de la continuité numérique.

Le *RE* joue un rôle essentiel dans la compréhension et la reconstruction de produits, de technologies et de concepts existants.

*La **Rétro-Ingénierie** (RE) dans le domaine du génie industriel, également dénommée « reverse engineering », est une activité qui consiste, à partir d'un composant réel ou de son support numérique 3D de bas niveau sémantique, à créer un modèle numérique sémantiquement riche (Várady et al., 2007)*

Le *RE* est traditionnellement mené sur des composants qui n'ont pas de supports CAO ou ayant un support 3D sémantiquement pauvre avec pour finalité la « reconstruction » d'un modèle 3D sémantiquement riche. Ce nouveau modèle ainsi créé offre des possibilités d'études, d'analyses, de simulations et de modifications pour les ingénieurs et experts pouvant intervenir durant les différentes phases du cycle de vie d'un produit manufacturé (Terzi et al., 2010). En d'autres termes, le *RE* est à l'origine une activité qui consiste, à partir d'un modèle existant (pièce réelle ou modèle numérique pauvre), à générer un modèle numérique généralement 3D pouvant être un modèle surfacique ou un modèle volumique de type CAO avec une arborescence.

Selon (Durupt, 2020), les principales finalités industrielles du *RE* sont les suivantes : (i) réaliser un contrôle qualité *in situ* d'un composant fabriqué, via le contrôle dimensionnel ou d'aspect du résultat de production ; (ii) refabriquer un composant en reconstruisant un modèle 3D interprétable par les activités de *CAE* ; et (iii) retrouver un modèle numérique ou une maquette numérique contenant la définition d'un produit ou système.

Cette troisième approche promeut la réutilisation de l'existant plutôt que la création « *from scratch »*. Elle consiste à exploiter l'information stockée dans les PDM ou les archives d'une entreprise en vue de retrouver ou recréer des modèles CAO ou maquettes numériques qui, au terme du processus de RE, contiennent non seulement des informations géométriques mais aussi fonctionnelles, et non uniquement la géométrie résultante d'un modèle.

En effet, selon le niveau de richesse sémantiques des données disponibles et exploitables, il est possible de réutiliser des esquisses à la base d'opérations de modélisation, des paramètres pilotant la géométrie, des contraintes géométriques et règles métiers, ou même des informations supplémentaires comme le tolérancement fonctionnel ou l'application de matériaux. Ces informations, retrouvées dans le SI, ne peuvent généralement pas être déduites par la seule analyse d'un modèle géométriquement pauvre sans l'apport d'une source supplémentaire d'informations.

C'est pourquoi, en plus d'outils de modélisation, les processus de *RE* intègrent des méthodes de recherche d'informations en bases pour rendre possible l'accès et l'utilisation de connaissances expertes d'ingénierie dans la réalisation de tâches complexes sur les modèles CAO.

Ainsi, pour ne pas restreindre le *RE* à la simple reconstruction de géométries CAO à partir du scan d'un objet physique, vision encore majoritaire dans la littérature scientifique, nous avançons la définition suivante :

*La **Rétro-Ingénierie** est un processus d'analyse et de compréhension d'un produit ou d'un système existant dans le but de recréer et retrouver des informations et connaissances relatives à* sa *définition.*

Deux domaines au départ distincts se retrouvent ainsi liés aux processus de *RE* : l'analyse de formes, et le *Knowledge-Based Engineering*.

L'analyse de forme (*SA* pour *Shape Analysis*) est une discipline qui se concentre sur l'extraction et l'interprétation des caractéristiques et des structures de la forme des modèles 3D pour en déduire des informations significatives (Laga et al., 2018). Les activités de *SA* permettent de comparer des modèles 3D et de les relier selon des critères purement géométriques ou encore par des associations sémantiques déduites de la forme et des fonctions des modèles 3D. Parmi les principales activités de *SA*, on retrouve tout d'abord le *matching*, qui consiste à comparer des formes 3D en quantifiant leur dissimilarité (Osada et al., 2002). Le *retrieval* est généralement perçu comme une extension du *matching*. Cette activité consiste à retrouver dans un ensemble les formes 3D les plus similaires à une forme analysée, selon des critères géométriques ou sémantiques (Gezawa et al., 2020; Tangelder & Veltkamp, 2008). De son côté, la *classification* est une activité de catégorisation de formes 3D, c'est-à-dire associer une forme à une famille prédéfinie (Bello et al., 2020; Xiao et al., 2020). Enfin, l'activité de *recognition* permet d'identifier une forme 3D comme une sous partie, ou composante, d'une forme 3D plus large (Laga et al., 2018).

Face aux problèmes de discontinuité numérique des données CAO dans les SI industriels, l'analyse de forme propose des solutions techniques afin de permettre à un utilisateur d'accéder rapidement et avec agilité à un grand volume de données dispersées et hétérogènes, de croiser, réconcilier et contextualiser les données, et enfin d'analyser les données pour en extraire de la valeur (Lupinetti et al., 2019).

Dans cette logique, certains éditeurs de solutions CAO/*PDM* ont développé des fonctionnalités avancées d'analyse de données 3D, dont voici quelques exemples :

- La suite logicielle *Exalead OnePart*[1] par Dassault Systèmes est une solution de recherche et d'exploration des données d'ingénierie. Elle intègre des fonctionnalités de *classification* automatisée des composants, d'identification des doublons, de recherche de modèles 3D et fichier existants (par requête textuelle et par similarité de forme), d'exploration des structures XAO et de comparaison géométrique des modèles CAO.
- *3D Part Finder*[2] par 3DSemantix permet la recherche les modèles CAO par la forme et la taille ce qui complète la recherche par mots-clés.
- *GeoSearch* par Cadenas[3] permet de rechercher un composant CAO dans une base de données et catalogues fabricants

Ces solutions se basent sur des technologies avancées d'analyse des métadonnées mais également des formes géométriques des modèles 3D, dans le but de simplifier un accès intelligent aux données et de promouvoir la réutilisation du capital existant dans les SI industriels.

Les systèmes de *Knowledge-Based Engineering* ont pour objectif de permettre une meilleure intégration de la connaissance aux activités d'ingénierie. Le *KBE* appartient à la famille des *Knowledge Technologies* (*KT*) dont l'objectif est de fournir une utilisation plus riche et intelligente de l'information. Pour cela, les *KT* ont pour objectif l'identification des connaissances importantes implémentées par les experts, la définition des méthodes de capture de l'information dans le cadre d'un problème donné, ainsi que les manières de représenter et stocker la connaissance afin de la capitaliser, y accéder, et surtout la réutiliser (Milton, 2008; Rocca, 2012). Dans sa définition, le *KBE* n'est pas une alternative aux outils de CAO, mais propose une

[1] https://www.3ds.com/fileadmin/PRODUCTS-SERVICES/EXALEAD/Resources-center/PDF/onepart-reuse-exalead-solution-brief-en.pdf (visité le 15/10/2023)
[2] https://www.3dpartfinder.com/ (visité le 15/10/2023)
[3] https://www.cadenas.de/en (visité le 15/10/2023)

intégration dans un même système des différentes briques telles que des technologies de CAO, de Programmation Orientée Objet (POO), et d'IA. Sa spécificité réside dans l'utilisation de *Knowledge Base* (*KB*), une forme de *DataSet* qui capitalise des connaissances, donc une forme d'information riche et structurée sur la base de concepts, d'attributs définis, de valeurs et de relations (Milton, 2008).

À l'origine, les *KBE* sont apparus en réponse à l'insuffisance des outils de CAO focalisés sur la modélisation géométrique (Chapman & Pinfold, 1999). Dès les années 1990-2000, divers outils de CAO ont commencé à intégrer des applications de *KBE* pour l'automatisation de tâches complexes d'ingénierie via un raisonnement automatique sur la base de systèmes de règles expertes et d'inférences sur les données (Chandrasegaran et al., 2013; Rocca, 2012). Le paradigme de la POO permet la structure et la définition de relations entre différents types de données. Dans les faits, ces solutions s'apparentent principalement à des systèmes experts (aussi appelés *Knowledge Based System*) qui fonctionnent sur la base de règles prédéfinies, et l'intégration de techniques de raisonnements avancés par IA n'y est pas toujours une évidence.

De même, les approches du *KBE* ont rapidement été appliquées aux problèmes de *RE*. Des approches comme (Fisher, 2004; Vilmart et al., 2018) stockent la connaissance sous forme de règles métiers et de contraintes géométriques afin de faciliter la reconstruction de modèles CAO paramétriques. D'autres approches comme (Bruneau et al., 2014; Durupt et al., 2019) s'intéressent à la capitalisation d'informations riches dans un formalisme défini permettant son identification au sein de larges bases de données et sa réutilisation dans le cadre de processus de *RE*. On appelle cela la signature des données, qui se base sur l'intégration de méthodes de *SA* au sein des systèmes de *KBE*.

Aujourd'hui, les avancées des techniques de *Machine Learning (ML)* donnent un nouvel élan aux systèmes *KBE* avec le développement de nouvelles méthodes de raisonnement et d'interprétation des données, notamment dans le cadre d'activités d'analyse de forme. Les capacités de traitement géométrique de la CAO, de structuration de l'information de la POO, et de traitement avancé de la connaissance du *ML* semble faire des systèmes de *KBE* une solution intéressante pour les processus de *RE* sur les données CAO d'un SI industriel.

Suite à l'analyse du contexte scientifique de nos recherches, la seconde partie de ce chapitre s'intéresse à l'analyse de son contexte industriel. Nous tenterons notamment d'identifier des cas d'applications concrets auxquels les processus de *RE* peuvent répondre. Par ailleurs, l'objectif de cette seconde partie est d'identifier les spécificités du contexte industriel afin d'affiner la définition d'une question de recherche et des problématiques associées, qui seront présentées au terme de ce chapitre.

II. Contexte Industriel

Les recherches présentées dans ce manuscrit ont été menées dans le cadre d'une thèse CIFRE au sein du département Conception collaborative et Management de la configuration de la direction technique de la société SAfran Aircraft Engines[4] (SAE). Ce département centralise les processus, méthodes et outils liés à la modélisation 3D, à la maquette numérique et à la gestion de configuration.

SAE, anciennement Snecma, est une entreprise française spécialisée dans la conception, le développement, la production, la vente et le maintien en condition opérationnelle de moteurs pour l'industrie aéronautique et spatiale. Partenaire de General Electric dans le cadre du programme CFM International[5], SAE est responsable des modules de soufflante (*i.e. fan*), compresseur basse pression (*i.e. low-pressure compressor*) et turbine basse pression *(i.e. low-pressure turbine*) des moteurs civils de type Turbofan CFM 56 et de leur nouvelle génération LEAP, schématisés dans la Figure 1[6]. Les *turbofan*, ou turboréacteur à double flux, se distinguent des Turboréacteurs par la présence d'un flux d'air secondaire dit « froid » issu de la poussée générée par la soufflante. Par rapport aux turboréacteurs à simple flux, les Turbofan présentent divers avantages comme une réduction de la consommation et du bruit à poussée équivalente. Les moteurs du programme CFM 56 équipent un large éventail d'avions civils Airbus et Boeing. SAE est également un acteur principal du domaine de l'aviation de combat avec le M88 et le M53 qui équipent respectivement les Dassault Rafale et les Mirage 2000.

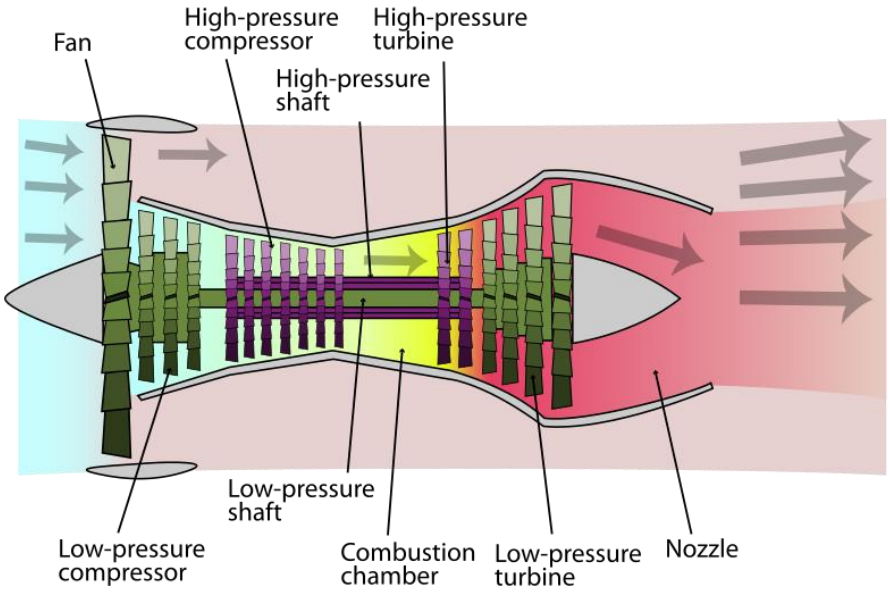


Figure 1 : schéma d'un moteur Turbofan et de ses différents modules

[4] https://www.safran-group.com/companies/safran-aircraft-engines
[5] https://www.safran-group.com/fr/societes/cfm-international
[6] https://commons.wikimedia.org/wiki/File:Turbofan_operation.svg)

Comme d'autres, l'industrie Aéronautique est confrontée aux problématiques de la continuité numérique des données de définitions de ses produits et systèmes.

La première raison de ces problématiques est évidemment la complexité des systèmes concernés. Le *turbofan* est intégré avec la nacelle, les suspensions et d'autres équipements pour former un système complexe appelé un *Integrated Power Plant* (*IPPS*).

> *Les **systèmes** sont délimités par des frontières et connectés à leur environnement ou à d'autres systèmes externes par des entrées et des sorties. Les changements apportés à des parties d'un système ou les modifications de leurs caractéristiques et/ou paramètres au cours d'une période caractérisent ce que l'on appelle la « dynamique du système ».*
> *On parle de système « **compliqué** » pour les systèmes avec un grand nombre de composants et de paramètres connectés.*
> *Un système, compliqué ou non, devient « **complexe** » lorsque les paramètres et les interactions de son système ou de ses parties sont soumis à une forte dynamique de changement.*
> *Définition extraite du manuscrit de thèse de (Vosgien, 2015)*

Un *IPPS* est caractérisé par un très grand nombre de sous-systèmes et de composants en interaction (Vosgien, 2015). Les évolutions de chaque élément des systèmes doivent être tracées et reportées sur les composants avec lesquels ils interagissent via la gestion des versions et évolutions des modèles CAO et données diverses qui contiennent la définition des produits au sein des articles des systèmes d'*EDM*.

On observe cette complexité à l'échelle même du découpage fonctionnel du composant et de son modèle CAO et de leur structuration dans le *PDM* illustrée par la Figure 2. Étant donnée la complexité des pièces, leurs différentes composantes (*i.e.* parties) sont modélisées séparément avant d'être assemblées en un modèle CAO « final ». Dans le *PDM*, un article « Composante » contient, entre autres, un modèle CAO qui contient lui-même des éléments géométriques, fonctionnels et une arborescence de construction. Un article « Assemblage » structure les articles « Composantes » ainsi que le modèle CAO « final » qui sera assemblé en *DMU*. Le modèle final est donc composé de "corps" solides avec pour toute information fonctionnelle un lien vers le modèle source. En théorie, la modification d'une composante se répercute sur le modèle final. Seulement, isoler ce dernier, c'est-à-dire exporter son modèle CAO hors de son article, implique la perte du lien vers ses composantes, et donc la perte d'un grand nombre d'informations géométriques et fonctionnelles ayant menée à sa définition. Ajoutons à cela la présence d'éléments géométriques de contexte, c'est-à-dire des références externes comme des interfaces issues d'autres modèles non présents sous l'article géométrie qui ne sont pas modifiables.

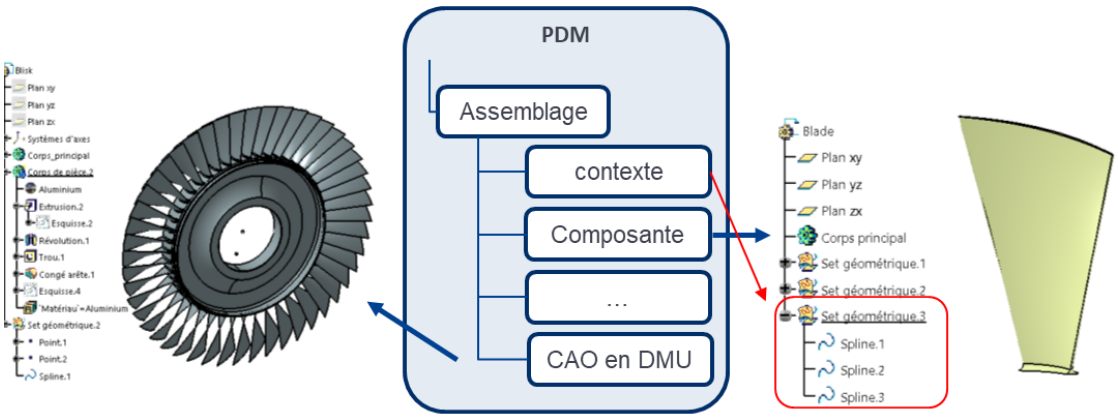


Figure 2: structure des articles dans le PDM

Les projets aéronautiques ont évolué grâce à des partenariats à grande échelle. Chaque partenaire produit une grande quantité de données pour lesquelles le besoin de contrôle des informations qu'elles contiennent implique souvent l'utilisation de modèles CAO pauvres par exemple. De même, la nature multidisciplinaire des projets de systèmes complexes comme les programmes aéronautiques se traduit par la collaboration des différentes disciplines d'ingénierie impliquées, comme illustré par la Figure 3 ci-dessous.

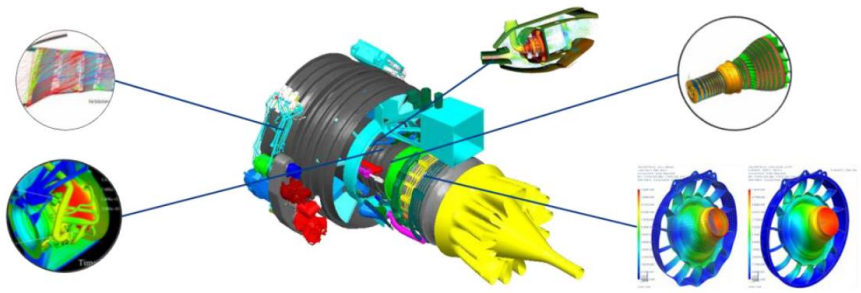


Figure 3: nature multidisciplinaire des Turbofan (Vosgien, 2015)

Les multiples représentations relatives aux domaines d'applications génèrent une grande diversité de données de conception, potentiellement gérée dans différentes briques du SI de l'entreprise. Concrètement, on parle de différents systèmes d'*EDM* contenant différentes représentations ou sous-composantes du même produit, comme les versions et évolutions des CAO, les modèles de simulation, les études aérodynamiques, acoustiques et thermiques par exemple. La traçabilité des évolutions et la capacité de relier toutes ces données hétérogènes au produit qu'elles concernent est complexe et souvent basée sur les liens relationnels des SI ou la métadonnée des articles. Par exemple, les aubages des modèles CAO sont formés de surfaces rigides ayant une fonction aérodynamique de guidage d'un fluide en écoulement. Un outil spécialisé dans le calcul aérodynamique est employé pour générer les éléments géométriques de définition des surfaces d'aubages. Ces dernières sont alors stockées dans un *PDM* spécifique aux activités des aérodynamiciens avant d'être intégrées au modèle de conception paramétrique sous la forme de surfaces implicites. L'utilisation d'un format de modélisation neutre pour l'intégration des surfaces d'aubages aux modèles natifs implique l'absence de leurs éléments de définition et une nécessité de maintenir le lien entre les systèmes *PDM*.

Enfin, une dernière cause de rupture de la continuité numérique des données de définition des modèles aéronautiques est liée à la durée de vie des produits dont le cycle de développement se compte à lui seul en dizaines d'années. Pendant ce temps, l'évolution des solutions *CAE* et des SI peut impliquer une obsolescence des formats de données et la rupture de leurs liens relationnels. Pour limiter l'obsolescence des modèles CAO et assurer l'archivage long terme, l'utilisation d'un format de donnée neutre permet le stockage des modèles et la capacité d'y accéder encore des années plus tard. Seulement, toutes données CAO antérieures à la norme *STEP AP242* sont nécessairement concernées par des manques sémantiques, ainsi que certaines données postérieures selon la mise en place des processus intégrant la norme et son intégration complète par les éditeurs.

Suite à l'identification de potentielles causes de rupture de continuité numérique dans un contexte de gestion des informations en environnement industriel, cinq scénarii d'applications (i.e. *usecases*) sont proposés pour l'exploitation de processus de *RE* en vue de maintenir l'intégrité des modèles CAO.

Usecase 1 : reconstruction de maillages

L'étape de vérification et de validation de la conformité d'un composant manufacturé passe par différentes analyses métrologiques exigeantes de ses dimensions. Dans certains cas, des écarts sont constatés et une analyse avancée de la pièce vise à déterminer si une autorisation d'exploitation peut tout de même être délivrée au composant. On appelle cela le traitement de dérogations. À partir du scan de l'objet réel, la reconstruction partielle ou globale d'un modèle 3D solide représentatif de la géométrie de la pièce physique permet alors d'observer les déviations locales avec le modèle de définition et mesurer les volumes et masses des différentes parties de la pièce. Seulement, le modèle CAO de définition identifié par ses métadonnées a parfois évolué entre temps et ne correspond plus parfaitement à la géométrie du composant manufacturé. L'analyse du modèle maillé peut permettre l'identification dans le *PDM* d'un modèle natif présentant une forme similaire comme illustré dans la Figure 4. Ce dernier permet de simplifier la reconstruction du maillage et de comparer les modèles pour la validation d'un composant manufacturé. De manière plus générale, l'analyse de formes permet de contextualiser des données isolées et de créer du lien entre des modèles 3D de sources et formats différents.

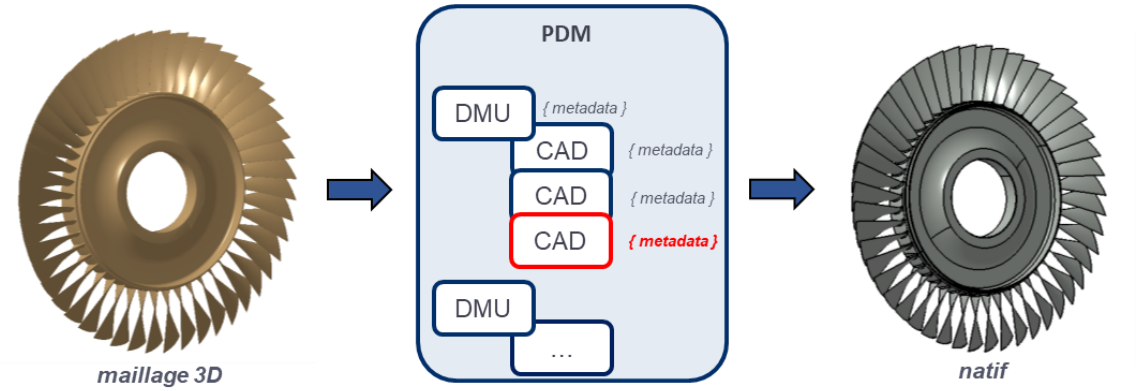


Figure 4: Usecase 1 - recherche dans le PDM d'un modèle CAO natif correspondant à la forme d'un maillage 3D

Usecase 2 : réutilisation de modèles CAO et de PMI

Qu'il soit issu d'un partenaire ou qu'il s'agisse d'un modèle archivé, un modèle CAO pauvre ne présente que peu d'intérêt pour les activités de CAE. Cependant, il est probable que le *PDM* de l'entreprise contienne des informations et connaissances à son sujet. La question est de savoir comment les identifier et les extraire. En identifiant des modèles similaires à l'objet d'étude dans le *PDM*, il est possible de récupérer et réutiliser des informations fonctionnelles telles que l'arborescence de construction, des cotations fonctionnelles, des paramètres et contraintes géométriques ou encore des spécifications de matériaux. Par ailleurs, la documentation associée aux modèles identifiés comme une nomenclature ou une gamme d'assemblage est aussi une source d'information précieuse qui peut être exploitée pour l'augmentation sémantique du modèle analysé. La Figure 5 illustre l'identification d'un modèle CAO riche et de documentations pertinentes dans le cadre d'une activité de *RE* sur un modèle sémantiquement pauvre grâce à la proximité géométrique entre les modèles.

En général, l'accès et la réutilisation de modèles CAO riches offre aux ingénieurs l'opportunité de capitaliser sur les connaissances implémentées dans d'anciens modèles de conception en examinant toutes les solutions préexistantes pour la modélisation d'une fonction donnée. L'accès au capital de l'entreprise permet de s'en inspirer lors du développement de nouveaux produits en plus de maintenir la définition d'anciens modèles malgré l'obsolescence de leur format de modélisation.

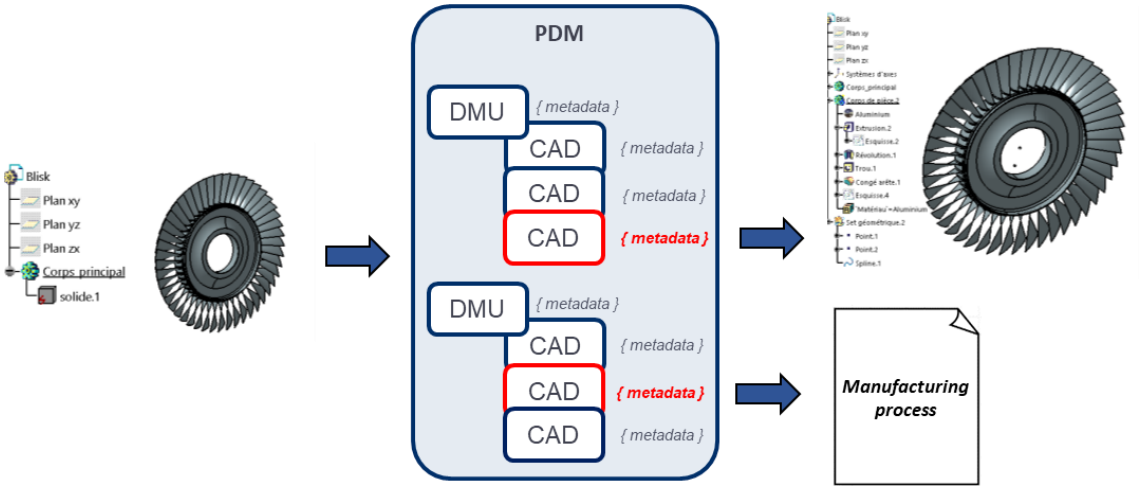


Figure 5 : Usecase 2 - recherche de modèles CAO riches et des PMI associées dans le PDM

Usecase 3 : identification de liens de parentés dans le PDM

Lorsqu'un modèle CAO est composé d'un solide issu d'un second modèle, sa capacité de modification et l'accès aux éléments géométriques de définition sont entièrement basés sur le lien de parenté qui existe entre eux. Un changement de format ou une cassure de ce lien implique la perte globale ou partielle des capacités de modification et une dissémination des informations fonctionnelles. Le lien de parenté entre ces données peut être rétabli grâce à l'analyse des éléments géométriques qui composent ces modèles et la reconnaissance de structures topologiques locales communes. Par exemple, la Figure 6 illustre le lien créé entre une composante multi-instanciée d'un modèle et son modèle d'origine.

De même, la reconnaissance d'une pièce dans un assemblage est un atout précieux pour la gestion des évolutions et la gestion de configuration des *DMU* et des modèles CAO qui les composent.

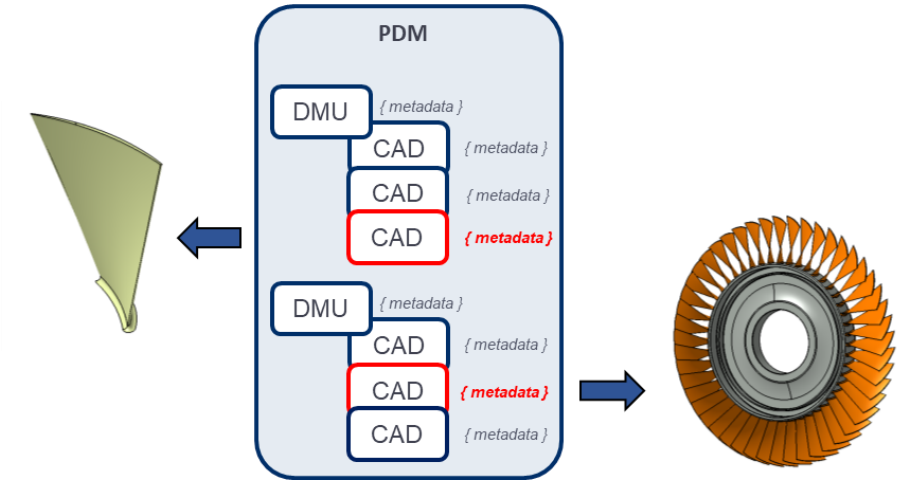


Figure 6 : Usecase 3 - recherche des liens de parentés entre les modèles CAO du PDM

Usecase 4 : Recherche transverse entre systèmes PDM

Dans certains cas, les informations fonctionnelles d'un modèle CAO sont disséminées dans différents systèmes *PDM* de l'entreprise. C'est le cas illustré par la Figure 7 où la définition géométrique d'une surface et le modèle CAO qui l'intègre sous forme de surface résultante à la modélisation d'un aubage sont contenus dans des systèmes *PDM* différents et déconnectés.

Ce problème d'environnements « silo » est à la base d'un grand nombre de problèmes de discontinuités numériques entre les multiples représentations des produits et systèmes. Les analyses transverses entre les éléments du SI permettent de croiser et de réconcilier des données stockées dans différentes bases de données et de dépasser le problème de « cloisonnement » de l'information dans des environnements disparates et déconnectés.

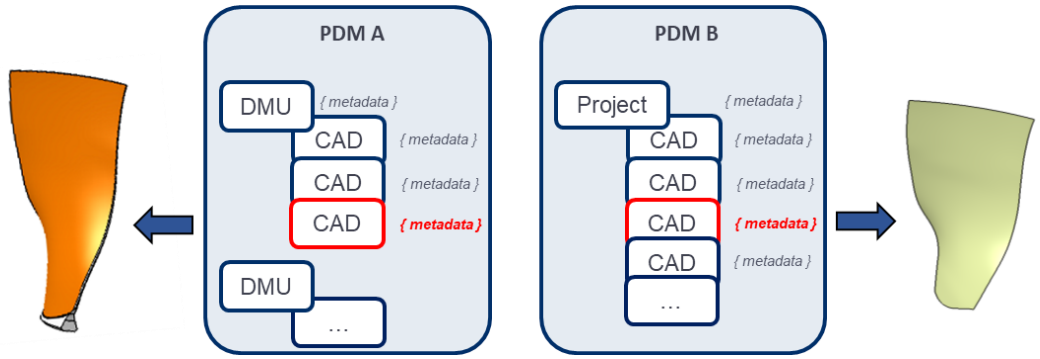


Figure 7 : Usecase 4 - recherche transverse de relations entre les modèles CAO de plusieurs PDM

Usecase 5 : identification des doublons et standardisation

Identifier des doublons en base permet de réduire considérablement les efforts de développement et d'industrialisation inutiles. Par exemple, un programme d'usinage peut être généré plusieurs fois au lieu d'être réutilisé si des modèles identiques n'ont pas été identifiés. C'est le cas de l'exemple illustré par la Figure 8 où deux surfaces identiques sont présentes dans le *PDM*. Si ces surfaces sont par la suite utilisées dans divers projets, certaines activités comme le tolérancement dimensionnel ou la simulation des performances aérodynamiques peuvent être réalisées plusieurs fois au lieu d'une.

De manière générale, on parle de standardisation pour qualifier le processus de rationalisation des bases de données par la suppression de données inutiles ou dédoublées. Ces pratiques permettent un gain de temps considérable, notamment dans le cas d'appareils complexes comportant de nombreuses pièces pouvant nécessiter un processus de conception ou de production complexe. La standardisation a par ailleurs un intérêt majeur concernant des composants standards pour la réduction du nombre de références de l'entreprise.

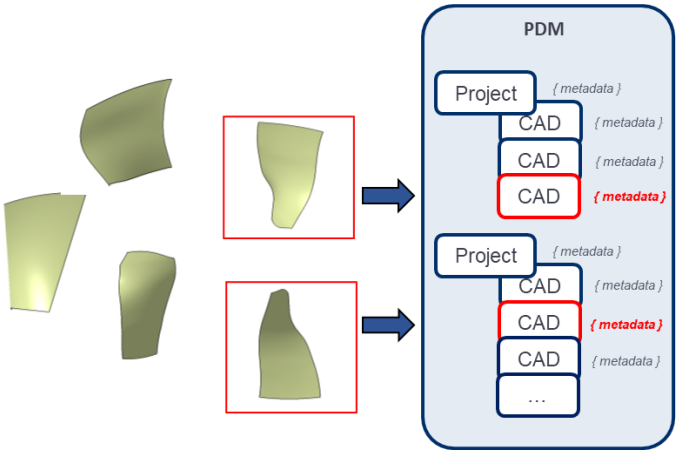


Figure 8 : Usecase 5 - recherche dans le SI de doublons entre les modèles CAO

Pour résumer, ces 5 scénarii d'applications mettent en avant un ensemble de défis significatifs pour la mise en œuvre de méthodes de RE à l'échelle industrielle, particulièrement sur le plan technique de l'analyse de formes, mais aussi en ce qui concerne la gestion des données techniques. Tout d'abord, nous constatons que l'hétérogénéité des formats 3D implique de pouvoir réconcilier des modèles 3D de divers formats, incluant des modélisations discrètes et continues. L'analyse d'un modèle 3D doit donc être indépendante des méthodes de modélisation et formats de fichiers. De plus, seule la forme résultante d'une géométrie peut être exploitée, puisque les modèles pauvres concernés par les applications de RE ne contiennent généralement pas ou peu d'informations fonctionnelles, et les métadonnées ne sont pas une option. Les applications de recherches transverses au sein du SI, et donc l'absence de liens relationnels entre les bases de données, confirment le besoin d'une analyse purement géométrique.

Puisque les relations entre modèles peuvent se trouver à différentes échelles, celle du modèle complet, de ses composantes ou de ses surfaces unitaires, le contexte aéronautique de notre étude implique un défis technique supplémentaire pour l'analyse de formes. En effet, les surfaces d'aubages étudiées peuvent présenter des formes résultantes très similaires, qui ne doivent cependant pas être confondues afin de garantir la pertinence des données et informations retrouvées. Un niveau de précision particulièrement élevé dans la comparaison des modèles est donc requis afin d'être en mesure de différentier des formes 3D malgré une forte similarité géométrique.

L'étude du contexte scientifique a permis de mettre en évidence des enjeux de continuité numérique rencontrés par les industriels et leurs impacts sur les modèles CAO. De plus, l'analyse du contexte industriel présente divers besoins auxquelles le *RE* semble, en théorie, pouvoir répondre. Avec l'augmentation de leur nombre et la complexité croissante des modèles CAO, il devient essentiel de développer des techniques avancées d'exploration et d'indexation pour offrir un accès unifié et un traitement haute performance de l'information. Ces techniques ne peuvent se baser uniquement sur la métadonnée et la structure des objets dans le SI, mais profitent de l'évolution des techniques d'analyse de forme afin de relier des modèles 3D entre eux et d'en extraire des informations utiles et pertinentes en se basant uniquement sur un attribut intrinsèque, leur géométrie.

Ainsi, face au défi du maintien de l'intégrité des modèles CAO industriels dans un contexte de discontinuité numérique des SI, nos recherches se tournent vers les processus de *RE* et les opportunités qu'apportent les méthodes de *SA* et les systèmes de *KBE*. La troisième et dernière partie de ce chapitre énonce la question de recherche ainsi que les différentes problématiques auxquelles nous tenterons de répondre via l'étude de la littérature scientifique et la définition d'une proposition scientifique.

III. Problématique et objectifs de recherche

Face aux problèmes de continuité numérique des SI affectant les modèles CAO industriels, ces travaux de recherche se concentrent sur le *Reverse Engineering* en tant que processus visant à récupérer et recréer des informations relatives à la définition des produits. Après analyse du contexte scientifique et industriel, nous proposons la problématique suivante pour positionner nos travaux :

La notion de dissimilarité entre formes 3D, tel qu'exploitée par les principales activités d'analyse de formes, ne permet pas de distinguer avec précision des modèles 3D proches mais non identiques.
L'absence d'une activité d'identification précise de formes identiques conduit à une incomplétude de l'ensemble des activités d'analyse de forme pour répondre aux besoins en termes d'extraction et de recherche d'information pour la Rétro-Ingénierie de modèles CAO.

Afin de répondre à cette problématique, deux postulats sont avancés. Le premier suppose que les informations manquantes se trouvent quelque part dans le SI de l'entreprise. Le second repose sur le principe que la forme seule d'un modèle 3D constitue, en tant que caractéristique intrinsèque, une base de réflexion suffisante pour son analyse. La question de recherche qui en découle est donc la suivante :

Face aux problématiques de discontinuité numérique des données, comment exploiter les activités d'analyse de forme dans le cadre des processus de Reverse Engineering afin de préserver l'intégrité des modèles 3D au sein d'un SI industriel complexe ?

Cette thèse poursuit ainsi deux macro-objectifs interdépendants : d'abord, identifier et extraire l'information pertinente, puis exploiter cette information en fonction des besoins. Pour répondre au premier, les activités de *SA* sont exploitées et les objectifs suivants définis :

Macro-objectif 1 : identifier et extraire l'information pertinente
• *Proposer une représentation de forme 3D permettant la comparaison de modèles CAO malgré des formats de représentations géométriques différents. Cette représentation doit être compatible avec les formes complexes des modèles aéronautiques et des surfaces aérodynamiques*
• *Raisonner sur les modèles CAO et les organiser sans dépendre de leurs métadonnées ou liens de dépendances dans un système de gestion données*
• *Retrouver dans les divers éléments d'un SI des modèles CAO de formes 3D globalement ou localement similaires, de manière robuste et efficace*
• *Identifier une forme de manière précise sans risque de confusion avec des formes similaires*

En ce qui concerne le second macro-objectif, il s'agit de mettre en place des processus de *RE* intégrant les activités de *SA*. Pour cela, les systèmes de Knowledge-Based Engineering sont exploités afin de gérer les connaissances relatives aux modèles CAO, en accord avec les objectifs suivants :

Macro-objectif 2 : capitaliser et exploiter les informations extraites
• *Contextualiser les applications de SA pour décliner un besoin métier en un processus de RE*
• *Structurer le traitement des informations manipulées, allant des différentes CAO et bases de données en entrée du processus à l'exploitation des connaissances déduites en sortie*
• *Capitaliser les connaissances relatives aux modèles CAO pour offrir une capacité de réutilisation sur le long terme*

Ces objectifs définissent le cadre de nos recherches visant à adresser la problématique industrielle de la continuité numérique par l'amélioration de la gestion des connaissances pour la préservation de l'intégrité des modèles CAO au sein d'environnements SI complexes.

Chapitre 2 : État de l'art de la littérature scientifique

Comme nous l'avons vu dans le chapitre précédent, les modèles CAO représentent le moyen le plus courant pour encapsuler les informations issues des processus de conception des produits mécaniques. Cependant, dans de nombreux cas, ces derniers ne sont pas accessibles aux concepteurs, n'existent tout simplement pas, ou bien ne correspondent plus à la géométrie réelle du produit manufacturé. Selon (Buonamici et al., 2018), fournir une représentation numérique consistante d'un objet physique demeure un problème persistant, en particulier dans le cadre de l'ingénierie mécanique qui nécessite généralement des méthodes capables de produire des résultats précis et proches du design original, obtenus en un temps limité, et ajustés aux besoins finaux des concepteurs.

Dans ce chapitre, un état de l'art de la littérature scientifique présente les concepts clés du *RE* et en explore les principales activités représentées dans la Figure 9. Dans cette synthèse bibliographique, le *RE* n'est pas restreint à la seule reconstruction de modèles CAO continus mais est analysé en tant que processus global dont l'objectif est de recréer ou retrouver des modèles CAO de haut niveau sémantique. Ainsi, les méthodes pour une meilleure intégration de la connaissance ainsi que la recherche d'informations dans le SI d'une entreprise sont considérées comme des composantes essentielles du *RE* moderne.

Cet état de l'art met en évidence l'évolution des défis et finalités des processus de *RE* ainsi que les perspectives offertes par l'évolution des technologies informatiques et de l'apprentissage automatisé.

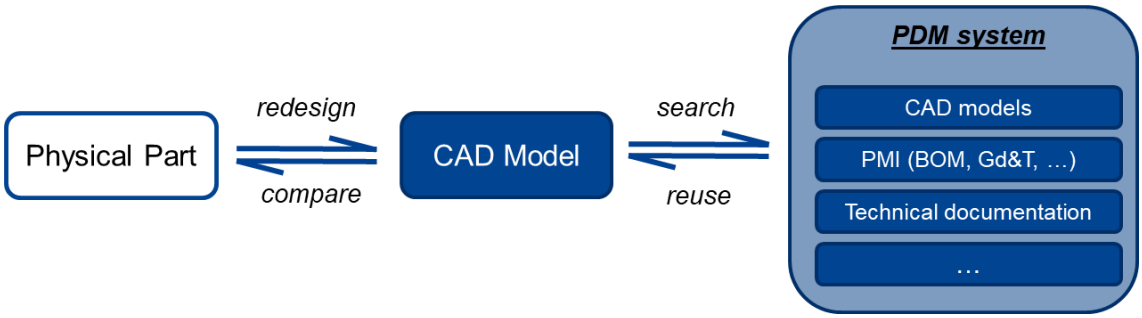


Figure 9 : principales activités des processus de RE de modèles CAO

I. De l'objet physique au modèle CAO: la reconstruction 3D

Dans une large part de la littérature scientifique, la Rétro-Ingénierie est vue comme le processus de reconstruction d'un modèle CAO sémantiquement riche à partir d'un objet physique.

Selon (Buonamici et al., 2018), le processus de *RE* est généralement constitué des quatre activités suivantes: la numérisation, le prétraitement, la segmentation et la modélisation.

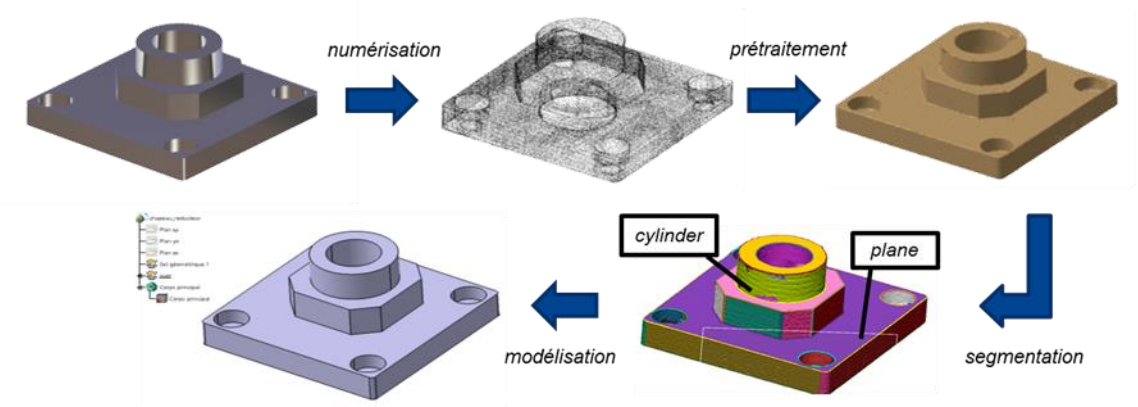


Figure 10 : le processus de RE (Buonamici et al., 2018)

*Numéri*sa*tion et prétraitement*

Numériser un objet physique consiste à en obtenir une représentation 3D numérique. Les travaux de (Barbero & Ureta, 2011) détaillent les différentes méthodes de numérisation existantes, telles que les méthodes de contact physique de la surface, ou l'utilisation de systèmes optiques comme une caméra ou un scanner laser. Les données acquises prennent généralement la forme de nuages de points définis par des coordonnées cartésiennes.

Les étapes de prétraitement (*i.e. preprocessing*), comme la décimation du nombre de points acquis, la génération d'un maillage, sa densification par subdivision des faces, le remplissage des trous ou encore le lissage de la surface sont des opérations disponibles dans la plupart des logiciels d'acquisition ou de *RE*. Cette « simplification » du modèle numérisé est essentielle pour les étapes subséquentes.

Segmentation

La segmentation est le processus de subdivision d'un maillage 3D (ou d'un nuage de points) en régions distinctes appelées des segments.

Les méthodes de segmentation sont généralement basées sur l'extraction directe de propriétés géométriques intrinsèques aux données du maillage, telles que les propriétés différentielles de bas ordre comme la densité du maillage, la courbure de surface ou ses propriétés inertielles. Les variations géométriques détectées entre les entités du maillage permettent de les séparer en différents groupes (Bello et al., 2020). La synthèse bibliographique de (A. Nguyen & B. Le, 2013) s'intéresse principalement aux méthodes « traditionnelles » de segmentation basées sur l'analyse des variations géométriques. Les travaux de (Theologou et al., 2015) intègrent quant à eux les méthodes de *ML* comme (Kalogerakis et al., 2010) pour la segmentation sémantique de maillages ou nuages de points.

On parle de segmentation sémantique quand chaque région identifiée peut être associée à une opération de construction ou une surface spécifique utile aux activités avales de reconstruction. Par exemple, on parle de méthodes *model-based* lorsque la définition analytique des primitives géométriques (i.e. leurs équations de formes) est utilisée pour regrouper les points d'une forme primitive en un segment labélisé (Kaiser et al., 2019). Les méthodes détectent le type de primitive et optimisent les paramètres du modèle via un algorithme de *fitting* (i.e. d'ajustement) comme *Random* S*ample Consensus* (*RAN*SA*C*) (Attene & Patanè, 2010; Fischler & Bolles, 1981; Schnabel et al., 2007). La nature quadratique des équations de forme permet de limiter les temps de calculs, et la forme canonique des modèles permettent d'en extraire les paramètres pilotant de la géométrie. La segmentation est alors très sensible à la qualité du maillage (densité, bruit de mesure, etc.) ainsi qu'aux seuils de segmentations. La Figure 11 illustre les résultats de segmentation de type *model-based* d'un maillage. À gauche, faute de prétraitement, un phénomène de sur-segmentation (*i.e. oversegmentation*) du maillage ne permet pas d'identifier correctement les formes primitives du modèle. Le résultat de segmentation à droite de la Figure 11 peut être considéré comme sémantique et plus facilement exploité lors de l'activité de reconstruction.

Puisque 85% des modèles CAO d'ingénierie peuvent être représentés par des associations de formes primitives (Chivate & Jablokow, 1993), les méthodes de segmentation sémantique de surfaces complexes n'ont pas été intensivement étudiées par la littérature scientifique. Récemment, des algorithmes de *ML* ont été développés pour la segmentation sémantique de maillage, aussi bien pour améliorer la détection de primitives (L. Li et al., 2019) que pour des formes plus complexes (Qi, Yi, et al., 2017).

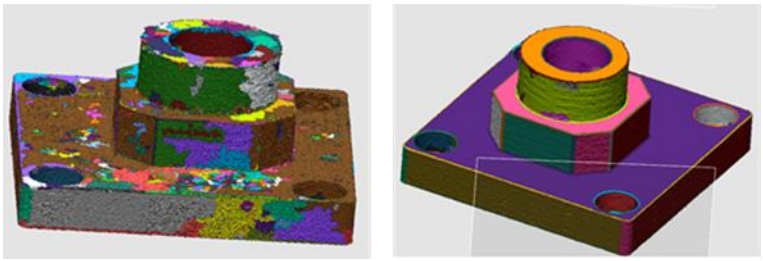


Figure 11 : résultats de segmentation d'un maillage avec le logiciel Geomagic Design X

Modéli*sa*tion

Les opérations de modélisation sont utiles pour générer un modèle 3D continu compatible avec les applications de CAE. On distingue généralement deux approches : *freeform* et *feature-based*.

L'approche *freeform* est une méthode de modélisation explicite qui consiste à décrire un solide par un ensemble de surfaces connexes, et résulte généralement en un solide "mort" (*i.e.* non modifiable via des paramètres). Des patchs de surfaces paramétriques sont ajustés sur les points du maillage via interpolation de *NURBS* et de surfaces *b-splines*, ce qui permet la reconstruction de surfaces complexes au plus proche des données du maillage avec un haut niveau de précision et de détails (Ben Makhlouf et al., 2019; Dimitrov et al., 2016). Si les approches *freeform* sont intéressantes car presque entièrement automatisées, elles résultent en un modèle CAO pauvre car entièrement figé, sans paramètres ni esquisses, et dont la topologie n'est pas représentative des surfaces et *form-features* d'un modèle de conception comme représenté dans la Figure 12.

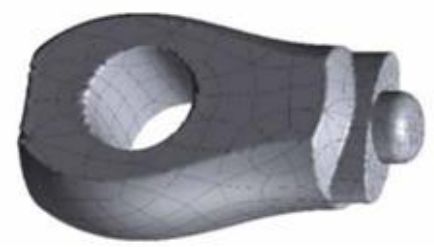


Figure 12 : reconstruction freeform d'un maillage (Buonamici et al., 2018)

D'autre part, le terme *feature-based* est utilisé pour décrire les méthodes de modélisation implicite qui cherchent à retrouver un modèle 3D paramétrique en appliquant une approche procédurale de séquences d'opérations de construction avec relations de parentés (J. Wang et al., 2013).

Parmi les principales méthodes, on retrouve le *primitive fitting* (Attene & Patanè, 2010; Bénière et al., 2013; Benkő et al., 2002) qui, basé sur les méthodes *model-based* de segmentation sémantique, consiste à reconnaître et ajuster des surfaces paramétriques de formes primitives aux segments du maillage. La Figure 13, extraite des travaux de (Bénière et al., 2013), illustre l'ajustement de surfaces primitives sur les segments d'un maillage, permettant ainsi la création d'éléments géométriques et de paramètres de contrôle en plus de la surface.

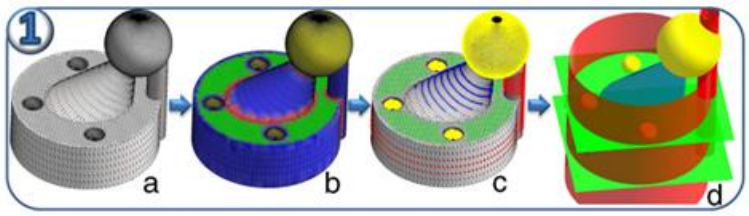


Figure 13 : reconstruction et ajustement de surfaces primitives (Bénière et al., 2013)

Les approches *Constructive Solid Geometry (CSG)* (Du et al., 2018) diffèrent légèrement des *primitive fitting* par le fait que des solides paramétriques de formes primitives sont directement ajustés dans le maillage et assemblés entre eux par des opérations booléennes.

Si les formes primitives représentent une large majorité de composants mécaniques (Kaiser et al., 2019), elles sont insuffisantes pour représenter des formes plus complexes. Les travaux de (Song et al., 2004; Vergeest et al., 2001) introduisent la notion de *form-feature* comme des formes génériques non primitives auxquelles sont associées des attributs et propriétés spécifiques. Les modèles 3D de ces *form-feature* peuvent être ajustés sur les segments via optimisation d'une transformation affine minimisant une fonction de distance. Selon (Vergeest et al., 2001), l'identification de formes non régulières dans un maillage brut rendrait les possibilités de réutilisation plus larges, mais les *form-feature* ne doivent pas dépendre d'une librairie prédéfinie. Pour (Kaiser et al., 2019), les futurs challenges dans le domaine consistent en l'amélioration de l'ajustement des surfaces en termes de complétude et de cohérence, en particulier avec l'introduction de formes plus complexes, qui devront rester génériques et non spécifiques aux données.

Enfin, les sections 2D sont une méthode incontournable pour la reconstruction de surfaces complexes pour lesquelles aucun *form-feature* n'est connu (Ke et al., 2006). En effet, les esquisses 2D sont à la base d'un grand nombre d'opérations de constructions (balayages, multi-sections, révolutions, extrusions, etc.) et peuvent être reconstruites sur des sections planes du maillage. Le choix des plans de section est souvent laissé à la charge de l'utilisateur et doit rester cohérent avec les opérations de construction qui suivent.

II. Vers une meilleure intégration de la connaissance au processus de RE

Selon (Buonamici et al., 2021), la majorité des recherches sur la reconstruction de modèles CAO s'est portée sur deux aspects : (i) retrouver des intentions de conception de haut niveau – qui se traduit généralement par l'introduction de relations géométriques entre les différents *features* CAO ; et (ii) l'étude d'algorithmes rapides et précis pour adapter des surfaces continues sur des données 3D. Pourtant, le choix des fonctions de construction, de leurs esquisses, ainsi que les paramètres et leurs relations représentent la réelle connaissance de haut niveau implémentée dans un modèle CAO. Plus précisément, on parle de « sémantique » d'un modèle CAO pour qualifier toute l'information contenue en addition de sa géométrie, comme ses dimensions, les paramètres pilotant la géométrie, les spécifications fonctionnelles, les méthodes de modélisation, les *product manufacturing informations* (PMI) et les intentions de conception. Encore aujourd'hui, assez peu de recherches adressent le problème de la reconstruction ou de la mise à jour de modèles CAO qui pourront ensuite être édités et plus facilement exploités lors des phases avales du processus de développement de produit (G. A. Shah et al., 2021).

S'il est disponible, l'utilisation d'un *template* facilite la reconstruction et assure l'obtention d'un modèle sémantiquement consistant puisque ce dernier contient déjà des connaissances utiles et structurées. La méthode dite *template-based* soulève ainsi la question de l'insuffisance sémantique des autres méthodes de modélisation qui, en fin de compte, retrouvent uniquement des informations géométriques. Les travaux de (Buonamici et al., 2021; G. A. Shah et al., 2021) proposent des méthodes de modifications automatisées des paramètres de contrôle des *templates* minimisant leurs déviations avec le maillage, permettant ainsi de maintenir les intentions de conceptions implémentées dans le *template*. De même, les travaux de (Hu et al., 2022) s'intéressent à la reconnaissance et à l'ajustement de *templates* dans des nuages de points.

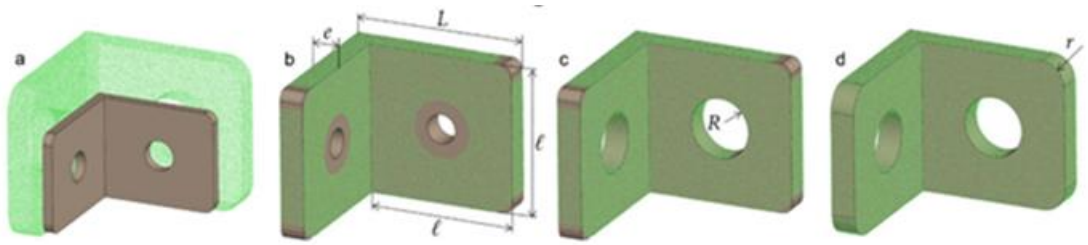


Figure 14 : ajustement d'un modèle paramétrique par la méthode template-based. (Shah et al., 2021)

En complément des connaissances expertes de conception préalablement citées, les processus de fabrication et les spécifications fonctionnelles d'un produit sont des informations qui peuvent être déduites de l'analyse d'un modèle CAO riche. Une approche uniquement géométrique n'est pas suffisante pour retrouver de telles connaissances sur le produit (Durupt et al., 2008; Fisher, 2004). Sur ce constat, le *Knowledge-Based Reverse engineering (KBRE)* se présente comme un ensemble de méthodes de *RE* qui consistent à capitaliser des informations en amont d'un travail de *RE* L'interprétation *in situ* par l'utilisateur de ces informations diverses apporte la connaissance nécessaire qui serait inaccessible par la seule approche géométrique.

Dans ce sens, (Durupt et al., 2019) propose un modèle de données implémentant des aspects fonctionnels et des contraintes de production dans la création des *features* de construction. Le projet **PHENIX** (Durupt et al., 2013) propose une méthodologie pour la capitalisation de références géométriques paramétriques qui peuvent être ajustées au maillage 3D. Le projet **METIS** (Bruneau et al., 2014; Ouamer-Ali et al., 2014) intègre l'utilisation de données hétérogènes (plans 2D, images, instructions, *templates* 3D) au processus de *RE*. La problématique principale du projet **METIS** concerne la gestion de la connaissance d'ingénierie dans son ensemble. Allant de la capture des connaissances à travers l'analyse des processus métier et la mise en place des connaissances formalisées par les experts jusqu'à son utilisation dans le processus de *RE*, le projet a pour principal enjeu la proposition de méthodes d'accès aux informations cohérentes et structurées stockées dans un SI. Une des propositions majeures avancée, illustrée par la Figure 15, correspond au concept de « signature », soit l'extraction d'informations utiles, dans un formalisme particulier, qui permet de relier de manière automatisée les données brutes aux informations capitalisées dans une *Knowledge Base (KB)*, facilitant ainsi leur réutilisation qui se conclut par un apport de connaissances sémantiques aux activités de *RE*.

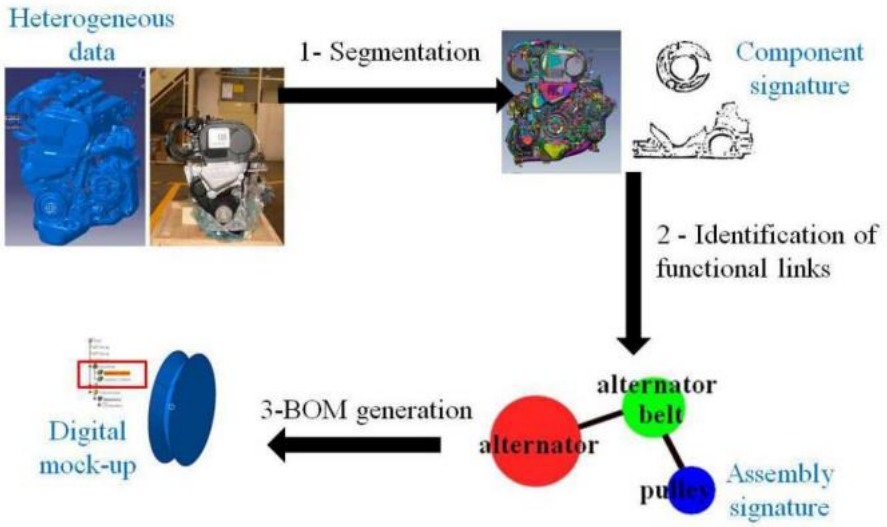


Figure 15: la méthodologie de reconstruction développée dans le cadre du projet METIS (Bruneau et al., 2014)

La tendance actuelle de la recherche autour des activités de *RE* semble donc chercher une plus grande intégration des données et connaissances existantes aux processus mis en place. Dans un contexte industriel, il semble en effet plus pertinent de s'intéresser aux moyens de capitaliser l'information et d'en faciliter l'accès, plutôt que de tout récréer *from scratch*. Il en découle que l'indexation, la recherche et l'extraction d'informations dans un SI industriel sont parmi les principaux défis actuels du *RE*. Lorsque les données considérées sont des modèles 3D, ces activités font parties du domaine de l'analyse de formes 3D.

III. L'analyse de forme: un moyen pour le RE et la gestion des connaissances

L'analyse de forme 3D, ou *Shape Analysis (*SA*)*, connait depuis deux décennies un renouveau dans l'intérêt que la recherche scientifique lui porte, notamment motivé par les avancées des technologies d'acquisition 3D, de modélisation, de visualisation, et bien sûr par l'amélioration substantielle des capacités de calcul et de stockage des outils informatiques (Laga et al., 2018).

La croissance rapide des bases de données 3D disponibles, tant dans le cadre privé des entreprises que dans le domaine public, apporte des opportunités sans précédent dans l'utilisation des méthodes de *SA* pour les processus de *RE*.

Tout d'abord, la variété des contenus 3D rend possible la réutilisation de modèles existants pour en produire de nouveaux, ou encore l'utilisation des informations fonctionnelles et connaissances relatives aux modèles de conception (Laga et al., 2018; Lupinetti et al., 2019). La recherche dans une base de données permet l'extraction de modèles 3D pertinents ainsi que des données hétérogènes associées à chacun. Cependant, la recherche de modèles 3D est complexe et ne peut généralement pas se baser sur de simples moteurs de recherches basés sur les métadonnées textuelles (Tangelder & Veltkamp, 2008).

Par ailleurs, des modèles supervisés de *ML* peuvent être entrainés sur ces larges référentiels de données à créer des raisonnements s'appuyant sur les propriétés et relations entre les formes plutôt que sur des règles ou instructions explicites (Laga et al., 2018). Ces raisonnements de haut niveau assistent l'humain dans l'organisation et la standardisation des bases de modèles 3D d'une entreprise (Lupinetti et al., 2019)

Cependant, ces opportunités s'accompagnent de défis auxquels la recherche scientifique se doit de répondre en vue d'exploiter pleinement le potentiel des activités de *SA* pour le *RE* de modèles CAO. La variété des formats de modèles CAO ainsi que les nombreuses typologies de formes qu'ils représentent nécessite le développement d'une technique de signature qui soit commune, compacte et optimisée pour les méthodes d'analyse. De plus, les volumes très importants de données impliquent le développement et l'optimisation de méthodes robustes, efficientes et précises pour l'exploration des bases de données et l'extraction d'informations pertinentes.

Dans la section qui suit, une introduction à l'analyse de formes permettra de préciser le concept de forme 3D, de présenter les principales activités de *SA*, ainsi que d'étudier les composantes et défis associés au développement d'une méthode de *SA*.

III.1. Introduction à l'analyse de forme

Définition

L'analyse de forme est une discipline qui se concentre sur l'extraction et l'interprétation des caractéristiques et des structures de la forme des modèles 3D pour en déduire des informations significatives.

*La **forme d'un objet** est définie comme son occupation spatiale dans un espace tridimensionnel. Les propriétés intrinsèques d'une forme ne prennent donc pas en considération* sa *position spatiale et son orientation, et sont indépendantes de* sa *méthode de représentation 3D (Laga et al., 2018)*

Les applications d'analyse de formes

L'analyse de formes trouve de nombreuses applications parmi les domaines du *Computer Graphics*, *Computer Vision*, *3D Modelling*, *Manufacturing*, et du *Reverse Engineering* notamment. Ci-dessous sont présentées les principales activités de *SA* qui se dégagent dans la littérature.

La première application de *SA* qu'il convient de citer est le *matching* de formes, qui consiste à déterminer leur niveau de ressemblance. Plus précisément, il s'agit de quantifier la dissimilarité entre les formes, le plus souvent par le calcul d'une mesure de distance entre elles. Les méthodes de *matching* sont généralement utilisées dans le cadre d'activités de *recognition*, de *retrieval* ou de *classification* de modèles 3D en bases de données (Osada et al., 2002). Par exemple, le *matching* de deux vis revient à en estimer la proximité de leurs formes globales.

L'indexation est une étape complémentaire qui consiste à structurer la base de données, c'est-à-dire référencer chacun des modèles 3D concernés par les activités d'analyse (Tangelder & Veltkamp, 2008). Dresser la liste complète de chaque objet de quincaillerie possédé, sans nécessairement les catégoriser, revient à les indexer.

Le *retrieval* de forme est l'un des premiers problèmes ayant attiré l'attention de la recherche scientifique autours des activités de *SA* (Ioannidou et al., 2017). Étant donné un objet 3D de requête, le *retrieval* consiste à identifier les objets 3D qui lui sont similaires (Gezawa et al., 2020; Tangelder & Veltkamp, 2008; Xiao et al., 2020). L'indexation et le *matching* de formes font généralement parti des méthodes de *retrieval*. En amont, les modèles 3D d'un ensemble sont indexés et une représentation adaptée est calculée. La forme requêtée est ensuite matchée avec chaque donnée indexée, permettant ainsi de déterminer les plus similaires (Laga et al., 2018; Tangelder & Veltkamp, 2008). Une méthode de *retrieval* intégrée dans un moteur de recherche en base de données permet d'aller au-delà de la requête par métadonnées pour la fouille, l'accès et la réutilisation de contenu 3D (Laga et al., 2018). Selon (Lupinetti et al., 2019), la capacité d'évaluer la similarité entre des modèles 3D d'un SI et l'accès aux informations qui en découle est devenue indispensable dans un contexte d'industrie 4.0. Pour reprendre les exemples de l'avant-propos, le *retrieval* d'une vis dans l'ensemble des objets connus consiste à identifier les objets de formes les plus similaires.

Avec le *retrieval*, la *classification* des formes est une activité principale de *SA* (Xiao et al., 2020). Il s'agit aussi d'une des premières applications pour lesquelles des méthodes de *ML* et *DL* ont été développées (Bello et al., 2020). Étant donné un objet 3D, une méthode de *classification*

permet de la catégoriser selon une liste de « familles » ou classes prédéfinies (Georgiou et al., 2020) permettant ainsi l'organisation de larges bases de modèles 3D (Laga et al., 2018). Un système de classification s'appelle une taxonomie. Il en existe autant que de points de vue possibles. Par exemple, déterminer qu'une vis est une « vis » revient à la classifier selon une taxonomie A. Cet objet pourrait être classifié comme « vis à tête hexagonale M6x30 » selon la taxonomie B ou simplement « élément de quincaillerie » selon le système de classification C.

La *segmentation* de forme vise à en discriminer ses parties ou composantes selon une décomposition sémantique ou en accord avec la topologie d'un modèle CAO de référence. Cette tâche joue un rôle important dans la compréhension des formes 3D représentées par un modèle 3D discret (Xiao et al., 2020). Selon (Bello et al., 2020), la *segmentation* de nuages de points est le regroupement de ces points en régions homogènes. Traditionnellement, l'analyse des variations des propriétés de surface telles que les normales, la courbure et l'orientation permettent de distinguer différentes régions. Plus récemment, des approches de *feature-based Machine/Deep Learning* ont été utilisées pour la *segmentation* des nuages de points. On distingue couramment la *segmentation* d'un objet selon ses composantes, appelée *part segmentation*, ou la *segmentation* de différentes catégories d'objets, appelée *semantic segmentation* (Bello et al., 2020). Étant donné un nuage de points ou un maillage, le but de la *semantic segmentation* est de séparer les données géométriques en sous-groupes ayant une signification sémantique, c'est-à-dire pouvant être classifiés selon une taxonomie définie (Y. Guo et al., 2020). La *semantic segmentation* inclut la *segmentation* de scènes 3D qui consiste à étiqueter chaque point d'une scène comme faisant partie d'un objet d'intérêt de premier plan ou d'une surface d'arrière-plan. Cette tâche est également connue sous le nom de *semantic segmentation and labeling* (K. Guo et al., 2016; Kalogerakis et al., 2010), et est étroitement liée à la *classification* d'objets 3D (Ioannidou et al., 2017; Laga et al., 2018). Par exemple, la *part segmentation* d'une vis revient à séparer sa tête de son fût et de son filet. Cette *segmentation* est sémantique si les différentes parties sont nommées. La *semantic segmentation* d'une table couverte d'objets divers revient à séparer et classifier chaque objet.

La détection de forme consiste à identifier la présence de caractéristiques ou fonctions spécifiques au sein de la forme, comme un congé, une forme primitive, une zone de forte courbure, ou encore un point d'intérêt appelé *keypoint* (Laga et al., 2018; Xiao et al., 2020). Les *keypoints* sont des points autours desquels de fortes variations des propriétés géométriques sont observées, et permettent par exemple de tracer les frontières d'une région dans les méthodes de segmentation traditionnelles. Leur détection est une étape critique pour diverses applications comme le *retrieval* ou la *recognition* (Ioannidou et al., 2017). La détection d'un filet sur un objet aide le raisonnement de l'humain ou de la machine à déterminer qu'il s'agit d'une vis.

La tâche de *recognition* d'objets 3D est de déterminer, en présence de bruit et d'occlusion du maillage, l'identité et la pose (c'est-à-dire la position et l'orientation) d'un objet d'intérêt dans une scène 3D. On l'appelle *shape recognition* (Laga et al., 2018). Un grand nombre d'applications sont développées dans les domaines de la robotique, des véhicules autonomes, de la réalité augmentée ou encore des lignes d'assemblage automatique (Laga et al., 2018). La *recognition* fait appel à plusieurs méthodes étudiées, comme la détection de *keypoints*, la *semantic segmentation*, la *classification* des segments ou encore le *matching* (Georgiou et al., 2020; Ioannidou et al., 2017). Selon (Y. Guo et al., 2020) et (Bello et al., 2020), la *detection* d'objets dans un nuage de points en entrée retourne les boîtes englobantes et la catégorie de chacun des objets détectés. En général, on parle de *scene understanding* pour qualifier l'ensemble des méthodes de *DL* pour la compréhension de scènes 3D, incluant les activités de *detection*, de *recognition*, de *semantic segmentation* ou encore de *classification* (Georgiou et al., 2020). La reconnaissance ne concerne pas exclusivement les scènes 3D, mais est équivalente au *partial retrieval* (Lupinetti et al., 2019). Cette activité consiste à identifier la présence d'une composante ou d'un segment dans une pièce complète. Les méthodes diffèrent selon le format du modèle

d'entrée, mais une application principale concerne l'*Automatic Feature Recognition* (AFR) (Alwswasi & Ivanov, 2019; Ding et al., 2021), qui fait référence à la reconnaissance d'opérations de construction spécifiques dans une pièce, comme des trous, des rainures ou des poches. La *recognition* correspond ainsi à identifier les vis sur une table, ou bien les têtes hexagonales d'un ensemble de vis. On notera enfin que certains auteurs utilisent le terme *object detection* pour signifier *recognition* (Bello et al., 2020; Georgiou et al., 2020), ou que certains travaux parlent de *recognition* pour des activités de *classification* et de *retrieval* conjointes (Shi et al., 2015; Su et al., 2015)

L'activité de *registration* consiste à recaler (*i.e.* aligner) des modèles 3D potentiellement incomplets. L'un des exemples les plus populaires est le problème de reconstruction 3D, dans lequel un objet 3D est généralement scanné par plusieurs capteurs positionnés à différents endroits autour de l'objet. Pour construire le modèle 3D complet de l'objet, il faut fusionner les scans partiels produits par chaque capteur. Cette opération nécessite un alignement correct, c'est-à-dire la *registration* des nuages de points ou maillages (Laga et al., 2018). Une autre application courante est l'alignement d'un scan 3D avec un modèle de définition pour étudier les déviations entre leurs surfaces. La méthode d'*Iterative Closest Point* (*ICP*) de (Besl & McKay, 1992) et ses dérivées (Laga et al., 2018) visent à calculer des paramètres de transformation rigide pour aligner les nuages de points. La méthode *Deep Closest Point* (DCP) de (Y. Wang & Solomon, 2019) utilise l'apprentissage profond pour améliorer les capacités de *registration* de la méthode *ICP* connue pour son manque de robustesse (Xiao et al., 2020). La *registration* de deux vis revient à les aligner dans le même sens pour observer une différence de longueur ou de diamètre.

À la suite du référencement des différentes méthodes de *SA*, une activité semble être ignorée bien qu'elle soit selon nous primordiale pour une compréhension de haut niveau des objets 3D. Nous proposons de définir comme *shape labeling* l'identification précise d'une forme en la reconnaissant comme parfaitement identique à une forme indexée. Le *labeling* d'un objet 3D consiste ainsi à identifier ses métadonnées (index, lien, référence) dans une base de données de modèles 3D de référence.

Il est possible de considérer le *labeling* comme une méthode de *retrieval* retournant un unique résultat dont la forme est strictement identique à celle de l'objet analysé. Dans le cas où aucun objet référencé ne correspond, l'activité de *labeling* ne retourne rien. Le terme strictement identique utilisé pour qualifier deux formes est équivalent à considérer leur mesure de dissimilarité comme nulle, ou inférieure à un seuil très exigeant.

Nous considérons cette activité de *SA* comme essentielle en raison du domaine d'application de nos recherches. En effet, les performances aérodynamiques des pièces aéronautiques dépendent largement de la géométrie des surfaces d'aubages, et sont affectées par la moindre variation ou déformation locale, même minime. Ainsi, il est nécessaire d'être exigeant en termes d'analyses des formes et de différencier deux surfaces similaires mais non identiques. D'une manière plus générale, la méthode de *labeling* est utile aux moteurs de recherche en base. Étant donnés les très grands volumes des ensembles de modèles 3D publics ou privés, la notion de « formes similaires » semble désormais insuffisante pour l'extraction d'informations pertinentes. Une méthode non suffisamment sélective peut rendre difficile l'identification de la réponse exacte ou pertinente. En termes de recherche d'informations, la méthode de *labeling* vise une précision aussi bien qu'un *recall* élevé (cf. Annexe 4 : Métriques d'évaluation)

On note un point d'attention en vue d'éviter les confusions : le terme *labeling* est parfois utilisé dans le cadre de la segmentation sémantique comme l'action de classifier des points pour les grouper en segment, ou classifier un segment formé de points.

Les challenges

Un humain peut facilement et naturellement abstraire la forme d'un objet et la décrire via ses attributs ou en langage naturel, et même grouper ou différencier des formes entre elles selon des critères géométriques ou sémantiques (Laga et al., 2018). L'analyse de forme vise à automatiser ces tâches triviales pour un humain mais complexe pour un ordinateur.

À la base d'une majorité d'applications de *SA*, on trouve les composantes suivantes (Laga et al., 2018) :

(i) **Une représentation mathématique adéquate des formes :** les méthodes de modélisation 3D sont utiles pour le stockage, la visualisation ou encore la manipulation et modification des objets 3D, mais pas forcément pour leur analyse de haut niveau. Le réel défi est de définir une représentation mathématique qui capture l'essence de la forme via des attributs qui lui sont propres.

(ii) **Une mesure qui quantifie les similarités et différences entre les formes:** le concept de similarité est spécialement ambigu car il dépend non seulement de notions géométriques mais aussi de sémantique, du contexte, des applications, mais aussi de la perception humaine (Laga et al., 2018; Tangelder & Veltkamp, 2008). Il est cependant nécessaire de définir une fonction de dissimilarité entre les descripteurs de deux formes A et B pour quantifier leurs différences. On illustre cette fonction comme une distance $d(A,B)$ répondant aux critères suivants :
   - non-négative : $d(A,B) > 0$
   - identité : $d(A,B) = 0\ si\ et\ seulement\ si\ A = B$
   - symétrique : $d(A,B) = d(B,A)$
   - respect de l'inégalité triangulaire : $d(A,B) + d(B,C) >= d(A,C)$

(iii) **Un algorithme** qui traduit les modèles 3D en descripteurs de formes et décline leur utilisation ainsi que la mesure de dissimilarité en résultats des activités de *SA.*

Les descripteurs de formes ***(shape descriptors)*** sont des représentations numériques qui capturent les caractéristiques et les propriétés géométriques des objets tridimensionnels. Ils agissent comme une « signature » intrinsèquement liée à la géométrie des formes, indépendamment de tout format de données ou métadonnées associées au fichier contenant l'information. Ils permettent de représenter des formes 3D dans un formalisme défini et optimisé pour les activités de *SA*.

Plusieurs critères, issus des travaux de (Laga et al., 2018; Tangelder & Veltkamp, 2008), ont un impact sur le choix et le développement d'un descripteur de forme. Ils sont présentés ci-après :

- **Le temps d'exécution** concerne aussi bien la rapidité de calcul d'un descripteur que son utilisation pour l'analyse de la forme qu'il représente. Un descripteur doit pouvoir être calculé *in-situ* ou en amont des activités sur de très larges ensembles de données. Le temps d'exécution est donc un critère primordial dans le choix d'un descripteur 3D.
  Nb : on note que le temps d'exécution dépend naturellement de la puissance de calcul disponible et donc de l'évolution des outils informatiques. Pour un descripteur et une méthode donnée, la valeur de ce critère n'est pas figée dans le temps et sert donc majoritairement d'élément de comparaison.

- **La compacité** du descripteur dépend de son format et de sa dimension. Une représentation compacte et efficace sur le plan computationnel permet de limiter la quantité de mémoire ou de ressources de calcul nécessaires pour être stockée ou traitée.

- **La distinctivité** du descripteur précise ses capacités discriminatoires. Ce dernier doit être capable de capturer les caractéristiques distinctives des formes, pour fournir des valeurs distinctes pour des formes différentes et des valeurs similaires pour des formes similaires. La génération d'un descripteur de forme est théoriquement une application injective, ce qui signifie qu'à chaque descripteur est associée une unique forme. L'application n'est pas bijective car plusieurs descripteurs peuvent représenter la même forme.
  Concrètement, la distinctivité d'un type de descripteur est évaluée selon sa capacité à estimer la dissimilarité entre les formes. Cependant, plusieurs auteurs soulèvent le fait que la notion de dissimilarité est subjective car dépendante de l'application et du jugement de l'utilisateur (Laga et al., 2018; Tangelder & Veltkamp, 2008). En effet, il n'est pas cohérent de comparer les capacités de *classification* d'un descripteur avec les capacités de *retrieval* d'un second. Dans le but de permettre une meilleure évaluation des descripteurs de formes, nous introduisons la notion de **niveaux de distinctivité** en vue de juger un descripteur dans le cadre d'une application spécifique plutôt que de manière globale. Trois niveaux sont définis : (1) le premier niveau de distinctivité dépend de la capacité d'un descripteur à capturer les caractéristiques distinctives communes aux formes d'une famille d'objets, et permet donc de juger un descripteur pour la *classification* de formes. (2) Le second niveau de distinctivité correspond à la capacité d'un descripteur à discriminer les formes en fonction de leur proximité géométrique, et permet de comparer différents descripteurs sur des activités de *retrieval*. Enfin, (3) le troisième niveau de distinctivité est atteint si un descripteur permet de différencier des formes très similaires. Un descripteur validant le 3^{e} niveau de distinctivité permet donc des activités de *labeling* d'objets 3D.

- **La robustesse** d'un descripteur fait référence à sa résistance aux variations et aux perturbations telles que le bruit, les zones manquantes dans le cas d'un scan d'objet et la position et l'orientation du modèle 3D. Le descripteur calculé doit rester cohérent malgré ces variations. On note qu'il serait plus juste de parler de robustesse des méthodes plutôt que des descripteurs. La robustesse de la méthode de calcul d'un descripteur signifie sa capacité à retourner le même résultat pour différentes variations géométriques et topologiques du même modèle 3D. Par ailleurs, la robustesse des méthodes d'analyse dépend naturellement de la qualité des descripteurs de formes, mais aussi de la mesure de dissimilarité. Tout ne dépend donc pas uniquement du descripteur, mais aussi de la manière dont il est manipulé.

- **L'invariance** d'un descripteur signifie qu'il est indépendant de la position spatiale de la forme représentée. Ainsi, les translations et rotations du modèle 3D n'ont pas d'impact sur le descripteur calculé. On note que certaines propriétés comme l'invariance peuvent être atteintes soit via des étapes de prétraitement de la forme avant calcul du descripteur (comme la normalisation de la position), soit par du calcul d'un descripteur satisfaisant ces propriétés, ou encore au niveau de la mesure de dissimilarité (Laga et al., 2018). Concrètement, le critère d'invariance d'un descripteur de forme rejoint celui de la robustesse à la position spatiale.

Le défi du développement d'une solution d'analyse de forme réside donc dans la proposition d'un descripteur compact permettant l'atteinte des trois niveaux de distinctivité avec une grande robustesse via des méthodes efficientes et précises. On note enfin que la sélection d'un descripteur de forme approprié dépend de la nature des formes à analyser et des objectifs spécifiques de l'application. Certains descripteurs sont plus adaptés aux formes simples, tandis que d'autres sont plus efficaces pour les formes complexes ou les formes présentant des variations d'échelle, de rotation ou de déformation. La partie suivante présente un ensemble de descripteurs de la littérature.

III.2. Les descripteurs de formes

Différentes taxonomies des descripteurs de forme existent (B. Li et al., 2015; Laga et al., 2018; Tangelder & Veltkamp, 2008; Georgiou et al., 2020). Dans notre cas, nous en distinguons trois types principaux : les descripteurs basés sur les caractéristiques géométriques, les descripteurs basés sur les graphes, et enfin les descripteurs basés sur le *Deep Learning*. La Figure 16 structure ces différents types sous la forme d'une taxonomie unique. Elle permet de mieux appréhender les différentes sections ci-après.

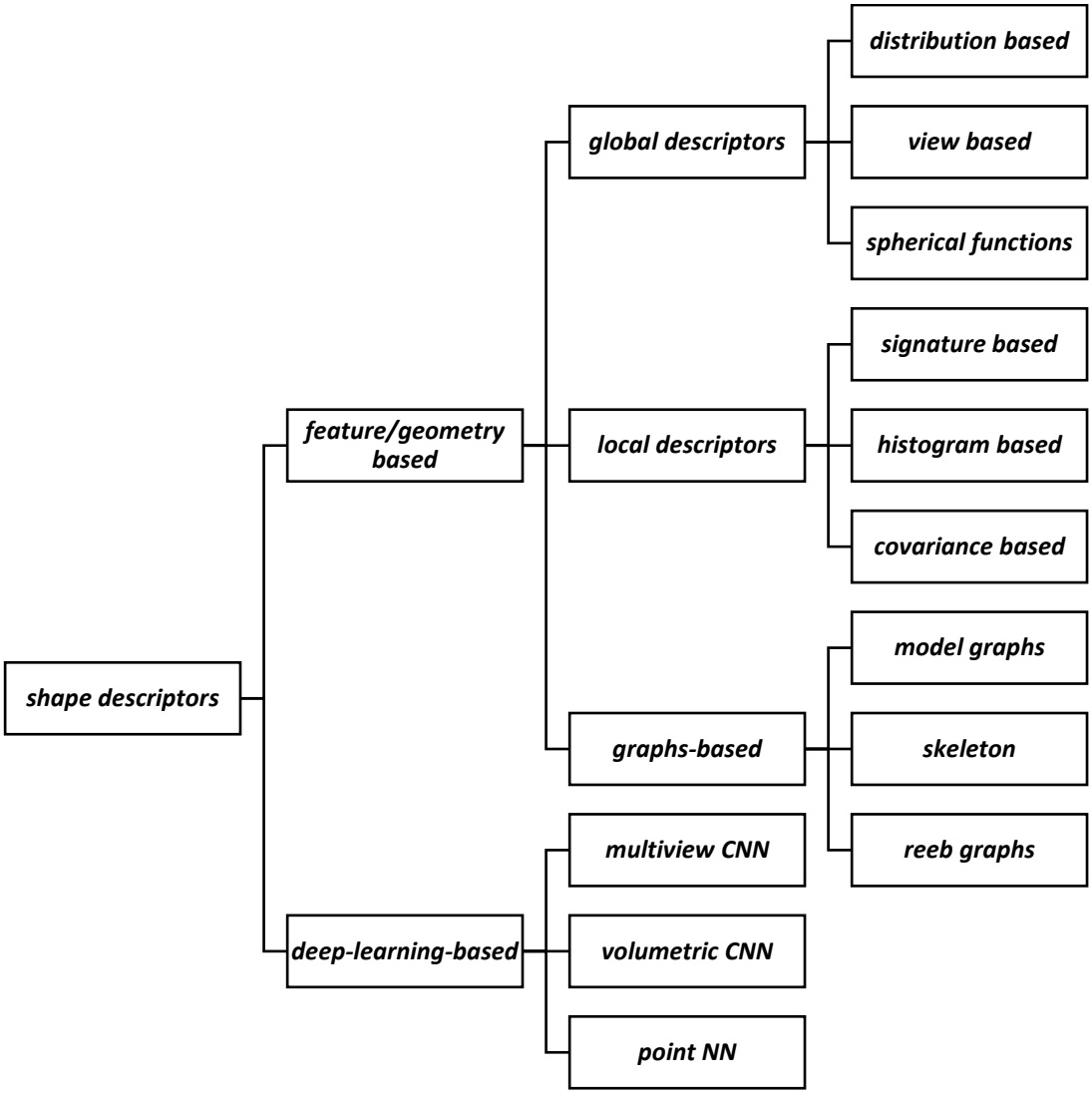


Figure 16 : taxonomie des descripteurs de formes

III.2.1. Les descripteurs basés sur les caractéristiques géométriques

Les *feature based*, ou *geometry based* descriptors prennent la forme de vecteurs de nd paramètres (souvent appelés *feature vector*) qui encodent la géométrie de la forme. Ils sont parfois appelés descripteurs « traditionnels » ou « *hand-crafted* » car directement basés sur l'extraction de caractéristiques spécifiques sans utilisation de techniques avancées d'apprentissage automatisé.

On distingue les descripteurs globaux des descripteurs locaux.

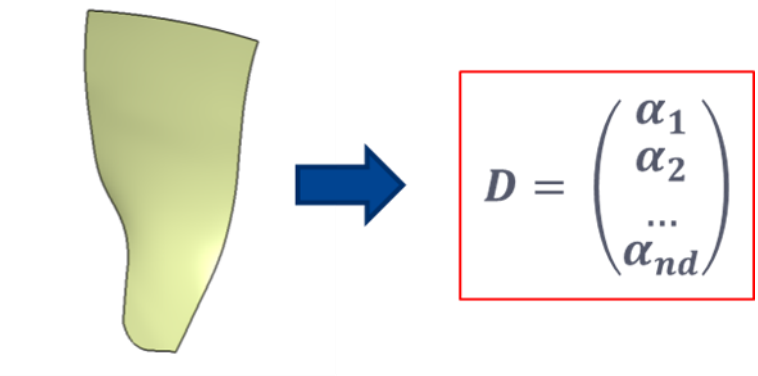


Figure 17 : feature based descriptors

Les descripteurs globaux

Historiquement, les descripteurs globaux ont été proposés comme un moyen pour décrire une forme dans son ensemble. Les plus connus sont ceux basés sur la distribution, ceux basés sur les vues, et ceux basés sur une représentation sphérique.

Les *Shape Distribution* (*SD*) sont des descripteurs proposés par (Osada et al., 2002) pour représenter la forme d'un objet 3D comme une distribution de probabilités de propriétés géométriques (angle, distance, aire ou volume) ou différentielles (normale ou courbure de surface) calculées à différentes localisations de la forme. Les *SD* se basent sur diverses fonctions de formes, les *shape functions*, pour calculer les propriétés sur un ensemble de données géométriques discrètes échantillonnées sur la forme. La Figure 18 illustre trois *shape functions* calculées sur les éléments géométriques discrets échantillonnés sur une forme. L'Annexe 3 référence un ensemble de *shapes functions* proposées depuis l'introduction des SD par (Osada et al., 2002).

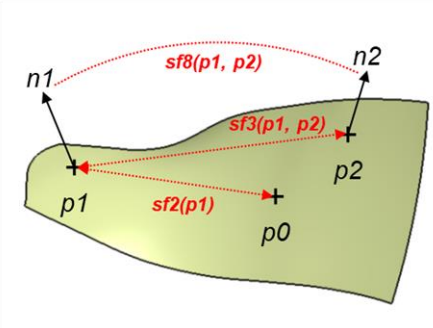


Figure 18 : exemple de shape functions des SD (Osada et al., 2002)

L'histogramme de distribution d'une fonction de forme est constitué des intervalles normalisés de la plage de valeurs calculées sur un échantillon de données. À chaque intervalle est associée la probabilité que la valeur de la fonction de forme calculée en un point aléatoire soit comprise dans cet intervalle. Le descripteur de forme associé est ainsi constitué des valeurs des probabilités pour chaque intervalle, comme illustré Figure 19. Ces derniers sont ensuite principalement utilisés dans des activités de *matching* et de *retrieval* en calculant une mesure de distance entre les descripteurs pour estimer la dissimilarité entre deux formes.

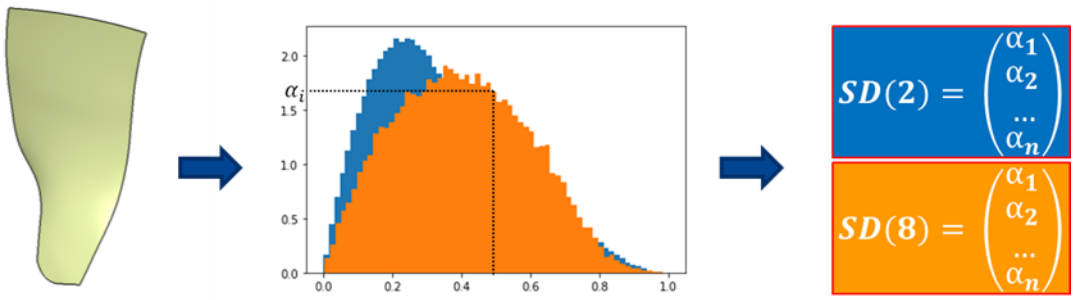


Figure 19 : création des descripteurs SD à partir des histogrammes de distribution de deux shape functions

On notera que la performance des SD dépend de la fonction de forme choisie, du nombre de mesures effectuées, et du nombre d'intervalles de l'histogramme de distribution. Si ce type de descripteur est compact et rapide à calculer, il est couramment admis qu'ils ne sont pas suffisamment discriminants (Laga et al., 2018).

Les descripteurs basés sur les vues représentent une forme 3D comme une collection d'images capturées selon différents points de vue. Ils se basent sur le principe que, si deux formes sont identiques, alors leurs projections 2D sont similaires. Le *Light Field Descriptor* (*LFD*) de (D.-Y. Chen et al., 2003) propose un système de caméras virtuelles en rotation autour d'un objet permettant de capturer des images selon un grand nombre de points de vue. Divers descripteurs 2D peuvent être calculés pour chaque image, comme les transformées de Fourrier ou le Zernike moment coefficient (D.-Y. Chen et al., 2003), puis agrégés en un descripteur global pour chaque prise de vues du système de caméras. La mesure de dissimilarité entre deux objets A et B est ainsi la distance minimum entre un ensemble de vues de A et chaque ensemble de vues de B (ou plus précisément les descripteurs globaux de chaque ensemble de vues).

Si les LFD sont relativement rapides à calculer, ont un bon potentiel discriminant et sont invariants et robustes, ils sont loin d'être compacts (Laga et al., 2018). Par exemple, le LFD de (D.-Y. Chen et al., 2003) est un vecteur de 100*45 coefficients.

Les travaux de (H. Laga et al., 2004; Laga et al., 2006) proposent la paramétrisation d'un modèle 3D comme une fonction définie sur un domaine sphérique (ou système de coordonnées sphérique). Parmi les principales fonctions de formes sphériques, on trouve, entre autres, les *Extended Gaussian Image* (*EGI*) et les *Complex Extended Gaussian Image* (*CEGI*) proposés par (B. K. P. Horn, 1984) et (S. B. Kang & K. Ikeuchi, 1993). Selon (Laga et al., 2018), ces descripteurs se limitent à un nombre réduit de modèles 3D ayant une topologie équivalente à celle d'une sphère (aussi appelés surfaces de genre 0). Ils ne sont donc pas compatibles avec l'analyse de surfaces non fermées (*i.e.* ne délimitant pas un volume)

Les descripteurs locaux

Alors que les descripteurs globaux représentent la forme d'un objet dans son ensemble, les descripteurs locaux caractérisent des régions locales. Ces derniers se montrent spécialement utiles pour des activités de *registration*, de *detection* ou de *recognition*, et permettent parfois de meilleurs résultats sur les activités de *matching* et de *retrieval*.

Les descripteurs locaux représentent la forme de *patchs* locaux autour des points spécifiques, appelés points d'intérêt, sa*lient points*, ou encore *keypoints*. Dans la plupart des cas, les points qui ne portent assez d'informations sont exclus pour éviter le calcul de descripteurs sur un trop grand nombre de points. Pour cela, des *feature detectors* sont introduits pour détecter les points dont le voisinage présente de fortes variations des propriétés géométriques, comme un changement rapide et brusque de courbure par exemple (H. Chen & Bhanu, 2007; Y. Guo et al., 2014). Ensuite, les descripteurs des *patchs* localisés sont calculés. Un aperçu complet est disponible dans la référence (Y. Guo et al., 2016). On notera en particulier les méthodes dites *point signature* qui rééchantillonnent la surface localisée autour d'un *keypoint*, puis utilise les attributs géométriques individuels des points pour générer le descripteur (Chua & Jarvis, 1997). Les méthodes basées sur les histogrammes représentent une région locale avec un histogramme de mesures géométriques ou topologiques (comme le nombre de points, l'aire ou la courbure de surface) sur un domaine spécifique (R. B. Rusu et al., 2008, 2009; Tombari et al., 2010).

Enfin, des méthodes comme le *bagg of features* de (Z. Lian et al., 2010) permettent d'agréger des descripteurs locaux en un descripteur global visant un caractère discriminant supérieur aux descripteurs globaux présentés précédemment.

On note que ces méthodes nécessitent bien plus de puissance de calcul et de mémoire que les descripteurs globaux (Y. Guo et al., 2016). De plus, elles n'ont de réel intérêt que lorsque la forme présente effectivement des *keypoints* remarquables, ce qui est souvent le cas pour les modèles 3D d'objets mais non évident pour leurs surfaces locales. La Figure 20 représente une échelle de couleurs des courbures calculée sur les maillages de pièces globales (Figure 20 (a) et (c)) et ceux d'une surface unitaire d'aubage (Figure 20 (b) et (d)). On observe que la détection de *keypoints* est bien plus intuitive sur les modèles complets que sur leurs surfaces locales.

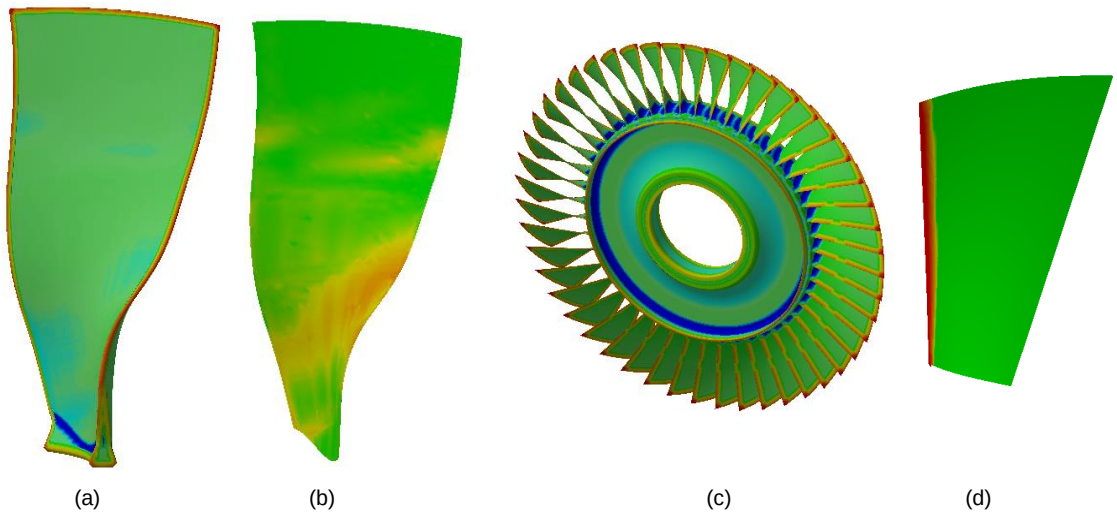


Figure 20: détection des keypoints dans les zones de fortes courbures (mean gaussian curvature)

III.2.2. Les descripteurs basés sur les graphes

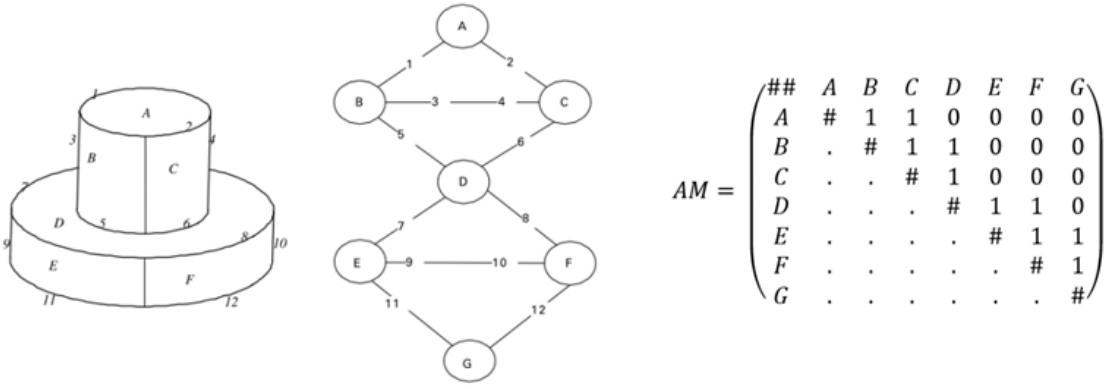
Selon (Tangelder & Veltkamp, 2008), les méthodes basées sur les graphs tentent de représenter la « signification » géométrique des formes 3D à l'aide de graphiques représentant les relations entre les composantes d'un modèle 3D. On distingue les *models graphs*, les squelettes et les *reeb graphs*.

Les models graphs

Les *models graphs* permettent de décrire un modèle au format b-rep comme un graphique, où les nœuds correspondent aux surfaces unitaires et les arrêtes aux relations de connexités. On parle de FAG pour *Face Adjacency Graph* (Lupinetti et al., 2019).

Les *models graphs* sont, en réalité, davantage des descripteurs topologiques des modèles CAO que des descripteurs de forme. On note qu'ils permettent aussi la description d'assemblages. Les travaux de (El-Mehalawi & Miller, 2003a, 2003b) comme ceux de (Lupinetti et al., 2018) concernent « l'étiquetage » des graphs qui deviennent alors des *Attributed Adjacency Graphs* (AAG) par l'ajout d'informations fonctionnelles aux nœuds et arrêtes du graphique, comme la nature de la surface, le type de connectivité ou encore un descripteur *feature-based*. Les AAG permettent une reconnaissance des structures communes parmi les pièces et assemblages, spécialement utiles dans le cadre d'activités d'AFR (Al-wswasi & Ivanov, 2019; Ding et al., 2021; Ma et al., 2010).

La matrice d'adjacence (*AM* pour *adjacency matrix*) est une représentation numérique pour le stockage et l'analyse des graphes d'adjacence. Elle permet de représenter les relations de connexités et de comparer différentes matrices pour identifier des structures communes (Ding et al., 2021). Les relations de connexités entre les $n > 1$ surfaces unitaires d'un groupe topologique sont représentées par une matrice diagonale de dimension $(n * n)$. Si les faces i et j sont adjacentes, alors $a_{i,j} = 1,\ a_{i,j} = 0$ sinon. La Figure 21 représente un modèle CAO de type b-rep, son *graph* de connectivité et la matrice d'adjacence associée.



$$AM = \begin{pmatrix} \#\# & A & B & C & D & E & F & G \\ A & \# & 1 & 1 & 0 & 0 & 0 & 0 \\ B & . & \# & 1 & 1 & 0 & 0 & 0 \\ C & . & . & \# & 1 & 0 & 0 & 0 \\ D & . & . & . & \# & 1 & 1 & 0 \\ E & . & . & . & . & \# & 1 & 1 \\ F & . & . & . & . & . & \# & 1 \\ G & . & . & . & . & . & . & \# \end{pmatrix}$$

Figure 21: modèle b-rep, FAG et AM associé. Figure en partie extraite des travaux de (El-Mehalawi & Miller, 2003a)

Les squelettes

Les squelettes des modèles 3D, parfois appelés squelettes topologiques ou squelettes médians, sont des représentations abstraites des modèles 3D qui capturent la structure essentielle ou la forme générale de l'objet tridimensionnel. (Sundar et al., 2003) les définit comme une “colonne vertébrale” de l’objet. L'objectif des squelettes est de réduire la complexité du modèle tout en conservant ses caractéristiques structurelles importantes. Ils servent à représenter la forme générale de l'objet sans tenir compte de ses détails géométriques et ne sont donc pas adaptés à l’analyse de surfaces complexes.

Les reeb graphs

Les *reeb graphs* sont similaires aux squelettes et décrivent la topologie d’un modèle via une méthode de *slicing* (i.e. tranchage) du maillage et la détermination du centre de gravité des « tranches », qui correspondent alors aux nœuds du graphe (Durupt, 2020; Herlem et al., 2014). Les arêtes du *graph* sont les liens d’adjacence entre les nœuds. Les *reeb graphs* décrivent ainsi la structure topologique globale d’une forme délimitant un volume fermé. De même que les squelettes, les *reeb graphs* ne semblent pas adaptés aux applications qui nécessitent un haut niveau de distinctivité.

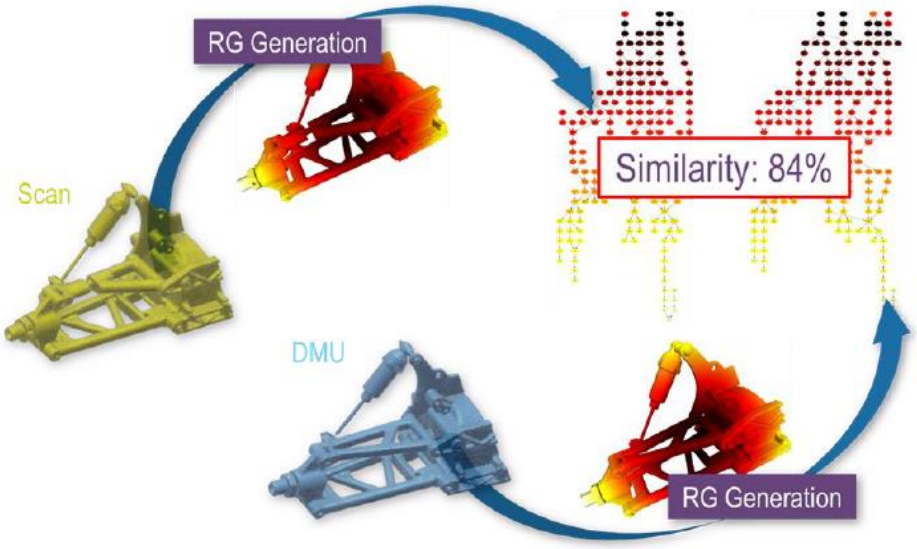


Figure 22: illustration du matching de modèles par reeb graphs. Figure extraite des travaux de (Herlem et al., 2014)

III.2.3. Les descripteurs basés sur le Deep-Learning

De manière globale, les descripteurs globaux et locaux produisent des résultats acceptables pour des activités de *retrieval*, *recognition* et *classification* sur de petits ensembles d'objets 3D (Laga et al., 2018). Seulement, chacun de ces descripteurs, qualifiés de « *hand-crafted* » (Ioannidou et al., 2017) ou « traditionnel » (Georgiou et al., 2020), a été conçu pour capturer certaines propriétés spécifiques des formes. Ils voient donc leurs capacités de ségrégation réduites lors de l'analyse de larges ensembles, et tout particulièrement s'ils présentent de fortes variabilités intra-classes et de fortes similarités inter-classes. Plus précisément, les descripteurs et méthodes d'analyses peinent à différencier des formes géométriquement proches, ou à regrouper des formes sémantiquement proches bien que géométriquement éloignées lors des activités de *classification* et de *retrieval*.

Les recherches sur le sujet se sont donc naturellement tournées vers les outils d'apprentissage supervisé pour les méthodes de *SA*. En effet, suite au succès des réseaux de neurones artificiels pour l'analyse d'images 2D, un grand nombre d'architectures de *DL* a été développé pour l'extraction de descripteurs représentatifs des données 3D et/ou la prédiction de résultats d'analyse de formes. De récents états de l'art des techniques de DL pour l'analyse de formes ont été réalisés par (Georgiou et al., 2020; Gezawa et al., 2020; Ioannidou et al., 2017; Xiao et al., 2020). Dans la suite ce cette partie, les principaux travaux de la littérature sont présentés.

Les Multi-view CNN

Un *Convolutional Neural Network (CNN)* est un réseau de neurones qui, à partir d'une image, produit un ensemble de scores qui correspond aux probabilités que l'image appartienne aux classes connues du modèle. De manière similaire à (D.-Y. Chen et al., 2003), le *multi-view CNN (MVCNN)* de (Su et al., 2015) utilise un ensemble de vues 2D généré par capture du modèle 3D sous différents angles. Chacune de ces images est traitée par un *CNN* entrainé à la *classification*, puis l'activation des neurones d'une couche définie du modèle est retournée comme descripteur de l'image qui lui a été donné en entrée. Un dernier *CNN* permet d'agréger les descripteurs extraits en un descripteur global de la forme. Une mesure de distance entre descripteurs permet des applications de *matching* et *retrieval*, et un modèle de *ML* peut être entrainé à classifier les descripteurs (et donc les formes) comme illustré Figure 23. On note également les travaux de (Qi et al., 2016) pour améliorer les résultats de *classification* via une stratégie de multi-résolution, soit l'augmentation des données d'apprentissage dans un format différent de l'image d'origine.

La très grande variété de *datasets* disponibles pour la *classification* d'images permet un pré-entrainement des *CNNs*, limitant le nombre de données d'entrainement à générer depuis les données 3D et simplifiant la mise en œuvre de cette méthode. Les *MVCNN* génèrent des descripteurs aux capacités de ségrégation supérieures aux méthodes *view-based*, et restent encore aujourd'hui parmi les méthodes obtenant les meilleurs résultats de *classification* (Xiao et al., 2020). Cependant, l'utilisation d'images implique inévitablement une perte d'informations géométriques (Xiao et al., 2020).

(Dekhtiar et al., 2018) propose l'utilisation d'un *CNN* pour classifier des images de composants mécaniques. Pour générer les données d'entrainement du modèle, une méthode de *data augmentation* est mise en place et consiste à générer un grand nombre de copies de chaque image en faisant varier certains paramètres tels que l'arrière-plan, la saturation des couleurs ou encore les contrastes de l'image. Cette étape permet de rendre le modèle invariant à certaines perturbations des images à labéliser. Par ailleurs, ces mêmes travaux comme ceux de (Su et al., 2015) proposent d'appliquer la méthode du *transfert learning*, soit le fait de pré-entrainer le *CNN* sur une des abondantes bases d'images disponibles publiquement, puis d'affiner les paramètres du modèle sur des données privées, permettant ainsi de limiter le nombre de donnés d'entrainement nécessaires.

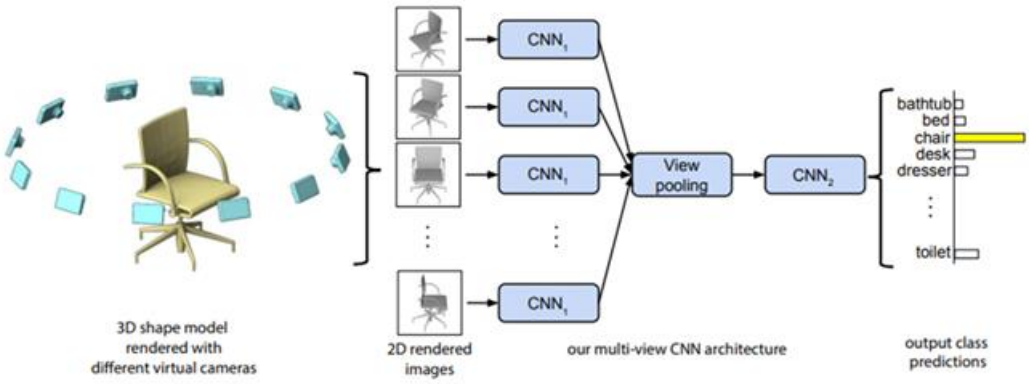


Figure 23 : MVCNN pour la classification et le retrieval d'objets 3D (Su et al., 2015)

Figure 24 : résultats de classification d'un modèle CAO par prédictions d'un CNN (Dekhtiar et al., 2018)

Les Volumetric CNN

Les *CNN 3D*, ou volumétriques, sont une extension de l'architecture des *CNN* opérant sur des images 2D à une architecture qui permet le traitement de modèles volumétriques. Dans ce cas, des voxels (terme issu de *volumetric pixels* (Bruneau, 2016)) représentent la forme comme une grille volumétrique en trois dimensions où chaque cube est défini par un statut binaire (occupé ou non) comme représenté dans la Figure 25.

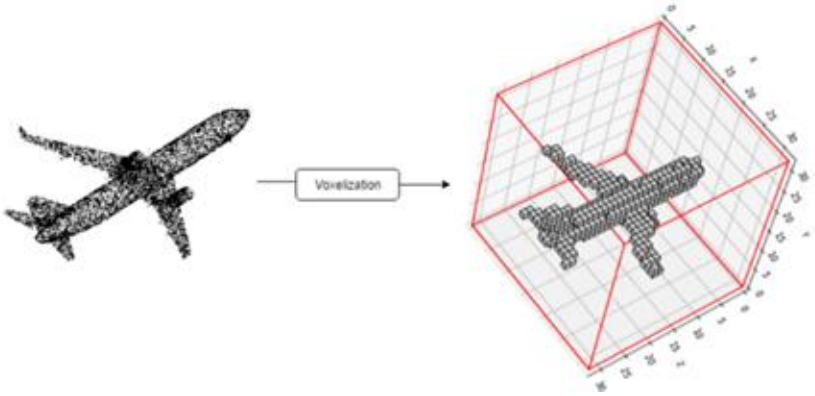


Figure 25 : représentation « voxélisée » d'un nuage de points dans une grille d'occupation volumétrique 30 × 30 × 30 (Bello et al., 2020)

De même que pour les *CNN* 2D, le résultat d'une couche du *CNN* 3D entraîné est utilisé comme descripteur de la forme qui lui est donnée en *input* (Maturana & Scherer, 2015; Wu et al., 2015). Ces modèles permettent ainsi des activités de *classification*, de *retrieval* et de *recognition* de forme.

Les travaux de (Qi et al., 2016) déterminent une supériorité des MV*CNN* sur les *volumetric CNN*, notamment due au fait que des voxels ne permettent pas de maintenir précisément les informations géométriques des surfaces. En effet, augmenter la résolution du modèle implique une diminution de la taille des voxels, résultant en une augmentation exponentielle de la mémoire nécessaire et du coût de calcul du *CNN* 3D. Les *volumetric CNN* sont donc restreints à l'analyse de modèles de basse résolution, soit de formes relativement simples. On notera aussi que, basés sur une représentation volumétrique des formes, les voxels ne sont pas adaptés aux surfaces non fermées (i.e. ne délimitant pas un volume).

Les travaux de (P.-S. Wang et al., 2017) utilisent une structure de donnée adaptive appelée *octree* pour réduire la dimension du modèle ainsi que celle du *CNN* 3D générant le descripteur de la forme. En plus des activités de *classification* et de *retrieval*, ce modèle permet la segmentation du maillage 3D ou nuage de points à l'origine du modèle volumétrique.

Les Point NN

Dû à leur caractère non-ordonné et irrégulier, peu de recherches s'étaient intéressées à l'analyse directe de nuages de points jusqu'à peu (Laga et al., 2018; Xiao et al., 2020). Le modèle *PointNet* de (Qi, Su, et al., 2017) est le premier à proposer la *classification* ainsi que la segmentation sémantique par traitement direct d'un nuage de points via l'extraction de *features* significatives relatives à chaque point. Son évolution *PointNet*++ (Qi, Yi, et al., 2017) permet une meilleure extraction de propriétés locales de points, et donc une meilleure prédiction de leurs classes pour la segmentation sémantique.

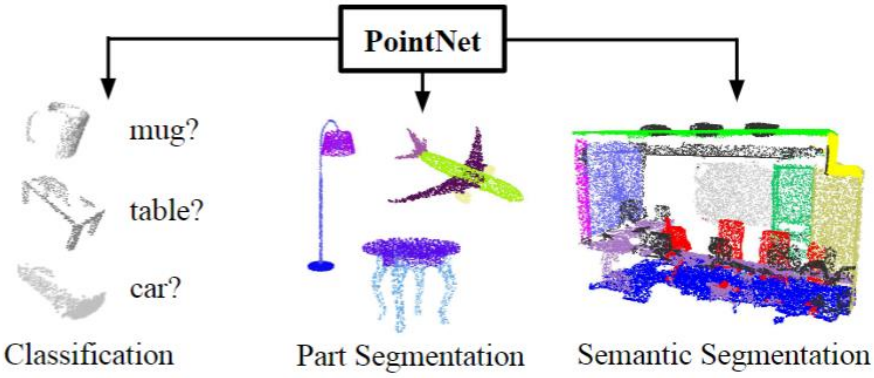


Figure 26 : les fonctionnalités de PointNet (Qi, Su, et al., 2017)

Toujours dans le but d'appliquer les technologies de *DL* aux nuages de points, (Y. Li et al., 2018) proposent ce qu'ils appellent une *X-transformation* permettant de réordonner le nuage de points, et de rendre possible l'utilisation d'un *CNN* classique pour générer un descripteur de forme. De son côté, (Hu et al., 2022) s'intéressent à la reconnaissance, l'identification et l'ajustement de *templates* CAO dans un nuage de points.

Le principal avantage des méthodes traitant les nuages de points est qu'elles nécessitent moins de mémoire qu'une représentation de forme par des voxels, et impliquent moins de perte d'informations géométriques que les projections 2D (Gezawa et al., 2020). Des synthèses des travaux de *DL* spécifiques à l'analyse de nuages de points ont été proposées par (Bello et al., 2020; Y. Guo et al., 2020).

IV. Bilan de l'état de l'art

Le deuxième chapitre de ce manuscrit de thèse a abordé les principaux concepts associés au *RE*, processus dont la finalité est de recréer ou retrouver des données et connaissances relatives à la définition d'un produit ou d'un système mécanique.

Le *RE* a traditionnellement été considérée comme un processus de reconstruction de modèles CAO continus à partir des données numérisées de pièces physiques. Aujourd'hui, de nombreuses méthodes permettent un ajustement semi-automatisé d'éléments géométriques dans les données de maillages 3D pour reconstruire des géométries primitives et des formes plus complexes. Des solutions de *RE* puissantes sont disponibles commercialement et répondent à la plupart des besoins de reconstruction (Buonamici et al., 2018).

Par la suite, l'intégration de connaissances fonctionnelles au modèle reconstruit est devenue essentielle pour répondre aux besoins industriels et l'utilisation du résultat de *RE* pour les activités avales de *CAE*. En vue d'obtenir davantage qu'une enveloppe géométrique, les approches *template-based* ont évolué pour permettre la reconstruction de modèles intégrant les connaissances des experts et leurs intentions de conception dans une arborescence de construction paramétrique et modifiable. De même, les recherches se sont tournées vers l'exploitation des systèmes de *Knowledge Based Engineering* permettant la capitalisation de connaissances dans un formalisme adapté à leurs récupérations. Ces recherches encouragent l'intégration de données diverses et non uniquement géométriques dans les processus de *RE* pour favoriser l'accès et l'exploitation du capital sémantique de l'entreprise, plutôt que de considérer la *RE* comme une activité « *from scratch* », unitaire et isolée. Cependant, ces approches ont pour prérequis la capacité de retrouver des informations pertinentes au milieu des importants volumes de données numériques des entreprises. Dans le domaine de la recherche d'informations en base de données, le concept de signature permet la capitalisation d'une donnée dans un formalisme permettant son identification. Lorsque la donnée concernée est un modèle CAO, les descripteurs de formes font généralement office de signatures pour la réalisation d'activités d'analyses de forme.

L'analyse de forme s'est imposée comme une discipline essentielle pour la compréhension des modèles 3D et la recherche d'informations en base de données. Les descripteurs de forme jouent un rôle crucial en représentant les formes des modèles de manière adaptée aux activités de *SA* comme le *matching*, la *classification*, le *retrieval*, la *segmentation*, la *recognition*, etc. Ces activités permettent une meilleure compréhension des modèles 3D afin de les comparer, les relier, les organiser, ainsi que les retrouver par l'exploration des bases qui les stockent.

Le Tableau 2 ci-dessous regroupe l'ensemble des articles de revue sur les méthodes de *SA* étudiés lors de l'état de l'art présenté dans ce chapitre. Pour chacun sont représentés les activités de *SA* couvertes par les articles scientifiques qui y sont référencés.

On observe tout d'abord que la majorité des méthodes développées se concentrent sur les activités de *classification*, *retrieval* et de *semantic segmentation* (qui inclue *part segmentation* et *scene segmentation* équivalent à *recognition*).

reference	*year*	*matching*	*classification*	*retrieval*	*part segmentation*	*detection*	*recognition*	*reconstruction*	*registration*	*labeling*
(Tangelder & Veltkamp, 2008)	2008	x		x						
(Ioannidou et al., 2017)	2017	x	x	x	x	x	x			
(Laga et al., 2018)	2018	x	x	x	x	x	x		x	
(Lupinetti et al., 2019)	2019	x	x				x			
(Georgiou et al., 2020)	2020		x	x	x		x			
(Xiao et al., 2020)	2020	x	x	x	x	x	x	x	x	
(Gezawa et al., 2020)	2020		x	x						
(Bello et al., 2020)	2020		x		x		x			
(Y. Guo et al., 2020)	2020		x		x		x			

Tableau 2 : Synthèse des activités traitées par les méthodes référencées dans des articles de revue sur l'analyse de formes

De nombreux descripteurs qualifiés de « *hand crafted* » ou « traditionnels » ont d'abord été développés par extraction de caractéristiques géométriques ou topologiques des formes. Ces descripteurs traitent principalement des activités de *matching* et de *retrieval* (Tangelder & Veltkamp, 2008). Ces méthodes de *retrieval* sont généralement évaluées en utilisant la métrique du score de précision (cf. Annexe 4). Cela signifie que les résultats de l'activité sont jugés corrects lorsqu'ils appartiennent à la même « famille » que l'objet analysé. Cette approche permet de prendre en compte des aspects fonctionnels et conceptuels, mais introduit également une dimension subjective qui peut influencer l'évaluation des résultats de manière significative puisque la notion de similarité étudiée dans ce contexte est de nature sémantique et arbitraire, plutôt que purement géométrique. La distinction entre similarité sémantique et géométrique est cruciale pour garantir la pertinence et l'exactitude des méthodes de *retrieval* dans des applications industrielles complexes et particulièrement pour l'analyse de modèles aéronautiques. On note enfin que l'efficacité des descripteurs « *hand crafted* » est limitée par la complexité des formes et la grande volumétrie des données, les rendant peu discriminants (Laga et al., 2018).

Les techniques de *Machine/Deep Learning* ont rapidement apporté des améliorations dans les méthodes de *SA*. Les succès des réseaux de neurones artificiels pour l'analyse d'images ont poussé les chercheurs à étendre leurs applications à l'analyse de modèles 3D grâce aux méthodes *multi-view*. Cependant, un ensemble de prises de vues 2D d'un objet 3D ne capture pas parfaitement ses détails géométriques, altérant la distinctivité des descripteurs calculés (Y. Guo et al., 2020; Xiao et al., 2020).

Les représentations volumétriques ont pour objectif d'étendre les possibilités d'utilisation des *CNN* au monde 3D sans passer par l'analyse de vues 2D. Cependant, conserver les détails dans une représentation par voxels implique une très grande résolution et un impact direct sur le temps de calcul d'un descripteur de formes. De plus, les grilles d'occupation volumétrique ne sont pas adaptées à la représentation de surfaces ouvertes (Xiao et al., 2020).

Les méthodes d'analyse de maillages ou de nuages de points sont spécialement intéressantes, car elles couvrent la plupart des formats 3D courants. En effet, le maillage 3D reste le standard des numérisations de pièces réelles ou des activités de simulations et de calculs, et

les méthodes de tessellation permettent d'obtenir des données géométriques discrètes à partir d'un modèle *b-rep*. Cependant, les méthodes d'analyse de maillages ou de nuages de points ne semblent pas couvrir l'activité de *retrieval*, pourtant essentielle aux scénarii de *RE* identifiés, comme le démontrent les travaux de revue de (Bello et al., 2020; Y. Guo et al., 2020).

De manière générale, les méthodes basées sur des modèles de *ML/DL* nécessitent de grandes quantités de données d'apprentissage étiquetées, c'est-à-dire des modèles 3D préalablement classés selon un taxonomie de « familles ». Bien que des ensembles de données publics existent (Chang et al., 2015; Kim et al., 2020; Wu et al., 2015), ils ne reflètent généralement pas les bases de données industrielles, et encore moins les typologies spécifiques des composants aéronautiques. Ainsi, bien qu'il soit possible d'entraîner des modèles sur ces ensembles de données publiques, la généralisation des résultats nécessite un affinement de l'apprentissage sur des typologies de modèles spécifiques. Cela est particulièrement complexe pour des éléments comme les surfaces gauches d'aubages, pour lesquels il n'existe pas de taxonomie prédéfinie. En conséquence, l'adaptation des modèles ML/DL à ces typologies spécifiques peut s'avérer difficile, voire impossible, sans un jeu de données représentatif et exhaustif.

Ainsi, en accord avec la définition donnée au *labeling* de forme, à savoir l'identification précise d'une forme en la reconnaissant comme parfaitement identique à une forme indexée, aucune recherche à notre connaissance ne traite d'un descripteur aux propriétés distinctives suffisantes pour le *labeling* de formes (cf. Tableau 2).

En conclusion, l'analyse de l'état de l'art nous permet de préciser nos objectifs de recherche vers la proposition d'un descripteur de forme couvrant un ensemble d'activités de SA, tout en étant adapté au contexte de notre étude.

Tout d'abord, ce descripteur doit permettre de caractériser des surfaces gauches ouvertes aussi bien que des volumes. Compte tenu de la volumétrie importante des données, ce descripteur doit rester compact et rapide à calculer.

Pour garantir une couverture exhaustive, il doit être applicable aux usages « classiques » de SA, c'est-à-dire adapté aux méthodes de *matching*, de *retrieval* et de *classification*. Une attention particulière sera portée à la proposition d'une méthode de retrieval performante en termes de distinctivité.

Enfin, développer une méthode de calcul d'une mesure de dissimilarité précise permettant de dépasser la simple notion de « similarité » est essentiel pour atteindre un niveau de distinctivité III. Cela permettra d'ajouter l'activité de *labeling* aux activités de SA, contribuant ainsi à l'avancement des recherches sur la compréhension des modèles CAO, l'exploration des bases de données et l'extraction pertinente des informations qu'elles contiennent.

Chapitre 3 : Proposition de méthodes et outils pour l'analyse de modèles CAO

Dans les chapitres précédents, l'analyse du contexte de recherche, des besoins industriels spécifiques ainsi que l'état de l'art de la littérature scientifique nous ont permis d'orienter nos travaux et de converger vers une proposition scientifique en deux axes.

La première partie de ce chapitre porte sur l'apport de nouvelles connaissances pour l'analyse de forme et l'exploration d'ensembles de modèles 3D :

- Un descripteur de forme est proposé comme représentation optimisée de surfaces complexes. Ce dernier a pour objectif de mieux s'adapter aux spécificités des typologies de formes des modèles aéronautiques.
- Des méthodes de *SA* - fondées sur des techniques issues de différents domaines comme la géométrie computationnelle, l'analyse numérique et les méthodes d'apprentissage automatisé - sont proposées dans le but de couvrir les activités de *matching*, de *retrieval* et de *classification*. Une nouvelle méthode de *labeling* est proposée afin de répondre au besoin d'identification précise de formes 3D. Ces méthodes ont pour objectifs de répondre aux principaux besoins en termes de compréhension et de comparaison des modèles CAO.
- En comparaison avec l'état de l'art actuel des technologies, les apports conjoints des descripteurs et méthodes proposés seront discutés.

Dans une seconde partie, un axe d'étude technique orienté SI s'intéresse au déploiement des activités de *SA* dans le cadre de processus de *RE* des données CAO en environnement *PDM* :

- La définition d'un modèle de données nommé le *Knowledge-Based Reverse Engineering model,* ou ***KBRE model***, permet de traduire les modèles CAO de sources diverses et de formats variés en objets interprétables par les méthodes de *SA*.
- Les méthodes de *SA* proposées dans la partie précédente sont spécifiquement intégrées au modèle de données proposé.
- L'apport du ***KBRE model*** en tant que brique technologique pour l'évaluation et le déploiement de méthodes de RE sur des modèles CAO contenus dans les SI d'une entreprise est discuté.

I. Proposition de méthodes et outils pour l’analyse de forme

L’étude relative à l’état de l’art de la littérature scientifique a permis de qualifier de nombreux descripteurs de forme, souvent basés sur la quantification d’attributs géométriques ou l'utilisation des technologies de *ML*, permettant la traduction des formats de modélisation des objets 3D dans un formalisme adapté à leur analyse.

Cependant, l’hypothèse de l’insuffisance de ces descripteurs pour l'analyse des typologies de formes des modèles aéronautiques a été soulevée, et plus précisément sur leurs capacités discriminatoires permettant la comparaison de formes proches, et surtout la reconnaissance et l'identification précise d'une surface aérodynamique complexe. En effet, les principales analyses couvertes par la recherche scientifique du domaine s’intéressent aux activités de *classification*, de *retrieval*, de *segmentation*, ou encore de *recognition*. Si ces activités permettent principalement de filtrer de larges ensembles de modèles 3D, aucune ne semble traiter le problème de *labeling* défini comme la capacité à identifier de manière précise et fiable une forme parfaitement identique dans un ensemble de données indexées. Cette dernière application représente néanmoins un besoin essentiel pour la compréhension des modèles CAO d’un système industriel, sans quoi l’accès efficient aux informations ciblées se base encore sur une analyse experte, chronophage et complexe pour un utilisateur humain.

Pour répondre à ces manques techniques, nous proposons un nouveau type de descripteur de forme, appelé les ***Algebraic Implicit Surface Models***. L'enjeu de cette proposition est d'explorer une nouvelle manière de caractériser une forme 3D qui soit plus adaptée à un ensemble d'activités d'analyses avancées de surfaces complexes que les descripteurs existants.

I.1. Proposition d'un descripteur de forme

Les Algebraic Implicit Surface Model

Un ***Algebraic Implicit Surface Model*** - ou ***AISM*** - est une forme de modèle mathématique qui représente une surface 3D par une équation algébrique dans un repère donné (Fryazinov et al., 2010; Seland & Dokken, 2007). Dans un espace cartésien, la surface est définie par un ***AISM*** f comme l'ensemble des points p satisfaisant la condition :

$$p(x,y,z) \mid f(x,y,z) = 0$$

Si les formes dites primitives ont des équations de surfaces connues sous la forme canonique d'un polynôme de degré 2, ce n'est pas le cas des formes plus complexes dites *freeform* qui, par ailleurs, ne peuvent généralement pas être interpolées par une équation polynomiale de degré 2.

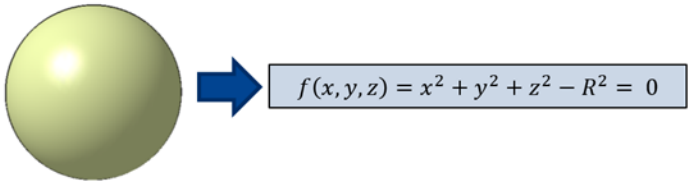


Figure 27: AISM de degré 2 d'une sphère

Pour généraliser les *AISM* à toutes les typologies de formes, nous formulons l'hypothèse que toute forme, aussi complexe soit-elle, peut être approximée par un modèle polynomiale implicite de degré $d \geq 2$. Une forme continue S est ainsi définie de la manière suivante :

$$S = \{p(x,y,z) \mid f_d(p) = 0 \pm \mathcal{E}\}, \qquad f_d : \mathbb{R}^3 \rightarrow \mathbb{R},\ f_d(x,y,z) = \sum_{d_x=0}^{d} \sum_{d_y=0}^{d} \sum_{d_z=0}^{d} \alpha_{d_x,d_y,d_z}\ x^{d_x} y^{d_y} z^{d_z}$$

$$\text{avec} \qquad d_x + d_y + d_z \leq d$$

Le terme ε correspond à la valeur du seuil empirique pour le maximum d'erreur acceptable en chaque point. Ce seuil est permis du fait de l'hypothèse d'une approximation, et non d'une interpolation qui impliquerait la prédiction stricte $f(x,y,z) = 0$ en tout point p de S.

L'erreur quadratique moyenne d'approximation d'un *AISM* sur un échantillon pc de n points de S est définie par :

$$mse_d = \frac{1}{n} \sum_{i=1}^{n} f_d(x_i, y_i, z_i)^2$$

Le degré d de l'*AISM* optimal est celui du modèle pour lequel mse_d est minimum. Des méthodes numériques stables telle la régression linéaire multiple par la méthode des moindres carrés permettent d'approximer un *AISM* sur un ensemble de points pc, uniformément échantillonné sur S, en optimisant les valeurs des paramètres α_{d_x,d_y,d_z} du modèle pour minimiser mse_d.

Proposition d'un descripteur de forme :

La proposition d'un nouveau type de descripteur de forme est définie de la manière suivante :

- Soit S une forme tridimensionnelle continue
- Soit $pc = \{\ p_i(x_i, y_i, z_i),\ i \in [1, n]\ \}$ un ensemble de n points échantillonnés sur S
- Soit f_d un *AISM* de degré $d \geq 2$ dont les nd paramètres $\alpha_j,\ j \in [1, nd]$ ont été approximés sur pc

⇨ **Le vecteur $\boldsymbol{D_d} = \begin{pmatrix} \boldsymbol{\alpha_1} \\ \boldsymbol{\alpha_2} \\ \dots \\ \boldsymbol{\alpha_{nd}} \end{pmatrix}$ est défini comme descripteur de forme de $\boldsymbol{S}$.**

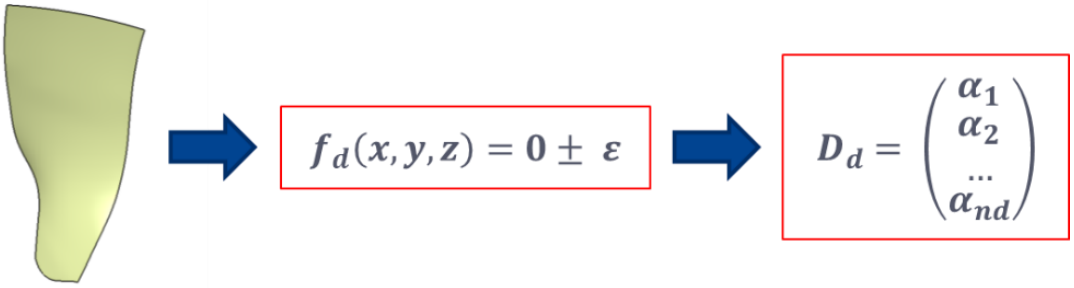


Figure 28 : extraction des paramètres d'un AISM approximé sur une surface comme descripteur de sa forme

Le choix des *AISM* comme descripteurs de forme pour l'analyse des modèles CAO repose sur plusieurs avantages répondant aux besoins spécifiques de notre étude.

Tout d'abord, en tant que modèle générique de surface, un *AISM* est en mesure d'approximer toute typologie de forme, aussi bien la surface résultante d'un solide qu'une surface complexe ouverte. La proposition des *AISM* se base sur l'hypothèse qu'un modèle polynomiale de haut degré permet d'approximer une forme avec suffisamment de précision pour que ce dernier soit unique et spécifique à la forme analysée. Autrement dit, pour deux formes S_1 et S_2 géométriquement proches mais non parfaitement identiques et leurs *AISM* approximés respectifs $D_{1,d}$ et $D_{2,d}$:

$$D_{1,d} \neq D_{2,d}$$

Par ailleurs, un *AISM* est directement approximé sur un nuage de points échantillonné sur une surface. Ce dernier doit être préalablement recalé dans une position normalisée pour assurer l'invariance du descripteur, par exemple via la méthode de *Principal Components Analysis (PCA)* (Kurita, 2019). Le descripteur de forme qui en est issu, un vecteur composé des nd paramètres approximés de l'*AISM*, est compact et rapide à calculer (entre 19 et 219 paramètres pour des modèles de degrés 3 à 9, pour un temps de calcul inférieur à la seconde selon l'implémentation qui sera présentée au Chapitre 4). La méthode proposée est donc assimilable à la catégorie des *« hand crafted » descriptors*, qui ne nécessite pas d'apprentissage machine, ni de données étiquetées.

Les *AISM* sont adaptés aux méthodes classiques de *SA*, telles que le *matching*, le *retrieval* et la *classification*, que nous étudierons dans la suite de ce chapitre. De plus, sa formalisation et plus particulièrement le calcul de l'erreur d'approximation, permettent la proposition d'une méthode de *labeling* basée sur un critère de dissimilarité précis, en vue de dépasser la notion de similarité géométrique.

Les descripteurs de formes de type ***AIgebraic Implicit Surface Models (AISM)*** ainsi définis cherchent à répondent aux exigences de l'analyse de formes dans le cadre de notre étude. Leur adaptabilité à toutes typologies de formes 3D, leur efficacité de traitement, ainsi que la couverture des principales activités de *SA,* sans oublier l'apport de l'activité de *labeling*, en font l'élément principal de la proposition scientifique de nos travaux de recherche.

Des études expérimentales présentées au chapitre suivant permettent de valider la proposition des *AISM* comme descripteur de forme. Avant cela, les méthodes d'analyses de formes 3D via l'utilisation de leurs descripteurs sont présentées dans la seconde partie de ce chapitre. Notons que l'ensemble des descripteurs manipulés lors d'une analyse spécifique sont tous du même type. Il peut donc s'agir d'*AISM* de n'importe quel degré d, tant que d est commun à chaque descripteur. Ces derniers seront génériquement nommés D_{index}.

I.2. Méthodes et outils d'analyse de forme

Pour rappel, la majorité des méthodes d'analyse de forme nécessitent les éléments suivants :

- Un descripteur de forme, ici les *AISM* présentés ci-dessus
- Une mesure de dissimilarité entre les formes, spécifique à chaque activité
- Une méthode qui décline la mesure de dissimilarité en résultats d'analyse

Parmi l'ensemble des applications d'analyses de formes existantes, les méthodes de *matching*, de *retrieval* et de *classification* sont tout d'abord présentées dans cette sous-partie.

La méthode de *labeling*, principal apport issu de la formalisation des *AISM* comme descripteurs de formes, est ensuite détaillée et critiquée.

Enfin, une méthode de *recognition* sera présentée comme illustration d'une utilisation conjointe des méthodes précédentes, appliquées à l'analyse de modèles CAO à différentes échelles.

Pour chaque méthode, nous précisions les données d'entrée, la mesure de dissimilarité utilisée ainsi que le résultat d'analyse. Les algorithmes d'implémentation de chaque méthode sont présentés dans la seconde partie de ce chapitre.

I.2.1. Matc*hing*

Le *matching* de deux formes consiste à évaluer leur proximité géométrique en calculant un score de dissimilarité entre elles. Plus le score est proche de zéro, plus les formes sont similaires.

Soient S_1 et S_2 deux formes :

(i) Les descripteurs $D_1 = \begin{pmatrix} \alpha 1_1 \\ \alpha 1_2 \\ \dots \\ \alpha 1_{nd} \end{pmatrix}$ et $D_2 = \begin{pmatrix} \alpha 2_1 \\ \alpha 2_2 \\ \dots \\ \alpha 2_{nd} \end{pmatrix}$ de S_1 et S_2 sont calculés

(ii) Le résultat de *matching*, appelé $match\_score$, est défini comme le résultat d'une fonction de distance entre D_1 et D_2 :

$$match\_score = dist\left(\begin{pmatrix} \alpha 1_1 \\ \alpha 1_2 \\ \dots \\ \alpha 1_{nd} \end{pmatrix}, \begin{pmatrix} \alpha 2_1 \\ \alpha 2_2 \\ \dots \\ \alpha 2_{nd} \end{pmatrix}\right)$$

avec $dist(a, b)$ la distance euclidienne entre les vecteurs a et b.

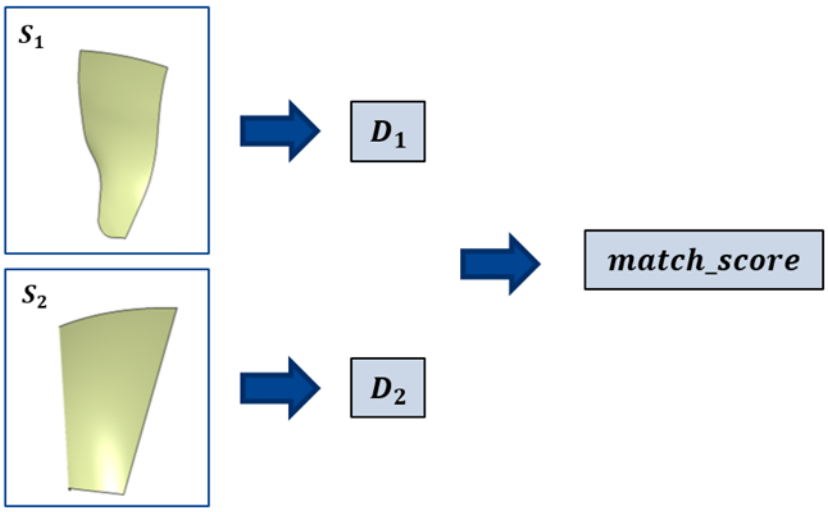


Figure 29 : matching de formes

Matcher deux formes via leurs descripteurs est simple, rapide, et peut être réalisé sans manipulation de modèles 3D, à condition que leurs descripteurs aient été préalablement calculés. Cependant, cette méthode reste une estimation et n'apporte que peu d'informations à l'utilisateur sur la dissimilarité entre les formes. En effet, mis à part une idée approximative de la proximité géométrique globale entre les formes, le *matching* ne permet pas de déterminer la nature de la divergence : déformation globale, locale, facteur d'échelle, etc. Dans les faits, nous verrons que cette méthode ne devient réellement intéressante que lorsqu'un grand nombre de formes sont à comparer, c'est-à-dire pour les activités de *retrieval*.

Si le *matching* ne concerne qu'un nombre réduit de modèles 3D, il lui sera souvent préféré *l'analyse de déviation.*

Disponible dans la plupart des logiciels de CAO, cette méthode consiste à superposer deux modèles 3D pour afficher les distances minimales entre leurs surfaces. Cette méthode manipule donc directement les modèles 3D de S_1 et S_2, via leurs représentations géométriques sous la forme de surface b-rep ou bien de maillage 3D, sans utilisation de descripteur.

Deux ensembles non ordonnés de points discrets pc_1 et pc_2 sont échantillonnés respectivement sur les modèles 3D de S_1 et S_2. À tout point $p_{1,i}$ de pc_1 est associé un point $p_{2,i}$ de pc_2, ce dernier étant le plus proche de $p_{1,i}$ de tout son ensemble. L'analyse de déviation correspond au vecteur contenant les distances euclidiennes entre les points de pc_1 et leurs homologues respectifs de pc_2, soit :

$$deviation = \begin{pmatrix} \sqrt{(p_{1,1} - p_{2,1})^2} \\ \sqrt{(p_{1,2} - p_{2,2})^2} \\ \dots \\ \sqrt{(p_{1,n} - p_{2,n})^2} \end{pmatrix}$$

Le résultat de l'analyse de déviation peut être observée graphiquement avec une échelle de couleurs comme sur la Figure 30. Elle permet de mieux apprécier les déviations locales et leur intensité, et ainsi de connaitre les zones de forte déviation. Notons cependant que l'analyse de déviation dépend du recalage (*i.e. « registration »*) des formes, qui consiste à les aligner selon leur meilleur score de déviation, donc dans leurs positions spatiales relatives qui minimisent la moyenne des valeurs de déviations. Si l'algorithme *Iterative Closest Point (ICP)* (Besl & McKay, 1992) permet d'effectuer le recalage automatiquement, nous verrons lors des expérimentations que son manque de fiabilité rend la méthode de recalage et l'analyse de déviation difficilement automatisable et plus complexe que le *matching* de formes.

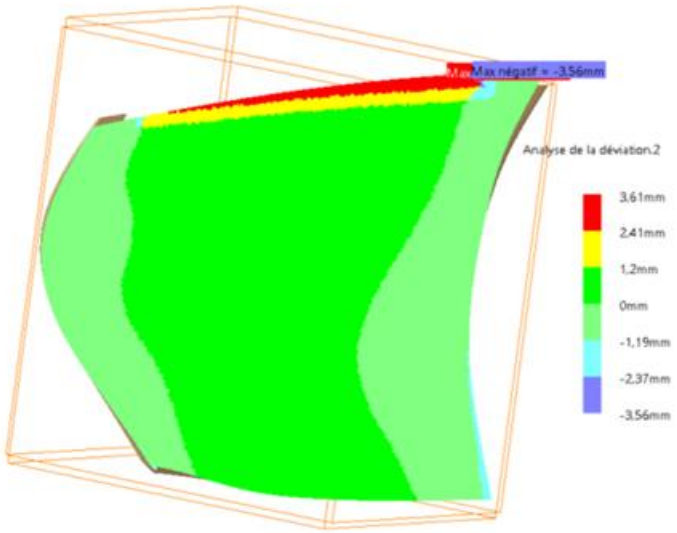


Figure 30 : analyse de déviation entre les modèles 3D de deux formes

I.2.2. Classification

La *classification* d'une forme consiste à l'associer à une classe ou catégorie, c'est-à-dire une famille de formes partageant toutes des caractères ou attributs communs.

Le choix des familles et les critères de *classification* sont des notions subjectives dépendant généralement d'un besoin spécifique ou d'un point de vue particulier. Par exemple, un utilisateur peut préférer classer un ensemble de vis selon les types de têtes, lorsqu'un autre peut les classer selon le diamètre ou la longueur. Enfin, un troisième utilisateur peut simplement souhaiter distinguer une vis d'un écrou. Si des taxonomies officielles font consensus pour les composants standards (ex. *International Classification for Standards*, ICS publiée par ISO[7]) ou encore pour les formes primitives (comme illustré par la Figure 31), la *classification* d'une base industrielle reste dépendante des vues métiers, et peut évoluer au cours du temps. Les classes existantes ainsi que les critères d'attribution d'une forme à sa classe sont dépendants de la finalité de l'application. Par conséquent, différentes catégorisations de la même forme sont possibles, toutes relatives à un certain point de vue renseigné par l'utilisateur. On considère donc le choix d'une taxonomie particulière comme un prérequis à la *classification* de formes 3D.

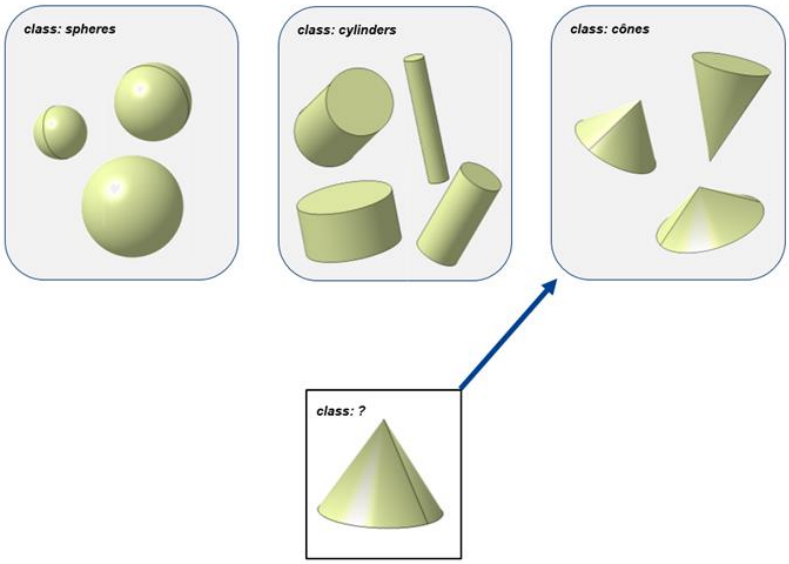


Figure 31 : classification d'une forme selon les catégories de formes primitives

Pour qu'un outil informatique soit en mesure de catégoriser des formes, ce dernier doit « apprendre » à associer leurs attributs aux classes existantes. Pour ce faire, un algorithme d'apprentissage automatisé supervisé peut être entrainé sur un ensemble de données d'apprentissage (Kubat & Kubat, 2017), c'est-à-dire un ensemble de formes ou leurs descripteurs préalablement classifiées (le plus souvent par un humain).

Parmi les nombreux algorithmes de *ML* existants, un modèle d'*ANN* appelé le *Multi-Layer Perceptron Classifier (MLP-C)* « apprend » une fonction non linéaire $f(x) : y,\ R^n \rightarrow R^o$ (n et o les dimensions de x et y respectivement) à partir d'exemples constitués des attributs $x = x_1, x_2, \dots, x_n$ et d'une « cible » y. La fonction approximée f permet au modèle de prédire la valeur y associée à tout nouveau x. L'Annexe 2 détaille plus amplement le fonctionnement des *MLP-C*.

[7] https://www.iso.org/publication/PUB100033.html

La méthode de *classification* par *MLP-C* se décline en trois étapes, (i) et (ii) illustrées par la Figure 32 et (iii) illustrée par la Figure 33 :

(i) Génération des données d'entrainement : on définit comme X_train le vecteur contenant l'ensemble des descripteurs D_i des formes S_i de l'ensemble d'entrainement, et Y_train le vecteur contenant les indices de classe associés à chaque D_i

(ii) Entrainement du *MLP-C* : l'algorithme effectue une optimisation automatique des paramètres internes du réseau pour minimiser les erreurs de prédictions $f(X\_train) = Y\_predict$ versus Y_train sur l'ensemble des données d'entrainement. Le modèle approximé peut être sauvegardé

(iii) Classification : le *MLP-C* permet maintenant de prédire la classe $class\_predict = y\_predict$ de tout nouveau descripteur D (et donc de la forme S associée) via le modèle approximée $f(D) = class\_predict$

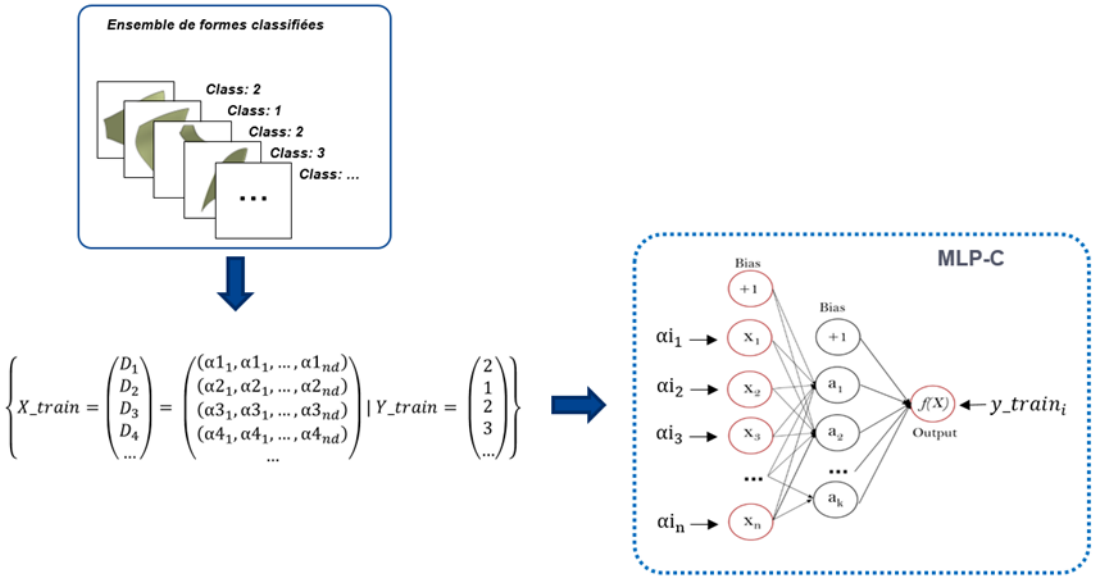


Figure 32 : entrainement d'un MLP-C pour la classification

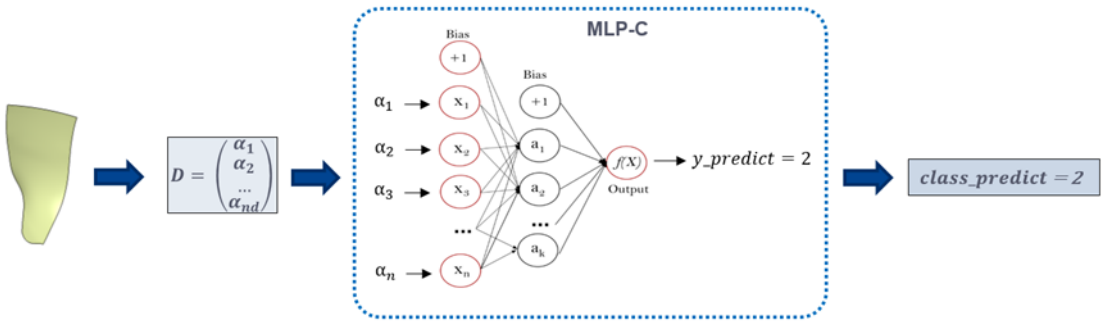


Figure 33 : prédiction d'un MLP-C entrainé pour la classification

La *classification* permet d'associer des formes selon des critères motivés par des besoins métiers et selon un point de vue subjectif. L'activité de *retrieval* associe des formes selon un critère unique, indépendant du contexte, et intrinsèque aux géométries de formes : leur proximité.

I.2.3. Retrieval

Le *retrieval* d'une forme dans un ensemble de données constitué de m formes consiste à retrouver les k formes de l'ensemble les plus « proches » de celle analysée.

Deux méthodes différentes sont présentées. La première base la notion de proximité entre les formes sur le score de dissimilarité de la méthode de *matching*, de même manière que la majorité des méthodes de *retrieval* de la littérature scientifique (Tangelder & Veltkamp, 2008). La seconde utilise les probabilités d'association d'un modèle de *ML* de manière similaire à la méthode de *classification*, à la différence que chaque forme de l'ensemble de données est unique dans sa propre classe.

Retrieval par matching

La Figure 34 illustre la méthode de *retrieval* par *matching* qui consiste à calculer les scores de *matching* entre une forme analysée S_query et chacune des formes S_i d'un ensemble. En amont de l'activité de *retrieval*, les formes S_i ont été indexées et leurs descripteurs respectifs D_i calculés.

Nb : notons que ces descripteurs peuvent être sauvegardés *pour ne pas avoir à les recalculer à chaque fois - on appelle cette étape la* ***capitali****sa****tion*** *d'un ensemble de données, dont nous parlerons plus en détail dans la seconde partie de ce chapitre.*

Un utilisateur requiert le résultat de *retrieval* de S_query dans l'ensemble $\{ S_i,\ i \in [1;\ m] \}$, soit les k formes de l'ensemble les plus similaires à S_query, avec $k < m$.

(i) D_query est calculé
(ii) Les m scores de *matching* $match\_score_i$ sont calculés entre D_query et chaque D_i
(iii) Les k scores les plus proches de 0 sont retenus. Leurs index permettent de retrouver les formes associées $S\_retrieved_j$ comme résultat de *retrieval*

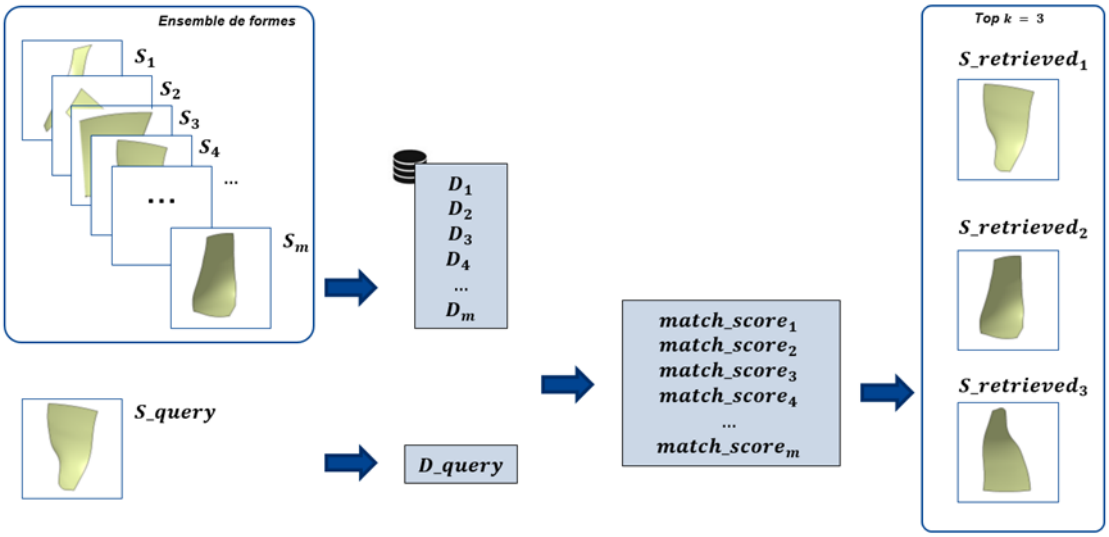


Figure 34 : retrieval de formes par la méthode de matching

En vue de compenser un potentiel manque de robustesse de la méthode de *retrieval* par *matching* présentée ci-dessus, une seconde méthode se base sur les capacités d'un *MLP-C* à identifier les descripteurs associés à une même forme malgré de légères variations de leurs paramètres. On parle de la méthode de *retrieval* par *MLP-C*.

Retrieval par MLP-C

La méthode consiste à utiliser un *MLP-C* pour la *classification* les formes d'un ensemble de données, non pas par famille de formes (*i.e.* un groupe de formes différentes réunies dans une classe), mais par index de forme (i.e. une forme est seule dans sa classe). Il y a donc autant de classes que de formes dans l'ensemble.

La méthode de *retrieval* par *MLP-C* est la suivante :

(i) *Data-augmentation* : Pour chaque S_i de l'ensemble $\{ S_i,\ i \in [1;\ m] \}$, p instances $S_{i,j}$ de son modèle 3D sont générées par transformation spatiale euclidienne aléatoire de S_i (translation et rotations rigides sans déformation ni facteur d'échelle). Il s'agit donc de la même représentation de forme, mais positionnée différemment dans l'espace. La Figure 35 illustre cette étape de *data augmentation*.

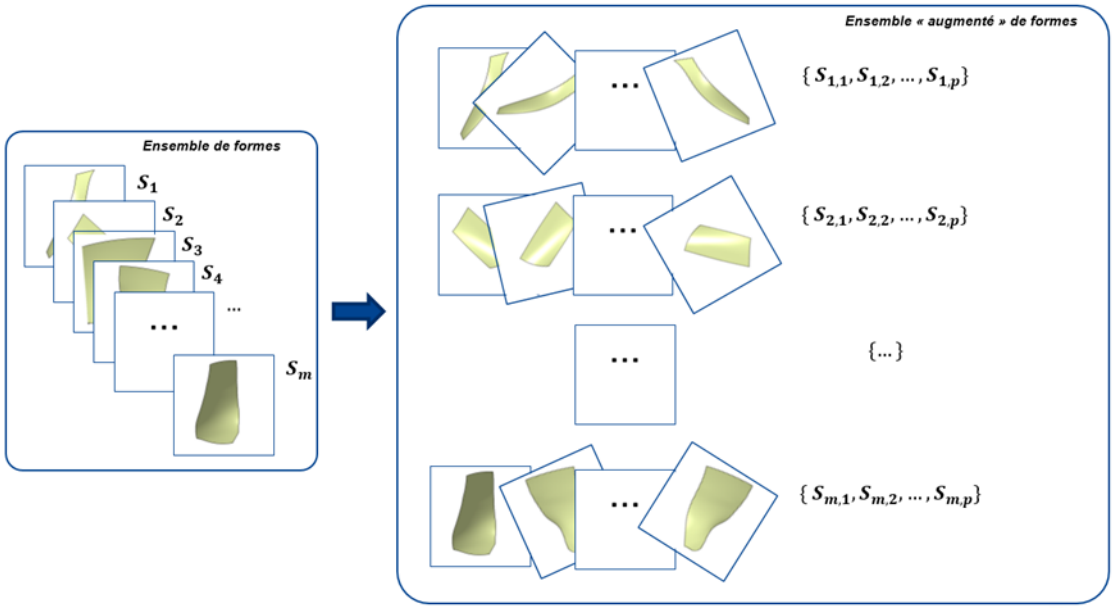


Figure 35 : data-augmentation des modèles 3D représentatifs des formes d'un ensemble

(ii) Génération des données d'entrainement : X_train est le vecteur contenant l'ensemble des descripteurs $D_{i,j}$ des formes $S_{i,j}$ de l'ensemble "augmenté" des données d'entrainement, et Y_train le vecteur contenant les index de la forme d'origine associée à chaque descripteur (toute instance $S_{i,j}$ a pour index $y_i = i$)

(iii) Entrainement du *MLP-C* : l'algorithme effectue une optimisation automatique des paramètres internes du réseau pour minimiser les erreurs de prédictions $f(X\_train) = Y\_predict$ versus Y_train sur l'ensemble des données d'entrainement. Le modèle approximé peut être sauvegardé. La Figure 36 ci-dessous illustre les étapes (ii) et (iii).

(iv) *Retrieval* : le *MLP-C* permet maintenant de prédire la classe (ici, l'index de forme) de tout nouveau descripteur D_query via la fonction approximée $f(D\_query) = y\_predict$. De plus, un *MLP-C* permet de prédire $proba_i$, la probabilité d'association de D_query à la classe i, et ce pour chacune des classes connues (avec $\sum_{i=1}^{m} proba_i = 1$). Le top k des $proba_i$ les plus élevées est retenu et les S_i associées sont reconnues comme résultat de *retrieval*, comme illustré dans la Figure 37.

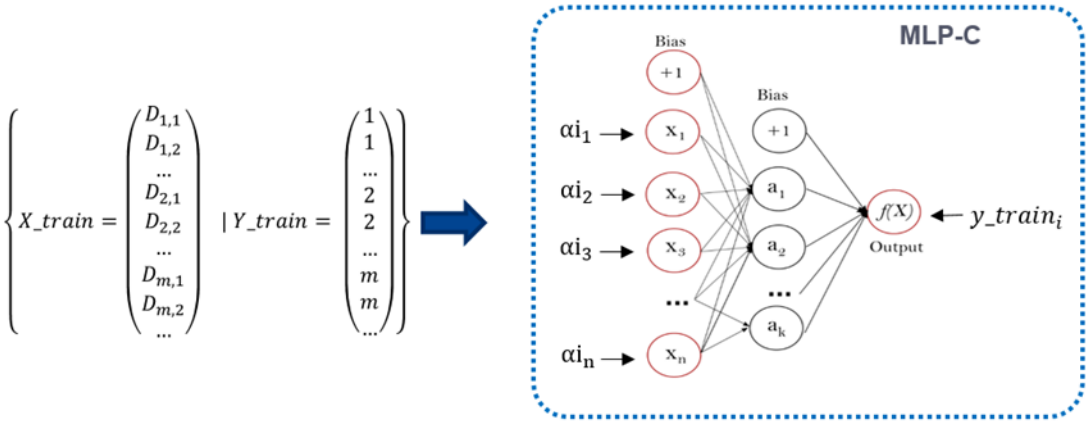


Figure 36 : entrainement d'un MLP-C pour le retrieval

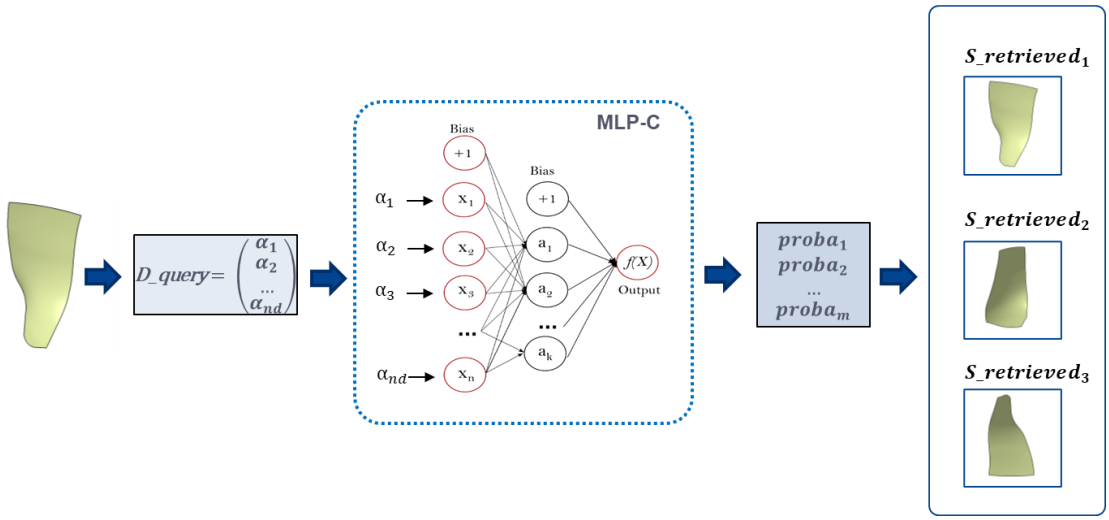


Figure 37 : prédiction d'un MLP-C entrainé pour le retrieval

Si la méthode de *retrieval* par *matching* présente l'avantage d'être simple, très rapide, et adaptée aux très larges ensembles, nous verrons dans le chapitre suivant qu'elle manque de robustesse. Les étapes chronophages de *data-augmentation* et d'apprentissage de la méthode de *retrieval* par *MLP-C* requièrent d'importantes ressources informatiques. De plus, l'utilisation d'un *MLP-C* rend la méthode moins dynamique puisque le réseau doit être entrainé de nouveau lorsqu'une nouvelle forme est ajoutée à l'ensemble. Enfin, la convergence du modèle dépend de la qualité des données d'apprentissage. Cela signifie que si deux descripteurs identiques sont appris sous différents index, ou si les descripteurs représentatifs d'une même forme sont très différents, l'apprentissage du *MLP-C* est plus complexe et ses prédictions moins fiables. Finalement, notons que le *retrieval* est une manière « brute » de retrouver des formes ressemblantes, et qu'il est difficile d'évaluer les résultats sans appréciation de l'utilisateur. Le score de *matching* entre descripteurs est insuffisamment précis pour apprécier la proximité entre les formes, en plus d'être sensible aux imperfections des descripteurs. La prédiction d'un *MLP-C* ne fournit quant à elle aucune information sur le niveau de proximité entre les formes, puisque la somme des prédictions sur toutes les classes du modèle est toujours égale à 1. Autrement dit, il n'est pour le moment pas possible de dire si les formes retrouvées sont des doublons parfaits, ou seulement des formes proches. Face à ces constats, une méthode permettant d'évaluer très précisément la similarité entre deux formes est nécessaire. Cette méthode s'appelle le *labeling*.

I.2.4. Labeling

Le *labeling* d'une forme consiste à l'identifier en retrouvant sa référence ou autre métadonnée. Il s'agit donc de déterminer s'il existe un modèle CAO indexé dont la forme est parfaitement identique à celle analysée.

Pour cela, une méthode qui prend avantage de la nature des *AISM* en tant que modèle algébrique de surface et de la possibilité de calculer l'erreur de prédiction d'un *AISM* sur une forme a été mise au point :

- Soit $pc0$ un ensemble de $n0$ points $p0_i(x0_i, y0_i, z0_i)$ échantillonnés sur le modèle 3D d'une forme $S0$.
- Soit f_d un *AISM* approximé sur $pc0$. L'erreur quadratique moyenne d'approximation de f_d sur $pc0$ est :

$$\text{mse0} = \frac{1}{\text{n0}} \sum_{i=1}^{n0} f_d(x0_i, y0_i, z0_i)^2$$

- Soit pc_query un ensemble de n points $p_i(x_i, y_i, z_i)$ échantillonnés sur une surface S_query. De même que pour $pc0$, on note mse l'erreur quadratique moyenne des prédictions de f_d sur pc_query.

⇨ **Le *fitting* de f_d sur un ensemble de n points pc_query échantillonnés sur le modèle 3D de S_query consiste à optimiser la transformation euclidienne appliquée à pc_query minimisant l'erreur de prédiction de f_d sur pc_query.**

Autrement dit, les paramètres de transformation rigide $(\theta_x, \theta_y, \theta_z, t_x, t_y, t_z)$ formant la matrice de transformation M sont optimisés pour minimiser le « cout » de *fitting* de f_d définit par :

$$fit\_cost = \frac{1}{n} \sum_{i=1}^{n} f_d(x_{t,j}, y_{t,j}, z_{t,j})^2$$

avec

$$\begin{pmatrix} x_t \\ y_t \\ z_t \\ 1 \end{pmatrix} = M.p = \begin{pmatrix} \theta_{xx} & \theta_{xy} & \theta_{xz} & t_x \\ \theta_{yx} & \theta_{yy} & \theta_{yz} & t_y \\ \theta_{zx} & \theta_{zy} & \theta_{zz} & t_z \\ 0 & 0 & 0 & 1 \end{pmatrix} . \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix}$$

La Figure 38 illustre les étapes de transformations successives lors du *fitting*.

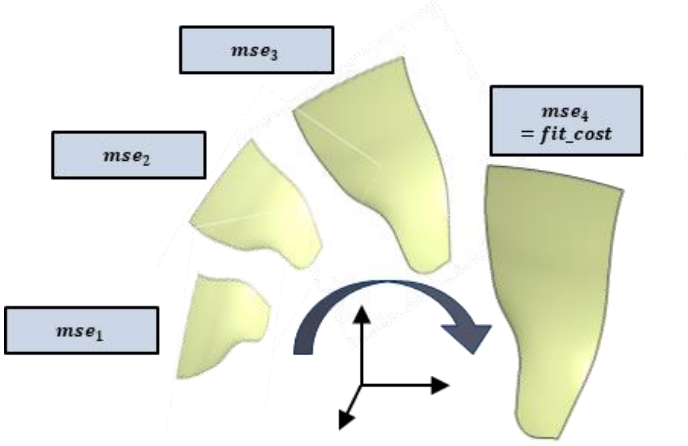


Figure 38 : transformations successives d'une forme au cours du fitting d'un AISM

La méthode de *fitting* de f_d sur S_query revient à recaler son modèle 3D dans la même position spatiale que le modèle 3D de $S0$ lors de l'approximation de f_d. Cette méthode compense la non-invariance des *AISM* et permet l'utilisation du score de *fitting* (i.e. l'erreur de prédiction d'un *AISM* sur un échantillon de points) comme mesure de dissimilarité entre la forme représentée par l'*AISM* et la forme analysée.

L'hypothèse suivante est formulée pour l'exploitation du score de *fitting* comme mesure de dissimilarité pour la méthode de *labeling* :

⇨ **Si $(fit\_cost - \mathbf{mse0})^2 < \boldsymbol{\varepsilon}$, alors S_query et $S0$ sont des formes strictement identiques et parfaitement recalées**

Le résultat final fit_cost de la méthode de fitting de l'*AISM* f_d représentatif d'une forme indexée $S0$ sur la forme analysée S_query détermine, par son écart avec $mse0$, si f_d peut être accepté comme un modèle représentatif de S_query. Dans ce cas, on considère par extension que S_query est parfaitement identique à $S0$, et que les surfaces ont donc le même label.

La méthode de *labeling* de S_query consiste à déterminer si, parmi un sous-ensemble de $n\_candidats$ formes $\{S\_candidat_1, S\_candidat_2, \dots, S\_candidat_{n\_candidats}\}$, il en existe une ou plusieurs parfaitement identique à S_query. Pour cela, le *fitting* des *AISM* respectifs des candidats est réalisé sur un ensemble de points échantillonnés sur le modèle 3D de S_query dans la cadre de la méthode de *labeling* illustrée par la Figure 39 :

(i) Un ensemble de points pc_query est échantillonné à partir d'un modèle 3D de la forme S_query

(ii) Un nombre $n\_candidats$ de formes candidates pour le *labeling* sont proposées. Pour chacune, un *AISM* $f_{d,i}$ et son erreur d'approximation $mse0_i$ sont disponibles

(iii) Le *fitting* de chaque $f_{d,i}$ est réalisé sur pc_query, et les résultats fit_cost_i sont retournés

(iv) Si $(mse0_i - fit\_cost_i)^2$ est inférieur à un seuil expérimental, S_query est considérée comme identique à $S\_candidat_i$, et $S_{label} = S\_candidat_i$

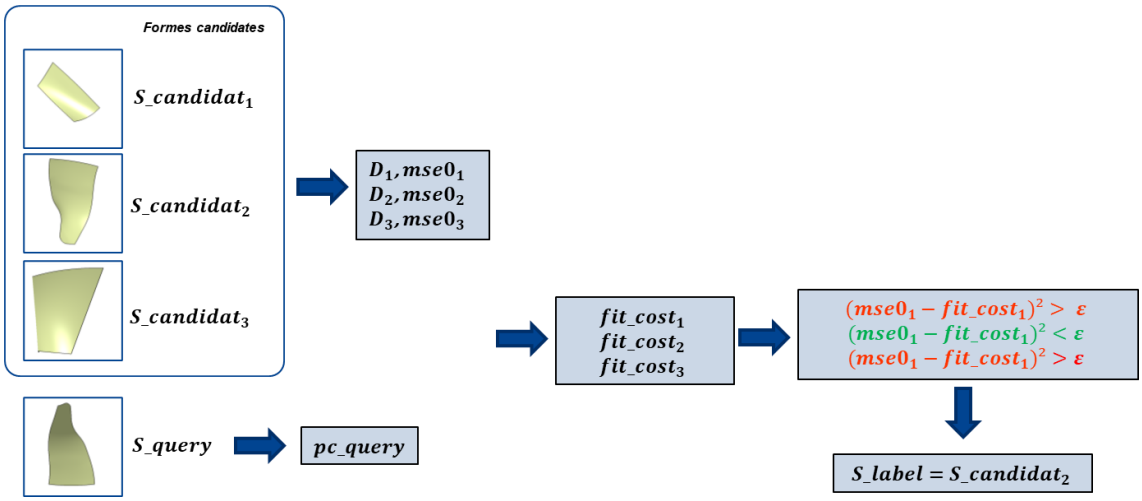


Figure 39 : labeling de forme

S'il était possible de considérer le meilleur score de *retrieval* comme label, l'activité de *labeling* prétend à un plus haut niveau de précision et de certitude dans l'identification d'une forme. Le *fitting* de leurs *AISM* sur une forme retourne une mesure de dissimilarité permettant de déterminer si un ou plusieurs candidats sont strictement identiques à la forme étudiée, s'ils sont simplement proches, ou encore si aucun des candidats n'est valide.

Par rapport au score de *matching*, la méthode de *fitting* est couteuse en puissance de calcul, rendant difficilement envisageable son utilisation à grande échelle. On considère donc l'activité de *retrieval* comme un moyen de sélectionner des candidats appropriés, soit d'agir comme un premier filtre pour limiter le nombre de *fitting* à réaliser.

Comme l'algorithme *ICP* (Besl & McKay, 1992), le *fitting* d'un *AISM* de $S0$ sur S_query agis comme un recalage de S_query sur $S0$. Cependant, contrairement à l'algorithme *ICP* pour lequel les modèles 3D de chaque candidat doivent être chargés, seul le modèle 3D de S_query est manipulé pour le *fitting* des *AISM* candidats.

Jusqu'ici, nous avons considérés les formes analysées comme des surfaces unitaires. Cependant, un modèle CAO est généralement composé de plusieurs surfaces unitaires connectées entre elles dans des relations de connexités. Maintenant qu'il nous est possible de reconnaitre précisément chacune, l'analyse qui suit s'intéresse à des assemblages de surfaces formant ensemble un modèle ou une sous-partie de modèle CAO, appelés *form-feature*.

I.2.5. Form-Feature recognition

Un *feature 3D*, ou *form-feature* est un ensemble d'éléments géométriques connectés auquel est associé des attributs et connaissances métier (Fontana et al., 1999; J. J. Shah & Mäntylä, 1995). L'activité de *feature recognition* (ou *feature détection*) consiste à identifier la présence ou non d'un *form-feature* particulier comme composante d'un modèle CAO. Par extension, le *feature retrieval* consiste à identifier les modèles d'un ensemble de données dans lesquels on retrouve le *form-feature* d'intérêt. Concrètement, détecter un *form-feature* consiste à identifier la présence de $n \geq 1$ surfaces unitaires connues et indexées, reliées entre elles par des relations de connexité spécifiques.

Les relations de connexités entre les $n > 1$ surfaces unitaires S_i d'un groupe topologique sont représentées par une matrice de dimension $(n * n)$, appelée matrice d'adjacence (AM pour *adjacency matrix*), comme présentée dans l'état de l'art.

AM est une matrice diagonale ($a_{i,j} = a_{j,i}$) avec $a_{i,j} = 1$ si S_i et S_j sont connexes, $a_{i,j} = 0$ sinon :

$$AM = \begin{pmatrix} \#\# & \#1 & \dots & \#n \\ \#1 & \# & a_{i,j} & a_{1,n} \\ \dots & a_{j,i} & \# & a_{i,n} \\ \#n & a_{n,1} & a_{n,i} & \# \end{pmatrix}$$

La méthode de détection d'un *form-feature* dans un modèle CAO P constitué de np surfaces consiste à identifier les composantes de P identiques à celles du *form-feature.* Les premières étapes illustrées par la Figure 40 consistent à calculer le descripteur topologique et les descripteurs de formes de *form-feature :*

(i) La Matrice d'adjacence $AM\_feature$ de *form-feature* est calculée
(ii) Un *AISM* D_i est calculé pour chaque surface S_i de *form-feature*

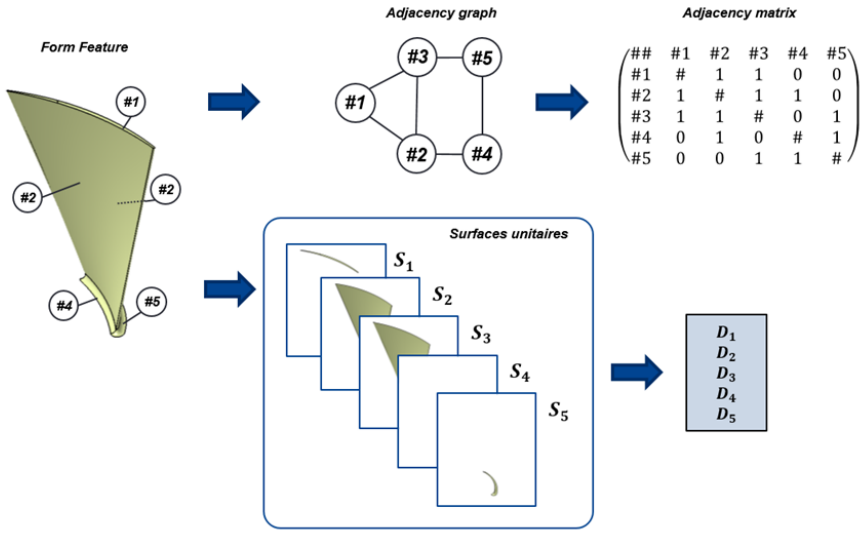


Figure 40 : calcul du descripteur topologique (la matrice d'adjacence) du form-feature et des descripteurs de formes de chaque surface

(iii) AM_part, la matrice d'adjacence de P est calculée

(iv) Les surfaces capitalisées $\{S_1, \dots, S_n\}$ de *form-feature* servent de candidats au *labeling* des Sp_i de P. Certaines surfaces de P sont donc identifiées, d'autres non, comme l'illustre la Figure 41 ci-dessous

(v) Seule les lignes et colonnes de AM_part correspondant aux surfaces labélisées $(Sp_i = S_j)$ sont conservées. Les structures équivalentes à $AM\_feature$ sont détectées dans les versions permutées de AM_part

(vi) Tout groupe topologique de P formé de surfaces labélisées $Sp_i \in \{S_1, \dots, S_n\}$ et dont une version permutée de la matrice d'adjacence est identique à $AM\_feature$ est ainsi reconnue comme identique à *form-feature*, comme illustré dans la Figure 42

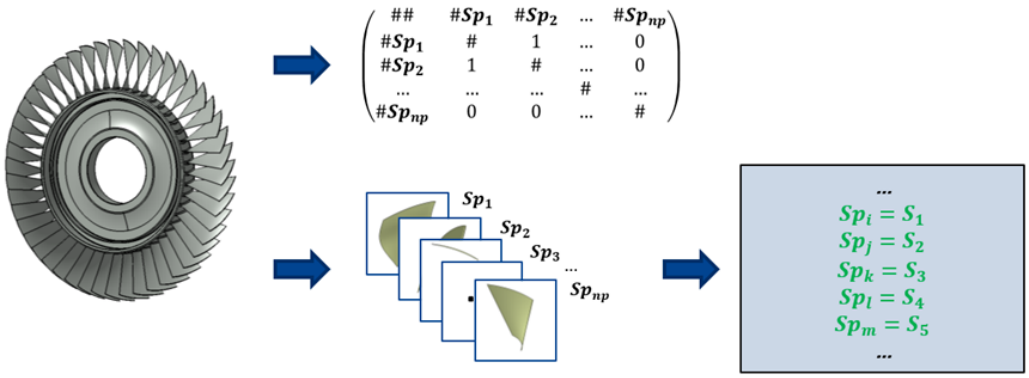


Figure 41 : labeling des surfaces unitaires d'un modèle CAO

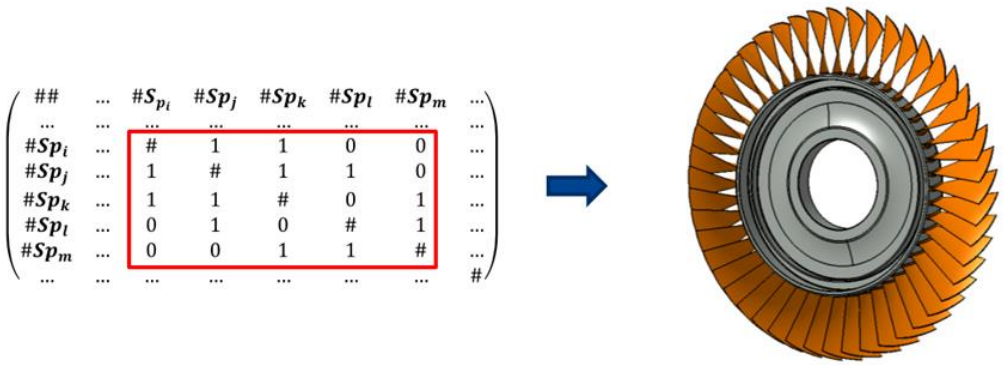


Figure 42 : identification des différents groupes topologiques d'un modèle CAO identiques à un form-feature analysé

L'analyse de *form-feature* est basée sur les capacités de *matching*, de *retrieval* et de *labeling* de surfaces unitaires. L'utilisation conjointe de ces méthodes avec l'analyse de la topologie d'un modèle complet, via sa matrice d'adjacence, permet de reconnaitre des sous assemblages locaux composés de surfaces identifiées.

Cette dernière activité permet une analyse avancée de modèles CAO et de leurs structures. Il est ainsi possible de comparer des modèles CAO entre eux à différentes échelles, sans se limiter à l'analyse de leurs formes globales. Par exemple, un modèle CAO multi-instancié dans un autre modèle est reconnu (et ses informations fonctionnelles avec), des modèles partageant des *form-feature* en commun reliés, ou encore les modifications entre deux évolutions d'un modèle identifiées. En résumé, l'analyse de *form-feature* est l'application des méthodes de *SA* à l'analyse de modèles CAO.

I.3. Bilan des méthodes d'analyse de forme proposées

Cette première partie du chapitre sur la proposition scientifique de nos travaux expose les méthodes et les outils développés pour l'analyse de forme 3D au moyen d'un descripteur de forme compact et optimisé, les ***Algebraic Implicit Surface Models***.

L'introduction des *AISM* vise à améliorer la représentation et l'analyse des surfaces complexes présentes dans les modèles CAO des pièces aéronautiques. Ces descripteurs permettent de réaliser diverses activités d'analyse, à savoir le *matching*, la *classification*, le *retrieval*, le *labeling* et la *recognition* de formes, comme illustré dans la Figure 43.

La proposition d'une méthode de *retrieval* basée sur un *Multi-Layer Perceptron Classifier* offre l'occasion d'évaluer les avantages et limites des modèles complexes de *ML* par rapport à la méthode classique basée sur une analyse numérique plus simple, le *matching*.

Le développement d'une méthode de *fitting* des *AISM* introduit une nouvelle mesure de dissimilarité qui permet l'identification précise de formes complexes, élargissant ainsi la portée de la méthode de *labeling* qui était initialement limitée aux formes quadratiques.

Ensemble, ces activités d'analyse sont utiles pour recréer et retrouver diverses données et informations à partir de modèles CAO, afin de déduire des connaissances pertinentes pour les activités d'ingénierie en plus de maintenir le niveau sémantique des modèles de définition des produits.

L'enjeu est maintenant de s'intéresser à l'exploitation de ces activités d'analyse de forme dans le cadre de processus de *RE* des modèles CAO en environnement *PDM*. Il s'agit donc de répondre aux objectifs autours de la gestion de l'information relative aux modèles CAO.

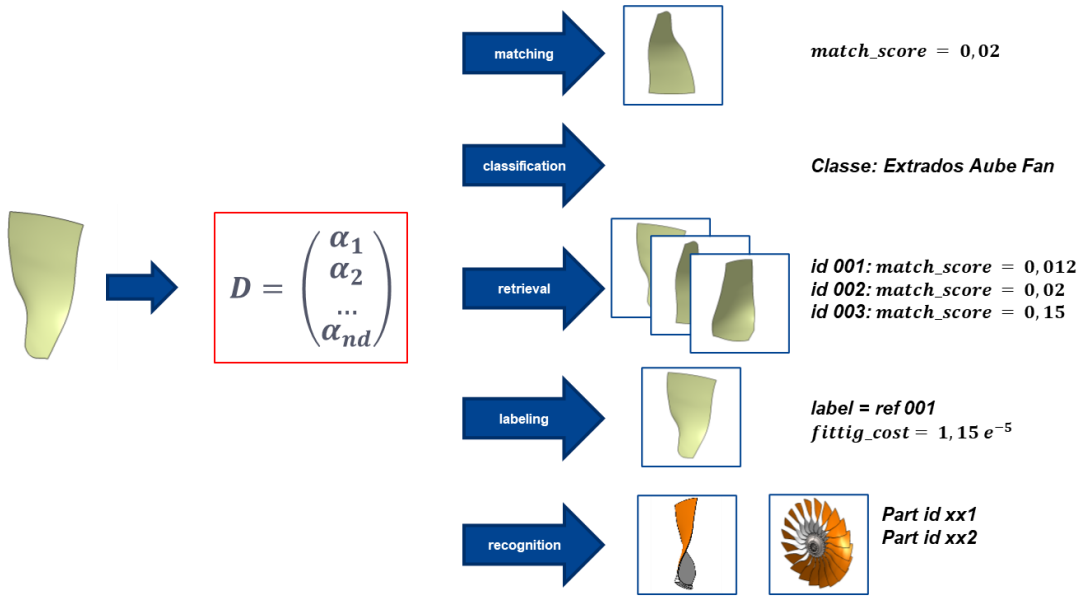


Figure 43 : analyses de formes réalisables avec les AISM

II. Proposition du Knowledge Based Reverse Engineering Model

Dans la partie précédente, les méthodes et outils permettant l'analyse de forme 3D à l'aide de descripteurs de forme ont été détaillés afin de répondre aux objectifs de création et d'extraction de connaissances sur des modèles 3D. Cependant, à ce stade de la proposition, certains éléments restent manquants pour en permettre l'exploitation dans les processus de *RE* pour la gestion des données CAO en environnement *PDM*.

En effet, la proposition doit permettre la mise en perspective du besoin avec les moyens techniques disponibles afin de le traduire en processus de *RE*. Ces processus incluent la communication avec les SI et le croisement de leurs données, comme « comparer a avec b » ; la définition d'un cadre aux analyses, tel que « chercher a dans b » ; et la contextualisation des informations pour les transformer en connaissances exploitables, comme « a de A correspond à b de B ». De plus, en raison de la complexité des activités de *SA* et leurs interdépendances, il est nécessaire de structurer le traitement de l'information au sein des méthodes. Cela englobe des aspects tels que le calcul, la sauvegarde et la récupération des descripteurs, ou encore l'augmentation des données et l'apprentissages des *ANNs*.

L'objectif de cette partie est donc de répondre aux objectifs en termes de création, de récupération, d'identification, d'organisation, de stockage, de diffusion, et surtout de mise en œuvre opérationnelle des données et connaissances relatives aux modèles 3D.

Pour permettre le déploiement de processus de *RE* pour le maintien de l'intégrité des modèles CAO d'une entreprise face aux problématiques de continuité numérique, une nouvelle brique des Systèmes d'information est proposée.

Le **Knowledge-Based Reverse-Engineering *model* (*KBRE model*)** est un modèle de données proposé comme un outil technique pour l'expérimentation autours de méthodes de *SA*.

Basé sur les principaux formats de modélisation 3D, les objets du modèle offrent des capacités de lecture, de manipulation et de modifications des modèles 3D.

Agrémenté d'une classe générique représentative du concept de forme 3D, indépendante de toute méthode et format de modélisation, le *KBRE model* intègre des fonctionnalités avancées de traitement et d'analyse des modèles CAO ainsi que des bases de données qui les renferment.

Une classe d'objets conçue pour représenter des ensembles de modèles 3D permet l'interopérabilité du KBRE model avec les différents éléments d'un SI industriel, en proposant une structure flexible et évolutive pour la gestion des informations issues de sources variées et de formats divers. Par ailleurs, son association à une *Knowledge Base (KB)* permet la capture et la capitalisation de connaissances expertes relatives aux produits. Ces connaissances pourront alors être utilisées dans la cadre de processus de *RE* des modèles CAO de l'entreprise.

La Figure 44 illustre les relations entretenues entre les différentes ressources mises en œuvre lors d'un processus de *RE* intégrant le *KBRE model*. Un utilisateur émet une requête à laquelle le *KBRE model* retourne diverses informations, dont l'interprétation résulte en un apport de connaissances utiles au processus de *RE*. Pour générer ces informations, des activités de *SA* intégrées au *KBRE model* sont réalisées sur des modèles CAO issus du SI. Le lien entre la donnée analysée et son emplacement dans les SI est maintenue par sa métadonnée. Les activités de *SA*, basées sur des techniques avancées de géométrie computationnelle, d'analyse statistique et d'apprentissage automatique, sont rendues possibles par un échange d'informations avec une base de connaissances.

Ainsi, le *KBRE model* est une nouvelle ressource au sein du SI d'une entreprise qui apporte à un utilisateur de nouvelles connaissances sur ses bases de modèles CAO pour la réalisation d'activités complexes dans le cadre de processus de *RE*.

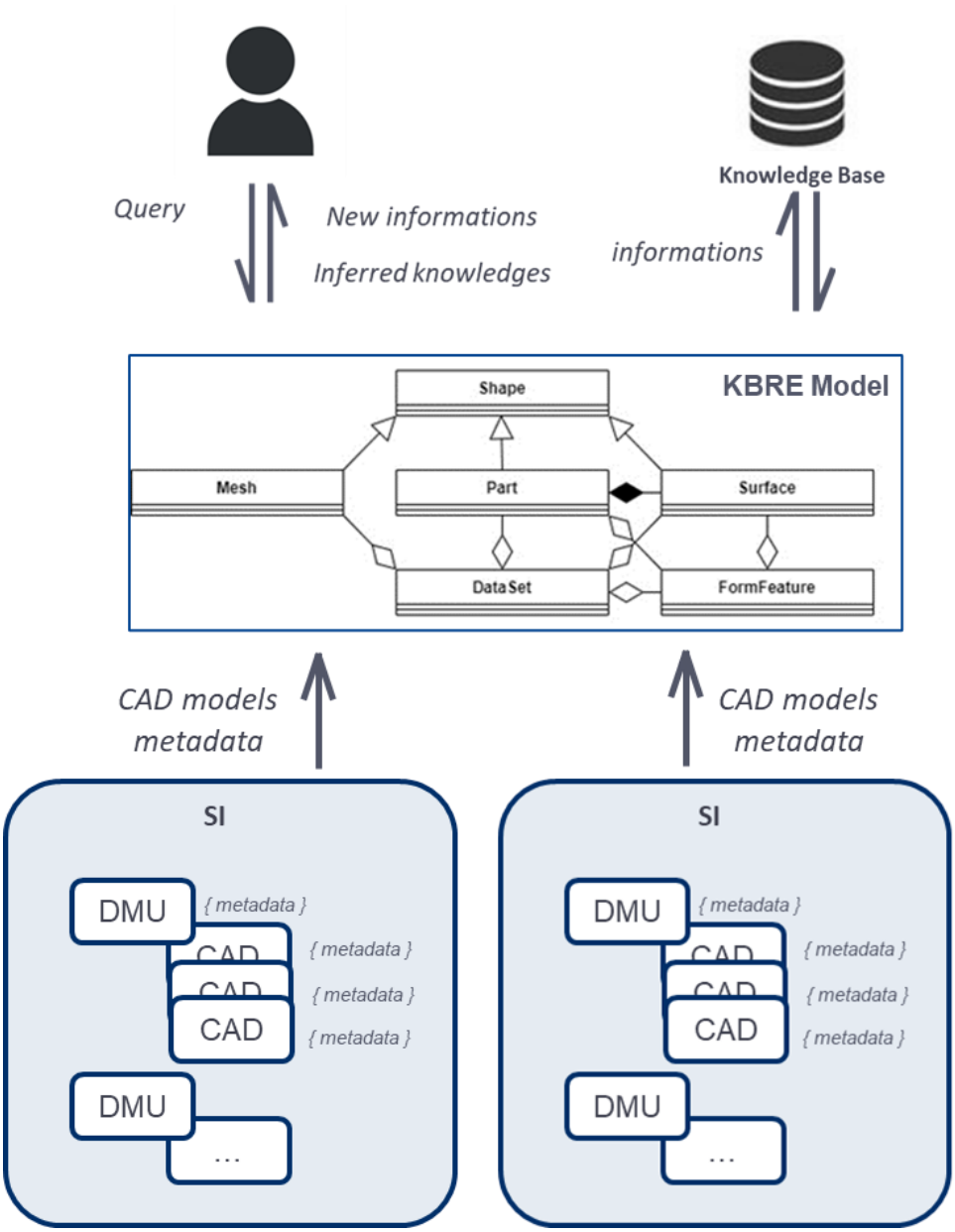


Figure 44 : liens entretenus par les éléments impliqués dans un processus de RE

II.1. Objectifs et précisions préalables

Un modèle de données est une description abstraite de la structure, des associations, des relations ainsi que des contraintes relatives aux données d'un système d'information (Elmasri & Navathe, 2011). Plus précisément, un modèle physique de données décrit les fichiers et types de données spécifiques qui seront traités dans le système présenté. Dans un modèle de données orienté objet, les données sont représentées par des entités spécialisées selon leur type que l'on appelle des objets. Le type d'un objet définit l'information qu'il contient et ses propriétés sous la forme d'attributs, ainsi que les capacités d'actions de l'objet et sur l'objet, soit ses comportements, sous la forme de fonctions et méthodes. Les interactions entre objets et la dynamique de création de l'information est aussi définie par le modèle (Elmasri & Navathe, 2011).

La proposition du *KBRE model* intègre la définition d'objets représentatifs des modèles 3D et des bases de données qu'ils composent pour catégoriser et différentier les activités d'analyse applicables selon leurs formats, structures, ou encore selon divers critères sémantiques. La contextualisation des analyses et l'interprétation des informations qui en sont extraites permet alors à un utilisateur d'en déduire diverses connaissances utiles aux activités avales du processus de *RE*.

Avant de présenter plus en détail la structure du *KBRE model*, il semble nécessaire d'apporter quelques précisions.

Les objets du modèle ne sont pas des formats géométriques de données 3D. Ces derniers ne peuvent être directement visualisés ou modifiés dans un *model*eur CAO.

Le modèle de données est indépendant de tout SI et format 3D propriétaire. La structure des objets diffère de celles des SI et des différents formats de modèles CAO. Ce sont des représentations « miroirs » des modèles 3D, optimisés pour l'analyse de leurs formes et la gestion des informations qui en sont issues. Le modèle de données vise l'interopérabilité entre différents SI par la définition d'objets communs et un échange dynamique de données et d'informations.

Inspiré des systèmes *KBE*, le modèle de données est basé sur les technologies de programmation orientée objet (POO), d'intelligence artificielle (IA) et de modélisation 3D-CAO (Chapman & Pinfold, 1999). Les systèmes *KBE* intègrent des fonctionnalités avancées de traitement et d'analyse des modèles CAO par l'intégration de méthodes pour la réalisation de tâches complexes d'ingénierie (Milton, 2008). Le lien avec une *Knowledge Base* permet de capturer les informations et connaissances expertes liées au produit et de les stocker dans une base de connaissance afin d'en faciliter l'utilisation dans la cadre d'activités d'ingénierie (Bruneau, 2016), ou de *RE* dans notre cas.

Enfin, le modèle n'est pas « fini », dans le sens ou des méthodes et objets peuvent être ajoutés/retirées sans affecter les autres fonctions. Dans le cadre de notre proposition, nous ne présentons que les méthodes pour les activités de *SA* étudiées. Cependant, d'autres méthodes/outils peuvent êtres implémentés dans une logique d'unification et de complétude des outils.

II.2. Structure du *KBRE model*

Avant de détailler chaque objet du *KBRE model* et les méthodes, les types d'objets ainsi que les relations qu'ils entretiennent sont présentés ci-après.

Un objet de type ***Shape*** incarne le concept de forme 3D, indépendant de toute interprétation de format de donnée, de méthode de modélisation géométrique et de position spatiale.

Un objet de type ***Part***, basé sur le modèle de donnée STEP, est la représentation d'un modèle CAO au format *b-rep*, c'est-à-dire un ensemble de surfaces continues entretenant des relations de connexités. Il permet l'analyse à l'échelle du modèle CAO dans son ensemble.

Un objet de type ***Surface*** est la représentation d'une surface continue unitaire d'un modèle CAO *b-rep*. Il permet l'analyse à l'échelle de l'entité topologique locale d'un modèle CAO.

Un objet de type ***FormFeature*** est la représentation d'une composante d'un modèle CAO, soit d'un groupe topologique de surfaces unitaires connexes du modèle continu. Il permet l'analyse à l'échelle d'une partie locale du modèle CAO.

Un objet de type ***Mesh***, basé sur le modèle de donnée STL, est la représentation d'un modèle 3D au format de maillage triangulaire. Il permet l'analyse et la manipulation de formes représentées par des éléments géométriques discrets. Notons qu'un objet ***Mesh*** contient la représentation géométrique d'une forme globale (*i.e.* sans découpage topologique) pouvant être celle d'un objet ***Part***, ***Surface*** ou **FormFeature**.

Un objet de type ***DataSet*** représente un ensemble de modèles 3D dans leurs descriptions globales et locales. C'est un groupement organisé d'objets des classes précédentes. Un objet ***DataSet*** agit sur tous les objets qu'il regroupe, et permet donc l'analyse à l'échelle d'ensembles de modèles 3D.

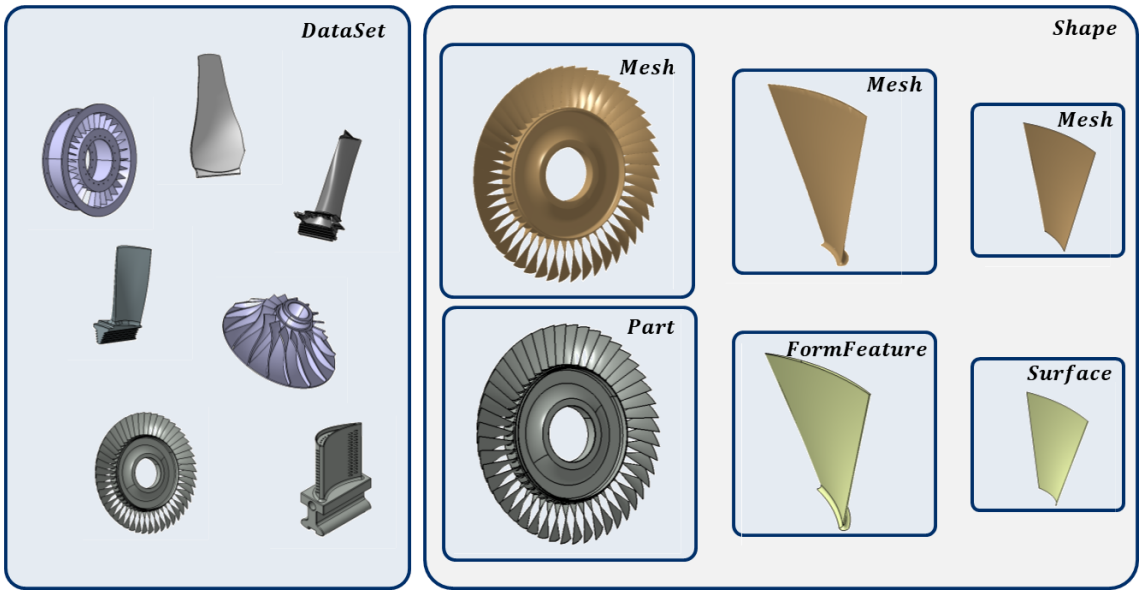


Figure 45 : les classes du KBRE model

Les différents objets du *KBRE model* sont liés entre eux par des relations d'agrégation, de composition et d'héritage :

Un objet *DataSet* est défini par un utilisateur pour agréger un ensemble d'objets *Part*, *Mesh*, *Surface* et *FormFeature*. Un *DataSet* peut représenter n'importe quel ensemble de modèles 3D, comme une *DMU*, un certain niveau d'arborescence du *PDM*, une librairie de maillages 3D, un catalogue de *form-features* 3D ou de *templates* métiers, etc. C'est un regroupement cohérent de modèles 3D selon un point de vue utilisateur pour un besoin métier. En offrant des possibilités d'analyses ciblées et la création d'informations relatives à des ensembles divers de modèles 3D, le *DataSet* permet de décliner les fonctionnalités d'analyses du modèle de données en applications métier spécifiques. Notons aussi que l'agrégation implique qu'un *DataSet* peut regrouper autant d'objets souhaités, et qu'un objet peut appartenir à plusieurs objets de type *DataSet*.

Un objet *Part* est directement relié à un modèle CAO b-rep. Il est composé de n objets de type *Surface* dont les relations de connexité définissent la topologie du modèle CAO. Les objets *Part* structurent les informations connues sur des modèle CAO continus et permettent l'analyse de leurs formes globales, ainsi que de chacune des surfaces unitaires qui les composent via les objets de type *Surface*. La création d'un objet *Part* depuis un modèle CAO a pour conséquence la création automatique de n objets *Surface*. Si un objet *Part* peut exister indépendamment de tout *DataSet* par « traduction » d'un modèle CAO isolé, ce n'est pas le cas des objets *Surface* qui n'existent que comme composante d'une *Part.* La suppression de ce dernier entraine la suppression de ses composantes.

Un objet de type *FormFeature* est créé manuellement depuis une instance d'objet *Part* par un utilisateur souhaitant en analyser une composante uniquement. Pour ce faire, l'utilisateur spécifie les objets *Surface* composants qui vont être agrégés dans un objet *FormFeature* nouvellement créé. Ce dernier est à son tour agrégé dans l'objet *Part* d'origine.

Un objet *Mesh* est directement relié à un modèle 3D au format de maillage triangulaire. Ses attributs principaux sont générés par traduction des éléments géométriques qui le composent, à savoir les faces, cotés et sommets du maillage.

Enfin, *Shape* est une classe abstraite, ce qui signifie que ses objets ne sont jamais instanciés. Cette classe permet de définir les méthodes communes aux objets de type *Mesh*, *Part*, *Surface* et *FormFeature* dont ils héritent.

La Figure 46 illustre les relations entre les classes du *KBRE model* dans un diagramme de classes *Unified Modeling Language* (*UML*).

La partie suivante décrit la définition de objets du *KBRE model*, c'est-à-dire les attributs et méthodes de chacun.

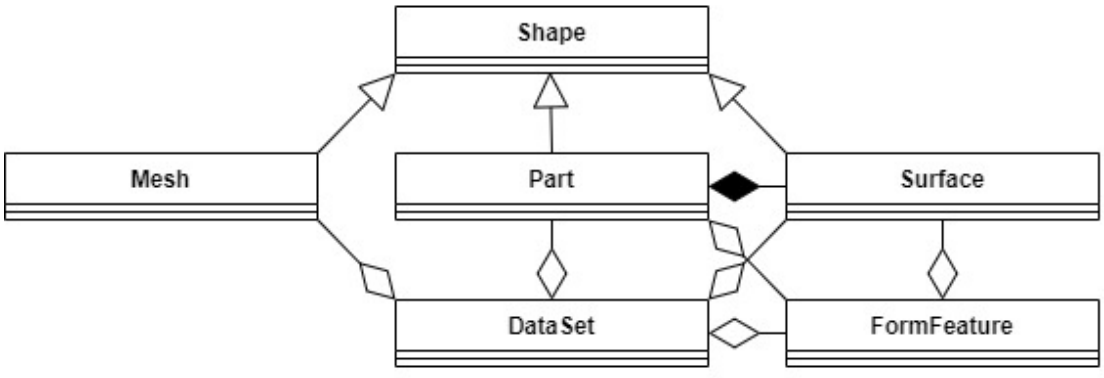


Figure 46 : Diagramme de classes UML simplifié du KBRE model

II.3. Définition des objets du *KBRE model*

II.3.1. Les objets de type Shape

Shape est une classe abstraite, ses objets ne sont donc jamais instanciés. Cette classe a pour fonction de définir les méthodes basées sur les descripteurs de formes qui ne nécessitent aucune manipulation de modèle 3D.

Un objet *Shape* peut contenir un ou plusieurs descripteurs de forme stockés dans ses attributs. Un descripteur est défini par son type, appelé d_type et un argument d_arg. d_type correspond à la nature du descripteur, soit s'il s'agit d'un *AISM* ou d'un *SD* par exemple. d_arg correspond au paramètre du descripteur, comme le degré de l'*AISM* ou la fonction de forme du *SD*. Dans le cas d'un *AISM*, l'erreur d'approximation est aussi stockée dans l'attribut. Le nom de l'attribut D_name contenant un descripteur spécifique est formé par son type et son argument. Par exemple, *AISM*_6 est un attribut contenant les paramètres d'un *AISM* de degré 6 et son erreur d'approximation, ou encore SD_2 est un attribut contenant les paramètres de distribution de la fonction de forme numéro 2. Notons enfin que l'objet ainsi structuré reste ouvert à l'ajout et l'utilisation de nouveaux descripteurs de formes.

Chaque méthode définie dans cette classe est applicable aux objets de type *Mesh*, *Part*, *Surface* et *FormFeature*. Il s'agit des méthodes $\boldsymbol{match}()$, $\boldsymbol{classify}()$, $\boldsymbol{retrieve\_match}()$ et $\boldsymbol{retrieve\_mlpc}()$.

La méthode $\boldsymbol{match}(\boldsymbol{shape}, \boldsymbol{D\_name})$ permet de rapidement comparer l'objet avec une autre instance d'objet $shape$ par le calcul du score de *matching* entre les descripteurs des deux objets. Les descripteurs comparés doivent naturellement être disponibles dans les attributs D_name des instances manipulées, faute de quoi la méthode retourne une alerte. Le score de *matching* retourné est un nombre réel qui détermine la dissimilarité entre les formes globales des objets analysés.

La méthode $\boldsymbol{retrieve\_match}(\boldsymbol{dataset}, \boldsymbol{k}, \boldsymbol{D\_name})$ compare l'objet analysé avec un ensemble d'objets agrégés dans un objet $dataset$ de type *DataSet*. Le score de *matching* est calculé entre le descripteur D_name de l'objet analysé et ceux de chaque objet agrégé par $dataset$, puis les k objets associés aux scores les plus bas sont retournés. Il convient encore de noter que les descripteurs de chaque objet agrégé dans $dataset$ doivent avoir été préalablement calculés. Le *retrieval* permet ainsi de retrouver les modèles d'un large ensemble avec les formes les plus similaires à celle analysée.

D'autres méthodes font appel à des outils informatiques de *ML*. On appelle $mlpc$ un objet de type *MLP-C*, externe au *KBRE model*. Les détails sur les étapes d'entraînement de $mlpc$ seront fournies lors de la définition de la classe *DataSet*. Pour le moment, supposons que les objets $mlpc$ utilisés par la suite ont été préalablement entraînés.

La méthode $\boldsymbol{classify}(\boldsymbol{mlpc}, \boldsymbol{D\_name})$ retourne un résultat de *classification* propre à une taxonomie spécifique. Dans ce contexte, $mlpc$ prédit les probabilités d'association de D_name à chacune des familles de formes qui définissent sa taxonomie, et retourne la plus probable. La famille prédite est alors stockée dans un argument appelé $category$, afin d'éviter toute confusion entre classe de forme et classe d'objet.

Dans le cadre de la seconde méthode de *retrieval* proposée, les classes connues du $mlpc$ ne font plus référence à des familles d'objets, mais à chaque objet individuel agrégé par un *DataSet*. Le $mlpc$ est ainsi entraîné à prédire la probabilité d'association d'un descripteur à chaque objet de l'ensemble. La méthode $\boldsymbol{retrieve\_mlpc}(\boldsymbol{mlpc}, \boldsymbol{k}, \boldsymbol{D\_name})$ permet retrouver les k objets associés aux plus hautes probabilités du $mlpc$ qui seront considérés comme les résultats de la recherche.

II.3.2. Les objets de type Mesh

Dans le *KBRE model*, un objet de type ***Mesh*** représente un modèle 3D au format de maillage triangulaire. Une instance d'objet *Mesh* est créée par traduction d'un fichier contenant les informations relatives aux entités géométriques d'un maillage 3D, comme le format stéréolithographie (.stl). Les paramètres suivants permettent de créer une instance de *Mesh* :

- $\boldsymbol{path}$: il s'agit du chemin d'accès vers le fichier stocké dans le SI
- $\boldsymbol{metadata}$: informations textuelles sur l'instance, comme sa provenance et son identifiant dans un SI

D'autres attributs permettent le stockage de données et d'informations relatives à l'instance :

- $\boldsymbol{vertices}$: cet attribut contient la liste des sommets du maillage, c'est à dire des points de l'espace définis par leurs coordonnées (x, y, z) dans un repère cartésien
- $\boldsymbol{faces}$: cet attribut correspond à la liste des facettes du maillage, c'est à dire des faces planes triangulaires définies par trois sommets
- $\boldsymbol{edges}$: il s'agit de la liste des côtés des faces du maillage qui sont définis par les deux sommets qu'ils relient
- $\boldsymbol{normals}$: à chaque facette est associée un vecteur normal défini par ses coordonnées (n_x, n_y, n_z). Les normales définissent l'orientation de la *surface*, soit le coté externe de chaque facette.
- $\boldsymbol{areas}$: cet attribut contient la liste des aires des facettes, automatiquement calculées
- $\boldsymbol{category}$: renseignée manuellement ou par la méthode $classify()$, il s'agit de la classe (*i.e.* la famille) à laquelle l'instance appartient selon une taxonomie définie. On note que, dans le cadre de nos travaux, chaque objet ne peut être associé qu'a une seule famille pour plus de simplicité. Cependant, $\boldsymbol{category}$ pourrait être un dictionnaire ou une liste et contenir les différentes taxonomies
- $\boldsymbol{index}$: identifiant unique de l'instance selon son type parmi les objets capitalisés. Des détails seront apportés avec la méthode $capitalize()$ des objets *DataSet*
- $\boldsymbol{D\_name}$: descripteur de forme de l'objet

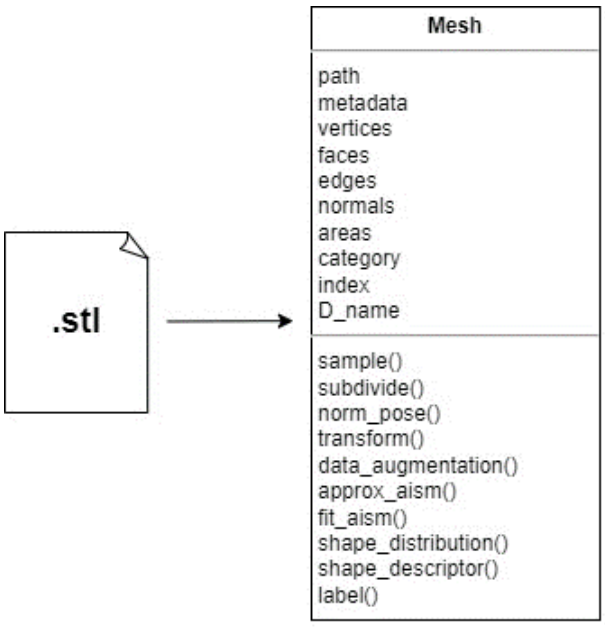


Figure 47: classe d'objet Mesh

Les objets de type *Mesh* sont à la base de toute activité de *SA* car ils contiennent les entités géométriques discrètes nécessaires au calcul de descripteurs de formes, ainsi qu'à la manipulation d'une géométrie dans l'espace.

Tout d'abord, des méthodes de prétraitement permettent de modifier les sommets, faces et cotés du maillage. Notons que le fichier source n'est pas modifié, seule l'instance d'objet *Mesh* associée l'est. Ces méthodes sont nécessaires pour la validation de divers critères empiriques relatifs aux éléments géométriques du maillage avant son analyse. Par exemple, la méthode $\boldsymbol{sample}(\boldsymbol{n})$ échantillonne le maillage à un nombre $\boldsymbol{n}$ de sommets, lorsque la méthode $\boldsymbol{subdivide}(\,)$ densifie le maillage en divisant chaque face en trois par l'ajout d'un sommet en son centre. La méthode $\boldsymbol{norm\_pose}(\,)$ normalise la position du maillage sur la base de l'algorithme de *Principal Component Analysis* (*PCA*). Pour cela, une transformation rigide est calculée et appliquée au maillage pour positionner son centroïde à l'origine et aligner ses axes d'inertie principaux avec les axes du repère. La méthode $\boldsymbol{transform}(\boldsymbol{M})$ modifie le positionnement spatial des sommets du maillage par une transformation euclidienne rigide définie par $\boldsymbol{M}$.

Les méthodes de prétraitement et de normalisation permettent de valider la conformité du maillage par rapport à des critères géométriques spécifiques à chaque méthode d'analyse ou de calcul de descripteurs, sous peine de résultats biaisés ou d'un temps d'analyse inutilement prolongé.

La méthode $\boldsymbol{data\_augmentation}(\boldsymbol{n})$ génère $\boldsymbol{n}$ instances de l'objet *Mesh* qui serviront à l'apprentissage des *MLP-C*. Les méthodes $subdivide(\,)$, $sample(n)$ et $transform(M)$ sont successivement appliquées à chaque instance générée avec $\boldsymbol{n}$ et $\boldsymbol{M}$ aléatoirement générés.

L'approximation d'un ***AISM*** de degré $\boldsymbol{d}$ sur les sommets du maillage est réalisée par la méthode $\boldsymbol{approx\_aism}(\boldsymbol{d})$ qui retourne $\boldsymbol{aism\_params}$, le vecteur des paramètres du modèle polynomial approximé. L'erreur d'approximation $\boldsymbol{mse}$ est aussi retournée.

La méthode $\boldsymbol{fit\_aism}(\boldsymbol{aism\_params})$ retourne le coût de *fitting* $\boldsymbol{fit\_cost}$ ainsi que la matrice de transformation optimisée $\boldsymbol{M}$ pour le *fitting* de l'*AISM* formé par $\boldsymbol{aism\_params}$ sur les sommets du maillage.

En vue de les comparer aux *AISM*, une méthode de calcul de descripteurs de type *Shape Distribution* (*SD*) (Osada et al., 2002) est applicable aux maillages. La méthode $\boldsymbol{shape\_distribution}(\boldsymbol{sfi})$ retourne $\boldsymbol{sd\_params}$**,** les paramètres de l'histogramme de distribution de la fonction de forme sfi (cf. Annexe 3 : Shapes Functions).

Grace aux méthodes précédemment définies, il est maintenant possible de calculer les descripteurs représentatifs de la forme des objets *Mesh*. La méthode $\boldsymbol{shape\_descriptor}(\boldsymbol{D\_name})$ calcule un descripteur spécifique sur les données géométriques discrètes de l'objet. Pour ce faire, l'utilisateur sollicite un descripteur spécifique grâce à l'argument $\boldsymbol{D\_name}$, comme ***AISM*_6** pour *AISM* de degré 6 ou **SD_2** pour la distribution de la fonction de forme numéro 2. Après prétraitement automatique du maillage selon des critères définis empiriquement, le descripteur calculé est retourné puis ajouté aux attributs de l'instance (cf. ***Algorithme 1***). On note que le calcul de nouveaux descripteurs peut être facilement ajouté à la méthode.

Les descripteurs de formes de l'objet pouvant désormais être calculés, des analyses de formes peuvent être faites. Par héritage de la classe *Shape*, les méthodes $\boldsymbol{match}(\,)$, $\boldsymbol{classify}(\,)$, $\boldsymbol{retrieve\_match}(\,)$ et $\boldsymbol{retrieve\_mlpc}(\,)$ peuvent être appliquées aux objets *Mesh*.

Une dernière méthode d'analyse permet de comparer précisément le maillage étudié avec d'autres modèles 3D via la méthode $\boldsymbol{label}(\boldsymbol{candidats}, \boldsymbol{D\_name})$, avec $\boldsymbol{D\_name}$ de type *AISM*. Pour chaque objet de la liste $\boldsymbol{candidats}$, un *AISM* représentatif de sa forme est fitté sur les sommets du maillage analysé. L'erreur de *fitting* est calculée pour chaque candidat et sert de mesure de dissimilarité. Un seuil d'erreur empirique permet de valider si le candidat associé peut être considéré comme identique au maillage analysé ou bien rejeté. Le meilleur candidat des retourné. L'***Algorithme 2*** représente de manière simplifiée la méthode de *labeling* des objets de type Mesh.

Algorithme 1 : méthode shape_descriptor( ) de la classe Mesh

```
inputs :
        mesh                                # objet Mesh
        D_type                              # type de descripteur, extrait de D_name
        D_arg                               # argument du descripteur, extrait de D_name
returns :
        D                                   # vecteur des paramètres du descripteur calculé
        mse                                 # erreur d'approximation si D_type = 'AISM', None sinon

mesh.norm_pose( )                           # normalisation de la position du maillage

while length(mesh.vertices) < n :           # tant que le nombre de sommets est inférieur à n :
   mesh.subdivide                           # subdivision du maillage
mesh.sample(n)                              # échantillonnage à n sommets

if D_type = 'AISM' :                        # si AISM requêté :
   D, mse = mesh.approx_AISM(d = D_arg      # approximation d'un modèle de degré d=D_arg
   mesh.D_name = (D, mse)                   # l'attribut D_name est renseigné

if D_type = 'SD' :                          # si SD requêté :
   mesh.shape_distribution(sfi = D_arg)     # calcul de la distribution de la fonction de forme sfi = D_arg
   mesh.D_name = (D, None)                  # l'attribut D_name est renseigné

return (D, mse)                             # retourne les résultats
```

Algorithme 2 : méthode label( ) de la classe Mesh

```
inputs :
        mesh                                # objet Mesh
        candidats                           # liste des objets candidats
        d                                   # degré des AISM, extrait de D_name
returns :
        label                               # objet label, ou nul

fit_error0, label = thr, None               # initialisation de l'erreur de fitting à la valeur empirique de seuil

mesh.norm_pose( )                           # normalisation de la position du maillage

while length(mesh.vertices) < n :           # tant que le nombre de sommets est inférieur à n :
   mesh.subdivide                           # subdivision du maillage
mesh.sample(n)                              # échantillonnage à n sommets

for candidat in candidats :                 # pour chaque objet candidat de la liste :
   D, mse = candidat.D_name                 # récupération des paramètres et de l'erreur de son AISM
   fit_score, M = mesh.fit_AISM(D)          # applique la méthode de fitting d'un AISM
   fit_error = (fit_score – mse)**2         # calcul de l'erreur de fitting
   if fit_error < fit_error0 :              # si erreur de fitting minimum des candidats et inférieure au seuil :
      label = candidat                      # le candidat est attribué au label

return label                                # retourne le label du maillage
```

La classe *Mesh* ainsi définie permet de manipuler des maillages 3D et d'en extraire des descripteurs de forme nécessaires aux activités d'analyse. Cependant, la majorité des données géométriques présentes dans un système *PDM* industriel sont modélisées par une représentation continue et prennent donc la forme de modèles CAO *b-rep*. La classe ***Part*** est définie pour en permettre l'analyse.

II.3.3. Les objets de type Part

Dans le *KBRE model*, un objet de type ***Part*** est la représentation d'un modèle 3D au format b-rep, c'est à dire un assemblage de surfaces continues unitaires connexes. Un objet *Part* est considéré de deux manière différentes : comme un ensemble topologique de formes unitaires qui sont chacune représentées par des objets de type *Surface*, ou comme une forme globale résultante. L'objet *Part* permet l'analyse de la géométrie globale des modèles CAO.

Une instance de *Part* est générée par traduction d'un modèle CAO *b-rep* dont la définition géométrique est contenue dans un fichier au format neutre STEP. Le paramètre suivant permet de créer une instance de *Part* :

- $\boldsymbol{path}$: il s'agit du chemin d'accès vers le fichier stocké dans le SI
- $\boldsymbol{metadata}$: informations textuelles sur l'instance, comme sa provenance et son identifiant dans un SI

D'autres attributs permettent le stockage de données et d'informations relatives à l'instance :

- $\boldsymbol{adjacency\_matrix}$ **:** il s'agit de la matrice d'adjacence du modèle CAO, qui définit sa topologie. Chacune des surfaces qui le compose est représenté par un tag (*i.e.* l'index de surface relatif à la pièce). La matrice d'adjacence représente les informations de connexité entre les tags des surfaces
- $\boldsymbol{surfaces}$ **:** la liste des instances d'objets de type *Surface* composant l'objet *Part*. La structure de ces objets est définie par la suite
- $\boldsymbol{form\_features}$ **:** la liste des instances d'objets de type *FormFeature* agrégés par l'objet *Part*. La structure de ces objets est définie par la suite
- $\boldsymbol{category}$: renseigné manuellement ou par la méthode $classify()$, il s'agit de la classe (*i.e*. la famille) à laquelle l'instance appartient selon une taxonomie définie
- $\boldsymbol{index}$: identifiant unique de l'instance selon son type parmi les objets capitalisés. Des détails sont apportés avec la méthode $\boldsymbol{capitalize}()$ des objets *DataSet*
- $\boldsymbol{D\_name}$: descripteur de forme de l'objet

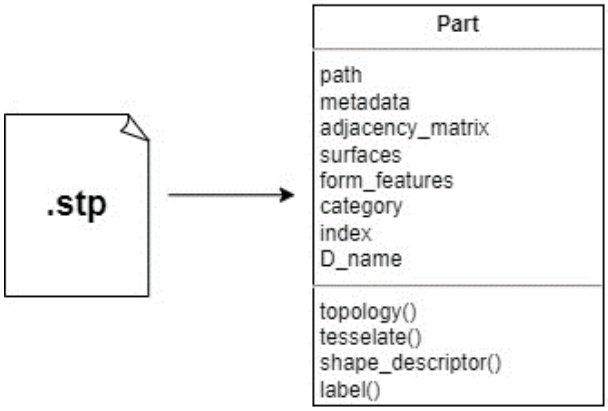


Figure 48: classe d'objet Part

Les objets de type *Part* peuvent être analysés par les méthodes suivantes :

La méthode $\boldsymbol{topology}()$ permet la lecture du fichier source et l'extraction des informations topologiques du modèle CAO. Cette méthode retourne les liste $\boldsymbol{tags}$ et $\boldsymbol{geometry\_types}$ qui permettront la création des instances de type *Surface* qui composent la Part.

Contrairement à un objet de type *Mesh*, un objet *Part* ne contient pas d'entités géométriques discrètes nécessaires au calcul des descripteurs de sa forme. Par conséquent, il est nécessaire de générer une représentation géométrique discrète représentative de la forme globale du modèle continu d'origine. Pour ce faire, la méthode $\boldsymbol{tesselate}(\boldsymbol{n})$ génère un maillage 3D de $\boldsymbol{n}$ sommets par tessellation du modèle CAO d'origine, et retourne l'instance d'objet de type ***Mesh*** associée après avoir supprimé le fichier généré. Cet objet *Mesh* est utilisé comme représentation géométrique intermédiaire et non permanente de l'objet *Part* pour la manipulation et l'analyse de sa forme. Il est important de noter que si $\boldsymbol{n} = \boldsymbol{None}$, la méthode $\boldsymbol{tesselate}(\boldsymbol{n})$ est paramétrée pour générer un maillage fidèle aux critères définis pour le calcul d'un descripteur de forme et la labélisation. Pour d'autres exploitations, un utilisateur peut renseigner $\boldsymbol{n}$ selon ses besoins.

La méthode $\boldsymbol{shape\_descriptor}(\boldsymbol{D\_name})$ appliquée à une instance d'objet *Part* est définie comme suit :

(i) La méthode $\boldsymbol{tesselate}()$ génère un maillage 3D par tessellation du modèle CAO b-rep source (pointé par l'attribut $\boldsymbol{path}$) et une instance d'objet *Mesh* est créée.
(ii) La méthode $\boldsymbol{shape\_descriptor}(\boldsymbol{D\_name})$ appliquée à l'objet *Mesh* retourne le descripteur souhaité.
(iii) Ce dernier est ajouté à l'attribut $\boldsymbol{D\_name}$ de l'objet *Part*.
(iv) L'objet *Mesh* est supprimé

Les descripteurs de formes de l'objet pouvant désormais être calculés, des analyses de formes peuvent être réalisées. Par héritage de la classe *Shape*, les méthodes $\boldsymbol{match}()$, $\boldsymbol{classify}()$, $\boldsymbol{retrieve\_match}()$ et $\boldsymbol{retrieve\_mlpc}()$ peuvent être appliquées aux objets *Part*.

Pour labéliser un objet *Part*, tout comme pour le calcul de ses descripteurs, il est nécessaire de disposer d'un maillage 3D pour le *fitting* des *AISM* candidats. Ainsi, la méthode $\boldsymbol{label}(\boldsymbol{candidats}, \boldsymbol{D\_name})$ utilise également la méthode $\boldsymbol{tesselate}()$ pour générer une instance d'objet de type *Mesh* comme représentation géométrique intermédiaire de l'objet. Le résultat de *labeling* de l'objet **Mesh** est retourné comme résultat de *labeling* de l'objet *Part* analysé.

Les objets de type ***Part*** ainsi structurés permettent l'analyse de la forme globale résultante d'un modèle 3D b-rep. La classe ***Surface*** est définie dans le but d'analyser séparément chacune des surfaces qui composent un modèle CAO.

II.3.4. Les objets de type Surface

La classe ***Surface*** permet des analyses de formes non plus sur un modèle CAO complet, mais sur les surfaces unitaires qui composent ce modèle. En tant que composante d'un objet *Part*, un objet *Surface* est directement relié à un modèle CAO au format b-rep, mais les instances ne peuvent être créées manuellement et le sont automatiquement et uniquement lors de la création d'une *Part*. Ces dernières sont alors regroupées dans la liste de l'attribut $\boldsymbol{surfaces}$ de l'objet *Part*.

Les éléments suivants sont extraits du fichier CAO source par la méthode $\boldsymbol{topology}()$ des objets *Part*, et renseignés comme paramètres de création d'une instance de *Surface* :

- $\boldsymbol{tag}$: l'index de la surface comme définie par la topologie du modèle CAO source
- $\boldsymbol{geometry\_type}$: il s'agit de la « nature » de la surface, soit s'il s'agit d'une primitive tel que « plan », « cylindre », « cône », « sphère » ou « tore », ou bien s'il s'agit d'une *surface* « *freeform* »

D'autres attributs permettent le stockage de données et d'informations relatives à l'instance :

- $\boldsymbol{part}$: correspond à l'objet *Part* que l'instance compose, pour permettre d'accéder à ses attributs
- $\boldsymbol{category}$: renseigné manuellement ou par la méthode $classify()$, il s'agit de la classe (i.e. la famille) à laquelle l'instance appartient selon une taxonomie définie
- $\boldsymbol{index}$: identifiant unique de l'instance selon son type parmi les objets capitalisés. Des détails seront apportés avec la méthode $\boldsymbol{capitalize}()$ des objets *DataSet*
- $\boldsymbol{D\_name}$: descripteur de forme de l'objet

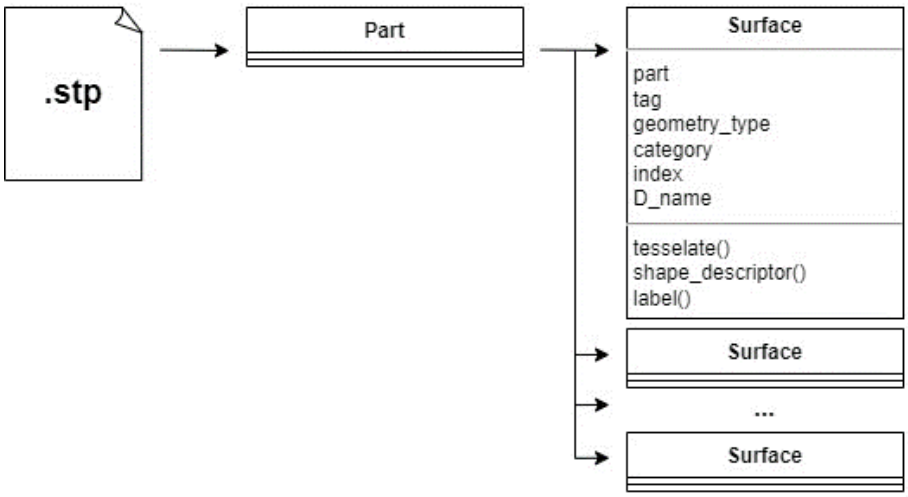


Figure 49: classe d'objet Surface

De la même manière que pour les objets *Part*, la méthode $\boldsymbol{tesselate}(\boldsymbol{n})$ permet de générer un maillage 3D qui sert de représentation géométrique intermédiaire de l'objet *Surface* analysé. Cette méthode isole la surface identifiable par son $\boldsymbol{tag}$ dans le fichier CAO source, et retourne un objet *Mesh* représentatif de sa forme.

La tessellation des objets *Surface* permet d'étendre les méthodes $\boldsymbol{shape\_descriptor(D\_name)}$ et $\boldsymbol{label(candidats,\ D\_name)}$ aux objets Surface. De plus, les méthodes $\boldsymbol{match}()$, $\boldsymbol{classify}()$, $\boldsymbol{retrieve\_match}()$ et $\boldsymbol{retrieve\_mlpc}()$ peuvent être appliquées aux objets de type Surface par héritage de la classe *Shape*.

Le type d'objet *Surface* du *KBRE model* permet ainsi l'analyse des composantes d'un modèle CAO et offre de nouvelles possibilités dans l'extraction d'informations à partir des formes de modèles 3D. Dans un contexte de développement de produits complexes, de nombreuses itérations et évolutions sont réalisées sur les modèles 3D. Analyser indépendamment les surfaces d'un modèle, et non plus sa forme globale, permet d'identifier des évolutions locales, d'identifier un modèle composant un modèle plus large, de repérer les modèles partageant des surfaces communes, etc. Dans le but de compléter les possibilités d'analyses locales de modèles CAO en s'intéressant à ses composantes, les objets ***FormFeature*** sont définis.

II.3.5. Les objets de type FormFeature

Dans un contexte d'analyse de pièces complexes, la définition des objets ***FormFeature*** permet de ne s'intéresser qu'à une sous-partie de ces modèles. Les *FormFeature* sont définis comme des groupements d'objets *Surface* entretenant des relations de connexité entre elles. Ils permettent des analyses relatives à des besoins d'ingénierie spécifiques pour lesquels analyser le modèle CAO dans son intégralité est une perte de temps ou simplement non pertinent.

Un objet *FormFeature* est généré manuellement depuis une instance de *Part* par un utilisateur qui spécifiera la nouvelle liste des $\boldsymbol{n}$ objets *Surface* agrégés par le Form*Feature.* Les paramètres suivants permettent de créer une instance d'objet *FormFeature* :

- $\boldsymbol{part}$: correspond à l'objet *Part* à partir duquel est créé l'instance
- $\boldsymbol{surfaces}$: sous-ensemble de l'attribut du même nom de l'objet *Part* constitué des $\boldsymbol{n}$ instances de Surface agrégées par l'objet

D'autres attributs permettent le stockage de données et d'informations relatives à l'instance :

- $\boldsymbol{adjacency\_matrix}$ **:** matrice d'adjacence recalculée qui définit la topologie de l'objet
- $\boldsymbol{category}$: renseigné manuellement ou par la méthode $classify()$, il s'agit de la classe (*i.e.* la famille) à laquelle l'instance appartient selon une taxonomie définie
- $\boldsymbol{index}$: identifiant unique de l'instance selon son type parmi les objets capitalisés. Des détails seront apportés avec la méthode $\boldsymbol{capitalize}()$ des objets *DataSet*
- $\boldsymbol{D\_name}$: descripteur de forme de l'objet

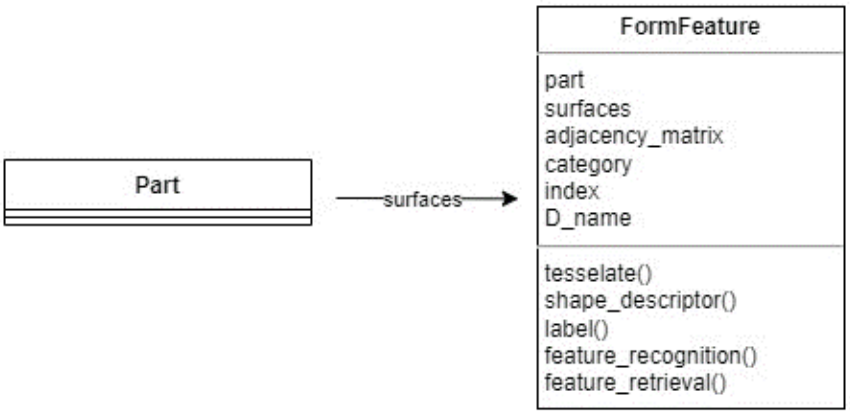


Figure 50 : classe d'objet FormFeature

De manière équivalente aux classes *Part* et *Surface*, les méthodes $\boldsymbol{tesselate(n)}$, $\boldsymbol{shape\_descriptor(D\_name)}$ et $\boldsymbol{label(candidats, D\_name)}$ s'appliquent aux objets *FormFeature*, ainsi que $\boldsymbol{match}()$, $\boldsymbol{classify}()$, $\boldsymbol{retrieve\_match}()$ et $\boldsymbol{retrieve\_mlpc}()$ par héritage de *Shape*.

Deux méthodes spécifiques aux *FormFeature* sont implémentées.

$\boldsymbol{feature\_recognition(part, D\_name)}$, avec D_name un *AISM*, permet d'identifier s'il existe dans l'instance $\boldsymbol{part}$ des groupes topologiques d'objets *Surface* identiques à l'objet *FormFeature*. Cela signifie que, pour un groupe topologique reconnu, chaque objet *Surface* qui le compose est strictement identique à un objet *Surface* agrégé par le *FormFeature*, et que leurs relations de connexités sont identiques. Cette méthode permet notamment d'identifier des répétitions dans un modèle. La méthode appliquée à une instance $\boldsymbol{form\_feature}$ est définie comme suit :

(i) Un descripteur $\boldsymbol{D\_name}$ est calculé pour chaque instance de *Surface* $\boldsymbol{S_i}$ agrégé par $\boldsymbol{form\_feature}$ avec la méthode $\boldsymbol{shape\_descriptor(D\_name)}$**.** On note que $\boldsymbol{D\_name}$ est obligatoirement de type *AISM*

(ii) Pour chaque instance $\boldsymbol{Sp_i}$ de $\boldsymbol{part}$:
  - (ii.a) Un descripteur $\boldsymbol{D\_name}$ est calculé
  - (ii.b) la méthode $\boldsymbol{match_{score_i} = Sp_i.match(S_j, D\_name)}$ est appliquée à $\boldsymbol{Sp_i}$ pour chaque $\boldsymbol{S_j}$ de $\boldsymbol{form\_feature}$. Si $\boldsymbol{match\_score_i}$ est inférieur à un seuil empirique, $\boldsymbol{S_j}$ est ajoutée à une liste $\boldsymbol{candidats_i}$
  - (ii.c) $\boldsymbol{label_i = Sp_i.label(candidats_i,\ D\_name)}$ est appliquée à $\boldsymbol{Sp_i}$

(iii) Une copie de $\boldsymbol{part.adjacency\_matrix}$**,** que l'on appelle $\boldsymbol{am\_part}$ est créée avec uniquement les lignes et colonnes correspondant aux $\boldsymbol{tag_i}$ des $\boldsymbol{Sp_i}$ pour lesquelles le résultat de *labeling* $\boldsymbol{label_i}$ n'est pas $\boldsymbol{None}$

(iv) Une fonction inspirée des *graph comparison* de (El-Mehalawi & Miller, 2003a, 2003b) permet d'identifier les structures communes entre $\boldsymbol{am\_part}$ et $\boldsymbol{am\_feature}$, la matrice d'adjacence de $\boldsymbol{form\_feature}$. Cette fonction retourne autant de listes de $\boldsymbol{n}$ surfaces $\boldsymbol{Sp_i}$ que de groupes topologiques identiques à $\boldsymbol{form\_feature}$ identifiés.

Ainsi, $\boldsymbol{feature\_recognition(part, D\_name)}$ appliquée à $\boldsymbol{form\_feature}$ retourne des groupes de $\boldsymbol{n}$ objets ***Surface*** composant l'instance $\boldsymbol{part}$.

Par extension, la méthode $\boldsymbol{feature\_retrieval(dataset, D\_name)}$ itère la méthode $\boldsymbol{feature\_recognition(part, D\_name)}$ appliquée à un objet *FormFeature* pour chaque instance d'objet de type *Part* agrégé dans un objet de type *DataSet*.

Jusqu'ici, les types d'objets ***Mesh***, ***Part***, ***Surface*** et ***FormFeature*** sont structurés pour l'analyse de modèles 3D à différents niveaux de leurs descriptions géométriques et topologiques. Pour permettre l'analyse d'ensembles de modèles, les objets de type ***DataSet*** déjà rencontrés dans certaines méthodes sont définis.

II.3.6. Les objets DataSet

Un objet de type ***DataSet*** agrège diverses instances d'objets des autres types du *KBRE model*. Cela permet de les rassembler et d'appliquer différentes méthodes d'analyse à tous les objets d'un ou plusieurs types, afin de généraliser les analyses à de larges ensembles de données plutôt que de répéter l'analyse sur chacune d'entre elles.

Les paramètres de création d'un objet **DataSet** sont les suivants :

- $\boldsymbol{dir}$: le chemin d'accès vers un répertoire contenant les fichiers des modèles 3D
- $\boldsymbol{name}$: le nom donné au *DataSet*
- $\boldsymbol{metadata}$: informations textuelles sur l'instance, comme sa provenance et son identifiant dans un SI

A la création d'une instance d'objet *DataSet*, les attributs suivants sont générés :

- $\boldsymbol{parts}$ **:** la liste des instances d'objets de type *Part* agrégés par l'objet
- $\boldsymbol{meshes}$ **:** la liste des instances d'objets de type *Mesh* agrégés par l'objet
- $\boldsymbol{surfaces}$ **:** la liste des instances d'objets de type *Surface* agrégés par l'objet
- $\boldsymbol{form\_features}$ **:** la liste des instances d'objets de type *FormFeature* agrégés par l'objet
- $\boldsymbol{mlpcs}$: la liste des *MLP-C* entrainés à la *classification* et au *retrieval* sur les objets agrégés
- $\boldsymbol{index}$: identifiant unique de l'instance selon son type parmi les objets capitalisés. Voir la méthode $\boldsymbol{capitalize}()$ ci-dessous

La première méthode, illustrée par la Figure 51, consiste en la création (*i.e*. l'initialisation) d'un objet de type *DataSet* grâce à une fonction $\boldsymbol{factory}(\boldsymbol{dir})$ qui, pour chaque fichier contenu dans le répertoire pointé par $\boldsymbol{dir}$, sélectionne et crée automatiquement le/les instances d'objets adaptées selon l'extension du fichier. De cette manière, si $\boldsymbol{path}$ pointe vers un fichier d'extension « .stl », un objet *Mesh* est créé. Si $\boldsymbol{path}$ pointe vers un fichier « .stp », un objet *Part* est créé ainsi que les $\boldsymbol{n}$ objets *Surface* qui le composent. L'ajout d'un objet *FormFeature* à un *DataSet* est réalisé manuellement. La Figure 51 synthétise graphiquement la création d'une instance d'objet *DataSet*.

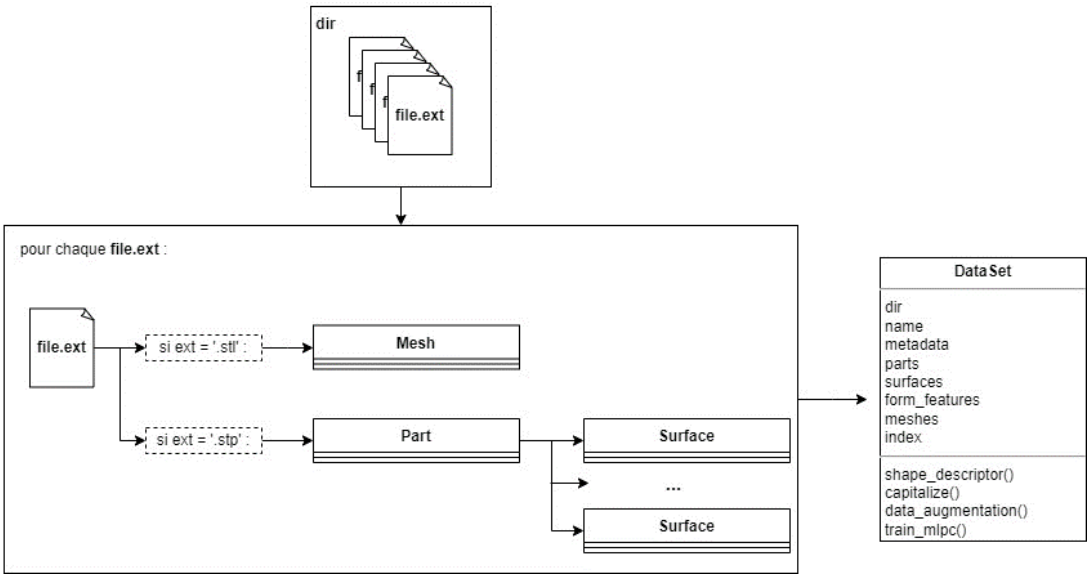


Figure 51: classe d'objet DataSet

La première méthode implémentée, $\boldsymbol{shape\_descriptor}(\boldsymbol{D\_name})$, applique la méthode du même nom à chaque objet agrégé par le *DataSet*. Cette méthode permet de s'assurer de disposer des descripteurs nécessaires aux activités d'analyse des objets.

Sur des grands ensembles, le calcul d'un descripteur de chaque objet agrégé est une opération relativement chronophage qu'il n'est ni utile ni souhaitable de réaliser plusieurs fois. Il est donc nécessaire de stocker les descripteurs calculés en vue de les récupérer rapidement pour des analyses ultérieures. La capitalisation est une méthode de stockage de l'information dans une base de connaissance. Elle permet de conserver les informations relatives aux objets générés à un instant donné pour y accéder ultérieurement via une méthode de chargement des objets, de leurs attributs, et surtout de leurs descripteurs sans besoin de les calculer à nouveau. Pour cela, on associe une base de connaissances, ou *Knowledge Base (KB)*, à chaque classe d'objet du *KBRE model* (excepté la classe abstraite *Shape*).

Ces *KB* permettent d'indexer l'ensembles des objets capitalisés et de stocker leurs attributs. Elles prennent simplement la forme de tableaux où les lignes correspondent aux instances d'objets, et les colonnes à leurs attributs, comme illustrés par les Tableaux 3 à 7 ci-dessous. De manière générale, les objets contenus dans des attributs sont remplacés par leur index attitré. Les *MLP-C* sont remplacés par le chemin d'accès vers un fichier de stockage de leurs paramètres. On note l'absence de certains attributs des objets dans les *KB*, comme les matrices d'adjacence des *Part* et *FormFeature* qui ne sont pas stockées, de même que l'ensemble des attributs relatifs aux données géométriques des maillages ($vertices$, $faces$, $edges$…). Des colonnes correspondent à chaque descripteur $\boldsymbol{D\_name_i}$ implémenté dans le modèle. Si un descripteur particulier n'est pas disponible pour une instance, la case associée reste vide.

index	dir	name	metadata	meshes	parts	surfaces	form_ features	mlpc
1	dir1	*Model*net40	*DataSet* public	i, j, k, …	i, j, k, …	i, j, k, …	i, j, k, …	mlpc_path1, mlpc_path2
2	dir2	aero6000	Base aubes aéro	l, m, n, …	l, m, n, …	l, m, n, …	l, m, n, …	mlpc_path1, mlpc_path2

Tableau 3 : KB_DataSet

index	path	category	surfaces	form_ features	$\boldsymbol{D\_name_1}$	$\boldsymbol{D\_name_2}$	…
1	path1	Blade	i, j, k, …	i, j, k, …	D, mse	$D, None$	
2	path2	Screw	l, m, n, …	l, m, n, …	D, mse		

Tableau 4 : KB_Part

index	part	tag	geometry_ type	category	$\boldsymbol{D\_name_1}$	$\boldsymbol{D\_name_2}$	…
1	i	12	freeform		D, mse	$D, None$	
2	j	35	freeform		D, mse		

Tableau 5 : KB_Surface

index	part	surfaces	$\boldsymbol{D\_name_1}$	$\boldsymbol{D\_name_2}$	…
1	i	i, j, k, …	D, mse	$D, None$	
2	j	l, m, n, …	D, mse		

Tableau 6 : KB_FormFeature

index	path	category	$\boldsymbol{D\_name_1}$	$\boldsymbol{D\_name_2}$	…
1	path1	Blase	D, mse	$D, None$	
2	path2	Screw	D, mse		

Tableau 7 : KB_Mesh

La méthode $\boldsymbol{capitalize}()$ appliquée à une instance de *DataSet* ajoute automatiquement sa ligne dans *KB_DataSet* ainsi que l'ensemble des lignes correspondant aux objets *Part*, *Surface*, *FormFeature* et *Mesh* qu'il agrège. De cette manière, l'ensemble d'un *DataSet* peut être chargé rapidement par la fonction $\boldsymbol{load\_dataset}(\boldsymbol{index})$ sans besoin de recalculer leurs descripteurs de formes. La capitalisation permet ainsi l'analyse différée de très larges bases de modèles 3D via le calcul de tous les descripteurs de formes en amont de l'application, ce qui est n'est pas envisageable *in situ*.

Les objets *DataSet* permettent aussi l'utilisation d'outils d'intelligences artificielle tels que les *ANNs* pour les activités d'analyse de forme. La méthode $\boldsymbol{data\_augmentation}(\boldsymbol{n}, \boldsymbol{object\_type})$ génère un ensemble de $\boldsymbol{n}$ objets *Mesh* aléatoirement transformés et échantillonnés pour chaque objet de type $\boldsymbol{object\_type}$ agrégé dans le *DataSet*. On dispose alors de $\boldsymbol{n}$ représentations géométriquement et topologiquement (au sens structure du maillage) différentes de la même forme, et donc de $\boldsymbol{n}$ descripteurs pour la même forme, permettant notamment l'apprentissage des *MLP-C* pour le *retrieval*.

La méthode $\boldsymbol{train\_mlpc}(\boldsymbol{analysis}, \boldsymbol{object\_type}, \boldsymbol{D\_type}, \boldsymbol{n})$ génère, entraine, sauvegarde et retourne $\boldsymbol{mlpc}$, un objet de type *MLP-C* issue d'une dépendance du *KBRE model*. L'argument $\boldsymbol{object\_type}$ permet l'apprentissage sur un seul type d'objet agrégé dans le *DataSet*, par exemple uniquement sur les objets de type Surface. Si l'argument $\boldsymbol{n}$ n'est pas nul, la méthode $\boldsymbol{data\_augmentation}(\boldsymbol{n}, \boldsymbol{object\_type})$ est appliquée au *DataSet* et retourne les données d'apprentissage dans un paramètre $\boldsymbol{X\_train}$. Pour chaque maillage, un descripteur $\boldsymbol{D\_type}$ est calculé. $\boldsymbol{analysis}$ prend pour valeur "$\boldsymbol{classification}$" ou "$\boldsymbol{retrieval}$" et définit si $\boldsymbol{mlpc}$ est entrainé pour la méthode $\boldsymbol{classify}()$ ou $\boldsymbol{retrieve\_mlpc}()$**.** Dans le premier cas, $\boldsymbol{Y\_train}$ est la liste des catégories associées à chaque objet de $\boldsymbol{X\_train}$, dans le second $\boldsymbol{Y\_train}$ contient leurs index. La fonction $\boldsymbol{train\_mlpc}(\boldsymbol{X\_train}, \boldsymbol{Y\_train}, \boldsymbol{D\_type})$ entraine et retourne $\boldsymbol{mlpc}$. Ce dernier est stocké dans l'attribut $\boldsymbol{mlpcs}$ du *DataSet*, en plus de ses métadonnées propres qui incluent $\boldsymbol{analysis}$**,** $\boldsymbol{object\_type}$, $\boldsymbol{D\_type}$ et $\boldsymbol{n}$**.** Les objets $\boldsymbol{mlpc}$ sont ainsi disponibles pour les méthodes $\boldsymbol{classify}(\boldsymbol{mlpc}, \boldsymbol{D\_name})$ et $\boldsymbol{retrieve\_mlpc}(\boldsymbol{mlpc}, \boldsymbol{D\_name})$ des objets enfant de *Shape* afin de prédire leurs associations à une des classes connues par le modèle, c'est à dire une catégorie ou un index de forme.

En résumé, le type d'objets ***DataSet*** ainsi structuré permet d'appliquer une méthode sur des ensembles d'objets du *KBRE model*, et de fixer un cadre aux analyses comme la possibilité de *retrieval* dans un ensemble de modèles 3D spécifique.

La Figure 52 présente la définition complète des objets du *KBRE model* présenté dans cette seconde partie de notre proposition scientifique. Avant de conclure sur les apports de la proposition, un bilan est dressé sur la définition du *KBRE model* par rapport à ses objectifs.

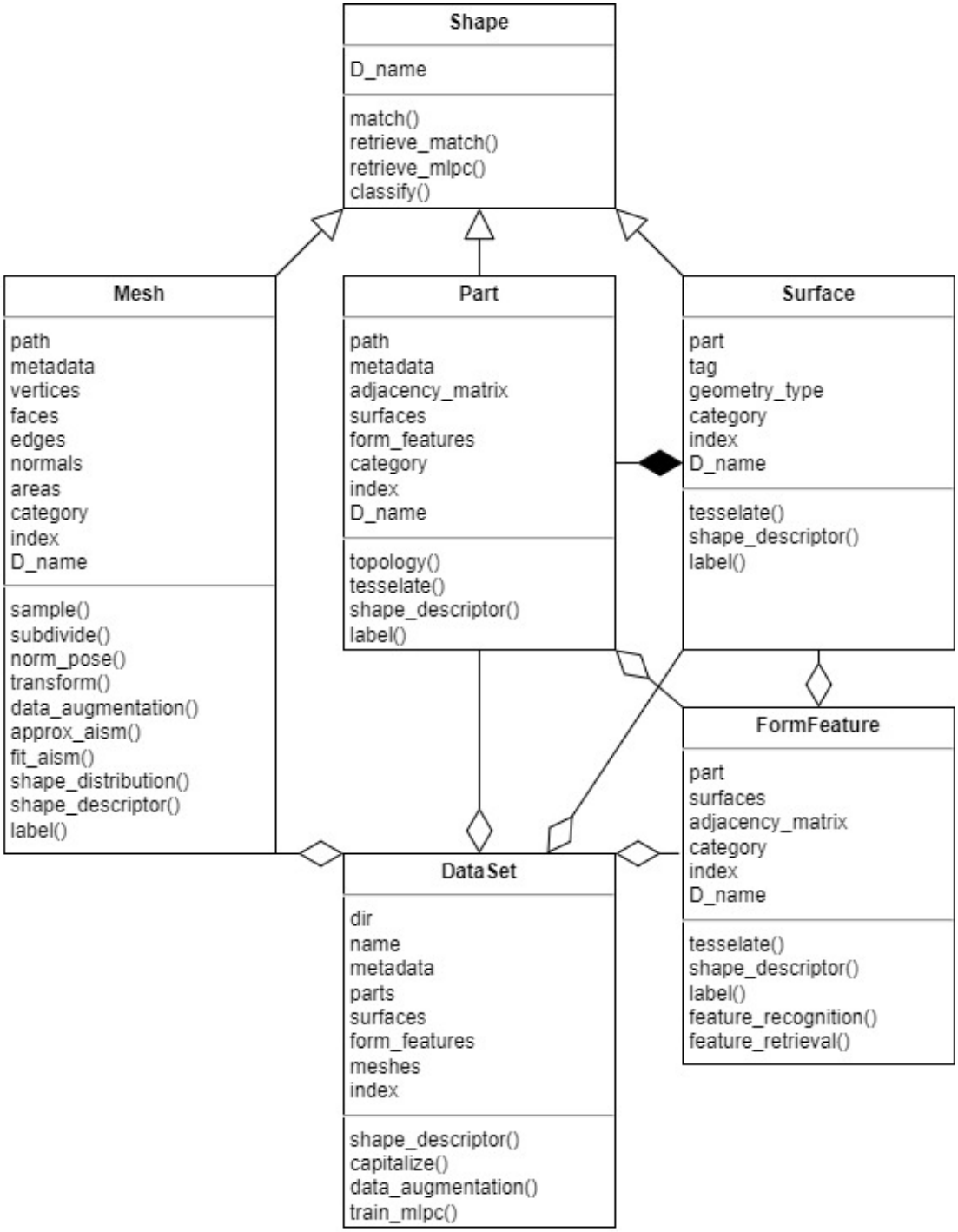


Figure 52: diagramme de classes UML complet du KBRE model

II.4. Bilan sur le *KBRE model*

Les méthodes de *SA* présentées dans la partie précédente permettent l'analyse de modèles 3D. Le *KBRE model* en permet l'exploitation dans le cadre de processus de *RE*.

Tout d'abord, la définition des classes d'objets et des méthodes applicables offre la structure nécessaire pour définir et appliquer des méthodes spécifiques en fonction des formats de modèles CAO. Cela permet une adaptation précise aux différents types de données.

Les objets **Mesh** représentent les maillages 3D, format géométrique très courant pour des activités de visualisation, de calcul, et aussi de reconstruction. Ces objets sont essentiels pour la manipulation d'éléments géométriques discrets et le calcul de descripteurs de formes.

Les objets de type **Part**, **Surface** et **FormFeature** sont des représentations de modèles CAO continus à plusieurs niveaux de leurs descriptions géométriques et topologiques. Ils permettent l'analyse à plusieurs échelles du format de modélisation géométrique le plus couramment utilisé par les activités d'ingénierie. La tessellation permet la traduction de ces objets pour permettre le calcul de descripteurs de forme et leur analyse.

L'interopérabilité entre les différents formats de modélisation 3D est possible grâce à l'utilisation de descripteurs de formes communs entre les types d'objets, permettant des analyses de *matching*, de *classification*, de *retrieval* et de *labeling* pour comparer et rapprocher les modèles CAO entre eux. De plus, le modèle de données implémente des techniques avancées de *ML* pour une meilleur robustesse des méthodes, qui résulte en une amélioration des capacités d'interprétation des résultats d'analyse.

La classe d'objet ***DataSet*** est conçu pour manipuler efficacement de vastes ensembles de modèles 3D et pour la réalisation d'actions communes sur les données agrégées. Par ailleurs, la contextualisation des analyses et l'interprétation des informations qui en sont extraites est essentielle pour la création d'une base riche et structurée de connaissances.

La capitalisation et le chargement d'objets depuis les ***Knowledge Base*** ouvre la possibilité aux activités d'analyse différées dans le temps sur de très larges ensembles de données, sans besoin de répéter diverses activités chronophages comme le calcul des descripteurs de forme ou l'apprentissage des *MLP-C*.

Enfin, on note que le *KBRE model* est extensible et peut être enrichi par d'autres descripteurs ou méthodes d'analyse. Il offre ainsi une plate-forme polyvalente pour le développement d'activités avancées de *RE*.

Pour résumer, le *KBRE model* a pour objectif de permettre l'expérimentation et l'utilisation des activités de SA dans le cadre de processus de *RE* sur des données CAO via la gestion des données et connaissances associées. La structure des objets en permet l'organisation, lorsque la création et l'indentification sont assurées par les méthodes. Le stockage et la récupération des données et connaissances associées sont permis par la capitalisation dans des *KB*, et leur diffusion est possible grâce à l'indépendance du modèle par rapport aux autres composantes du SI et logiciels d'édition spécifiques. Enfin, la mise en œuvre opérationnelle nécessite l'intégration du *KBRE model* au SI de l'entreprise ainsi que le développement d'une interface graphique pour le dialogue avec les utilisateurs.

III. Conclusion de la proposition

Dans un contexte de 4e révolution industrielle et de transformation numérique des entreprises, ces dernières font face à des problématiques de continuité numérique et doivent s'organiser afin de prévenir la perte de cohérence sémantique qui impacte la définition de leurs produits et systèmes. À la suite de ce constat, le maintien de l'intégrité des modèles CAO, principal support et aboutissement des processus d'ingénierie pour le développement de produits mécaniques, s'impose comme une nécessité.

Le *Reverse Engineering*, processus d'analyse et de compréhension des produits, a été identifiée comme un moyen contre l'obsolescence des modèles CAO d'une entreprise. En effet, le *RE* intègre de nombreuses activités visant à une meilleure gestion des données et connaissances associées aux modèles CAO, notamment par le développement de techniques d'analyse et d'exploration des bases de modèles 3D d'une entreprise. Nous proposons ainsi d'exploiter les processus de *RE* pour le maintien de l'intégrité des CAO face aux problèmes de continuité numérique des modèles CAO en environnement *PDM*.

Deux axes d'études ont été identifiés dans les phases amont de nos travaux de recherche. Tout d'abord, l'analyse de forme 3D est explorée en tant qu'ensemble d'activités pour l'extraction d'informations depuis des modèles CAO potentiellement incomplets. Ensuite les systèmes de *Knowledge Base Engineering* répondent aux besoins d'intégration des activités de *SA* et de gestion des données, informations et connaissances associées aux processus de *RE*.

L'analyse de forme 3D offre la capacité de créer ou recréer différentes informations relatives aux modèles 3D étudiés. Le *retrieval* et le *labeling* permettent de retrouver des modèles 3D en base autrement qu'avec des métadonnées textuelles potentiellement indisponibles ou invalides. Avec la *classification*, ces activités génèrent de nouvelles connaissances sur les relations entretenues entre modèles 3D en se basant uniquement sur leurs formes, un attribut intrinsèque et incorruptible d'un modèle CAO. Le concept de form-*feature*, utilisé par l'activité de *feature-recognition*, permet des analyses avancées des éléments géométriques et topologiques des modèles 3D auxquels sont attachés des connaissances sémantiques relatives à un besoin métier. Le premier enjeu au développement de méthodes et outils de *SA* concerne les descripteurs de formes et leur capacité à représenter des surfaces complexes. La proposition des *Algebraic Implicit Surface Model* vise au développement et à la maitrise des paramètres de création d'un descripteur de forme adapté aux surfaces complexes, typologie de forme largement ignorée dans la littérature scientifique.

Un deuxième enjeu concerne le développement de méthodes de *SA* robustes face à l'imperfection des descripteurs en raison de variations géométriques, topologiques, et de la position spatiale des formes qu'ils représentent. Des techniques basées sur les *ANNs* sont proposées pour exploiter les capacités du *ML* à produire des modèles de prédictions précis et robustes. Ces modèles permettent un raisonnement sur les propriétés et relations des formes 3D sans se baser sur un système de règles strictes. L'activité de *retrieval* permet notamment la comparaison d'une méthode intégrant un *Multi-Layer Perceptron Classifier* avec une méthode basée sur une analyse numérique simple, dans le but d'évaluer les apports et limites en termes de performances et de temps d'exécution des modèles complexes de *ML*.

Enfin, aucune méthode à notre connaissance ne s'intéresse au problème de comparaison de formes très similaires pour une identification précise de la référence en base d'un modèle CAO ou d'une de ses surfaces. On appelle cette activité le *labeling* de forme, qui vise à exempter l'utilisateur d'une appréciation experte des résultats de *classification* ou de *retrieval*. La méthode de *fitting* des *AISM* développée fournit une mesure de dissimilarité précise dont l'objectif est d'atteindre ce que l'on appelle le niveau III de distinctivité des descripteurs de formes, c'est-à-

dire la capacité à différentier des formes pourtant très similaires. Sur cette base, la proposition d'une méthode de *labeling* représente une avancée dans les techniques d'analyse et d'exploration de bases de modèles CAO en généralisant les capacités d'identification précises aux formes complexes qui peuvent désormais être précisément reconnues, sans se limiter au *retrieval* de formes proches.

Une fois des méthodes de *SA* fiables et robustes disponibles, reste la question de la gestion des données ainsi que des informations et connaissances créées pour intégrer les activités de *SA* aux processus de *RE* des modèles CAO. La piste suivie concerne les systèmes de *Knowledge Based Engineering (KBE)*.

Dans leur définition, les systèmes de *KBE* intègrent les technologies de Programmation Orienté Objet, de modélisation 3D CAO, ainsi que d'Intelligence Artificielle, couvrant ainsi la majorité des aspects techniques nécessaires à la proposition. Par ailleurs, la spécificité de systèmes *KBE* réside dans l'utilisation d'une base de connaissances pour la capitalisation de la connaissance produite, ce qui correspond au besoin de sauvegarde et d'utilisation des descripteurs de formes et modèles de *ML* pour l'analyse des CAO.

Pour pleinement répondre aux besoins de création, de récupération, d'identification, d'organisation, de stockage, de diffusion, et surtout de mise en œuvre opérationnelle des données et connaissances relatives aux modèles 3D, la proposition d'un *Knowledge Based Reverse Engineering model* se doit d'intégrer plusieurs fonctionnalités essentielles.

La structure des objets du *KBRE model* assure un traitement cohérent de l'information entre les éléments en œuvre dans les processus de *RE*, de même qu'au sein des méthodes spécifiques aux objet. La création, l'organisation et l'identification des connaissances est ainsi assurée pour de vastes ensembles de données CAO issues de sources diverses.

De plus, l'utilisation de *KB* permet la capitalisation des connaissances relatives aux modèles CAO en vue de les exploiter ultérieurement.

Un algorithme de traduction entre différents formats de modélisation 3D ainsi que l'utilisation de descripteurs communs permet l'interopérabilité entre différentes représentations des produits. L'interopérabilité entre les SI de l'entreprise est quant à elle assurée par l'indépendance du *KBRE model* aux formats et structures de données spécifiques à chaque élément des SI.

En résumé, le *KBRE model* représente une solution technique robuste pour les processus de *RE* en intégrant des fonctionnalités de traduction de formats, d'analyse et de gestion des connaissances des modèles CAO. Le modèle de données représente une base solide pour le développement et l'expérimentation de techniques avancées d'analyse de modèles 3D, tout en facilitant l'accès et le traitement performant de l'information issue de sources diverses. Cette approche ouvre la voie à une meilleure compréhension des modèles 3D et à des possibilités d'exploration plus approfondie des bases de données CAO pour maintenir l'intégrité de ces données essentielles aux entreprises face à leurs problématiques de continuité numérique.

Pour achever cette conclusion relative à notre proposition, concernant les apports scientifiques, il faut préciser que l'exploitation des connaissances apportées par l'analyse de modèles 3D est relative à chaque besoin métier. Il est donc impossible de couvrir l'ensemble des processus de *RE*. Cependant, le *KBRE model* offre de nombreuses fonctionnalités qui répondent à l'ensemble des besoins identifiés par les 5 scénarii de *RE* lors de l'analyse du Contexte Industriel de ces travaux de recherche.

En effet, face au scénario définit dans le ***Usecase 1 : reconstruction de maillages***, une instance d'objet *Mesh* peut représenter le maillage à reconstruire. La méthode de *retrieval* permet alors de retrouver de potentiels objets, représentatifs de modèles natifs présents dans un *DataSet* capitalisé, et dont la forme est proche de celle d'un maillage. Un modèle paramétrique retrouvé simplifie la reconstruction du maillage avec une méthode de *template-based*, comme présenté dans la seconde partie du Chapitre 2 : État de l'art de la littérature scientifique. De même, représenter un modèle CAO « pauvre » au format neutre par un objet de type *Part* permet d'explorer le SI à la recherche de modèles natifs de formes semblables ou identiques. Pour répondre aux besoins définis par le ***Usecase 2 : réutilisation de modèles CAO et de PMI***, la méthode de *retrieval* identifie des modèles à la géométrie similaire, quand la méthode de *labeling* affine le résultat en retournant uniquement les modèles de forme parfaitement identique. Les objets *FormFeature* sont particulièrement adaptés au ***Usecase 3 : identification de liens de parentés dans le ,*** puisqu'ils permettent l'analyse de sous-composantes de modèles. La méthode de *feature_retrieval* identifie la présence du groupe topologique analysé dans divers modèles CAO afin de les relier entre eux. Pour ce qui est du ***Usecase 4 : Recherche transverse entre systèmes*** , la création d'objets *DataSet* représentatifs des « silos » d'information permet le croisement des données qu'ils contiennent. Enfin, s'il est possible d'itérer la méthode de *retrieval* et de *labeling* sur chaque objet agrégé par un *DataSet* pour en identifier de potentiels doublons, le ***Usecase 5 : identification des doublons et standardisation*** présente des perspectives d'évolution et d'amélioration du *KBRE model*. En effet, l'implémentation de nouvelles méthodes comme le *clustering* est une piste d'optimisation de la méthode d'identification de doublons et la standardisation des bases de modèles 3D.

Ainsi, il est possible de conclure que l'analyse de modèles CAO offre une perspective prometteuse pour relever les défis de la gestion des données et connaissances associées aux modèles CAO d'un environnement industriel impacté par les défis de la continuité numérique.

Le chapitre suivant a pour objectif de valider les apports effectifs de la proposition par la réalisation d'un ensemble d'expérimentations et la présentation de la synthèse associée.

Chapitre 4 : Évaluation des méthodes d'analyse du KBRE model

Ce chapitre est consacré à la validation des éléments de la proposition scientifique, exposée dans le chapitre précédent. Cette étape de nos travaux de recherche permet d'apporter les connaissances nécessaires pour l'analyse de la valeur ajoutée des ***Algebraic Implicit Surface Model*** proposés comme descripteurs de formes. À travers cette démarche d'évaluation, nous cherchons à proposer des éléments tangibles et mesurables qui permettront de mieux comprendre l'apport et les limites des *AISM* pour chacune des activités d'analyse de modèles de CAO.

En parallèle de cette évaluation des ***AISM***, nous procédons à une réflexion critique sur le ***Knowledge-Based Reverse Engineering Model*** en tant de brique technologique des SI pour le déploiement de processus de Reverse Engineering.

Ce chapitre est structuré en deux parties. Tout d'abord, la méthodologie d'évaluation est détaillée selon les points suivants :

- Des précisions sont apportées sur l'implémentation informatique du *KBRE model*,
- Un plan d'expérience est défini dans le but de spécifier les expériences successives à réaliser pour valider puis évaluer les divers aspects de la proposition,
- Des ensembles de modèles CAO sont assemblés en objets *DataSet* qui serviront de données de tests pour les expérimentations. Plusieurs contextes d'applications possibles sont couverts par les *DataSet* qui regroupent les modèles 3D selon des typologies de formes définies.

Dans un second temps, chaque expérimentation est détaillée selon le plan suivant :

- Rappel technique sur la méthode évaluée,
- Énoncé des hypothèses et critères d'évaluation des résultats,
- Précision des paramètres évalués,
- Présentation sous forme algorithmique de la méthode de test,
- Présentation et critique des résultats,
- Bilan sur la méthode.

I. Méthodologie d'évaluation

Cette première partie apporte les précisions nécessaires sur le cadre méthodologique et technique des expérimentations réalisées lors de la phase d'évaluation des travaux de recherche.

On y précise en premier quelques détails techniques sur l'implémentation informatique du *KBRE model* en vue de disposer de l'ensemble des fonctionnalités présentées au chapitre précédent. Ensuite, la chronologie des expérimentations pour la validation successive des méthodes d'analyses de formes est détaillée. Enfin, les ensembles de modèles 3D sur lesquels seront réalisés les différentes expérimentations sont présentés.

I.1. Implémentation du *KBRE model*

Suite à sa définition présentée au chapitre précédent, une implémentation informatique du *KBRE model* est réalisée dans le langage de programmation orienté objet Python. Les fonctions et méthodes des objets du modèle de données se basent sur divers modules externes, parmi lesquels :

- La librairie GMSH (Geuzaine & Remacle, 2009) permet la lecture de fichiers CAO aux formats neutres (.step, .igs), et intègre un algorithme de tessellation des surfaces continues du modèle CAO.
- La librairie Trimesh (Dawson-Haggerty et al., 2019) permet la lecture et l'extraction des éléments géométriques discrets d'un fichier maillage 3D au format stéréolithographie (.stl). Les objets Trimesh intègrent diverses méthodes de manipulation et de modification des entités géométriques des maillages, dont héritent les objets de type *Mesh*.
- Les librairies scientifiques Scipy (Virtanen et al., 2020) et Scikit-Learn (Buitinck et al., 2013; Pedregosa et al., 2011) sont utilisées par les méthodes d'optimisation et d'apprentissage automatisée.

Les expérimentations sont réalisées sur un PC Intel Precision 7920 munis d'un CPU Intel(R) Xeon(R) Gold 5222 @3.8GHz, 4 cœurs. Mémoire RAM 128 Go.

I.2. Plan d'expérience

Le plan d'expérience est conçu comme une séquence d'expérimentations, où chacune a pour objectif l'évaluation et la validation des éléments de la proposition à travers les méthodes associées implémentées dans le *KBRE model*. Chacune de ces expérimentations consiste elle-même en un ensemble de tests avec différents paramétrages de sa méthode. L'ordre établi pour ces expérimentations vise à valider et à optimiser rigoureusement les méthodes qui constituent un prérequis aux étapes suivantes.

S'ils sont applicables, les critères usuellement requis pour un descripteur de forme (Laga et al., 2018; Tangelder & Veltkamp, 2008) sont utilisés pour dresser un bilan de chaque méthode.

Le **critère de compacité** concerne l'espace de stockage nécessaire. Il ne s'agit donc pas seulement de critiquer la taille du descripteur de forme utilisé, mais aussi de considérer le potentiel besoin de stockage associé au fonctionnement de chaque méthode d'analyse, comme les modèles d'apprentissage supervisés par exemple.

Le **temps d'exécution** est évalué indépendamment pour chaque méthode et concerne toutes les étapes nécessaires au bon fonctionnement de la méthode. De manière générale, le temps d'exécution dépend de l'implémentation informatique ainsi que du matériel mis en œuvre. Ce critère, principalement donné à titre indicatif, permet de comparer les méthodes du *KBRE model* entre elles.

La **distinctivité** d'un descripteur ne peut être évaluée de manière globale, mais est à considérer séparément pour chaque méthode d'analyse. En effet, les résultats d'analyses ne dépendent pas seulement du type de descripteur de forme utilisé, mais aussi de la mesure de dissimilarité utilisée et l'algorithme d'implémentation de la méthode. La distinctivité d'un descripteur sera donc évaluée selon trois niveaux de précisions:

(i) le **premier niveau** est atteint si le descripteur capture les caractéristiques distinctives communes aux formes d'une famille d'objets, et sera évalué selon les résultats de la méthode de *classification*. Par exemple, le niveau I de distinctivité revient à classifier un objet en tant que vis parmi un ensemble de catégories d'objets de quincaillerie.

(ii) le **second niveau** de distinctivité correspond à la capacité d'un descripteur à discriminer les formes en fonction de leur proximité géométrique, et sera évaluée sur les activités de *retrieval*. Par exemple, le niveau II de distinctivité revient à retrouver les vis de même longueur et diamètre que celle analysée parmi un ensemble de vis en vrac.

(iii) le **troisième niveau** est atteint si un descripteur permet de différencier des formes très similaires. Ce dernier niveau sera évalué sur les résultats des tests de *labeling*. Par exemple, le niveau III de distinctivité revient à identifier précisément la vis analysée en la comparant avec un ensemble de vis référencées dans un catalogue.

Les capacités discriminatives de niveaux I et II des *AISM* seront comparées à celles de différents SD de (Osada et al., 2002). Puisque l'activité de *labeling* n'est, à notre connaissance, couverte par aucuns travaux de recherche, comme évoqué en conclusion de notre état de l'art. L'atteinte du 3^{e} niveau de distinctivité est l'enjeu principal lié à la proposition des *AISM* comme descripteurs de formes.

La **robustesse** des méthodes d'analyses aux transformations géométriques rigides rejoint le critère d'invariance des descripteurs. Du fait du manque de robustesse connu de la méthode de normalisation appliquée avant approximation des *AISM* (Laga et al., 2018), il n'est pas possible d'affirmer leur invariance. La robustesse de chaque méthode à la position initiale des modèles 3D sera donc déduite des niveaux de distinctivités atteints par comparaison avec ceux des SD. Par ailleurs, la robustesse à l'échantillonnage et à de légères perturbations topologiques sera discutée.

On note que, dans le cadre de nos travaux, la robustesse au bruit de mesure et aux occlusions partielles ne sont pas étudiées car elles concernent majoritairement l'analyse de maillages 3D issus de numérisations. La robustesse aux transformations géométriques non rigides (déformations et/ou facteur d'échelle) n'est pas pertinente non plus car elle va à l'encontre d'une capacité de distinctivité de niveau trois. En effet, dans un contexte d'analyse de modèles aéronautiques, nous considérons que la moindre variation même locale entre deux surfaces doit être détectée par la méthode de *labeling*.

Les expérimentations 1 et 2 ont pour objectif la validation et l'optimisation des méthodes d'*approximation* et de *fitting* des *AISM*, prérequis aux méthodes qui suivent. Les expérimentations 3, 4 et 5 permettent d'évaluer respectivement les méthodes de *classification*, de *retrieval* et de *labeling*.

Exp-1 : Approximation des AISM

La première expérimentation concerne la validation de la méthode d'*approximation* des *AISM* sur des échantillons de points. Il s'agit d'une étape fondamentale au calcul des descripteurs de formes nécessaires à toutes les méthodes d'analyse proposées.

L'expérimentation consiste en ensembles de tests d'approximation sur les formes de plusieurs *DataSet*, avec la variation systématique de certains paramètres de la méthode. L'analyse des résultats, c'est-à-dire l'examen des erreurs d'approximation moyennes en fonction des paramètres ciblés, a pour principal objectif de confirmer que l'algorithme converge de manière fiable vers une solution. Cela permet de valider que l'approximation d'un *AISM* en tant que modèle représentatif d'une forme est toujours réalisable. Par ailleurs, l'analyse de la dispersion des résultats permet d'identifier des incohérences ou aberrations potentielles en plus de définir empiriquement le meilleur paramétrage de la méthode.

Au-delà de ces aspects de validation, cette expérimentation permet d'anticiper les valeurs des critères de compacité et de temps d'exécution du calcul des *AISM* en tant que descripteurs de formes. Des observations statistiques des résultats permettent d'identifier des variations de performances selon les typologies de formes étudiées, sans qu'il soit pour autant possible de conclure sur les niveaux de distinctivités des *AISM*.

Exp-2 : Fitting des AISM

L'expérimentation suivante se concentre sur la méthode de *fitting* des *AISM* sur des échantillons de points, qui permet d'optimiser la transformation spatiale minimisant les erreurs de prédiction de l'*AISM* sur le nuage de points. La validation de cette méthode est essentielle pour assurer le bon recalage d'un modèle 3D dans la même position normalisée que lors de l'approximation de son *AISM*. Cette méthode permet de garantir l'invariance des *AISM*, prérequis sur lequel est basée la méthode de *labeling* développée.

L'expérimentation consiste en de nombreux tests de *fitting* d'*AISM* sur des copies aléatoirement transformées des maillages qu'ils approximent. Les résultats permettent d'analyser de manière quantitative la qualité des recalages en fonction des valeurs de différents paramètres de la méthode. De plus, l'expérimentation permet de déterminer le temps nécessaire à l'exécution de la méthode ainsi qu'à évaluer sa robustesse vis-à-vis de certains paramètres. Enfin, l'impact de la typologie des formes analysées sur les performances de la méthode représente un point d'attention particulier dans l'évaluation de la méthode de *fitting*.

Exp-3 : Classification

La troisième expérimentation permet d'évaluer la distinctivité de niveau 1 des *AISM* pour la *classification* de formes. Les *AISM* sont comparés aux *SD* (Osada et al., 2002) dans le cadre de la méthode de *classification* par *MLP-C* implémentée. De plus, les résultats obtenus sont confrontés à ceux issus de récentes méthodes de *DL* présentés dans la littérature scientifique.

Ne disposant pas d'ensembles de modèles 3D catégorisés et étiquetés du domaine aéronautique, les expérimentations sont réalisées sur deux *DataSets* publiques.

Exp-4 : Retrieval

L'évaluation de la méthode de *retrieval* par *MLP-C* permet l'étude des avantages et limites actuelles d'un modèle complexe d'apprentissage automatisés par rapport à une approche d'analyse numérique plus « simple », telle que la méthode de *retrieval* par *matching*. Cette expérience contribuera à évaluer la robustesse intrinsèque des *AISM* envers les légères variations de la position initiale des maillages 3D analysés, ainsi qu'à de légères variations topologiques, et ce dans la cadre de chacune des deux méthodes de *retrieval* étudiées.

Par ailleurs, les capacités discriminatives de niveau 2 des *AISM* sont confrontées à celles des *SD* en tenant compte des diverses typologies de formes des *DataSet* de tests.

Tout en comparant les méthodes sur les critères du temps d'exécution global et de la compacité de la méthode, un aspect clé de cette investigation concerne la capacité d'un *ANN* à apporter plus de robustesse contre les variations spatiales et topologiques des maillages analysés. Cette série de tests permet de déterminer si la mesure de dissimilarité par *MLP-C* offre des capacités de distinctivité de niveau 2 supérieure à la méthode de *matching*.

Exp-5 : Labeling

Les tests de *labeling* ont pour objectif d'évaluer si la méthode du même nom permet de reconnaître de manière précise une forme parmi plusieurs candidats. Basée sur la méthode de *fitting* des *AISM* des formes candidates, la méthode de *labeling* cherche à différentier des formes malgré leur très forte ressemblance. Cette notion de distinction parmi des formes hautement similaires définit le 3^{e} niveau de distinctivité des descripteurs de formes, enjeu majeur de la proposition scientifique de ces travaux.

Cette expérimentation consiste à labéliser un très grand nombre de copies de maillages aléatoirement transformés, afin d'étudier le taux de réussite de la méthode. On note que les candidats à la labélisation de la forme analysée correspondent aux résultats de *retrieval* par *matching* auxquels est ajouté la forme source analysée si absente. L'efficacité de la méthode de *labeling* sera également discutée en tenant compte des critères de temps d'exécution et de robustesse vis-à-vis des différentes typologies de formes.

I.3. Les données de test

Les ensembles de modèles 3D suivant sont capitalisés sous forme de *DataSet* dans le *KBRE model*, et servent de données de tests pour les expérimentations. La comparaison des résultats sur chaque ensemble nous permet d'évaluer l'impact de divers attributs des *DataSet* sur les méthodes d'analyses, comme sa taille et la typologie des formes qu'il agrège.

- **test20** est un jeu de données regroupant 20 *form-features* extraites manuellement de modèles CAO de composants aéronautiques. Ces *form-features* représentent des surfaces et groupes topologiques d'aubages et de congés de raccordement des modèles 3D. Ce jeu de données est utile pour réaliser divers tests préliminaires à petite échelle et pré-valider les résultats avant d'analyser de plus larges ensembles de formes.

- **aero6000** regroupe 6000 surfaces unitaires issues de la base de calcul et de développement des aubages aérodynamiques. On y retrouve exclusivement des surfaces aérodynamiques qui sont ensuite retravaillées et intégrées aux modèles CAO de conception. Ce jeu de données présente un défi particulier pour les activités d'analyse de formes car les surfaces qu'il contient ont des géométries très similaires

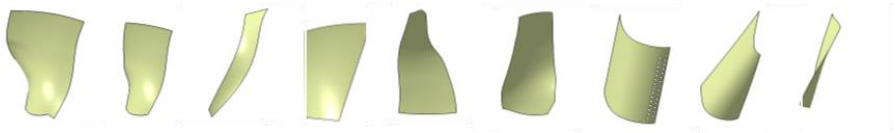


Figure 53 : exemple désensibilisé des surfaces du DataSet aero6000

- **parts1759** est un ensemble de 1759 modèles CAO extraits de la base d'archivage des modèles aéronautiques. On y retrouve des pièces spécifiques de moteurs regroupées avec des pièces mécaniques standard. 4913 surfaces *freeform* uniques d'une grande variété composent les modèles de cet ensemble et sont regroupées dans un *DataSet* appelé **surfaces4913** représentatif d'une typologie de forme similaire à **aero6000**

Figure 54 : exemple désensibilisé des modèles du DataSet parts1759

- **mcbA** (Kim et al., 2020) est un ensemble public de maillages 3D de pièces mécanique. Ce jeu de données regroupe 58696 pièces réparties entre 68 classes définies par la taxonomie de l'International Classification for Standards. **mcbA** permet l'expérimentation sur des typologies de formes non issues du domaine aéronautique.

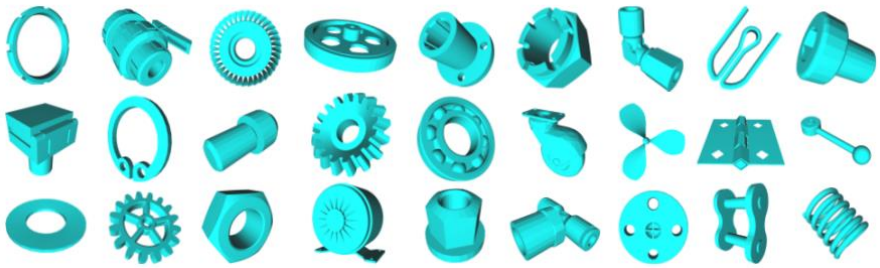


Figure 55 : exemple des modèles du DataSet mcbA de (Kim et al., 2020)

- ***model*net40** (Wu et al., 2015) est un ensemble de 11879 maillages 3D de modèles divers répartis en 40 classes. Créé en 2015 à l'université de Princeton, ce *DataSet* est une référence dans le monde de la recherche avec un grand nombre d'algorithmes de *classification* ayant été testés dessus.

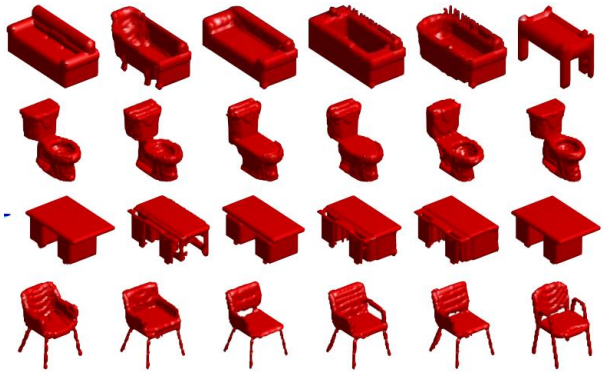


Figure 56 : exemple des modèles de modelnet40 de (Wu et al., 2015)

II. Expérimentations et résultats

Cette seconde partie du chapitre d'évaluation de la proposition scientifique examine en détail chaque expérimentation, présente ses résultats, et discute des conclusions qu'il est possible d'en extraire.

L'objectif principal de cette partie est avant tout de démontrer l'apport des *AISM* pour l'analyse de formes complexes, et plus spécifiquement l'atteinte du 3e niveau de distinctivité des descripteurs de formes qui ouvre de nouvelles perspectives pour l'exploration et l'analyse des bases de modèles CAO. La robustesse de chaque méthode aux paramètres qui l'influencent est un enjeu auquel l'optimisation des méthodes tente de répondre. La Figure 57 illustre le *workflow* des expérimentations ainsi que les objectifs fixés en termes de distinctivité et de robustesse pour chacune des méthodes analysées.

Diverses connaissances sont également apportées par les expérimentations, telles que la spécificité d'un descripteur à une typologie de forme, ou encore l'intérêt et les limites des modèles d'apprentissage automatisés. À la fin de ce chapitre, nous pourrons tirer des conclusions sur la contribution scientifique de la proposition ainsi que sur les orientations possibles de futures recherches.

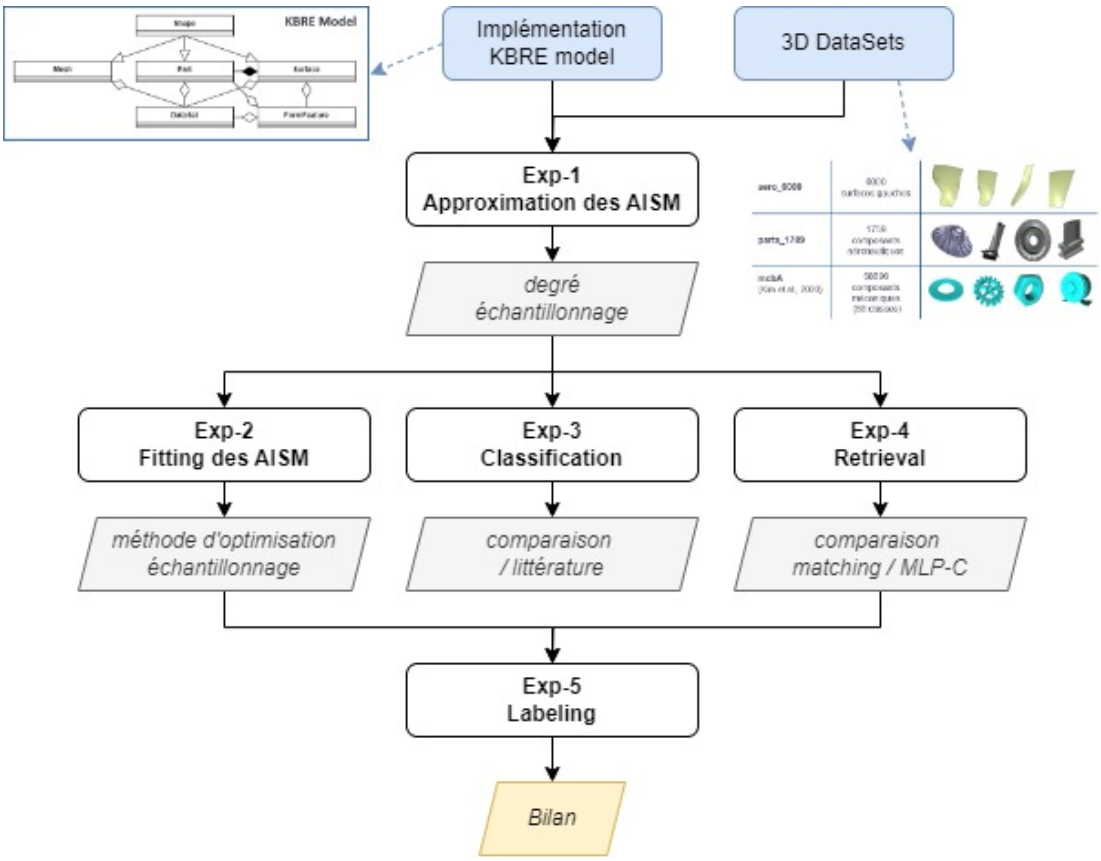


Figure 57 : workflow des expérimentations

II.1. Exp-1 : Validation de la méthode d'approximation

Dans cette première sous-partie, l'étude expérimentale visant à valider la méthode d'approximation des *AISM* sur des formes 3D est détaillée.

Appliquée à un objet de type *Mesh* du *KBRE model*, la méthode $approx\_aism(d)$ optimise les nd paramètres de la fonction polynomiale $f_d(x, y, z)$ pour minimiser l'erreur quadratique moyenne d'approximation (appelée simplement erreur d'approximation) sur l'ensemble des n sommets $p_i \in pc$ du maillage :

$$mse_d = \frac{1}{n}\sum_{i=1}^{n} f_d(x_i, y_i, z_i)^2$$

On rappelle que cette méthode retourne le vecteur des nd paramètres de l'*AISM* approximé dans un vecteur appelé $aism\_params$, ainsi que mse_d. Par ailleurs, l'erreur des prédictions du modèle approximé sur un nouveau maillage se calcule via la méthode $mse\_aism(aism\_params)$.

II.1.1. Hypothèses et critères de validation

Soit $mesh0$ un maillage dense de $n0$ sommêts, et $mesh$ une copie échantillonnée de $mesh0$ de $n < n0$ sommets.

Nous formulons l'hypothèse que la forme S représentée par $mesh$ peut-être approximée par un *AISM* f_d de degré $d > 2$ avec $mse_d < \varepsilon$. Plus ε est proche de 0, plus le modèle est précis. Dans ce cas, l'erreur des prédictions de f_d sur $mesh0$ est notée $mse0_d$. Si $mse0_d < \varepsilon$, alors f_d est considéré comme un *AISM* valide de S.

La valeur de ε est donc le principal critère pour l'optimisation et la validation de la méthode. Cette dernière n'admettant pas de valeur empirique connue pour le moment, l'expérimentation consiste à déterminer les paramètres de la fonction d'approximation minimisant ε sur divers ensembles de formes.

II.1.2. Paramètres de test

Différents paramètres influencent le résultat d'approximation d'un *AISM* sur un maillage 3D :

- **le degré d du polynôme** détermine le nombre de combinaisons possibles entre les coordonnées x, y, z et donc le nombre nd de paramètres α_i à régresser. Il s'agit de déterminer le paramètre $\boldsymbol{d}$ qui minimise ε ainsi que le temps de calcul moyen sur des ensembles de formes.

$\boldsymbol{d}$	2	3	4	5	6	7	8	9
$\boldsymbol{nd}$	9	19	34	55	83	119	164	219

Tableau 8 : dimension des AISM en fonction de leur degré

- **La dimension n de pc** détermine le nombre de points échantillonnés sur le modèle 3D pour l'approximation de son *AISM*. L'objectif est de déterminer n minimum à partir duquel ε ne diminue plus.

  Nb : Les tests étant effectués sur une grande variété de formes pour lesquelles l'aire de surface varie entre quelques dizaines de mm² à plusieurs m², il serait potentiellement plus cohérent de définir le paramètre d'échantillonnage en fonction de la densité du maillage plutôt que comme un nombre maximum de points. Cependant, pour étudier l'impact des autres paramètres sur le temps de calcul, nous fixons l'échantillonnage à un nombre de point spécifique sans considération des différences de dimensions entre les surfaces.

II.1.3. Méthode de test

L'***Algorithme 3*** ci-dessous détaille la méthodologie de génération des résultats de test de la méthode d'approximation des *AISM* sur les objets d'un *DataSet*. On note que, en amont de l'expérimentations, des maillages denses de 2^e5 sommets représentatifs des objets agrégés ont été générés et stockés dans l'attribut ***meshes*** de l'instance de *DataSet*.

Algorithme 3 : test de la méthode d'approximation des AISM sur les formes d'un DataSet

inputs :

```
        DataSet                                  # objet DataSet
        degrees = [ 2 : 9 ]                      # degrés des AISM à tester
        sample_values = [ 5e^2 , 1e^3 , 5e^3 , 1e^4 , 5e^4 , 1e^5 ]   # paramètres d'échantillonnages à tester
```

returns :

```
        test_results                             # retourne les résultats dans un tableau de la forme id/d/n/mse0/time
```

```
1.  meshes = DataSet.meshes                      # liste des maillages

2.  for mesh0 in meshes :                        # pour chaque maillage :
3.      id = mesh0.id                            # récupération de son index
4.      mesh0.norm_pose()                        # normalisation de la position du maillage

5.      for n in sample_values :                 # pour chaque paramètre d'échantillonnage :
6.          mesh = mesh0.copy()                  # copie de mesh0
7.          mesh.sample(n)                       # échantillonnage du maillage à n sommets

8.          for d in degrees :                   # pour chaque degré de modèle :
9.              AISM_params, mse = mesh.approx_AISM(d)   # approximation d'un AISM de degré d
10.             mse0 = mesh0.mse_AISM(AISM_params)       # calcul de l'erreur de prédiction sur les 2^e5 points du maillage original
11.             add line id/d/n/mse0/time to test_results   # les information du test sont ajoutées aux résultats
12. return tests_results                         # tableau récapitulatif des tests réalisés
```

II.1.4. Résultats

Les tests d'approximation sont réalisés sur les instances de *DataSet* **test20**, **aero100** (sous *DataSet* contenant les 100 premières instances de *Surface* du *DataSet* **aero6000**), et **parts100** (sous *DataSet* contenant les 100 premières instances de *Part* du *DataSet* **parts1759**)
L'analyse des résultats est détaillée pour **test20**. Les résultats sur les autres objets *DataSet* sont ensuite synthétisés.

Résultats de tests sur le DataSet test20

La première analyse concerne l'influence du nombre de points échantillonnés sur le résultat d'approximation. La Figure 58 représente l'évolution de la moyenne des $mse0_d$ pour l'ensemble des maillages du *DataSet* en fonction du paramètre d'échantillonnage n, et ce pour chaque degré d de modèle. On rappelle que $mse0_d$ est l'erreur de prédiction d'un *AISM* sur l'ensemble des 2e5 points du maillage d'origine, et non sur les n points échantillonnés uniquement.

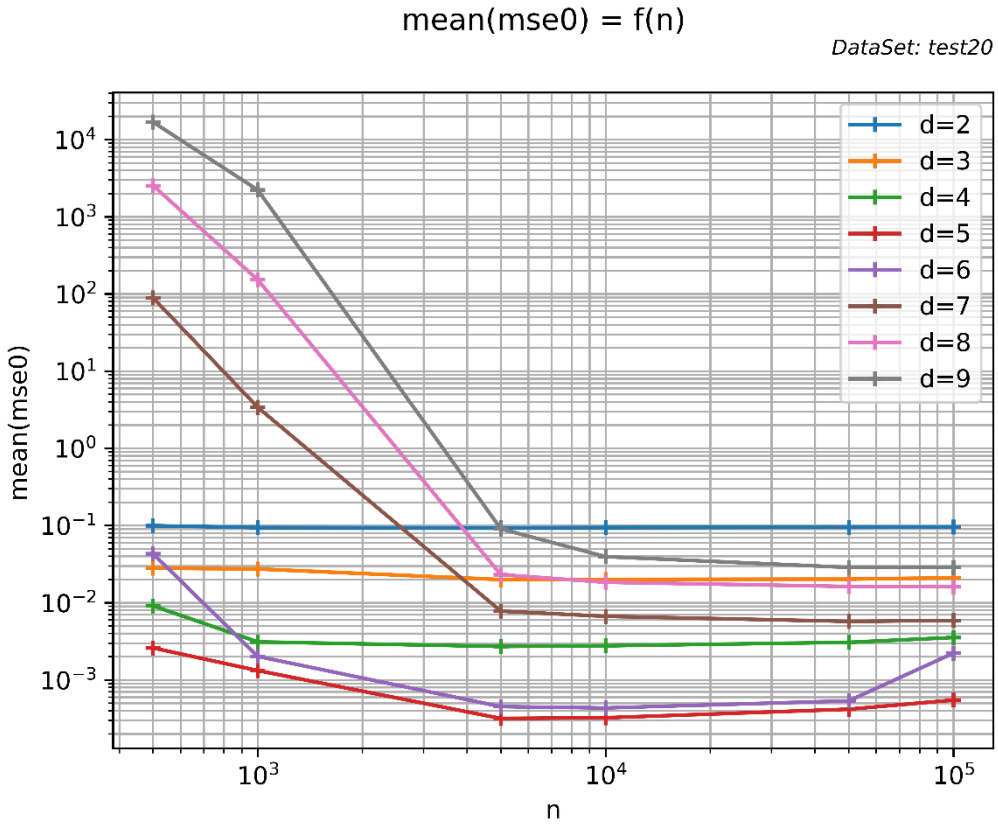


Figure 58 : évolution de la moyenne des erreurs de prédictions en fonction du paramètre d'échantillonnage sur les formes du DataSet test20

L'observation du graphique de la Figure 58 nous montre que les meilleurs résultats d'approximations sont atteints pour les modèles de degrés 4, 5 et 6 (courbes verte, violet et rouge). sans surprise, un *AISM* de degré 2 ne permet pas d'approximer les formes complexes avec une grande précision comme le démontrent les valeurs de $mse0_2$ (courbe bleue). On remarque par ailleurs que les modèles de degrés 8 et 9 ont en moyenne des erreurs de prédictions plus élevées que les modèles de degrés intermédiaires, ce qui peut être entre autres la conséquence de leur trop grand nombre de paramètres à régresser (cf. Tableau 8).

Pour chaque degré de modèle, $mean(mse0)$ converge aux alentours de $n = 5000$ points. Il serait donc potentiellement inutile d'approximer les *AISM* sur un plus grand nombre de points, observation qui est confirmée par les tests suivants.

La seconde analyse consiste à étudier l'évolution du temps moyen d'approximation en fonction de n pour chaque degré d'*AISM*. La Figure 59 illustre le résultat :

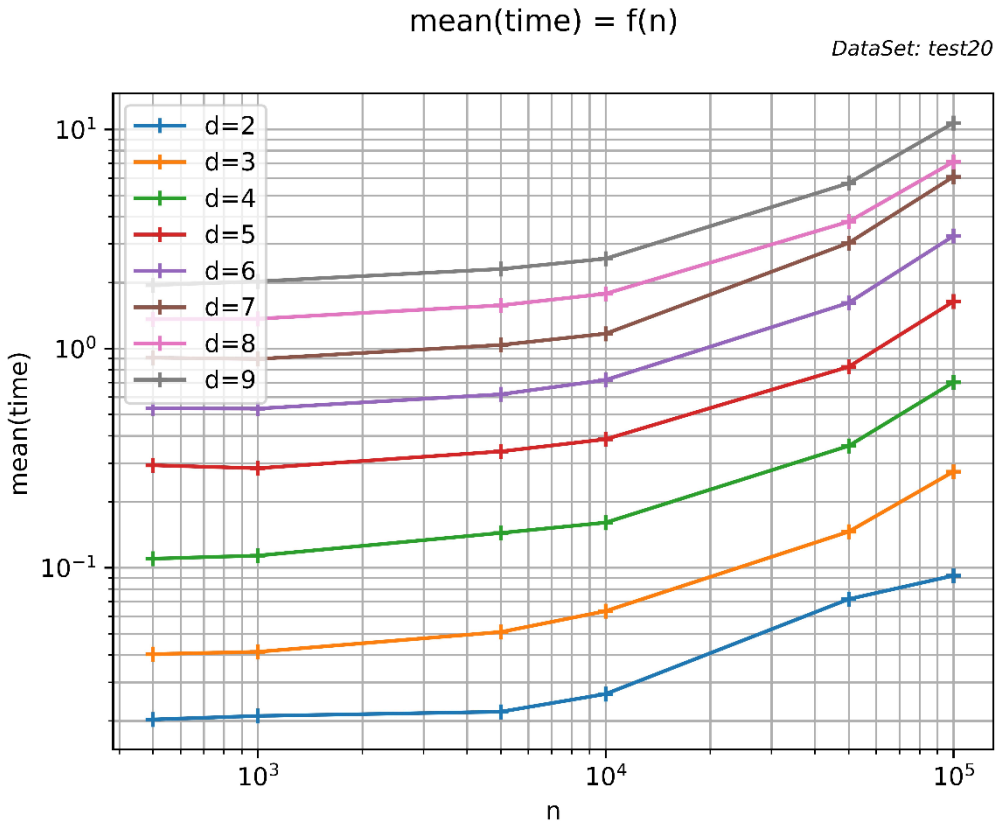


Figure 59: évolution de la moyenne du temps d'approximations en fonction du paramètre d'échantillonnage sur les formes du DataSet test20

Comme nous pouvions le supposer, le temps nécessaire à l'approximation d'un *AISM* augmente avec le degré du modèle. Cela s'explique simplement par le fait qu'un polynôme de degré $d + 1$ est composé de plus de paramètres à régresser qu'un polynôme de degré d. De même, l'augmentation du nombre de points échantillonnés implique une augmentation du temps d'approximation. Dans le but de minimiser au maximum le temps de calcul sans pour autant dégrader les résultats d'approximation des *AISM*, deux conclusions sont tirées :

(a) si les modèles de degrés d et $d + 1$ admettent sensiblement les mêmes erreurs d'approximation, les *AISM* de degré d seront préférés
(b) si les erreurs d'approximation d'un *AISM* de degré d approximé sur $n1$ et $n2$ points (avec $n1 < n2$) sont proches, on préfèrera alors $n1$ comme paramètre d'échantillonage

Pour confirmer l'influence du degré du modèle sur la qualité d'approximation, les dispersion des $mse0_d$ sur l'ensemble des formes du *DataSet* sont analysées séparément pour les degrés 3 à 7 (les degrés 2, 8 et 9 sont ignorés pour plus de lisibilité du graphique).

On se réfère à la Figure 60 pour la lecture des graphiques de dispersions. La Figure 61 représente l'analyse de dispersion des erreurs de prédictions pour le paramètre d'échantillonnage $n = 5000$.

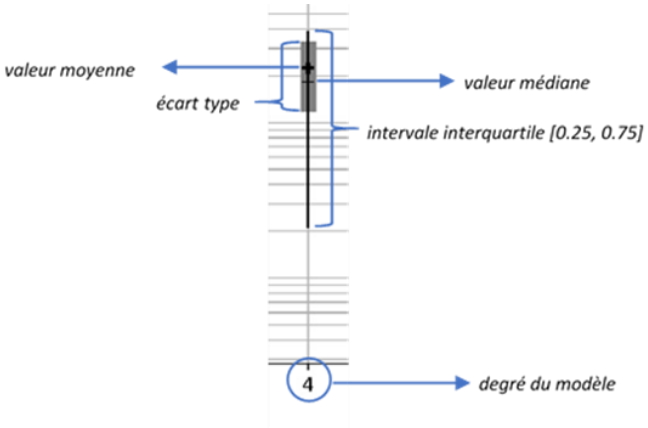


Figure 60: détails pour la lecture des graphiques de dispersions des erreurs d'approximations

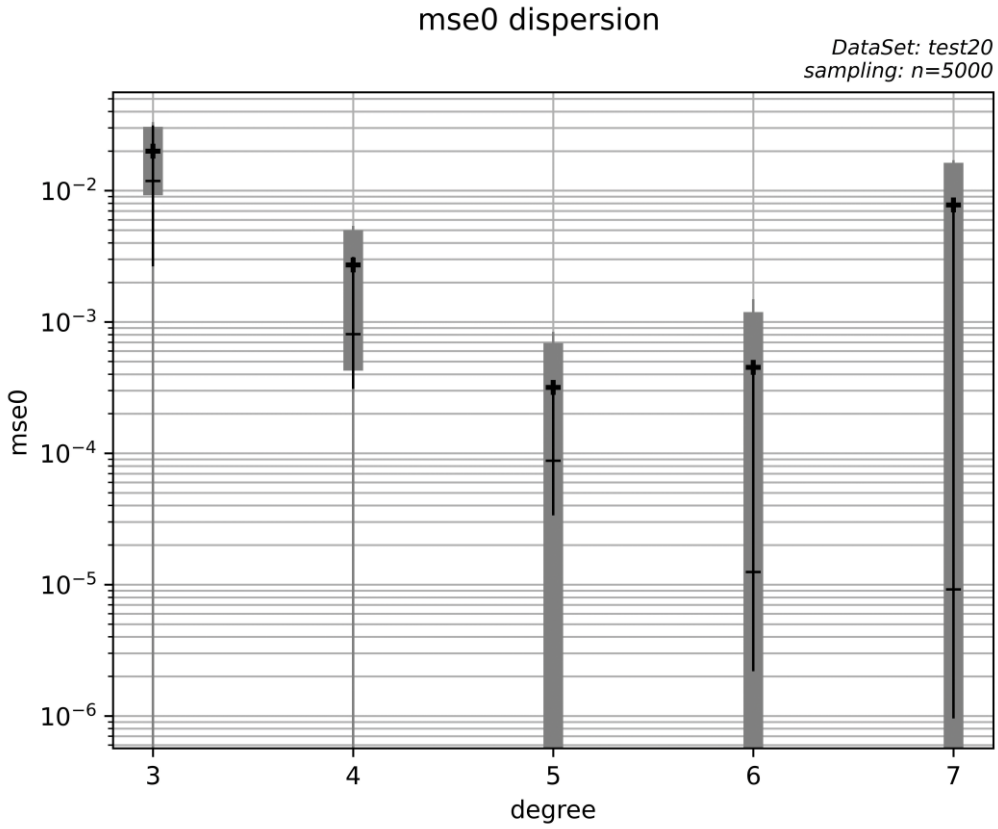


Figure 61 : dispersion des erreurs de prédictions en fonction du degré du modèle sur les formes du DataSet test20

Avec des $mse0_d$ moyen de l'ordre de 1e-4, les modèles de degrés 5 et 6 présentent à première vue les meilleurs résultats d'approximations. La valeur médiane pour les modèles de degrés 7 est la plus faible de toutes avec une valeur d'environ 9e-6, ce qui signifie qu'un modèle de degré 7 admet de meilleures approximations de formes dans la moitié des cas. Cependant, l'observation des écarts types et intervalles interquartiles démontre une plus grande dispersion

des résultats et la présence d'extrémums plus importants. Ainsi, l'analyse des résultats de tests pour le *DataSet* **test20** pousse à préférer les *AISM* de degrés 5 et 6 pour leurs précisions et la plus faible variabilité des résultats.

Avant toute conclusion sur l'expérimentation, les mêmes analyses sont réalisées sur les résultats de tests des *DataSet* **aero100** et **parts100**.

Résultats de tests sur les DataSet aero100 et parts100

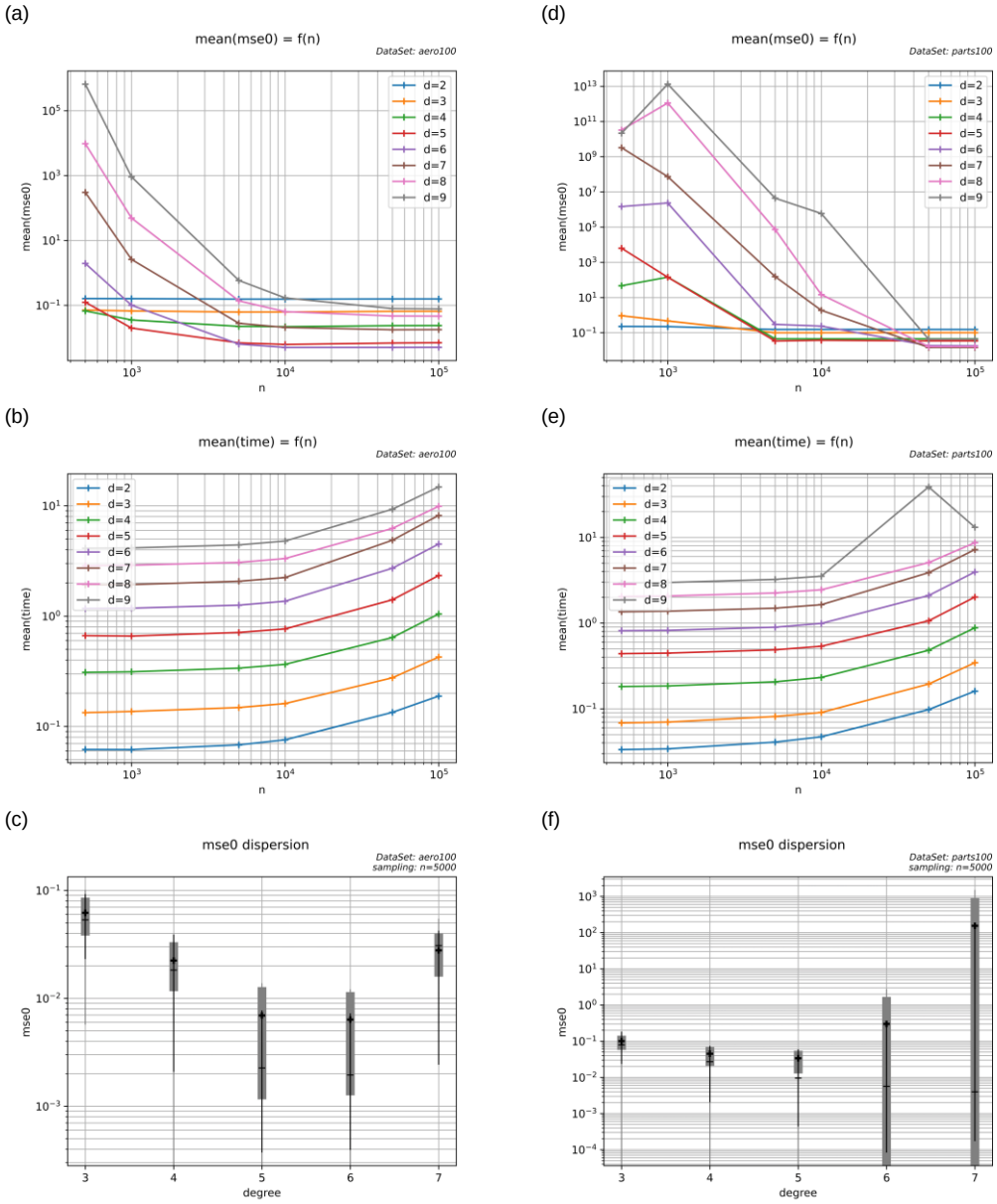


Figure 62 : analyse des résultats de tests d'approximation sur les DataSet aero100 et parts100

Les mêmes graphiques que pour l'analyse des résultats de tests sur **test20** sont présentés en Figure 62 pour **aero100** (colonne de gauche) et **parts100** (colonne de droite). On remarque avant tout les mêmes évolutions du temps d'approximation en fonction du degré du modèle et du paramètre d'échantillonnage (Figure 62 (b) et (e)), confirmant l'intérêt de préférer les modèles de

plus bas degrés, ainsi qu'un plus petit échantillonnage dans la mesure où cela n'impacte pas la qualité de l'approximation.

De manière générale, les modèles de degrés 4, 5 et 6 présentent les meilleurs résultats d'approximations, c'est-à-dire les valeurs les plus faibles et les moins dispersés (Figure 62 (c) et (f)). Il semble logique qu'un modèle de degré trop peu élevé ne permette pas la bonne approximation d'une forme complexe, du fait de son faible nombre de paramètres. La très forte dispersion des erreurs d'approximations des modèles de degrés supérieurs à 7 est probablement la conséquence du phénomène d'*overfitting* (ou surajustement) des modèles de régression. Ce dernier agit lors des tentatives de régressions d'un modèle inutilement complexe sur un échantillon trop faible. Ce risque permet d'affirmer que les *AISM* de degrés 4 à 6 sont les meilleurs candidats.

Sur l'ensemble des tests, l'échantillonnage des maillages à un nombre de points supérieur à $n = 5000$ n'a pas d'impact significatif sur l'amélioration des résultats (Figure 62 (a) et (d)), mais un impact non négligeable sur le temps de calcul. Le paramètre d'échantillonnage peut ainsi être arbitrairement fixé à $n = 5000$.

Une observation majeure, soulevée par l'analyse des graphiques de dispersion des erreurs d'approximations, est que les *AISM* approximent plus précisément les formes des objets de type *Surface*, aux variations de courbures plus faibles et continues, que la forme résultante des objets de type *Part* présentant des angles vifs. La Figure 63 permet de visualiser les erreurs d'approximations en chaque point d'un *AISM* de degré 6 sur des formes d'objets de type *Part* (colonne de gauche) ainsi que celles d'objets de type *Surface* (colonne de droite). On y observe que la typologie de forme impacte fortement la précision de son approximation par un *AISM*. Celles présentant des comme des surfaces sont mieux approximées que les formes résultantes de modèles entiers.

II.1.5. Bilan de la méthode

Cette première expérimentation a prouvé la possibilité d'approximer un *AISM* un nuage de points, et ainsi d'utiliser les paramètres du modèle comme descripteur de forme.

En fixant de manière arbitraire le paramètre d'échantillonnage et le degré du modèle, et dans les conditions spécifiées pour l'expérimentation, le temps moyen d'exécution de la méthode est proche de la seconde. Le temps de calcul des *AISM* comme descripteurs de formes est compatible avec son utilisation à grande échelle. De plus, un *AISM* approximé est stocké sous la forme d'un vecteur de 34, 55 ou 83 paramètres pour les modèles de degrés 4, 5 et 6. La compacité des descripteurs rend possible la capitalisation et le stockage de larges ensembles.

Les *AISM* dépendent de la position spatiale du modèle 3D qu'ils approximent. Bien qu'une normalisation de la position soit appliquée au maillage en amont, de légères variations dues à la topologie du maillage impactent le résultat d'approximation. Le descripteur qui en est extrait n'est donc pas invariant. La conséquence sur la robustesse des méthodes d'analyses est déduite des résultats des expérimentations qui suivent. Si les erreurs d'approximations sur l'ensemble des tests semblent cohérentes (car tendant vers 0), aucune hypothèse sur la distinctivité de ce descripteur ne peut être faite. En effet, le caractère injectif du modèle n'étant pas prouvé pour le moment, nous ne pouvons pas affirmer qu'un *AISM* approximé sur S_1 n'approxime pas aussi bien une forme S_2.

Enfin, on note que le choix arbitraire de fixer le paramètre d'échantillonnage au même nombre de points n'est probablement pas optimisé. Par exemple, il est probable que l'approximation d'un *AISM* pour un modèle 3D présentant de fortes variations de courbure nécessite plus de points que pour une surface ouverte plus lisse. La suite de cette expérimentation se tourne ainsi vers la détermination d'une fonction qui définirait le paramètre d'échantillonnage en fonction de certains attributs de la forme, comme son aire de surface ou ses variations de courbure par exemple.

En conclusion, la possibilité d'approximer un *AISM* comme descripteur de forme de modèles 3D est confirmée par la première expérimentation. Les prochaines expérimentations visent à en démontrer l'intérêt.

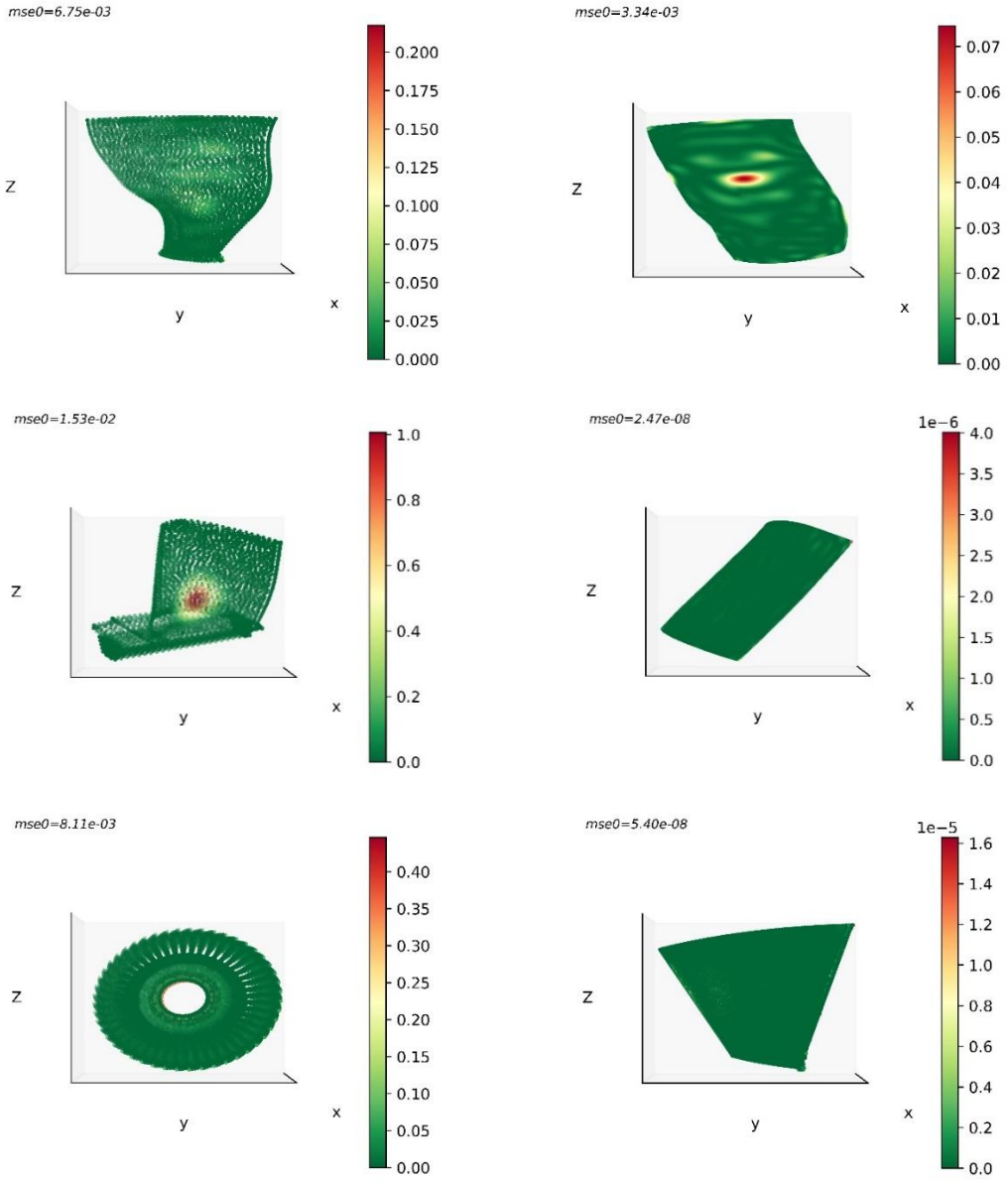


Figure 63: erreurs d'approximation d'un AISM de degré 6

II.2. Exp-2 : Validation de la méthode de *fitting*

Dans cette seconde sous-partie, l'étude expérimentale visant à valider la méthode de *fitting* des *AISM* sur des formes 3D est détaillée.

Appliquée à un objet de type *Mesh* du *KBRE model*, la méthode $fit\_aism(aism\_params, method)$ optimise les paramètres de transformation rigide du maillage permettant de minimiser le coût de *fitting*, c'est-à-dire l'erreur de prédiction de l'*AISM* f_d sur les n sommets du maillage transformé.

Autrement dit, le *fitting* de f_d sur pc consiste à optimiser les paramètres de transformation rigide (θ_x, θ_y, θ_z, t_x, t_y, t_z) pour minimiser :

$$fit\_cost_d = \frac{1}{n}\sum_{j=0}^{n} f_d(x_{t,j},\ y_{t,j},\ z_{t,j})^2$$

avec $p_{t(}x_t,\ y_t,\ z_t)$ la transformée de $p(x, y, z)$ par M, la matrice de transformation générée avec les paramètres (θ_x, θ_y, θ_z, t_x, t_y, t_z). La méthode $fit\_aism(aism\_params, method)$ retourne les paramètres optimisés fit_cost_d ainsi que M.

II.2.1. Hypothèses et critère de validation

Soit $mesh0$ un maillage et f_d un *AISM* de degré d approximé sur $mesh0$. On appelle $mse0_d$ l'erreur d'approximation de f_d sur $mesh0$.

Soit $mesh$ une copie de $mesh0$ ayant été transformée par $M2$. On appelle fit_cost_d le coût de fitting de f_d sur $mesh$ et M la matrice de transformation optimisée.

On en déduit alors $fit_{error_d} = (mse0_d - fit\_cost_d)^2$ et M_{error_d} la norme Euclidienne, ou *L2 norm*, entre les matrices $M2$ et M^{-1}, appelés respectivement erreur de *fitting* et erreur de recalage. Nous formulons l'hypothèse que, si $fit\_error_d < \varepsilon_1$ et $M\_error_d < \varepsilon_2$, alors $mesh$ est parfaitement recalée avec $mesh0$. Le *fitting* de f_d sur $mesh$ est alors considéré comme réussi.

Les valeurs ε_1 et ε_2 sont des seuils empiriques pour le moment inconnus. L'expérimentation consiste donc à déterminer les paramètres de la méthode de fitting pour lesquels les valeurs moyennes de ε_1 et ε_2 sont minimales. Un second critère d'évaluation est le temps de calcul. Plus le *fitting* des *AISM* est rapide, plus il sera envisageable de l'utiliser pour le *labeling* de forme.

II.2.2. Paramètres de test

Différents paramètres influencent le résultat de la méthode de *fitting* sur une forme 3D :

- **le degré d du polynôme** détermine le nombre de paramètres du modèle, et influence donc le temps et le résultat d'optimisation.
- **La dimension n de pc** détermine le nombre de points échantillonnés sur lesquels le modèle est fitté. Nous cherchons à déterminer n minimum à partir duquel ε_1 et ε_2 ne diminuent plus.
- **La méthode d'optimisation :** Trois méthodes de minimisation d'une fonction scalaire de plusieurs paramètres disponibles dans l'algorithme d'optimisation implémenté[8] sont comparées en termes de temps d'optimisation et de résultat. Il s'agit des méthodes d'optimisation POWELL, SLSQP et L-BFGS-B.

[8] https://docs.scipy.org/doc/scipy/reference/generated/scipy.optimize.minimize.html

II.2.3. Méthode de test

L'algorithme suivant détaille la méthodologie de génération des résultats de test de la méthode de *fitting* des *AISM* sur les objets d'un *DataSet*. On note que, en amont de l'expérimentations, des maillages représentatifs des objets agrégés ont été générés et stockés dans l'attribut *meshes* de l'instance.

Algorithme 4 : test de la méthode de fitting des AISM sur les formes d'un DataSet

```
inputs :
        DataSet                                  # objet DataSet
        degrees = [ 4, 5, 6 ]                    # degrés des AISM à tester
        sample_values = [ 1e^2 , 5^e2 , 1e^3 , 5^e3 , 1e^4]   # paramètres d'échantillonnages à tester
        methods = [ 'SLSQP', 'POWELL', 'L-BFGS-B' ]   # méthodes d'optimisation à tester
returns :
        test_results                             # retourne les résultats dans un tableau de
                                                 la forme id/d/n/fit_error/M_error/time

1.  meshes = DataSet.meshes                      # liste des maillages

2.  for mesh0 in meshes :                        # pour chaque maillage :
3.      id = mesh0.id                            # récupération de son index
4.      mesh0.norm_pose()                        # normalisation de la position du maillage

5.      for n in sample_values :                 # pour chaque paramètre d'échantillonnage :
6.          mesh = mesh0.copy()                  # copie du maillage d'origine
7.          mesh.sample(n)                       # échantillonnage de la copie à n sommets
8.          M2 = random_matrix()                 # génération d'une matrice de transformation
                                                 aléatoire
9.          Mesh.apply_transform(M2)             # transformation rigide de la copie

10.         for d in degrees :                   # pour chaque degré de modèle :
11.             AISM_params, mse0 = mesh0.approx_AISM(d)   # approximation d'un AISM de degré d sur le
                                                 maillage d'origine
12.             for method in methods :          # pour chaque méthode de fitting :
13.                 fit_cost,  M  =  mesh.fit_AISM(AISM_params,   # fitting de l'AISM du maillage d'origine sur la
    method)                                      copie transformée
14.                                              # calcul de l'erreur de fitting
15.                 fit_error = (mse0 – fit_cost)**2   # calcul de l'erreur de recalage
16.                 M_error = (M -  M2^-1)**2    # les information du test sont ajoutées aux
17.                 add line id / d / n / fit_error / M_error / time to   résultats
                    test_results
18.
    return test_results                          # tableau récapitulatif des tests réalisés
```

II.2.4. Résultats

Les tests de *fitting* sont réalisés sur les instances de *DataSet* **test20**, **aero100** (sous *DataSet* contenant les 100 premières instances de *Surface* de **aero6000**), et **parts100** (sous *DataSet* contenant les 100 premières instances de *Part* de **parts1759**)

L'analyse des résultats est détaillée pour **test20**. Les résultats sur les autres *DataSet* sont ensuite synthétisés.

Résultats de tests sur le DataSet test20

La première analyse consiste à analyser l'influence qu'ont le degré de l'*AISM*, le paramètre d'échantillonnage ainsi que la méthode d'optimisation sur les erreurs de *fitting* et de recalage.
La Figure 64 présente l'évolution de la médiane des erreurs de *fitting* en fonction du paramètre d'échantillonnage pour chaque combinaison possible entre degré de modèle et méthode d'optimisation étudiée.

Nb : la médiane est utilisée plutôt que la moyenne pour plus de lisibilité

La Figure 65 présente quant à elle l'évolution de la médiane des erreurs de recalage (*i.e.* l'erreur entre la matrice optimisée par la méthode de *fitting* et la matrice de recalage théorique) en fonction du paramètre d'échantillonnage pour chaque combinaison possible entre degré de modèle et méthode d'optimisation étudiée.

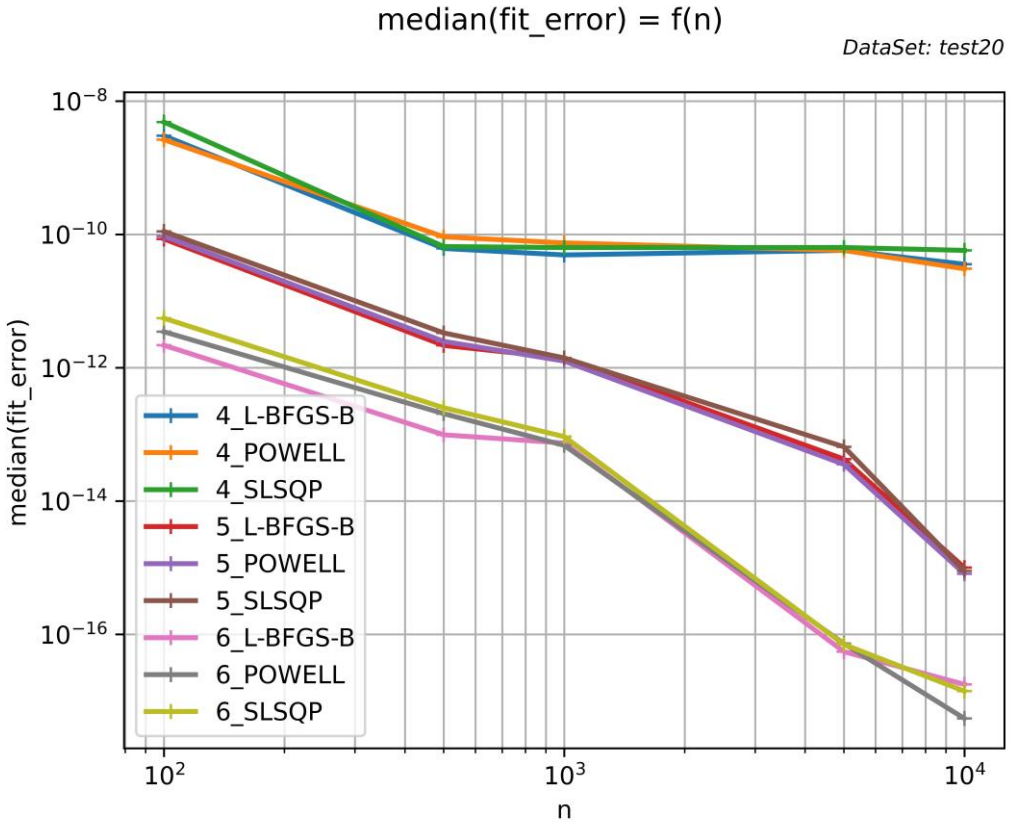


Figure 64: évolution de la médiane des erreurs de fitting en fonction du paramètre d'échantillonnage sur les formes du DataSet test20

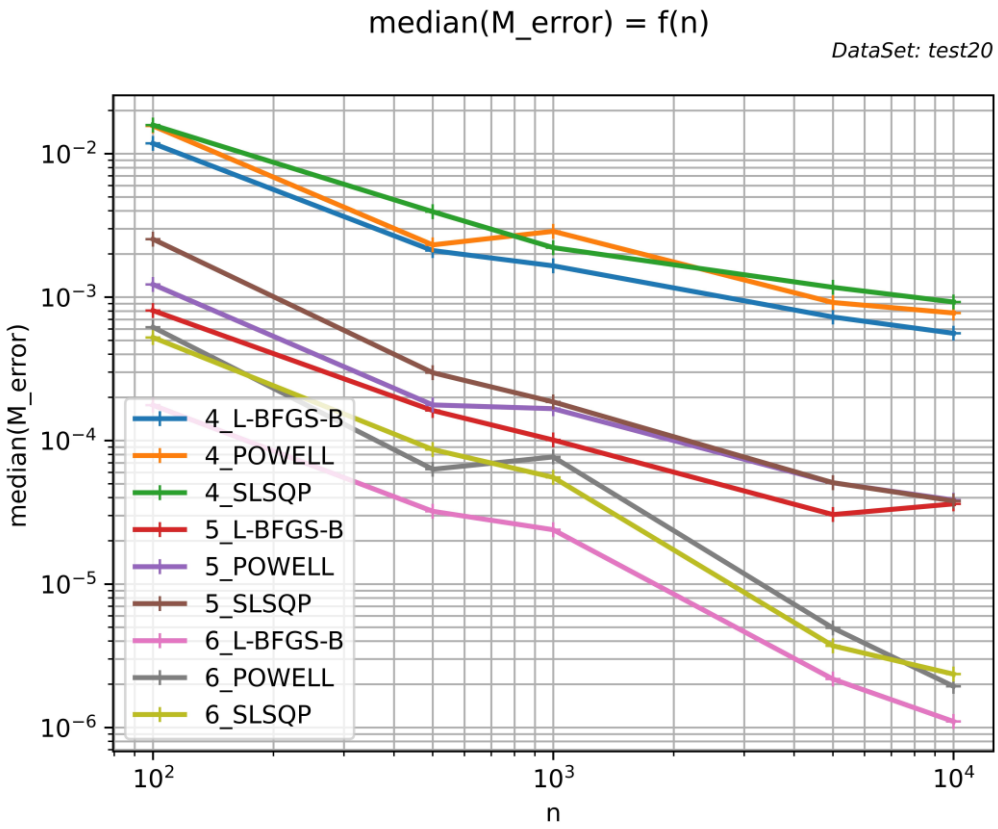


Figure 65 : évolution de la médiane des erreurs de recalage en fonction du paramètre d'échantillonnage sur les formes du DataSet test20

On observe dans un premier temps que les *AISM* de degrés 6 admettent les meilleurs résultats de *fitting*, ce qui se confirme avec l'analyse de l'erreur de recalage des modèles de degrés 6 qui est toujours plus faible que celles des *AISM* de degrés 4 et 5 à iso-échantillonnage.

La superposition relative des courbes représentatives des trois méthodes d'optimisations pour un degré de modèle donné démontre l'impact négligeable de ce paramètre sur les résultats. Ainsi, les méthodes POWELL, SLSQP et L-BFGS-B n'admettent pas de différence significative en termes de performances de *fitting*.

Enfin, la corrélation entre l'augmentation du nombre de points échantillonnés sur une forme et l'amélioration des résultats de *fitting* est mise en évidence par la Figure 64 et la Figure 65. En effet, les erreurs de *fitting* diminuent à mesure que le nombre de points échantillonnés augmente, et semblent commencer à converger entre $n = 5e3$ et $n = 1e4$.

Pour décider des paramètres optimaux, l'évolution du temps de calcul nécessaire au *fitting* des *AISM* en fonction de paramètre d'échantillonnage n est observée sur le graphique Figure 66. On y observe une augmentation quasi linéaire du temps moyen de *fitting* en fonction de n, et ce pour l'ensemble des degrés/méthodes étudiés. L'optimisation étant chronophage, il est par conséquent nécessaire de limiter au maximum le nombre de points échantillonnés sur les formes avant *fitting* d'un *AISM* sur cette dernière. On remarque aussi sur la Figure 66 que la méthode SLSQP est, de manière non négligeable, plus rapide que les méthodes POWELL et L-BFGS-B. Par exemple, le *fitting* d'*AISM* degré 6 sur $n = 1e4$ points avec la méthode SLSQP est en moyenne de 55 secondes versus 210 secondes avec la méthode POWELL et 140 secondes avec L-BFGS-B. Puisque les méthodes n'influent pas significativement sur le résultat, la méthode SLSQP est pour le moment favorite.

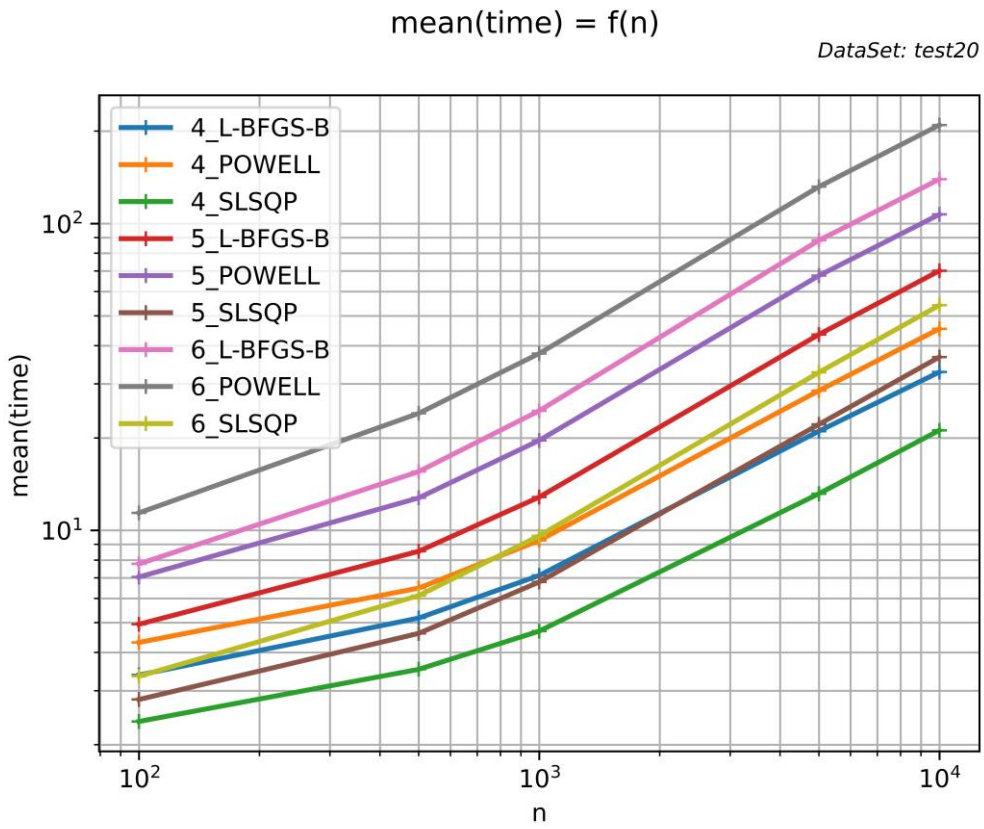


Figure 66 : évolution de la moyenne du temps de fitting en fonction du paramètre d'échantillonnage sur les formes du DataSet test20

Dans le but de valider les observations précédentes, les mêmes analyses sont réalisées sur les formes de deux autres *DataSet*. La Figure 67 regroupe les graphiques d'analyse des résultats de tests de *fitting* sur **aero100** (colonne de gauche) et **parts100** (colonne de droite).

Résultats de tests sur les DataSet aero100 et parts100

Les résultats sont compilés dans la Figure 67. Ces nouvelles informations confirment l'observation qu'aucune méthode d'optimisation ne surpasse les autres. Cependant, la méthode SLSQP est préférée car généralement bien plus rapide que les deux autres (Figure 67 (c) et (f)).

De manière globale, les résultats commencent à converger entre $n = 5e^3$ et $n = 1e^4$ (Figure 67 (a) et (d)). Dans certains cas, des améliorations notables des résultats pour un paramètre d'échantillonnage plus important doivent être considérées, mais la nécessité de limiter le temps de calcul nous pousse à fixer $n = 5e^3$ comme compromis performance / temps d'exécution de la méthode.

Pour une méthode et un échantillonnage donné, le *fitting* de modèles de degré 6 est incontestablement meilleur en termes d'erreur de *fitting* et d'erreur de recalage. Cependant, le temps de *fitting* des *AISM* de degrés 6 étant plus élevé que celui des modèles de degrés 4 et 5, ces trois modèles seront évalués pour la méthode de *labeling*.

Si la méthode de *fitting* est performante pour le recalage de formes représentatives de surfaces d'aubages (La Figure 67 (b)), l'erreur médiane de recalage est plus élevée pour les formes représentatives de modèles entier du *DataSet* **parts100** (La Figure 67 (e)). L'impact de la typologie de forme sur le *fitting* d'un *AISM*, et donc sur la méthode de *labeling*, est étudié dans l'expérimentation 6.

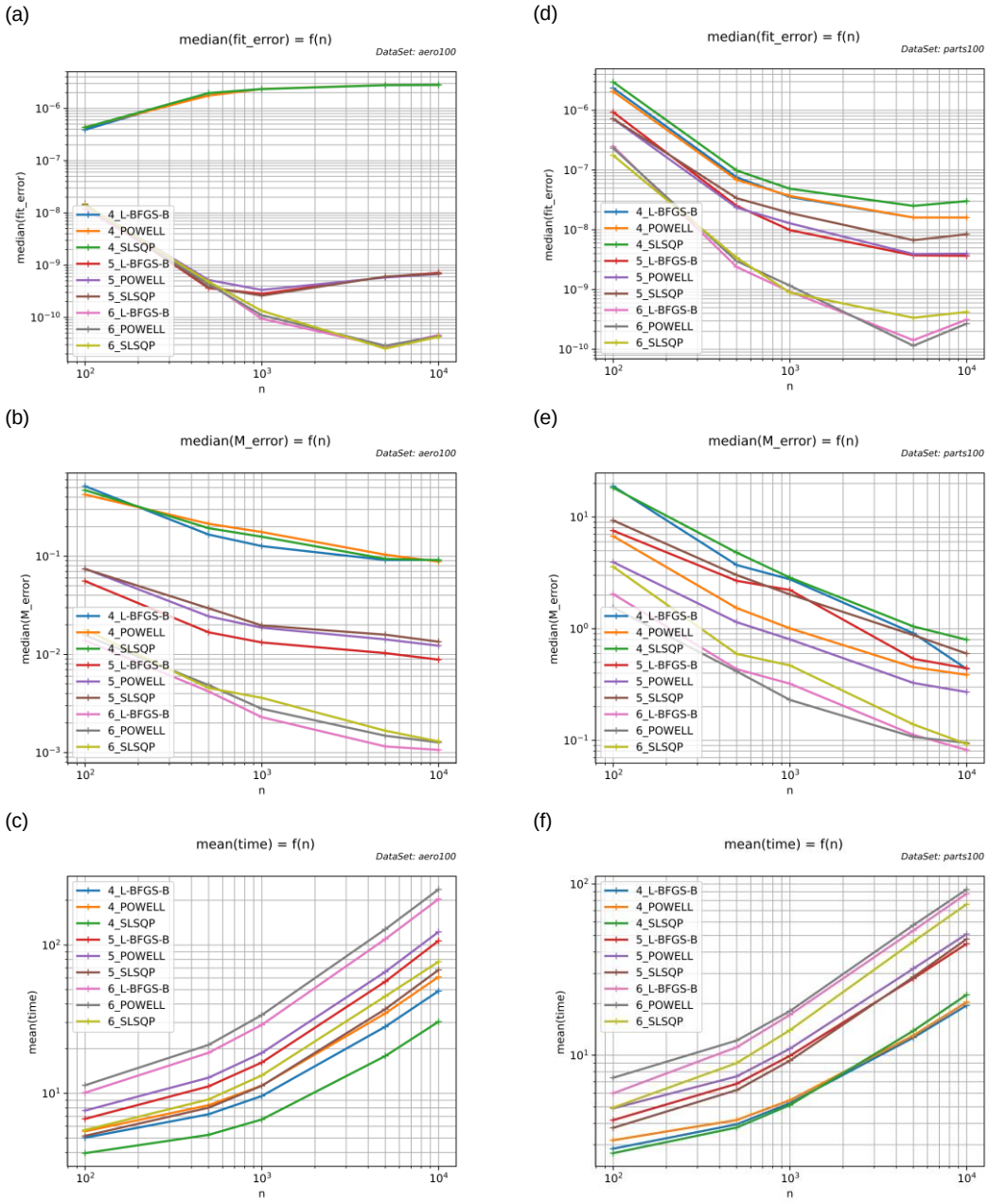


Figure 67: analyse des résultats de tests de fitting sur les DataSet aero100 et parts100

II.2.5. Bilan de la méthode

La seconde expérimentation a permis de valider la méthode de *fitting* d'un *AISM* sur un nuage de points échantillonné sur un maillage 3D. En fixant de manière arbitraire le paramètre d'échantillonnage, le degré du modèle ainsi que la méthode d'optimisation, et dans les conditions spécifiées pour l'expérimentation, le temps moyen d'exécution de la méthode se situe entre 30 et 60 secondes. Puisque la méthode de *fitting* d'un *AISM* n'a pas pour vocation d'être réalisé à grande échelle, elle est considérée comme compatible avec le critère de temps d'exécution pour son utilisation dans le cadre de la méthode de *labeling*. Il convient cependant de limiter le nombre de *fitting* d'*AISM* sur une forme pour son *labeling*. Par ailleurs, la méthode n'implique aucun besoin de stockage de donnée supplémentaire.

Puisque l'approximation des *AISM* sur un maillage retourne un modèle non invariant à la position spatiale, la méthode de *fitting* est développée pour recaler un maillage analysé dans la même position normalisée que lors de l'approximation de l'*AISM* en question, s'il s'agit de la même forme. Au vu des résultats de l'expérimentation, la robustesse de la méthode à la position initiale ainsi qu'aux variations topologiques des maillages est validée. On note cependant une faiblesse pour le recalage de formes résultantes de modèles globaux de pièces.

Si cette expérimentation valide le prérequis à la méthode de labélisation, nous ne pouvons cependant toujours pas affirmer l'injectivité des *AISM* à ce stade. L'atteinte du niveau III de distinctivité sera prouvée par l'expérimentation 6 sur la méthode de *labeling*. Avant cela, trois expérimentations intermédiaires sont présentées dans le but d'évaluer les niveaux de distinctivités I et II des *AISM* dans le cadre des méthodes de *classification* et de *retrieval*.

II.3. Exp-3 : Évaluation de la méthode de classification

L'expérimentation qui suit permet d'évaluer des résultats de *classification* par prédiction d'un *MLP-C* entrainé à la *classification* de forme.

Appliquée à un objet dont le descripteur de forme D_{name} a été calculé en amont, la méthode $classify(mlpc, D_{name})$ prédit la catégorie (i.e. famille de formes) à laquelle l'objet peut être associé.

Cette expérimentation permet de comparer plusieurs descripteurs et d'évaluer les capacités distinctives de niveau I dans le cadre de la méthode implémentée dans le *KBRE model*. Les résultats sont aussi comparés avec ceux de trois outils de la littérature scientifique.

II.3.1. Hypothèses et critère de validation

Soit $dataset$ un objet *DataSet* qui agrège n objets de type *Shape* (ou d'une classe fille). Ces objets sont répartis dans deux ensembles X_train et X_test constitués respectivement de n_{train} et n_{test} objets. Un descripteur D_name est calculé pour chaque objet de X_train et X_test.

On note Y_train la liste des catégories y_i connues de chaque x_i de X_train. y_i prend une valeur l_i de L, l'ensemble fini des m familles de formes existantes parmi X.

Un objet $mlpc$ est entrainé via la fonction $train\_mlpc(X\_train, Y\_train, D\_name)$ à prédire la bonne valeur $\hat{y}_i = y_i$ associée au descripteur D_name de chaque x_i.

On appelle maintenant Y_test la liste des catégories y_i des n_{tests} objets x_i de X_test.

$\hat{Y}$ est l'ensemble des résultats $\hat{y}_i$ de la méthode $classify(mlpc, D\_name)$ appliquée à chaque x_i de X_test. $\hat{y}_i$ prend nécessairement une valeur l_i de L.

Le résultat de *classification* d'un objet x_i est correct si $\hat{y}_i = y_i$, c'est-à-dire si la catégorie prédite est identique à la « vrai » catégorie connue de l'objet en question.

Le score d'***Accuracy*** est utilisé comme critère d'évaluation principal des tests de *classification* pour chaque descripteur :

$$accuracy\_score(Y\_test, \hat{Y}) = \frac{1}{n_{test}} \sum_{i=1}^{n_{test}} 1(\hat{y}_i = y_i)$$

$$\text{Avec } 1(a) := \begin{cases} 1 \; if \; a = True \\ 0 \; if \; a = False \end{cases}$$

II.3.2. Paramètres de test

Différents paramètres influencent les résultats de la méthode de *classification* :

- **Le type de descripteur** utilisé pour représenter les formes. Six descripteurs sont évalués, à savoir les *AISM* de degrés 4, 5 et 6, ainsi que les SD 2, 3 et 8 (cf. Annexe 3)
- **Les hyper paramètres du *MLP-C***, soit le paramétrage interne des fonctions du réseau de neurones. En amont de cette expérimentation, la méthode *Grid Search* (Ngoc et al., 2021) implémentée dans la librairie Scikit-Learn (Buitinck et al., 2013; Pedregosa et al., 2011) a permis de fixer certains hyper paramètres du ML, à savoir la fonction d'activation, la structure du réseau, ainsi que la tolérance.
- **Le nombre de données d'entrainement n_{train}** : seules les données d'origine des *DataSet* sont utilisés dans le cadre de l'expérimentation, sans recours à l'augmentation de données.

II.3.3. Méthode de test

L'algorithme suivant retourne les résultats de test de la méthode de *classification* sur les formes d'un ***DataSet*** pour un descripteur de forme donné.

On note que le descripteur D_name de chaque objet agrégé est calculé en amont.

Algorithme 5 : test de la méthode de classification sur les formes d'un DataSet

```
inputs :
        X_train, X_test          # ensembles d'objets d'entrainement et de test
        Y_train, Y_test          # listes des catégories des objets de X_train et X_test
        D_name                   # nom du descripteur de formes utilisé

returns :
        accuracy                 # taux de réussite des tests de classification

1.  mlpc = train_mlpc(X_train, Y_train, D_name)   # entrainement d'un objet MLP-C
2.  score = 0                                     # initialisation du score

3.  for i in range(n_test) :                      # avec n_test le nombre d'objets de X_test
4.      x = X_test[ i ]                           # attribution du ie objet de X_test à x
5.      y = Y_test[ i ]                           # y prend la i-e valeur de Y_test
6.      y^ = x.classify(mlpc, D_name)             # méthode de classification de x

7.      if y^ = y :                               # si la prédiction est identique à la « vraie »
8.          score = score + 1                     # catégorie, le score est incrémenté de 1
                                                  # calcul du résultat d'accuracy
9.  accuracy = score / n_test
                                                  # retourne les résultats du test
10. return accuracy
```

II.3.4. Résultats

Les tests de *classification* ont été effectués sur les deux *DataSet* publics de la littérature ***model*net40** et **mcbA**, constitués de maillages préalablement catégorisés. Ces derniers sont découpés en un ensemble de données d'entrainement et un ensemble de données de test.

On peut noter qu'il s'agit de maillages 3D représentatifs de modèles complets et non de surfaces unitaires. L'absence de jeu de données privé constitué de modèles 3D catégorisés n'a pas permis de mener l'expérimentation sur des données aéronautiques.

Le Tableau 9 présente les résultats de tests sur les six descripteurs étudiés. Les performances de trois méthodes de *classification* de la littérature scientifique, à savoir Point*CNN* (Y. Li et al., 2018), PointNet++ (Qi, Yi, et al., 2017) et Spider*CNN* (Y. Xu et al., 2018), sont présentées à titre de comparaison. Chacune de ces méthodes est basée sur une représentation sous forme de nuage de points. Les résultats sont extraits de (Kim et al., 2020).

En plus de la métrique ***Accuracy***, les métriques ***Average Precision*, *Average Recall* et *F1-Score*** sont fournies dans le tableau ci-dessous. Ces dernières permettent d'analyser plus finement les résultats en considérant les variations entre les différentes catégories. L'Annexe 4 fournit les précisions nécessaires concernant les métriques d'évaluation.

DataSet	Descriptor / method	Accuracy	Avg Precision	Avg Recall	Avg F1-score	training time (s)
mcbA	PointCNN	93,89%	90,13%	81,85%	0,84	NA
	PointNet++	87,45%	73,45%	73,68%	0,75	NA
	SpiderCNN	93,59%	86,64%	79,70%	0,81	NA
	AISM_4	70,61%	50,44%	45,95%	0,46	97,8
	AISM_5	71,32%	48,90%	43,71%	0,45	66,3
	AISM_6	73,04%	52,75%	45,31%	0,47	60,2
	SD_2	76,05%	52,83%	45,47%	0,46	48,4
	SD_3	77,28%	57,12%	51,75%	0,53	59,5
	SD_8	88,20%	68,72%	63,75%	0,65	76,2
modelnet40	PointCNN	92,20%	NA	NA	NA	NA
	PointNet++	91,90%	NA	NA	NA	NA
	SpiderCNN	92,40%	NA	NA	NA	NA
	AISM_4	29,03%	23,22%	20,03%	0,18	6,0
	AISM_5	27,11%	24,06%	18,76%	0,18	8,8
	AISM_6	25,18%	25,06%	17,18%	0,16	5,5
	SD_2	42,88%	37,25%	33,72%	0,32	6,9
	SD_3	27,19%	18,45%	19,02%	0,16	2,9
	SD_8	36,17%	30,64%	27,40%	0,26	5,7

Tableau 9 : résultats de classification sur les DataSet mcbA et modelnet40

Le temps d'exécution de la méthode de *classification* est principalement impacté par la phase d'apprentissage du *MLP-C*. On remarque que, sur les 47000 données d'entrainement du *DataSet* **mcbA**, cette dernière ne dépasse pas les 100 secondes. L'apprentissage sur les 10000 données de ***model*net40** ne dépasse pas les 10 secondes. La prédiction par un *MLP-C* entrainé nécessite quant à elle moins d'une seconde.

Les meilleurs scores *d'Accuracy* sont atteints avec les descripteurs de type *SD*, non concurrencés par les *AISM* sur la typologie de formes analysée.

Si les résultats *d'Accuracy* entre 70% et 88% de la méthode de *classification* s'approchent de ceux des modèles Point*CNN*, PointNet++ et Spider*CNN* sur **mcbA**, il n'en est pas de même

sur ***model*net40** avec des résultats ne dépassant pas les 43%, atteint avec le descripteur SD_2. Plusieurs hypothèses sont avancées pour expliquer cette différence. Tout d'abord, les descripteurs ont été calculés sur les modèles 3D en l'état, sans étape de prétraitement amont. Or, les maillages de ***model*net40** présentent une densité de sommets à la fois faible et non-uniforme, pouvant avoir un impact sur le calcul des descripteurs. De plus, la faible quantité des données d'entrainement (moins de 10000) pour 40 catégories peut quant à elle limiter l'apprentissage du *MLP-C*. Une étape de *data-augmentation* pourrait permettre de régler ce problème.

II.3.5. Bilan de la méthode

L'objectif de cette expérimentation est d'évaluer les capacités des descripteurs de formes à en capturer les caractéristiques distinctives et celles d'un outil d'apprentissage supervisé à reconnaitre les caractéristiques communes à une famille de forme.

Puisque la phase d'apprentissage n'est à réaliser qu'une seule fois, et que le temps de prédiction d'un *MLP-C* entrainé est inférieur à la seconde, le temps d'exécution de la méthode rend son utilisation tout à fait envisageable.

La méthode implémentée dans le *KBRE model* présente des résultats prometteurs, bien que ne surpassant pas celles de la littérature scientifique. Plusieurs axes d'amélioration sont par ailleurs envisageables.

En premier, la validation des données d'apprentissage et de test est nécessaire pour ne pas analyser des résultats potentiellement biaisés. En effet, des descripteurs calculés sur des maillages de densités non uniforme ne sont pas parfaitement représentatifs de cette dernière. Un prétraitement des maillages serait donc nécessaire selon des critères empiriques permettant de confirmer la robustesse des descripteurs calculés.

L'optimisation des paramètres du MLPC ainsi que l'augmentation des données d'entrainement sont des pistes d'améliorations non négligeables pour permettre l'apprentissage d'un MLPC avec de meilleures capacités à reconnaitre les caractéristiques distinctives communes des descripteurs de forme d'une même famille.

L'expérimentation permet d'observer une distinctivité de niveau I des *AISM* inférieure à celle des SD sur les typologies de formes des *DataSet* **mcbA** et ***model*net40**. La création d'un ***DataSet*** catégorisé constitué de formes de surfaces unitaires permettrait une plus grande complétude de l'expérimentation afin d'en tirer des conclusions plus exhaustives sur les *AISM* et la méthode de *classification*.

On remarquera enfin l'intérêt du *KBRE model* qui, en proposant la fonctionnalité de capitalisation et de sauvegarde de descripteurs, permet de récupérer ces données et de s'en servir pour l'entrainement et l'apprentissage d'un modèle d'apprentissage supervisés. À la suite de cet apprentissage, toute forme analysée peut être traduite en objet du *KBRE model* afin de prédire son appartenance à une catégorie spécifique d'une taxonomie de familles de formes connue et référencée dans un *DataSet* particulier. Cette information acquise concernant la forme analysée permet l'inférence de nouvelles connaissances par un utilisateur dans le cadre d'activités de *RE*.

II.4. Exp-4 : Évaluation de la méthode de *retrieval*

Dans cette sous-partie, l'étude expérimentale visant à valider la méthode de *retrieval* par *MLP-C* est détaillée.

Appliquée à un objet dont le descripteur de forme D_name a été calculé en amont, la méthodes $retrieve\_mlpc(mlpc, D\_name, k)$ permet d'identifier les index des k modèles 3D agrégés dans un objet de type *DataSet* dont les formes sont les plus similaires à celle de l'objet analysé.

Cette expérimentation permet de comparer plusieurs descripteurs et d'évaluer leurs capacités distinctives de niveau II dans le cadre de la méthode de *retrieval par MLP-C* implémentée dans le *KBRE model*.

Par ailleurs, la confrontation des résultats de la méthode de *retrieval* par *MLP-C* avec ceux de la méthode « classique » de *retrieval* par *matching* permet de déterminer les avantages et limites de l'utilisation d'un modèle complexe d'apprentissage par rapport à une méthode d'analyse plus simple.

II.5.1. Hypothèses et critère de validation

Soit $dataset$ un objet *DataSet* qui agrège n objets de type *Shape* (ou d'une classe fille). On appelle X l'ensemble des n objets agrégés par $dataset$.

Les ensembles X_train et X_test sont chacun constitués de $n_{samples} = n * n\_copies$ maillages 3D subdivisés, aléatoirement transformés et échantillonnés de chaque x_i de X, générés par la méthode $data\_augmentation(n_{copies})$ appliquée au *DataSet*. Pour chacun, un descripteur de forme de type D_name est calculé par la méthode $shape\_descriptor(D\_name)$.

Soient Y_train et Y_test les ensembles regroupant les index de formes y_i associées à chaque x_i de X_train et Y_train. y_i prend une valeur l_i de L, l'ensemble fini des n index des objets capitalisés de X.

Un objet $mlpc$ est entrainé via la fonction $train\_mlpc(X\_train, Y\_train, D\_name)$ à prédire la bonne valeure $\hat{y}_i = y_i$ associée au descripteur D_name de chaque x_i de X_train.

$\hat{Y}$ est l'ensemble des $n_{samples}$ listes $\hat{y}_i$ des $k > 1$ résultats de la méthode $retrieve\_mlpc(mlpc, D\_name, k)$ appliquée à chaque x_i de X_test. On a alors $\hat{y}_i = (\hat{y}_{i,1}, \dots, \hat{y}_{i,k})$ avec $\hat{y}_{i,j}$ un index l_i de L. Le résultat de *retrieval* d'un objet x_i de X_{test} est correct si l'index de son objet source associé y_i se trouve dans $\hat{y}_i$.

Le score de ***Top-k Accuracy*** est utilisée comme critère d'évaluation des résultats de tests de la méthode de *retrieval* :

$$top - k\_accuracy\_score(Y, \hat{Y}) = \frac{1}{n_{samples}} \sum_{i=1}^{n_{samples}} \sum_{j=1}^{k} 1(\hat{y}_{i,j} = y_i)$$

$$\text{Avec } 1(a) := \begin{cases} 1 \; if \; a = True \\ 0 \; if \; a = False \end{cases}$$

Nb : De manière similaire, le score de ***Top-k Accuracy*** est calculé pour la méthode $retrieve\_match(dataset, D\_name, k)$ sur l'ensemble X_{test}.

II.5.2. Paramètres de test

Différents paramètres influencent l'apprentissage d'un *MLP-C* et donc les résultats de la méthode de *retrieval* évaluée :

- **Le type de descripteur** utilisé pour représenter les formes. Six descripteurs sont évalués, à savoir les *AISM* de degrés 4, 5 et 6, ainsi que les *SD* 2, 3 et 8
- **le nombre *k*** de résultats de *retrieval* retournés, fixé arbitrairement à 5 pour tous les tests
- **Les hyper paramètres du *MLP-C***, soit le paramétrage interne des fonctions du réseau de neurones. En amont de cette expérimentation, la méthode *Grid Search* (Ngoc et al., 2021) implémentée dans la librairie Scikit-Learn (Buitinck et al., 2013; Pedregosa et al., 2011) a permis de fixer certains hyper paramètres du ML, à savoir la fonction d'activation, la structure du réseau, ainsi que la tolérance.
- **Le nombre de données d'entrainement** $n_{samples}$ générées par la méthode $data\_augmentation(n_{copies})$ des objets de type *Mesh* du *KBRE model, avec* n_{copies} fixé arbitrairement à 100

II.5.3. Méthode de test

L'algorithme suivant retourne les résultats de test de la méthode de *retrieval* par *MLP-C* sur les formes d'un ***DataSet*** pour un descripteur de forme donné.

On note que les données augmentées sont générées en amont, et que les descripteur D_name de chacune de ces données sont calculés.

Algorithme 6 : test de la méthode de retrieval par MLP-C sur les formes d'un DataSet

```
inputs :
        X_train, X_test          # ensembles augmentés d'objets d'entrainement et de test
        Y_train, Y_test          # listes des index des objets de X_train et X_test
        D_name                   # nom du descripteur de formes utilisé

returns :
        top_k_accuracy           # taux de réussite des tests de retrieval

1.  k=5                          # nombre de résultats de retrieval fixés à 5
2.  score = 0                    # initialisation du score

3.  mlpc = train_mlpc(X_train, Y_train, D_name)   # entrainement d'un objet MLP-C

4.  for i in range(n_test) :      # avec n_test le nombre d'objets de X_test
5.      x = X_test[ i ]          # attribution du ie objet de X_test à x
6.      y = Y_test[ i ]          # y prend la i-e valeur de Y_test
7.      y^ = x.retrieve_mlpc(mlpc, D_name, k)   # méthode de retrieval par MLP-C de x

8.      if y is in y^ :          # si l'index d'objet source se trouve dans le résultat de
                                 retrieval :
9.          score = score + 1    # le score est incrémenté de 1

10. top_k_accuracy = score / n_tests   # calcul du résultat de top-k accuracy

11. return top_k_accuracy        # retourne le résultat du test
```

II.5.4. Résultats

La méthode de *retrieval* est testée sur les *DataSet* **aero6000**, **parts1759** et **surfaces4913**. Chaque *DataSet* ne contient qu'un seul type d'objet du ***KBRE model*** dans le but d'étudier l'impact du descripteur utilisé en fonction de la typologie de formes à retrouver.

Les résultats de la méthode de retrieval *par MLP-C sont* présentés dans le Tableau 10. Pour comparaison, les colonnes grisées du côté droit du tableau correspondent aux résultats de la méthode de *retrieval* par *matching*. On note que les tests n'ont pas été réalisés sur les *DataSet* ***model*net40** et **mcbA** pour des raisons qui seront discutées ultérieurement.

DataSet	Descriptor	Top-k Accuracy	Accuracy	training time (h)	Top-k Accuracy	Accuracy
aero6000	AISM_4	94%	65%	10,48	30%	15%
	AISM_5	91%	62%	6,75	32%	16%
	AISM_6	96%	69%	11,30	32%	16%
	SD_2	9%	3%	2,96	2%	1%
	SD_3	77%	50%	5,69	19%	8%
	SD_8	59%	32%	6,76	20%	9%
parts1759	AISM_4	98%	84%	1,29	31%	18%
	AISM_5	98%	83%	1,05	28%	15%
	AISM_6	99%	85%	0,83	31%	18%
	SD_2	96%	72%	0,55	79%	48%
	SD_3	99%	85%	0,37	86%	63%
	SD_8	98%	82%	0,33	94%	69%
surfaces4913	AISM_4	95%	66%	3,06	56%	33%
	AISM_5	94%	64%	4,62	53%	31%
	AISM_6	95%	65%	6,02	49%	28%
	SD_2	40%	15%	2,18	15%	5%
	SD_3	78%	43%	2,59	34%	14%
	SD_8	78%	42%	4,26	47%	22%

Tableau 10 : résultats de retrieval par MLP-C de différents descripteurs sur les formes des DataSet de tests

La première observation concernant les résultats de tests ci-dessus est que la méthode de *retrieval* par *MLP-C* présente des résultats incontestablement meilleurs que la méthode par *matching* de formes, et ce pour chacun des descripteurs. En effet, des scores de ***Top-k Accuracy*** de plus de 95% sont atteints sur l'ensemble des *DataSet*, avec un léger avantage des ***AISM_6***. On en déduit que, si le manque de robustesse des descripteurs à certaines variations spatiales et/ou topologiques de la forme analysée limite ses capacités de distinctivité de niveau 2 via la méthode de *matching*, la robustesse d'un *MLP-C* permet de capturer et de reconnaitre les caractéristiques communes aux descripteurs d'une même forme et ainsi d'identifier des formes similaires.

On remarque une très nette supériorité des *AISM* par rapport aux *SD* pour le *retrieval* de formes parmi les *DataSet* **aero6000** et **surfaces4913**, lorsque les *SD* sont tout aussi performants que les *AISM* pour le *retrieval* de formes parmi le *DataSet* **parts1759**. S'ils ne présentent pas ou peu d'amélioration au retrieval de modèles volumiques, les *AISM* prouvent de meilleures performances pour l'analyse de surfaces complexes aéronautiques.

En augmentant le nombre de résultats retournés par la méthode (ici, $k = 5$), une amélioration significative de ***Top-k Accuracy*** est probable. Cependant, cela induirait un *post-processing* aval ou une validation des résultats plus chronophage pour l'utilisateur.

Le temps d'apprentissage des *MLP-C* est très variable, et ne suit pas une loi linéaire en fonction du nombre de données d'entrainement. Si l'étape n'est à réaliser qu'une fois, et que la prédiction d'un *MLP-C* entrainé nécessite moins d'une seconde (de même que le temps d'exécution de la méthode par *matching*), l'impact sur le déploiement de la méthode est à considérer. La Figure 68 présente l'évolution des résultats de *top-k accuracy* et du temps d'apprentissage d'un *MLP-C* en fonction du nombre de données d'entrainement générées pour chaque objet du *DataSet* (nombre total de données d'entrainement $n_{samples} = n * n_{copies}$)

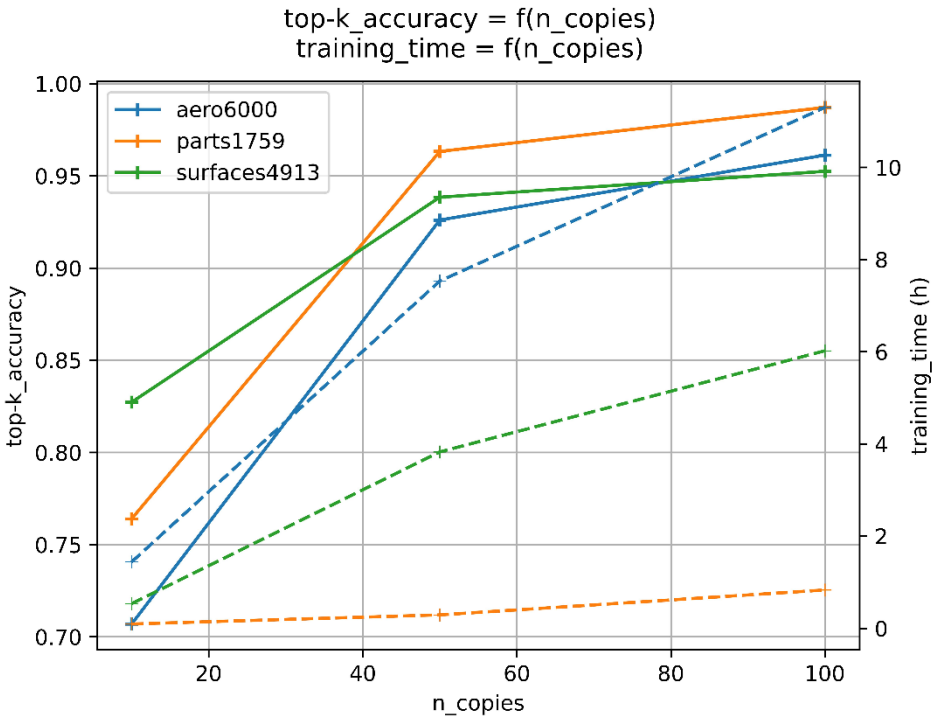


Figure 68 : évolution de top-k accuracy (courbes continues) et du temps d'apprentissage (courbes pointillées) en fonction du nombre de données d'entrainement générées (pour le descripteur AISM_6)

On observe sur le graphique ci-dessus que le nombre de données d'entrainement généré a un très fort impact sur les capacités d'un *MLP-C* pour le *retrieval* de formes, mais aussi sur son temps d'apprentissage. Pour éviter un phénomène d'*underfitting* (i.e. sous-apprentissage, qui décrit un modèle « qui pourrait mieux faire »), un compromis performances / temps d'apprentissage est donc à trouver pour fixer le paramètre de *data-augmentation*.

L'étape amont de *data-augmentation*, comme implémentée dans le *KBRE model*, est quant à elle très chronophage. Bien que dépendante des ressources informatiques disponibles et de possibles méthodes de parallélisation, une approximation simple permet d'anticiper le temps d'exécution de la méthode de *data-augmentation* (incluant le calcul du descripteur) dans le cadre de notre étude : 1 seconde par donnée d'apprentissage, soit n_{sample} secondes. Ainsi, pour le **aero6000**, $6000 * 100 = 6e5$ secondes, soit près de sept jours nécessaires au calcul des données d'entrainement pour un descripteur. Naturellement, diverses optimisations permettraient d'optimiser le processus, et tout dépend de l'implémentation informatique et des ressources allouées. Cependant, cette limitation ne nous a pas permis d'effectuer les tests exhaustifs de *retrieval* par *MLP-C* sur les *DataSet* ***model*net40** et **mcbA** composés de respectivement 11879 et 58696 maillages 3D.

Enfin, et pour terminer l'analyse des résultats d'expérimentation, le score ***d'Accuracy*** est donné à titre indicatif. Ce dernier analyse le taux de tests pour lesquels le meilleur résultat de

retrieval est correct (i.e. équivalent à ***Top-k Accuracy*** avec k=1). Un score *d'**Accuracy*** élevé indiquerait de bonnes capacités discriminatives de niveau III des descripteurs dans la cadre de la méthode. On note que si des scores *d'accuracy* intéressants sont atteints dans certains cas, la méthode ne semble toujours pas fiable pour considérer le meilleur score de *retrieval* comme résultat de *labeling* avec un haut niveau de confiance sur des ensembles de surfaces complexes.

II.5.5. Bilan de la méthode

Les descripteurs, en capturant les caractéristiques distinctives des formes d'un ensemble, permettent d'en estimer les ressemblances et ainsi retourner à un utilisateur les formes les plus semblables à celle de l'objet analysé.

Cette expérimentation a prouvé que si aucun SD n'atteint un niveau de distinctivité acceptable pour le *retrieval* de surfaces complexes aéronautiques, les *AISM* ont des capacités distinctives de niveau II bien supérieures malgré une très forte proximité des formes en question. À contrario, les SD permettent mieux capturer les caractéristiques distinctives des formes résultantes de modèles complets.

Comme nous l'avons appris par les expérimentations précédentes, les descripteurs étudiés sont très sensibles aux variations géométriques et topologiques (subdivision, échantillonnage) des modèles 3D qu'ils représentent, en plus de ne pas être invariant à leur position spatiale. Ainsi, une mauvaise qualité des modèles sources (bruit, occlusions, insuffisance d'éléments géométriques discrets, etc.) et une variation dans la position normalisée peut conduire au calcul d'un descripteur corrompu et non représentatif de la forme. La robustesse et la flexibilité des outils d'apprentissage automatisé peuvent être utilisés pour compenser ce manque de robustesse des descripteurs de forme. Comme pour la *classification*, un modèle comme le *MLP-C* a la capacité d'apprendre à associer des descripteurs à une classe connue et à rapprocher des descripteurs présentant des caractéristiques communes. C'est cette capacité qui est exploitée par la méthode de *retrieval* par *MLP-C* pour prédire les associations entre descripteurs de formes proches, malgré une potentielle lacune en termes de représentativité. Par conséquent, et au vu des résultats de l'expérimentation, la méthode de *retrieval* par *MLP-C* permet d'atteindre un niveau II de distinctivité des *AISM* avec une grande confiance dans les résultats.

Bien que très prometteuse, on note néanmoins plusieurs limites à cette méthode. Premièrement, la nécessité de générer les données d'apprentissage par data-augmentation rend la méthode très chronophage et couteuse en ressources informatiques, en plus de limiter ses possibilités d'applications sur de très larges *DataSet*. De plus, les *MLP-C* entrainés demandent à être stockés, donc la compacité de la méthode ne se limite pas aux descripteurs capitalisés. Enfin, la flexibilité de la méthode est questionnable. En effet, l'ajout ou le retrait de modèles d'un *DataSet* implique une nouvelle étape d'augmentation et un nouvel apprentissage, lorsqu'une simple étape de capitalisation des nouveaux descripteurs suffit à la méthode de *matching*. On notera néanmoins qu'une étude plus approfondie sur l'optimisation du nombre de données d'entrainement ainsi que les paramètres du *MLP-C* permettrait assurément d'optimiser le temps d'exécution de la méthode.

De manière générale, les scores *d'accuracy* très faibles démontrent que la méthode ne permet en aucun cas de considérer le meilleur résultat de *retrieval* comme celui de *labeling*, et qu'une méthode d'analyse plus avancée basée sur un critère de dissimilarité plus précis est nécessaire.

Par sa structure, le *KBRE model* simplifie le déploiement de solution de *ML/DL* pour l'analyse de modèles CAO de divers formats. La génération des données d'entrainement, l'apprentissage des modèles et le lien entre un *MLP-C* et les données d'un *DataSet* sont autant de fonctionnalités pour l'exploration de bases de modèles CAO et l'interprétation de connaissances à leurs sujets.

II.5. Exp-5 : Évaluation de la méthode de *labeling*

La dernière expérimentation réalisée pour l'évaluation des apports scientifiques de nos travaux de recherche se concentre sur la méthode de *labeling* de formes.

Basée sur la méthode de *fitting* des *AISM* des objets candidats sur le maillage 3D représentatif d'une forme, la méthode $label(candidats, D\_name)$ permet de déterminer sur la base d'une mesure de dissimilarité fiable et précise si un objet indexé présente une forme parfaitement similaire à celle analysée.

Cette expérimentation permet de comparer plusieurs *AISM* entre eux et d'évaluer les capacités distinctives de niveau III dans le cadre de la méthode implémentée dans le *KBRE model*.

II.6.1. Hypothèses et critère de validation

Soit $dataset$ un objet *DataSet* qui agrège n objets de type *Shape* (ou d'une classe fille).

On appelle $X0$ l'ensemble des n objets agrégés par $dataset$. Pour chacun, un descripteur de forme de type D_name est capitalisé.

Pour chaque objet $x0_i$ de $X0$, un ensemble de n_{copies} maillages $x_{i,j}$ représentatifs de la forme de $x0_i$ sont générés puis aléatoirement transformés et échantillonnés.

L'ensemble X regroupe les $n_{samples} = n * n_{copies}$ maillages $x_{i,j}$ générés. À chaque $x_{i,j}$ de X est associé son objet source $y_{i,j} = x0_i$ dans l'ensemble Y.

C est l'ensemble des $n_{samples}$ listes c_i des $k > 1$ résultats de la méthode $retrieve\_match(dataset, D\_name, k)$ appliquée à chaque x_i de X. On a alors $c_i = (c_{i,1}, \dots, c_{i,k})$. Pour chaque c_i de C, y_i de Y est ajouté à la liste c_i si ce n'est pas déjà le cas. L'ensemble C est utilisé comme candidats pour la méthode de *labeling* des x_i de X.

$\hat{Y}$ est l'ensemble des résultats $\hat{y}_i$ de la méthode $label(c_i, D\_name)$ appliquée à chaque x_i de X. Le résultat de *labeling* d'un objet x_i est correct si $\hat{y}_i = y_i$, c'est-à-dire si le label prédit est identique au « vrai » label connu de l'objet en question.

Le score d'***accuracy*** est utilisé comme critère d'évaluation principal des tests de *labeling* pour chaque descripteur :

$$accuracy\_score(Y_{test}, \hat{Y}) = \frac{1}{n_{test}} \sum_{i=1}^{n_{test}} 1(\hat{y}_i = y_i)$$

$$\text{Avec } 1(a) := \begin{Bmatrix} 1 \; if \; a = True \\ 0 \; if \; a = False \end{Bmatrix}$$

II.6.2. Paramètres de test

Différents paramètres influencent le résultat de la méthode de *labeling* :

- **Le degré des *AISM* candidats** fittés sur le maillage à labéliser.
- **le nombre k** de candidats en argument de la méthode de *labeling*. Pour chaque maillage de test, son objet d'origine est ajouté aux 5 meilleurs résultats de *retrieval* par *matching* (s'il ne s'y trouve pas déjà) pour former la liste des candidats.
- **Les paramètres de la méthode de fitting des *AISM*,** fixés selon les valeurs optimales empiriquement déterminées en 0

II.6.3. Méthode de test

L'algorithme suivant retourne les résultats de test de la méthode de *labeling* sur les formes d'un *DataSet*. On note que le descripteur évalué est capitalisé en amont pour chaque objet du *DataSet*.

Algorithme 7 : test de la méthode de labeling sur les formes d'un DataSet

```
inputs :
        DataSet                                  # objet de type DataSet
        X0                                       # liste des objets agrégés par DataSet
        D_name                                   # nom du descripteur de formes utilisé

return :
        accuracy                                 # taux de réussite des tests de labeling

1.  k=5                                          # nombre de résultats de retrieval fixés à 5
2.  score = 0                                    # initialisation du score

3.  for i in range(n) :                          # avec n le nombre d'objets de X0
4.      x = X0[ i ]                              # attribution du ie objet de X0 à x
5.      y = X0[ i ]                              # attribution du ie objet de X0 à y
6.      x = x.tesselate( )                       # génération d'un objet Mesh représentatif (si
                                                 pas déjà le cas)
7.      do n_copies times :                      # réaliser n_copies tests pour chaque maillage :
8.          x.subdivide()                        # subdivision des faces du maillage
9.          x.random_sampling()                  # échantillonnage aléatoire
10.         x.random_transform()                 # transformation aléatoire
11.         x.shape_descriptor(D_name)           # calcul du descripteur de forme
12.         candidats = x.retrieve_match(DataSet, D_name, k)   # retrieval par matching de x dans DataSet

13.         if y not in candidats :              # si y ne se trouve pas dans candidats :
14.             add y to candidats               # y est ajouté aux candidats

15.         y^= x.label(candidats, D_name)       # méthode de labélisation de x

16.     if y^ = y :                              # si le résultat de labeling est identique à l'objet
                                                 source :
17.         score = score + 1                    # le score est incrémenté de 1

18. accuracy = score / (n * n_copies)            # calcul du résultat de top-k accuracy

19. return accuracy                              # retourne le résultat du test
```

II.6.4. Résultats

La méthode de *labeling* est testée sur l'ensemble des *DataSet* de tests, c'est-à-dire **aero6000**, **parts1759** et **surfaces4913**, ***model*net40** et **mcbA**. Chaque *DataSet* ne contient qu'un seul type d'objet du *KBRE model* dans le but d'étudier l'impact du descripteur utilisé en fonction de la typologie de formes à retrouver. On note que 10 tests de *labeling* ont été réalisés pour chaque données des *DataSet* étudiés.

À titre de comparaison, les résultats *d'accuracy* de la méthode de *retrieval* par *MLP-C* sont ajoutés aux cotés des résultats de *labeling* (colonne grisée). Comme pour l'expérimentation 3, les métriques *Average Precision*, *Average Recall* et *F1-Score* sont fournies pour une analyse plus fine des résultats. L'Annexe 4 fournit les précisions nécessaires concernant les métriques d'évaluation.

DataSet	Descriptor	Accuracy	Accuracy (retrieval MLP-C)	Avg Precision	Avg Recall	Avg F1-score	Avg Time
aero6000	AISM_4	80,30%	65,13%	44,64%	36,46%	39,06%	15,52
	AISM_5	89,95%	61,63%	65,06%	58,73%	60,91%	26,53
	AISM_6	78,40%	69,23%	42,63%	34,78%	37,46%	34,07
parts1759	AISM_4	62,75%	84,02%	32,97%	24,43%	26,64%	22,11
	AISM_5	57,20%	83,01%	29,40%	20,02%	22,53%	33,68
	AISM_6	49,70%	85,22%	26,35%	17,72%	19,87%	46,54
surfaces4913	AISM_4	74,10%	66,00%	41,20%	32,37%	35,04%	20,02
	AISM_5	71,85%	63,58%	38,91%	28,80%	31,54%	32,54
	AISM_6	64,95%	65,33%	35,11%	24,18%	27,03%	46,57
modelnet40	AISM_4	62,69%		23,40%	16,42%	18,41%	17,46
	AISM_5	61,08%		23,43%	15,52%	17,53%	22,80
	AISM_6	59,35%		23,15%	14,92%	17,01%	30,77
mcbA	AISM_4	79,81%		35,81%	29,12%	31,35%	17,45
	AISM_5	80,19%		39,55%	32,31%	34,70%	25,20
	AISM_6	79,63%		35,17%	29,66%	31,21%	37,99

Tableau 11: résultats de labeling de différents descripteurs sur les formes des DataSet de tests

À la suite de l'optimisation de la méthode de *fitting* des *AISM*, le temps d'exécution moyen de la méthode de *labeling* se situe aux alentours de 20 secondes pour les *AISM* de degré 4, 30 secondes pour le degré 5, et 40 pour le degré 6. L'utilisation de la méthode de *labeling* est donc tout à fait possible pour des analyses unitaires, mais plus difficilement envisageable à grande échelle.

Au vu des résultats, il est difficile de conclure sur la supériorité d'un degré d'*AISM* en particulier. Bien que présentant les meilleurs résultats de *fitting*, les *AISM* de degrés 6 ne prouvent pas de réels avantages pour la méthode de *labeling*. Cela nous indique une possible généralisation de l'approximation des modèles de trop haut degré, qui se traduit par un *AISM* qui approxime avec la même erreur plusieurs formes pourtant différentes. Autrement dit, un *AISM* f_d approximé sur S_1 aura le même résultat de fitting sur S_1 que sur une forme S_2 similaire mais non

identique. Par conséquent, l'erreur de *fitting* utilisée comme mesure de dissimilarité ne permettra pas de discriminer les formes. Un autre inconvénient déjà évoqué est l'allongement du temps d'exécution de la méthode de manière non négligeable.

On identifie de meilleures performances de la méthode à labéliser les formes de surfaces unitaires, plutôt que les formes de modèles résultants. En effet, le meilleur résultat avec presque 90% *d'accuracy* est atteint par ***AISM*_5** sur les formes du *DataSet* **aero6000**, pourtant géométriquement très proches. Avec près de 80% de succès, les tests de *labeling* des formes de **mcbA** prouvent les performances de la méthode pour la labélisation de formes issues de modèles solides.

Les scores *d'accuracy* atteints par les trois degrés d'*AISM* confondus avec la méthode de *labeling* présentent une augmentation de près de 10% sur **surface4913** et 20% sur **aero6000** par rapport à ceux de la méthode de *retrieval* par *MLP-C*. D'un autre côté, le score *d'accuracy* est en baisse de plus de 20% sur **parts1759**. Cependant, la méthode de *labeling* ne nécessite ni étape de data-augmentation ni phase d'apprentissage, ce qui lui donne l'avantage sur le critère du temps d'exécution.

II.6.5. Bilan de la méthode

Des activités d'analyses de formes étudiées, celle présentant la plus grande complexité est probablement le *labeling* qui consiste à reconnaitre précisément une forme, sans la confondre avec d'autres pourtant très proches. Si un descripteur et une méthode permettent conjointement le calcul d'une mesure de dissimilarité suffisamment précise, ils atteignent ensemble des capacités distinctives de niveau III.

Les expérimentations précédentes ont démontré que ce 3e niveau de distinctivité n'était atteint ni par un score de *matching*, ni par les prédictions d'un *MLP-C* entrainé à reconnaitre les variations du descripteur d'une même forme.

Inspirée des méthodes de *fitting* de modèles canoniques de formes primitives (Attene & Patanè, 2010; Fischler & Bolles, 1981; Kaiser et al., 2019; Schnabel et al., 2007), la méthode de *fitting* d'un *AISM* représentatifs d'une forme indexée sur la forme analysée permet le calcul d'un critère de similarité fiable et précis entre elles.

La méthode de *labeling*, dérivée de la méthode de *fitting*, utilise ce critère de dissimilarité pour reconnaitre une forme parmi plusieurs choix possibles, ou encore toutes les rejeter.

Dépendamment des moyens alloués, la méthode de *labeling* reste assez chronophage. Elle a cependant l'avantage de ne nécessiter aucune étape d'apprentissage préalable. Grâce à la capitalisation des descripteurs des formes indexés dans le *DataSet*, seule la représentation 3D sous forme de maillage de la forme analysée est nécessaire.

Si la méthode évaluée manque encore de robustesse, de plus amples expérimentations permettraient sans doute de fiabiliser ses résultats pour pleinement valider le critère de distinctivité de niveau trois des *AISM* comme descripteurs de formes, permettant ainsi l'intégration de techniques de *labeling* aux activités de *RE*.

III. Bilan des expérimentations et discussions

Les expérimentations réalisées ont permis de mettre en lumière plusieurs aspects importants pour l'évaluation de l'apport des *AISM* comme descripteurs de formes pour les activités de *classification*, de *retrieval* et de *labeling*.

Tout d'abord, les prérequis concernant l'approximation et le *fitting* des *AISM* ont été validés avec succès. Ces étapes sont essentielles pour garantir la précision et la robustesse des *AISM* par rapport à la position spatiale des modèles 3D ainsi qu'aux variations topologiques des formes représentées.

En ce qui concerne la *classification*, nos expérimentations confirment le potentiel d'un *MLP-C* pour extraire et reconnaître les caractéristiques communes des descripteurs d'une famille de formes. Cependant, les tests n'ont prouvé aucun avantage de distinctivité des *AISM* par rapport aux SD pour la *classification* de modèles 3D de type *Part*. Par ailleurs, la méthode implémentée reste inférieure aux méthodes de *DL* de la littérature.

Dans le domaine du *retrieval*, les *AISM* surpassent les SD pour identifier des formes proches sur des typologies de surfaces complexes. En revanche, pour les formes de modèles entiers, aucun descripteur ne se révèlent plus performants. Bien que la méthode de retrieval par matching soit intéressante en raison de sa rapidité d'exécution et de sa simplicité de mise en œuvre, son manque de robustesse ne permet pas d'atteindre le niveau de distinctivité suffisant. L'utilisation d'un *MLP-C* offre ainsi un avantage compétitif grâce à sa capacité à reconnaître des caractéristiques distinctives de chaque forme. Cependant, les performances s'accompagnent d'une complexité accrue par la génération de données d'entrainement, l'apprentissage du réseau et le manque de flexibilité de la méthode. Il est important de noter que, dans tous les cas, le 3^{e} niveau de distinctivité des descripteurs n'a pas encore été atteint par les méthodes de *retrieval*.

Grâce à la méthode de fitting des *AISM* développée, l'activité de *labeling* tend à atteindre des capacités distinctives de niveau 3. Cette approche offre un moyen précis d'identifier une forme, bien que des améliorations soient encore nécessaires pour renforcer sa robustesse. En effet, si de bons résultats sont obtenus pour le *DataSet* ***aero6000***, les performances sont moins convaincantes sur d'autres *DataSets*, où le retrieval par MLP-C s'avère parfois supérieur. Les AISM semblent donc particulièrement adaptés à la typologie des surfaces d'aubages aérodynamiques, et leur généralisation à d'autres formes exige de poursuivre les recherches et d'optimiser les méthodes. De futures expérimentations pourraient renforcer davantage la fiabilité des résultats et valider pleinement le critère de distinctivité de niveau 3 des AISM en tant que descripteurs de formes.

Les *AISM* représentent un apport significatif pour l'amélioration des méthodes d'analyse de formes, notamment en ce qui concerne des typologies de surfaces complexes telles que celles des modèles aéronautiques. En plus des activités de classifications et de *retrieval* largement couvertes par les travaux de recherche du domaine, les *AISM* permettent des activités de *labeling* pour le moment réservées à aux primitives géométriques.

Conclusion et perspectives

Conclusion

À l'ère du numérique et de l'information, les problèmes de continuité numérique au sein des Systèmes d'Informations industriels ont un impact sur les entreprises dans la gestion des données, informations et connaissances relatives à leurs produits et systèmes. Les acteurs des différentes disciplines d'ingénierie impliquées dans le développement et le maintien des produits font alors face à une difficulté d'accès à une information qui soit structurée, pertinente et exploitable.

Ce phénomène, accentué par l'explosion des volumes de données et la diversité des formats, affecte particulièrement les modèles de Conception Assistée par Ordinateur, principal support et aboutissement des processus d'ingénierie pour le développement de produits mécaniques. Au cours des étapes du cycle de vie des produits, on constate une dissémination des modèles CAO dans des environnements cloisonnés du SI ainsi qu'une incomplétude des modèles voire une obsolescence de leurs formats.

Pour lutter contre la perte d'information et maintenir l'intégrité des modèles CAO, nos travaux de recherches se sont concentrés sur le processus de *Reverse Engineering*, dont l'objectif principal est de recréer ou reconstruire des informations et connaissances relatives aux produits altérés par une perte sémantique.

Le contexte industriel de ce projet de thèse en partenariat avec safran Aircraft Engines a permis d'identifier différents scénarios d'application liés aux modèles CAO de composants aéronautiques. Ces scénarios présentent deux défis majeurs, l'accès aux informations pertinentes au sein du système d'information (SI) et l'efficacité de leur exploitation. Deux axes de recherche ont donc été explorés. D'une part, l'analyse de forme pour l'exploration des bases de données et l'accès aux modèles CAO, de l'autre, l'utilisation de systèmes de type Knowledge-Based Engineering (*KBE*) pour maximiser leur exploitation dans le cadre des processus de Reverse Engineering.

Tout d'abord, les activités d'analyse de forme 3D ont été étudiées pour la recherche et l'extraction d'informations relatives aux modèles CAO. Quatre objectifs ont été identifiés, auxquelles les éléments de la proposition scientifique s'efforcent de répondre.

Le premier concerne le développement d'une méthode de représentation des formes 3D qui soit optimisée pour la représentation de formes complexes comme celles des aubages aérodynamiques. Dans cette optique, des descripteurs de forme ont été introduits dans la littérature en tant que signatures numériques des objets 3D. Les *AISM* ont été proposés en tant que descripteurs qui capturent les caractéristiques distinctives des surfaces complexes des modèles aéronautiques. Ces derniers ont été conçus et optimisés pour s'adapter à l'ensemble des activités d'analyse étudiées.

Le deuxième objectif s'intéresse aux méthodes de raisonnement pour l'organisation des données. Par la suite, cet objectif a été précisée comme correspondant au premier niveau de distinctivité des descripteurs pour les activités de *classification*. L'utilisation d'un *MLP-C* a été explorée pour l'identification des caractéristiques communes entre les formes de larges ensembles, dans le but de les catégoriser sans dépendre exclusivement de métadonnées potentiellement altérées ni de règles explicitement définies. Les expérimentations ont révélé le potentiel des *AISM* et de la méthode de classification, bien que certains outils identifiés dans la littérature scientifique demeurent plus performants dans le cadre de nos expérimentations.

Le troisième objectif concerne la capacité à discriminer des formes similaires en vue de retrouver et de réutiliser des modèles CAO dans une base de données. Il fait référence au deuxième niveau de distinctivité des descripteurs de forme dans le cadre d'activités de *retrieval*. Deux méthodes ont été proposées pour comparer différents descripteurs, démontrant ainsi la supériorité des *AISM* pour le *retrieval* de formes complexes. Une fois de plus, l'utilisation d'un *MLP-C* se révèle être un atout précieux pour les activités de *SA*. Par rapport à une analyse numérique simple, nos expérimentations ont clairement établi la nette supériorité des *MLP-C* pour l'identification de formes en fonction de leur similarité. Bien qu'efficaces, il est important de noter que ces techniques d'apprentissage supervisé peuvent être chronophages et exiger des ressources informatiques importantes, ce qui pose des contraintes en termes de simplicité d'utilisation et de rapidité d'exécution pour une utilisation pratique.

Enfin, le quatrième et dernier objectif vise à atteindre le troisième niveau de distinctivité des descripteurs, soit la capacité d'identifier des formes avec une grande précision, sans être trompé par les fortes similarités géométriques entre les modèles 3D. Pour relever ce défi, le développement d'une méthode de *fitting* a permis de tirer pleinement parti des *AISM* et de proposer une mesure de dissimilarité précise et prometteuse pour distinguer des formes proches. La méthode de *labeling* qui en découle contribue de manière significative à l'amélioration des méthodes avancées d'analyse de modèles 3D.

Les activités de *SA* offrent un potentiel exploitable pour explorer le capital d'une entreprise dans le but de retrouver des modèles CAO riches, de comparer différentes solutions de conception, de détecter des doublons, mais aussi d'extraire des connaissances exploitables dans le cadre des processus de rétro-ingénierie. Néanmoins, les données CAO industrielles présentent une grande complexité et diversité. Au cours des expérimentations réalisées sur les activités de *SA*, nos recherches ont révélé des variations de performances des descripteurs en fonction des caractéristiques des formes, mettant en évidence l'importance du choix de méthodes et d'outils appropriés en fonction du contexte.

Les méthodes de *SA*, lorsqu'elles ne sont pas intégrées, ne sont pas facilement exploitables pour la *RE* de modèles CAO en environnement *PDM*. Afin de pleinement répondre à la question de recherche, trois nouveaux objectifs ont été formulés.

Le premier aborde la manière de contextualiser les applications de *SA* afin de traduire un besoin métier en un processus de *RE*. Le deuxième se penche sur la structuration du traitement des informations manipulées, depuis les modèles CAO et bases de données en entrée du processus, jusqu'à l'exploitation des connaissances déduites en sortie. Enfin, le dernier objectif porte sur la capitalisation des connaissances relatives aux modèles CAO, permettant ainsi une réutilisation à long terme.

Nos recherches se sont alors tournées vers les systèmes de *KBE*. Ces derniers combinent les capacités de traitement géométrique des outils de CAO avec les possibilités de structuration de l'information de la Programmation Orientée Objet, les techniques avancées du *machine learning* pour le raisonnement, ainsi que le stockage d'informations riches, structurées et exploitables, autrement dit de la connaissance, au sein de *Knowledge Base*.

La proposition du *Knowledge-Based Reverse-Engineering* (*KBRE*) a pour objectif de résoudre ces problématiques en intégrant les méthodes de *SA* dans le processus de *RE* grâce à la gestion globale des informations et connaissances manipulées.

Tout d'abord, la définition des classes d'objets fournit la structure nécessaire pour définir et appliquer des méthodes spécifiques en fonction des types de données. Ces objets permettent de représenter différents formats et différentes méthodes de modélisation géométrique à plusieurs niveaux de la structure topologique des modèles, s'adaptant ainsi en fonction des besoins. De plus, des objets représentatifs des ensembles de modèles CAO permettent de contextualiser les applications et de manipuler simplement de très larges ensembles comme il en existe dans les *PDM* d'une entreprise.

Au cœur de la définition des objets, la tessellation permet la traduction des modèles CAO, favorisant ainsi l'interopérabilité grâce à l'utilisation de descripteurs de forme communs. S'ensuit l'implémentation des méthodes de *SA* définies dans la première partie de la proposition pour que le *KBRE model* intègre les fonctionnalités souhaitées.

La capitalisation des descripteurs et des modèles de ML est possible par l'utilisation d'une *Knowledge Base* qui structure la connaissance acquise, permet son stockage ainsi que la récupération des objets. Par ailleurs, l'indépendance du *KBRE* modèle vis-à-vis des systèmes d'information et logiciels spécifiques permet les analyses transverses essentielles pour la recherche d'informations disséminées.

Enfin, pour une mise en œuvre opérationnelle du *KBRE model* dans le cadre de processus de *RE*, il sera essentiel de développer une interface graphique pour faciliter les interactions avec les utilisateurs qui pourront alors explorer les modèles CAO dispersés, les comparer et en extraire les connaissances recherchées.

Perspectives

À l'issue de cette thèse, diverses perspectives s'ouvrent pour l'approfondissement et l'extension de nos travaux de recherche.

Tout d'abord, du côté de l'implémentation informatique de la solution proposée, des axes d'optimisation se profilent. Une amélioration des méthodes du *KBRE model* pourrait accroître l'efficacité et les performances des techniques de *SA*. Par exemple, l'amélioration de la robustesse du *retrieval* et de la précision du *labeling* sont des enjeux cruciaux. De manière plus générale, les optimisations concernent la simplicité d'utilisation, la rapidité d'exécution et la consommation de ressources informatiques liée au modèle de données. Par ailleurs, le potentiel d'adaptation du modèle *KBRE* à divers scénarii pourrait être étendu par l'intégration de nouvelles méthodes, telles que la segmentation sémantique, la reconstruction surfacique, et la détection de défauts, faisant ainsi évoluer cet outil vers une solution de *RE* plus complète. De plus, le développement d'une interface graphique se pose comme une nécessité pour son intégration industrielle. Une telle interface apporterait aux utilisateurs des possibilités de visualisation des modèles 3D analysés, un accès ergonomique aux méthodes applicables, et un moyen d'évaluation des résultats de *SA*. En outre, la visualisation des *DataSet*, de leurs caractéristiques, et des modèles CAO qui les composent est indispensable pour l'exploration du SI et pour l'interprétation humaine des connaissances qu'il renferme. Enfin, l'établissement de protocoles de communication et d'échange dynamique de données entre le *KBRE model* et l'ensemble des éléments du SI demeure un enjeu, impliquant notamment la définition du format de traduction et d'échange des modèles CAO ainsi que la gestion des mises à jour des données capitalisées dans les KB par rapport aux évolutions du SI.

Les perspectives d'amélioration des méthodes et outils de *SA* sont également nombreuses. L'évaluation et la comparaison de descripteurs de forme de la littérature aux *AISM* est une piste principale pour la poursuite de ces recherches. Par exemple, l'examen de descripteurs locaux par rapport à descripteurs globaux pourrait être approfondi, notamment dans la reconnaissance des zones d'intérêt au sein de maillages 3D. De même, l'exploration de techniques de ML telles que le Deep Learning et l'apprentissage non supervisé ouvre de nouvelles perspectives pour la génération et l'exploitation de descripteurs par les méthodes de *SA*.

D’autres perspectives concernent une recherche plus étendue des moyens pour le maintien de l’intégrité des modèles CAO au sein du SI. Nos travaux s’intéressent aux méthodes de *SA* pour croiser et relier des modèles CAO entre eux, sans considération pour leurs métadonnées et liens relationnels dans le SI. Le développement de méthodes d’analyse hybrides, prenant en considération aussi bien les caractéristiques intrinsèques des modèles 3D que leur contextualisation dans le SI, est une piste qu’il semble nécessaire d’analyser. Une exploration plus approfondie des méthodes de gestion des connaissances, notamment en termes de structuration à travers l'observation de systèmes de base de connaissances avancés, est également souhaitable.
Enfin, l'extension de nos recherches à d'autres secteurs industriels où le *RE* est tout aussi important représente de nouvelles perspectives de création de connaissances scientifiques. L'évaluation des descripteurs de forme et des méthodes de *SA* sur des typologies de modèles 3D issus de l'industrie automobile ou du bâtiment permettrait de mieux appréhender la diversité des modèles 3D et d'optimiser les solutions proposées. Cette démarche permettrait également d'adapter nos travaux aux spécificités de différents secteurs et aux modèles 3D associés, contribuant ainsi à une meilleure compréhension des besoins propres à chaque domaine d'application.

Pour terminer, la diffusion des résultats de nos travaux sont prévus via des actions de communication scientifique, avec la rédaction d’articles pour une publication dans des journaux internationaux.

Dans un contexte de transformation numérique par les technologies de l’ingénierie 4.0, nous sommes convaincus que l'analyse de forme jouera un rôle toujours plus important dans les processus de Reverse Engineering. Ces domaines de recherche dévoilent des potentiels jusqu’alors inexploités au sein des entreprises du secteur industriel, offrant des opportunités pour explorer et préserver leurs ressources essentielles, tout en tentant de solutionner les problèmes de discontinuité numérique et la perte de connaissance liés aux produits et systèmes.

Cette thèse apporte ainsi sa contribution aux domaines de la Rétro-Ingénierie et de l'Analyse de Forme 3D dans un monde où les enjeux de gestion de l'information deviennent de plus en plus complexes, mais où les perspectives technologiques s'annoncent d'autant plus grandes.

Publications scientifiques

Article de conférence publié dans Advances in Integrated Design and Production II - Proceedings of the 12th International Conference on Integrated Design and Production, CPI 2022
https://link.springer.com/book/10.1007/978-3-031-23615-0

Reverse Engeneering for Aeronautical Products : State of the Art and proposition, Williatte Philippe, Durupt Alexandre, Remy Sébastien, Bricogne Matthieu
10.1007/978-3-031-23615-0_39

Article de journal publié dans Computer Aided Design and Applications, volume 20, number 3
https://www.cad-journal.net/

Reverse Engineering for Aeronautics: Study on Parts Semantic Segmentation Philippe Williatte, Alexandre Durupt, Sébastien Remy and Matthieu Bricogne
10.14733/cadaps.2023.557-573

Bibliographie

A. Nguyen & B. Le. (2013). 3D point cloud segmentation : A survey. *2013 6th IEEE Conference on Robotics, Automation and Mechatronics (RAM)*, 225-230. https://doi.org/10.1109/RAM.2013.6758588

Alemanni, M., Destefanis, F., & Vezzetti, E. (2011). Model-based definition design in the product lifecycle management scenario. *The International Journal of Advanced Manufacturing Technology*, *52*, 1-14.

Al-wswasi, M., & Ivanov, A. (2019). A novel and smart interactive feature recognition system for rotational parts using a STEP file. *The International Journal of Advanced Manufacturing Technology*, *104*, 261-284.

Attene, M., & Patanè, G. (2010). Hierarchical Structure Recovery of Point-Sampled Surfaces. *Computer Graphics Forum*, *29*(6), Article 6. https://doi.org/10.1111/j.1467-8659.2010.01658.x

B. K. P. Horn. (1984). Extended Gaussian images. *Proceedings of the IEEE*, *72*(12), 1671-1686. https://doi.org/10.1109/PROC.1984.13073

Bahloul, K., Buzon, L., & Bouras, A. (2008). Archival Initiatives in the Engineering Context. In X.-T. Yan, W. J. Ion, & B. Eynard (Éds.), *Global Design to Gain a Competitive Edge* (p. 313-321). Springer London. https://doi.org/10.1007/978-1-84800-239-5_31

Barbero, B. R., & Ureta, E. S. (2011). Comparative study of different digitization techniques and their accuracy. *Computer-Aided Design*, *43*(2), 188-206. https://doi.org/10.1016/j.cad.2010.11.005

Bello, S. A., Yu, S., Wang, C., Adam, J. M., & Li, J. (2020). Deep learning on 3D point clouds. *Remote Sensing*, *12*(11), Article 11.

Ben Makhlouf, A., Louhichi, B., Mahjoub, M. A., & Deneux, D. (2019). Reconstruction of a CAD model from the deformed mesh using B-spline surfaces. *International Journal of Computer Integrated Manufacturing*, *32*(7), 669-681. https://doi.org/10.1080/0951192X.2019.1599442

Bénière, R., Subsol, G., Gesquière, G., Le Breton, F., & Puech, W. (2013). A comprehensive process of reverse engineering from 3D meshes to CAD models. *Computer-Aided Design*, *45*(11), 1382-1393. https://doi.org/10.1016/j.cad.2013.06.004

Benkő, P., Kós, G., Várady, T., Andor, L., & Martin, R. (2002). Constrained fitting in reverse engineering. *Computer Aided Geometric Design*, *19*(3), 173-205. https://doi.org/10.1016/S0167-8396(01)00085-1

Besl, P. J., & McKay, N. D. (1992). Method for registration of 3-D shapes. *Sensor fusion IV: control paradigms and data structures*, *1611*, 586-606.

Borrmann, A., & Berkhahn, V. (2018). Principles of Geometric Modeling. In A. Borrmann, M. König, C. Koch, & J. Beetz (Éds.), *Building Information Modeling* (p. 27-41). Springer International Publishing. https://doi.org/10.1007/978-3-319-92862-3_2

Bruneau, M. (2016). *Une méthodologie de Reverse Engineering à partir de données hétérogènes pour les pièces et assemblages mécaniques* [PhD Thesis]. Université de Technologie de Compiègne.

Bruneau, M., Durupt, A., Roucoules, L., Pernot, J.-P., & Rowson, H. (2014). *Methodology of reverse engineering for large assemblies products from heterogeneous data*.

Buitinck, L., Louppe, G., Blondel, M., Pedregosa, F., Mueller, A., Grisel, O., Niculae, V., Prettenhofer, P., Gramfort, A., Grobler, J., Layton, R., VanderPlas, J., Joly, A., Holt, B., & Varoquaux, G. (2013). API design for machine learning software : Experiences from the scikit-learn project. *ECML PKDD Workshop: Languages for Data Mining and Machine Learning*, 108-122.

Buonamici, F., Carfagni, M., Furferi, R., Governi, L., Lapini, A., & Volpe, Y. (2018). Reverse engineering modeling methods and tools : A survey. *Computer-Aided Design and Applications*, *15*(3), Article 3. https://doi.org/10.1080/16864360.2017.1397894

Buonamici, F., Carfagni, M., Furferi, R., Volpe, Y., & Governi, L. (2021). Reverse engineering by CAD template fitting : Study of a fast and robust template-fitting strategy. *Engineering with Computers*, *37*(4), 2803-2821. https://doi.org/10.1007/s00366-020-00966-4

Chandrasegaran, S. K., Ramani, K., Sriram, R. D., Horváth, I., Bernard, A., Harik, R. F., & Gao, W. (2013). The evolution, challenges, and future of knowledge representation in product design systems. *Computer-aided design*, *45*(2), 204-228.

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. (2015). *ShapeNet : An Information-Rich 3D Model Repository* (arXiv:1512.03012 [cs.GR]). Stanford University — Princeton University — Toyota Technological Institute at Chicago.

Chapman, C. B., & Pinfold, M. (1999). Design engineering—A need to rethink the solution using knowledge based engineering. *Knowledge-based systems*, *12*(5-6), 257-267.

Chen, D.-Y., Tian, X.-P., Shen, Y.-T., & Ouhyoung, M. (2003). On Visual Similarity Based 3D Model Retrieval. *Computer Graphics Forum*, *22*(3), 223-232. https://doi.org/10.1111/1467-8659.00669

Chen, H., & Bhanu, B. (2007). 3D free-form object recognition in range images using local surface patches. *Pattern Recognition Letters*, *28*(10), 1252-1262. https://doi.org/10.1016/j.patrec.2007.02.009

Chivate, P. N., & Jablokow, A. G. (1993). Solid-model generation from measured point data. *Special Issue Uncertainties in geometric design*, *25*(9), 587-600. https://doi.org/10.1016/0010-4485(93)90074-X

Chua, C. S., & Jarvis, R. (1997). Point Signatures : A New Representation for 3D Object Recognition. *International Journal of Computer Vision*, *25*(1), 63-85. https://doi.org/10.1023/A:1007981719186

Dawson-Haggerty et al. (2019). *Trimesh* (Version 3.2.0) [Logiciel]. https://trimsh.org/

De Courcy, R. (1992). Les systèmes d'information en réadaptation. *Québec, Réseau international CIDIH et facteurs environnementaux*, *5*(1-2), 7-10.

Dekhtiar, J., Durupt, A., Bricogne, M., Eynard, B., Rowson, H., & Kiritsis, D. (2018). Deep learning for big data applications in CAD and PLM – Research review, opportunities and case study. *Computers in Industry*, *100*, 227-243. https://doi.org/10.1016/j.compind.2018.04.005

Dimitrov, A., Gu, R., & Golparvar-Fard, M. (2016). Non-Uniform B-Spline Surface Fitting from Unordered 3D Point Clouds for As-Built Modeling. *Computer-Aided Civil and Infrastructure Engineering*, *31*(7), 483-498. https://doi.org/10.1111/mice.12192

Ding, S., Feng, Q., Sun, Z., & Ma, F. (2021). MBD Based 3D CAD Model Automatic Feature Recognition and Similarity Evaluation. *IEEE Access*, *9*, 150403-150425.

DS, D. (2010). PLM (Product Lifecyle Management). *PLM Glossary*.

Du, T., Inala, J. P., Pu, Y., Spielberg, A., Schulz, A., Rus, D., Solar-Lezama, A., & Matusik, W. (2018). InverseCSG: Automatic Conversion of 3D Models to CSG Trees. *ACM Trans. Graph.*, *37*(6). https://doi.org/10.1145/3272127.3275006

Durupt, A. (2020). Rétro-ingénierie par intégration de données et d'expertises hétérogènes, contribution à l'ingénierie mécanique'. *Habilitation à Diriger les Recherches, Université de Technologie de Compiègne*, 1-154.

Durupt, A., Bricogne, M., Remy, S., Troussier, N., Rowson, H., & Belkadi, F. (2019). An extended framework for knowledge modelling and reuse in reverse engineering projects. *Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture*, *233*(5), 1377-1389. https://doi.org/10.1177/0954405418789973

Durupt, A., Remy, S., Bricogne, M., & Troussier, N. (2013). PHENIX: product history-based reverse engineering. *International Journal of Product Lifecycle Management*, *6*(3), 270-287.

Durupt, A., Remy, S., Ducellier, G., & Eynard, B. (2008). From a 3D point cloud to an engineering CAD model : A knowledge-product-based approach for reverse engineering. *Virtual and Physical Prototyping*, *3*(2), 51-59. https://doi.org/10.1080/17452750802047917

Elmasri, R., & Navathe, S. B. (2011). *Fundamentals of database systems*. Addison-Wesley.

El-Mehalawi, M., & Miller, R. A. (2003a). A database system of mechanical components based on geometric and topological similarity. Part I: representation. *Computer-Aided Design*, *35*(1), 83-94.

El-Mehalawi, M., & Miller, R. A. (2003b). A database system of mechanical components based on geometric and topological similarity. Part II: indexing, retrieval, matching, and similarity assessment. *Computer-Aided Design*, *35*(1), 95-105.

Eynard, B., Durupt, A., & Le Duigou, J. (2023). Ingénierie 3D et gestion du cycle de vie de produits manufacturés. *Le BIM, nouvel art de construire: Données communes et modélisations partagées*, 43.

Eynard, B., Gallet, T., Roucoules, L., & Ducellier, G. (2006). PDM system implementation based on UML. *Mathematics and Computers in Simulation*, *70*(5-6), 330-342.

Feng, G., Cui, D., Wang, C., & Yu, J. (2009). Integrated data management in complex product collaborative design. *Computers in Industry*, *60*(1), 48-63.

Fischler, M. A., & Bolles, R. C. (1981). Random Sample Consensus : A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography. *Commun. ACM*, *24*(6), Article 6. https://doi.org/10.1145/358669.358692

Fisher, R. B. (2004). Applying knowledge to reverse engineering problems. *Computer-Aided Design*, *36*(6), 501-510. https://doi.org/10.1016/S0010-4485(03)00158-1

Fontana, M., Giannini, F., & Meirana, M. (1999). A free form feature taxonomy. *Computer Graphics Forum*, *18*(3), 107-118.

Fryazinov, O., Pasko, A., & Comninos, P. (2010). Fast reliable interrogation of procedurally defined implicit surfaces using extended revised affine arithmetic. *Computers & Graphics*, *34*(6), 708-718.

Georgiou, T., Liu, Y., Chen, W., & Lew, M. (2020). A survey of traditional and deep learning-based feature descriptors for high dimensional data in computer vision. *International Journal of Multimedia Information Retrieval*, *9*(3), Article 3. https://doi.org/10.1007/s13735-019-00183-w

Geuzaine, C., & Remacle, J.-F. (2009). Gmsh : A 3-D finite element mesh generator with built-in pre-and post-processing facilities. *International journal for numerical methods in engineering*, *79*(11), 1309-1331.

Gezawa, A. S., Zhang, Y., Wang, Q., & Yunqi, L. (2020). A review on deep learning approaches for 3d data representations in retrieval and classifications. *IEEE access*, *8*, 57566-57593.

Guérineau, J., Bricogne, M., Rivest, L., & Durupt, A. (2022). Organizing the fragmented landscape of multidisciplinary product development : A mapping of approaches, processes, methods and tools from the scientific literature. *Research in Engineering Design*, *33*(3), 307-349.

Guo, K., Zou, D., & Chen, X. (2016). 3D Mesh Labeling via Deep Convolutional Neural Networks. *ACM Trans. Graph.*, *35*(1). https://doi.org/10.1145/2835487

Guo, Y., Bennamoun, M., Sohel, F., Lu, M., & Wan, J. (2014). 3D object recognition in cluttered scenes with local surface features : A survey. *IEEE transactions on pattern analysis and machine intelligence*, *36*(11), 2270-2287.

Guo, Y., Bennamoun, M., Sohel, F., Lu, M., Wan, J., & Kwok, N. M. (2016). A Comprehensive Performance Evaluation of 3D Local Feature Descriptors. *International Journal of Computer Vision*, *116*(1), 66-89. https://doi.org/10.1007/s11263-015-0824-y

Guo, Y., Wang, H., Hu, Q., Liu, H., Liu, L., & Bennamoun, M. (2020). Deep learning for 3d point clouds : A survey. *IEEE transactions on pattern analysis and machine intelligence*, *43*(12), Article 12.

H. Laga, H. Takahashi, & M. Nakajima. (2004). Geometry image matching for similarity estimation of 3D shapes. *Proceedings Computer Graphics International, 2004.*, 490-496. https://doi.org/10.1109/CGI.2004.1309252

Herlem, G., Adragna, P. A., Ducellier, G., & Durupt, A. (2014). An extension of the core product model for the maturity management of the digital mock up : Use of graph and knowledge to describe mechanical parts. *International Journal of Product Lifecycle Management*, *7*(1), 94. https://doi.org/10.1504/IJPLM.2014.065462

Hu, S., Polette, A., & Pernot, J.-P. (2022). SMA-Net : Deep learning-based identification and fitting of CAD models from point clouds. *Engineering with Computers*, *38*(6), 5467-5488.

Ioannidou, A., Chatzilari, E., Nikolopoulos, S., & Kompatsiaris, I. (2017). Deep Learning Advances in Computer Vision with 3D Data : A Survey. *ACM Computing Surveys*, *50*. https://doi.org/10.1145/3042064

ISO, I. (2008). Iso/iec 15288 : Systems and software engineering—System life cycle processes. *ISO, IEC*, 24748-1.

Kaiser, A., Ybanez Zepeda, J. A., & Boubekeur, T. (2019). A Survey of Simple Geometric Primitives Detection Methods for Captured 3D Data. *Computer Graphics Forum*, *38*(1), 167-196. https://doi.org/10.1111/cgf.13451

Kalogerakis, E., Hertzmann, A., & Singh, K. (2010). Learning 3d mesh segmentation andlabeling. *ACM Transactions on Graphics-Proceedings of ACM SIGGRAPH2010. ACM*.

Ke, Y., Fan, S., Zhu, W., Li, A., Liu, F., & Shi, X. (2006). Feature-based reverse modeling strategies. *Computer-Aided Design*, *38*(5), 485-506. https://doi.org/10.1016/j.cad.2005.12.002

Kim, S., Chi, H., Hu, X., Huang, Q., & Ramani, K. (2020). A large-scale annotated mechanical components benchmark for classification and retrieval tasks with deep neural networks. *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XVIII 16*, 175-191.

Kubat, M., & Kubat, J. (2017). *An introduction to machine learning* (Vol. 2). Springer.

Kurita, T. (2019). Principal Component Analysis (PCA). In *Computer Vision : A Reference Guide* (p. 1-4). Springer International Publishing. https://doi.org/10.1007/978-3-030-03243-2_649-1

Laga, H., Guo, Y., Tabia, H., Fisher, R. B., & Bennamoun, M. (2018). *3D Shape analysis : Fundamentals, theory, and applications*. John Wiley & Sons.

Laga, H., Takahashi, H., & Nakajima, M. (2006). Spherical parameterization and geometry image-based 3D shape similarity estimation (CGS 2004 special issue). *The Visual Computer*, *22*(5), 324-331. https://doi.org/10.1007/s00371-006-0010-x

Laube, P. (2020). *Machine Learning Methods for Reverse Engineering of Defective Structured Surfaces*. Springer Fachmedien Wiesbaden. https://doi.org/10.1007/978-3-658-29017-7

Li, B., Lu, Y., Li, C., Godil, A., Schreck, T., Aono, M., Burtscher, M., Chen, Q., Chowdhury, N. K., Fang, B., & others. (2015). A comparison of 3D shape retrieval methods based on a large-scale benchmark supporting multimodal queries. *Computer Vision and Image Understanding*, *131*, 1-27.

Li, L., Sung, M., Dubrovina, A., Yi, L., & Guibas, L. J. (2019). Supervised fitting of geometric primitives to 3d point clouds. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2652-2660.

Li, Y., Bu, R., Sun, M., Wu, W., Di, X., & Chen, B. (2018). Pointcnn : Convolution on x-transformed points. *Advances in neural information processing systems*, *31*.

Lu, Y. (2017). Industry 4.0 : A survey on technologies, applications and open research issues. *Journal of Industrial Information Integration*, *6*, 1-10. https://doi.org/10.1016/j.jii.2017.04.005

Lupinetti, K., Giannini, F., Monti, M., & Pernot, J.-P. (2018). Multi-criteria retrieval of CAD assembly models. *Journal of Computational Design and Engineering*, *5*(1), 41-53.

Lupinetti, K., Pernot, J.-P., Monti, M., & Giannini, F. (2019). Content-based CAD assembly model retrieval : Survey and future challenges. *Computer-Aided Design*, *113*, 62-81.

Ma, L., Huang, Z., & Wang, Y. (2010). Automatic discovery of common design structures in CAD models. *Computers & Graphics*, *34*(5), 545-555.

MacLean, M., & Davis, B. H. (1998). *Time & bits : Managing digital continuity*. Getty Publications.

Maturana, D., & Scherer, S. (2015). Voxnet : A 3d convolutional neural network for real-time object recognition. *2015 IEEE/RSJ international conference on intelligent robots and systems (IROS)*, 922-928.

Milton, N. R. (2008). *Knowledge technologies* (Vol. 3). Polimetrica sas.

Ngoc, T. T., Dai, L., & Phuc, D. (2021). Grid search of multilayer perceptron based on the walk-forward validation methodology. *Int. J. Electr. Comput. Eng. IJECE*, *11*(2), 1742.

Nzetchou, S., Durupt, A., Eynard, B., & Remy, S. (2021). Semantic Enrichment of 3D Models Based on Ontology Integration. In L. Roucoules, M. Paredes, B. Eynard, P. Morer Camo, & C. Rizzi (Éds.), *Advances on Mechanics, Design Engineering and Manufacturing III* (p. 341-346). Springer International Publishing. https://doi.org/10.1007/978-3-030-70566-4_54

Nzetchou, S., Durupt, A., Remy, S., & Eynard, B. (2019). Review of CAD Visualization Standards in PLM. In C. Fortin, L. Rivest, A. Bernard, & A. Bouras (Éds.), *Product Lifecycle Management in the Digital Twin Era* (Vol. 565, p. 34-43). Springer International Publishing. https://doi.org/10.1007/978-3-030-42250-9_4

Osada, R., Funkhouser, T., Chazelle, B., & Dobkin, D. (2002). Shape Distributions. *ACM Trans. Graph.*, *21*(4), Article 4. https://doi.org/10.1145/571647.571648

Ouamer-Ali, M.-I., Laroche, F., Bernard, A., & Remy, S. (2014). Toward a methodological knowledge based approach for partial automation of reverse engineering. *Procedia CIRP*, *21*, 270-275.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., & Duchesnay, E. (2011). Scikit-learn : Machine Learning in Python. *Journal of Machine Learning Research*, *12*, 2825-2830.

Piegl, L., & Tiller, W. (1996). *The NURBS book*. Springer Science & Business Media.

Pratt, M. J. & others. (2001). Introduction to ISO 10303—The STEP standard for product data exchange. *Journal of Computing and Information Science in Engineering*, *1*(1), 102-103.

Qi, C. R., Su, H., Mo, K., & Guibas, L. J. (2017). Pointnet : Deep learning on point sets for 3d classification and segmentation. *Proceedings of the IEEE conference on computer vision and pattern recognition*, 652-660.

Qi, C. R., Su, H., Nießner, M., Dai, A., Yan, M., & Guibas, L. J. (2016). Volumetric and multi-view cnns for object classification on 3d data. *Proceedings of the IEEE conference on computer vision and pattern recognition*, 5648-5656.

Qi, C. R., Yi, L., Su, H., & Guibas, L. J. (2017). PointNet++ : Deep Hierarchical Feature Learning on Point Sets in a Metric Space. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, & R. Garnett (Éds.), *Advances in Neural Information Processing Systems* (Vol. 30). Curran Associates, Inc. https://proceedings.neurips.cc/paper_files/paper/2017/file/d8bf84be3800d12f74d8b05e9b89836f-Paper.pdf

R. B. Rusu, N. Blodow, & M. Beetz. (2009). Fast Point Feature Histograms (FPFH) for 3D registration. *2009 IEEE International Conference on Robotics and Automation*, 3212-3217. https://doi.org/10.1109/ROBOT.2009.5152473

R. B. Rusu, N. Blodow, Z. C. Marton, & M. Beetz. (2008). Aligning point cloud views using persistent feature histograms. *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 3384-3391. https://doi.org/10.1109/IROS.2008.4650967

Rocca, G. L. (2012). Knowledge based engineering : Between AI and CAD. Review of a language based technology to support engineering design. *Advanced Engineering Informatics*, *26*(2), 159-179. https://doi.org/10.1016/j.aei.2012.02.002

S. B. Kang & K. Ikeuchi. (1993). The complex EGI: a new representation for 3-D pose determination. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *15*(7), 707-721. https://doi.org/10.1109/34.221171

Sarcar, M., Rao, K. M., & Narayan, K. L. (2008). *Computer aided design and manufacturing*. PHI Learning Pvt. Ltd.

Schnabel, R., Wahl, R., & Klein, R. (2007). Efficient RANSAC for Point-Cloud Shape Detection. *Computer Graphics Forum*, *26*(2), Article 2. https://doi.org/10.1111/j.1467-8659.2007.01016.x

Seland, J. S., & Dokken, T. (2007). Real-time algebraic surface visualization. *Geometric modelling, numerical simulation, and optimization: applied mathematics at SINTEF*, 163-183.

Shah, G. A., Polette, A., Pernot, J.-P., Giannini, F., & Monti, M. (2021). Simulated annealing-based fitting of CAD models to point clouds of mechanical parts' assemblies. *Engineering with Computers*, *37*(4), 2891-2909. https://doi.org/10.1007/s00366-020-00970-8

Shah, J. J., & Mäntylä, M. (1995). *Parametric and feature-based CAD/CAM: concepts, techniques, and applications*. John Wiley & Sons.

Shi, B., Bai, S., Zhou, Z., & Bai, X. (2015). Deeppano : Deep panoramic representation for 3-d shape recognition. *IEEE Signal Processing Letters*, *22*(12), 2339-2343.

Shin, H., & Sohn, K. (2006). 3D Face Recognition with Geometrically Localized Surface Shape Indexes. *2006 9th International Conference on Control, Automation, Robotics and Vision*, 1-6. https://doi.org/10.1109/ICARCV.2006.345192

Song, Y., Vergeest, J. S. M., & Bronsvoort, W. F. (2004). Fitting and Manipulating Freeform Shapes Using Templates. *Journal of Computing and Information Science in Engineering*, *5*(2), 86-94. https://doi.org/10.1115/1.1875592

Stark, J. (2022). Product lifecycle management (PLM). In *Product Lifecycle Management (Volume 1) 21st Century Paradigm for Product Realisation* (p. 1-32). Springer.

Su, H., Maji, S., Kalogerakis, E., & Learned-Miller, E. (2015). Multi-view convolutional neural networks for 3d shape recognition. *Proceedings of the IEEE international conference on computer vision*, 945-953.

Sundar, H., Silver, D., Gagvani, N., & Dickinson, S. (2003). Skeleton based shape matching and retrieval. *2003 Shape Modeling International.*, 130-139.

Tangelder, J. W., & Veltkamp, R. C. (2008). A survey of content based 3D shape retrieval methods. *Multimedia tools and applications*, *39*, 441-471.

Terzi, S., Bouras, A., Dutta, D., Garetti, M., & Kiritsis, D. (2010). Product lifecycle management–from its history to its new role. *International Journal of Product Lifecycle Management*, *4*(4), 360-389.

Theologou, P., Pratikakis, I., & Theoharis, T. (2015). A comprehensive overview of methodologies and performance evaluation frameworks in 3D mesh segmentation. *Computer Vision and Image Understanding*, *135*, 49-82. https://doi.org/10.1016/j.cviu.2014.12.008

Tombari, F., Salti, S., & Di Stefano, L. (2010). Unique Signatures of Histograms for Local Surface Description. In K. Daniilidis, P. Maragos, & N. Paragios (Éds.), *Computer Vision – ECCV 2010* (p. 356-369). Springer Berlin Heidelberg.

Várady, T., Facello, M. A., & Terék, Z. (2007). Automatic extraction of surface structures in digital shape reconstruction. *Computer-Aided Design*, *39*(5), 379-388.

Venkiteswaran, A., Hejazi, S. M., Biswas, D., Shah, J. J., & Davidson, J. K. (2016). Semantic Interoperability of GD&T Data Through ISO 10303 Step AP242. *Volume 2B: 42nd Design Automation Conference*, V02BT03A018. https://doi.org/10.1115/DETC2016-60133

Vergeest, J. S. M., Spanjaard, S., Horva´th, I., & Jelier, J. J. O. (2001). Fitting Freeform Shape Patterns to Scanned 3D Objects. *Journal of Computing and Information Science in Engineering*, *1*(3), 218-224. https://doi.org/10.1115/1.1419197

Vilmart, H., Léon, J.-C., & Ulliana, F. (2018). From CAD assemblies toward knowledge-based assemblies using an intrinsic knowledge-based assembly model. *Computer-Aided Design and Applications*, *15*(3), 300-317. https://doi.org/10.1080/16864360.2017.1397882

Virtanen, P., Gommers, R., Oliphant, T. E., Haberland, M., Reddy, T., Cournapeau, D., Burovski, E., Peterson, P., Weckesser, W., Bright, J., van der Walt, S. J., Brett, M., Wilson, J., Millman, K. J., Mayorov, N., Nelson, A. R. J., Jones, E., Kern, R., Larson, E., … SciPy 1.0 Contributors. (2020). SciPy 1.0 : Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, *17*, 261-272. https://doi.org/10.1038/s41592-019-0686-2

Vosgien, T. (2015). *Model-based system engineering enabling design-analysis data integration in digital design environments : Application to collaborative aeronautics simulation-based design process and turbojet integration studies* [PhD Thesis]. Ecole Centrale Paris.

Wang, J., Gu, D., Gao, Z., Yu, Z., Tan, C., & Zhou, L. (2013). Feature-Based Solid Model Reconstruction. *Journal of Computing and Information Science in Engineering*, *13*(011004). https://doi.org/10.1115/1.4023129

Wang, P.-S., Liu, Y., Guo, Y.-X., Sun, C.-Y., & Tong, X. (2017). O-cnn : Octree-based convolutional neural networks for 3d shape analysis. *ACM Transactions On Graphics (TOG)*, *36*(4), 1-11.

Wang, Y., & Solomon, J. M. (2019). Deep closest point : Learning representations for point cloud registration. *Proceedings of the IEEE/CVF international conference on computer vision*, 3523-3532.

Wu, Z., Song, S., Khosla, A., Yu, F., Zhang, L., Tang, X., & Xiao, J. (2015). 3d shapenets : A deep representation for volumetric shapes. *Proceedings of the IEEE conference on computer vision and pattern recognition*, 1912-1920.

Xiao, Y.-P., Lai, Y.-K., Zhang, F.-L., Li, C., & Gao, L. (2020). A survey on deep geometry learning : From a representation perspective. *Computational Visual Media*, *6*(2), Article 2. https://doi.org/10.1007/s41095-020-0174-8

Xu, X., Lu, Y., Vogel-Heuser, B., & Wang, L. (2021). Industry 4.0 and Industry 5.0—Inception, conception and perception. *Journal of Manufacturing Systems*, *61*, 530-535. https://doi.org/10.1016/j.jmsy.2021.10.006

Xu, Y., Fan, T., Xu, M., Zeng, L., & Qiao, Y. (2018). Spidercnn : Deep learning on point sets with parameterized convolutional filters. *Proceedings of the European conference on computer vision (ECCV)*, 87-102.

Z. Lian, A. Godil, & X. Sun. (2010). Visual Similarity Based 3D Shape Retrieval Using Bag-of-Features. *2010 Shape Modeling International Conference*, 25-36. https://doi.org/10.1109/SMI.2010.20

Annexes

Annexe 1. Méthodes de modélisation géométrique

Le chapitre 2 du livre Principles of Geometric *Model*ing de (Borrmann & Berkhahn, 2018) décrit les méthodes de modélisation géométrique des volumes tridimensionnels. La modélisation explicite, qui décrit un volume par la définition de ses surfaces, aussi connue sous le terme de *Bourndary Representation* (BRep), et la modélisation implicite qui emploie une séquence d'étapes de construction, communément appelé approche procédurale.

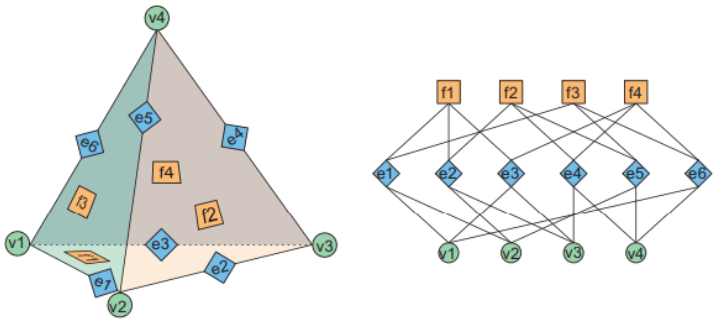
Modélisation explicite

On distingue les modèles BRep des maillages 3D.

- **Les Boundary Représentations**

Les modèles BRep sont définis par la hiérarchie d'éléments de « frontière » qui comprennent les corps (Body), les faces (Face), les côtés (Edge) et les sommets (Vertex).

Chaque élément est décrit pas ceux du niveau inférieur, définissant ainsi la topologie de modèle qui peut être décrite par un graphique comme sur la Figure 69.



solid	faces
1	1,2,3,4

face	edges
1	1, 2, 3
2	2, 4, 5
3	1, 5, 6
4	3, 4, 6

vertex	coordinates
1	2, 0, 0
2	0, 0, 0
3	3, 0, 0
4	1, 1, 3

edge	vertices
1	1, 2
2	2, 3
3	3, 1
4	3, 4
5	2, 4
6	1, 4

Figure 69: structure d'un modèle bRep (Borrmann & Berkhahn, 2018)

La définition d'un modèle BRep est augmentée par la nature géométrique des faces. Les surfaces dites primitives font références aux plans, cylindres sphères et tores. Des équations quadratiques et des éléments géométriques de références suffisent à les définir. On parle de surfaces *freeform* pour toutes celles nécessitant une définition géométrique plus complexe basée sur les B-Splines ou les Nurbs. Pour plus de détails, se référer à (Borrmann & Berkhahn, 2018; Piegl & Tiller, 1996).

- **Les maillages 3D (3D mesh)**

À la différence des modèles BRep, les maillages 3D sont constitués uniquement de faces, cotés et sommets. Dans le cas des maillages triangulaires par exemple, chaque face est plane, définie par trois sommets reliés par des côtés.

Le format stéréolithographie (stl) est l'un des plus largement utilisé par les applications de visualisation, d'animation et même de simulation 3D. La Figure 70 représente la structure du format de donné stl. Chaque face est définie par trois sommets, eux même définis par leurs coordonnées (x, y, z) dans un repère cartésien. Le vecteur normal de la face est défini par ses coordonnées (n_x, n_y, n_z) et indique le sens de la face. Le maillage est dit *watertight* s'il est fermé, c'est-à-dire que la surface délimite un volume. La longueur des mailles définit sa densité. Un maillage plus dense permet de mieux approximer une surface complexe, avec pour contrepartie une donnée de taille plus importante.

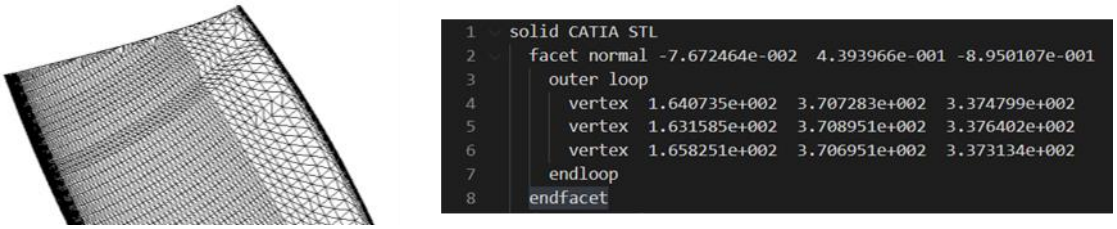


Figure 70: format stl de maillage triangulaire

Modéli*sa*tion implicite

Les méthodes de modélisation implicite stockent l'historique de la création d'un corps 3D. Elles représentent une approche alternative aux méthodes explicites décrites ci-dessus, qui stockent uniquement le résultat d'un processus de modélisation qui aurait pu être long et complexe.

- **Constructive Solid Geometry**

Une approche classique de la description procédurale des géométries 3D est la méthode Constructive Solid Geometry (CSG), qui utilise des objets de base prédéfinis qu'on appelle les primitives, comme des cubes, des cylindres ou les pyramides. Ces derniers sont assemblés par opérateurs booléens tels que l'union, l'intersection ou la différence pour créer davantage d'objets complexes.

Ce processus de combinaison aboutit à un arbre de construction qui décrit la génération du corps 3D (voir Figure 71). L'utilisation d'un ensemble fini de formes primitives limite les possibilités de modélisation de modèles complexes par la méthode CSG.

- **Les balayages réguliers**

La majorité des solutions de modélisation permettent la création de géométries par balayages réguliers à partir de géométries 2D construites dans des esquisses. Les principales méthodes sont illustrées dans la Figure 72. Les balayages réguliers permettent de créer une infinité de volumes paramétriques, c'est-à-dire modifiables via des paramètres qui pilotent la géométrie. L'arborescence de construction représente l'historique des opérations et les relations de parentés entre les opérations de constructions, parfois appelées *modeling features*. Les intentions du concepteur sont induites par le choix, l'ordre et la cotation des opérations. Pour cette raison, les balayages réguliers sont majoritaires pour la modélisation de composants mécaniques.

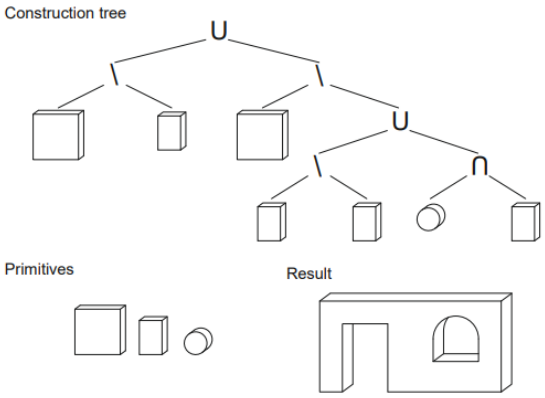


Figure 71 : Construction d'un modèle CSG (Borrmann & Berkhahn, 2018)

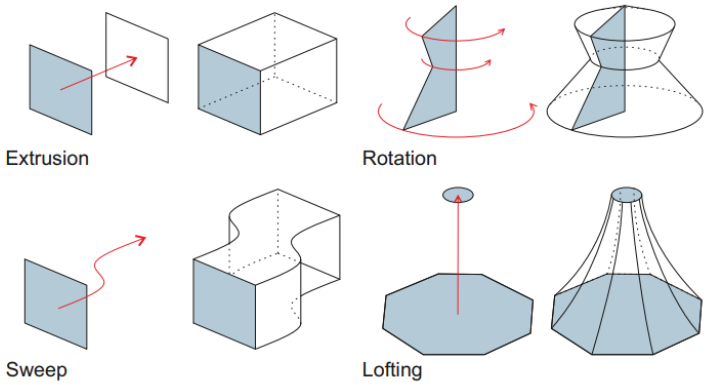


Figure 72 : Principaux balayages réguliers (Borrmann & Berkhahn, 2018)

La modélisation paramétrique est souvent réalisée dans un logiciel commercial avec un format de donnée natif propriétaire. Les échanges entre logiciels de CAO sont possibles par la traduction du modèle dans un format neutre, comme le format STEP issu de la norme ISO 10303 ou encore les formats IGS et JT.

Cependant, une perte globale ou partielle des informations fonctionnelles (i.e. arborescence de construction, paramètres et contraintes, etc.) est induite par la traduction de format par manque d'interopérabilité entre les solutions natives et les formats neutres. Ainsi, le modèle CAO au format neutre issu de la traduction d'un modèle paramétrique natif est généralement un modèle BRep qui ne contient plus que la définition explicite de la géométrie. Pour plus de détails sur la norme ISO 10303 et l'interopérabilité des formats CAO, se référer à (Nzetchou et al., 2019; Venkiteswaran et al., 2016).

Annexe 2. Documentation sur les MLP

Les éléments ci-dessous sont issus de la documentation de l'API scikit-learn[9] (Pedregosa et al., 2011)

Multi-layer Perceptron (MLP) is a supervised learning algorithm that learns a function $f(\cdot) : R^m \rightarrow R^o$ by training on a dataset, where m is the number of dimensions for input and o is the number of dimensions for output. Given a set of features $X = x_1, x_2, \ldots, x_n$ and a target y, it can learn a non-linear function approximator for either classification or regression. It is different from logistic regression, in that between the input and the output layer, there can be one or more non-linear layers, called hidden layers. Figure 1 shows a one hidden layer MLP with scalar output.

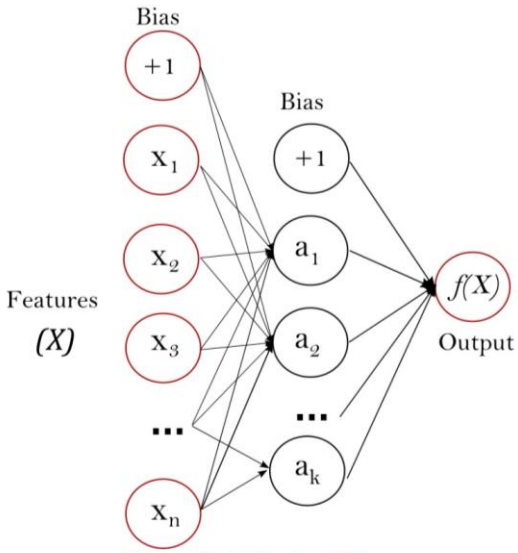


Figure 1 : One hidden layer MLP.

The leftmost layer, known as the input layer, consists of a set of neurons $\{x_i | x_1, x_2, \ldots, x_n\}$ representing the input features. Each neuron in the hidden layer transforms the values from the previous layer with a weighted linear summation $w_1x_1 + w_2x_2 + \ldots + w_mx_m$, followed by a non-linear activation function $g(\cdot) : R \rightarrow R$ - like the hyperbolic tan function. The output layer receives the values from the last hidden layer and transforms them into output values.

The module contains the public attributes `coefs_` and `intercepts_`. `coefs_` is a list of weight matrices, where weight matrix at index i represents the weights between layer i and layer $i + 1$. `intercepts_` is a list of bias vectors, where the vector at index i represents the bias values added to layer $i + 1$.

The advantages of Multi-layer Perceptron are:

- Capability to learn non-linear models.
- Capability to learn models in real-time (on-line learning) using `partial_fit`.

The disadvantages of Multi-layer Perceptron (MLP) include:

- MLP with hidden layers have a non-convex loss function where there exists more than one local minimum. Therefore different random weight initializations can lead to different validation accuracy.
- MLP requires tuning a number of hyperparameters such as the number of hidden neurons, layers, and iterations.
- MLP is sensitive to feature scaling.

[9] https://scikit-learn.org/stable/modules/neural_networks_supervised.html

MLP trains using Stochastic Gradient Descent, Adam, or L-BFGS. Stochastic Gradient Descent (SGD) updates parameters using the gradient of the loss function with respect to a parameter that needs adaptation, i.e.

$$w \leftarrow w - \eta(\alpha \frac{\partial R(w)}{\partial w} + \frac{\partial Loss}{\partial w})$$

where η is the learning rate which controls the step-size in the parameter space search. $Loss$ is the loss function used for the network.

Suppose there are n training samples, m features, k hidden layers, each containing h neurons - for simplicity, and o output neurons. The time complexity of backpropagation is $O(n \cdot m \cdot h \cdot o \cdot i)$where i is the number of iterations. Since backpropagation has a high time complexity, it is advisable to start with smaller number of hidden neurons and few hidden layers for training.

Given a set of training examples $(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)$ where $x_i \in R^n$ and $y_i \in \{0, 1\}$, a one hidden layer one hidden neuron MLP learns the function $f(x) = W_2 g(W_1^T x + b_1) + b_2$ where $W_1 \in R^m$ and $W_2, b_1, b_2 \in R$ are model parameters. W_1, W_2 represent the weights of the input layer and hidden layer, respectively; and b_1, b_2 represent the bias added to the hidden layer and the output layer, respectively. $g(\cdot) : R \rightarrow R$is the activation function, set by default as the hyperbolic tan. It is given as,

$$g(z) = \frac{e^z - e^{-z}}{e^z + e^{-z}}$$

For binary classification, $f(x)$ passes through the logistic function $g(z) = 1/(1 + e^{-z})$ to obtain output values between zero and one. A threshold, set to 0.5, would assign samples of outputs larger or equal 0.5 to the positive class, and the rest to the negative class.

If there are more than two classes, $f(x)$ itself would be a vector of size (n_classes,). Instead of passing through logistic function, it passes through the softmax function, which is written as,

$$softmax(z)_i = \frac{\exp(z_i)}{\sum_{l=1}^{k} \exp(z_l)}$$

where z_i represents the i th element of the input to softmax, which corresponds to class i, and K is the number of classes. The result is a vector containing the probabilities that sample x belong to each class. The output is the class with the highest probability.

In regression, the output remains as $f(x)$; therefore, output activation function is just the identity function.

MLP uses different loss functions depending on the problem type. The loss function for classification is Average Cross-Entropy, which in binary case is given as,

$$Loss(\hat{y}, y, W) = -\frac{1}{n}\sum_{i=0}^{n}(y_i \ln \hat{y}_i + (1 - y_i) \ln (1 - \hat{y}_i)) + \frac{\alpha}{2n}||W||_2^2$$

where $\alpha||W||_2^2$ is an L2-regularization term (aka penalty) that penalizes complex models; and $\alpha > 0$ is a non-negative hyperparameter that controls the magnitude of the penalty.

For regression, MLP uses the Mean Square Error loss function; written as,

$$Loss(\hat{y}, y, W) = \frac{1}{2n}\sum_{i=0}^{n}||\hat{y}_i - y_i||_2^2 + \frac{\alpha}{2n}||W||_2^2$$

Starting from initial random weights, multi-layer perceptron (MLP) minimizes the loss function by repeatedly updating these weights. After computing the loss, a backward pass propagates it from the output layer to the previous layers, providing each weight parameter with an update value meant to decrease the loss.

In gradient descent, the gradient $\nabla Loss_W$ of the loss with respect to the weights is computed and deducted from W. More formally, this is expressed as,

$$W^{i+1} = W^i - \epsilon \nabla Loss_W^i$$

where i is the iteration step, and ϵ is the learning rate with a value larger than 0.

The algorithm stops when it reaches a preset maximum number of iterations; or when the improvement in loss is below a certain, small number.

Annexe 3. Shapes Functions

Cette annexe présente un ensemble de shape functions synthétisées par (Laube, 2020), pouvant être utilisées pour la création de descripteurs de type Shape Distribution (SD) de (Osada et al., 2002).

- **sf1** (A3) Point angles : Angles between two vectors spanned by three random points. The histogram ranges from 0 to 180 degrees
- **sf2** (D2) Point distances : Euclidean distance δ between two random points
- **sf3** (D1) Centroid distances : Euclidean distance of random points to the bounding box centroid
- **sf4** (D3) Triangle areas : Square root of the triangle areas of random three-tuples of points
- **sf5** (D4) Tetrahedron volumes : Cubic root of the tetrahedron volume V of four random points
- **sf6** Cube cell count : Number of points contained in 8 equally sized cells, which result from a uniform subdivision of the point cloud's bounding box. This feature is not invariant to rotation
- **sf7** K-Median points : This is strongly related to the well knownk-means algorithm. From the x, y, and z coordinates three coordinate-histograms are computed and concatenated. k=32clusters are used
- **sf8** Normal angles : Angles between two normals at two random points
- **sf9** Normal directions : Coordinates of unit normals at all points
- **sf10** Principal curvatures : κ1, κ2 are computed by polynomial fitting of osculating jets
- **sf11** Mean curvatures: H=1/4(κ1+κ2)
- **sf12** Gaussian curvatures: K=κ1κ2
- **sf13** Curvature ratios:|κ1/κ2|
- **sf14** Curvature changes : Absolute difference between a random point's principal curvatures and those of its nearest neighbour
- **sf15** Curvature angles: Angles between the two corresponding principal curvature directions v1, v2 at two random points
- **sf16** Curvature directions: Coordinates of the two normalized principal curvature directions v1, v2 at all points
- **sf17** Curvature differences: Absolute differences of the principal curvatures, the Gaussian curvature, and the mean curvature at two random points
- **sf18** Shape index as defined in (Shin & Sohn, 2006)
- **sf19** Surflet pairs: The tuple (α, β, γ, δ) for two random points (R. B. Rusu et al., 2009)
- **sf20** Triple combination of the best point-, normal-, and curvature-features : sf5, sf8, and sf15
- **sf21** Simple surflet combination of sf5, sf8, and sf19
- **sf22** Extended surflet combination of sf19 and sf20
- **sf23** All features combination of features sf1 to sf19

Annexe 4. Métriques d'évaluation

Soit la méthode $f(x) = y$ retournant une prédiction y associée à x. On appelle $X = (x_1, x_2, \dots, x_{n\_samples})$ un ensemble de données test. $Y = (y_1, y_2, \dots, y_{n\_samples})$ est l'ensemble des "vraies" valeurs de prédictions y_i (i.e. prédiction théorique) associées à chaque donnée x_i de X, et $\hat{Y} = (\hat{y}_1, \hat{y}_2, \dots, \hat{y}_{n\_samples})$ l'ensembles des prédictions de f pour chaque x_i.

On note $L = (l_1, l_2, \dots, l_{n\_labels})$ l'ensemble des labels possibles, soit les valeurs que peuvent prendre chaque y_i et $\hat{y}_i$. Y_l est le sous-ensemble des y_i de Y de valeur l (de manière similaire, $\hat{Y}_l$ est le sous-ensemble de $\hat{Y}$ avec $\hat{y}_i = l$).

Les métriques d'évaluation de $\hat{Y}$ par rapport à Y sont définies comme suit :

- $mean\_square\_error$ mesure la moyenne des erreurs quadratiques entre les valeurs prédites par le modèle et les valeurs théoriques associées. Notons que dans ce cas, y_i et $\hat{y}_i$ sont des valeurs numériques.

$$mean\_square\_error(Y, \hat{Y}) = \frac{1}{n\_samples} \sum_{i=1}^{n\_samples} (\hat{y}_i - y_i)^2$$

- $accuracy\_score$ mesure la proportion de prédictions correctes sur le nombre total de données de test. Si $\hat{y}_i$ est la valeur prédite de la i-eme donnée de test, et y_i est la valeur valeur théorique correspondante, alors la proportion des prédictions correctes sur les $n\_samples$ tests est:

$$accuracy\_score(Y, \hat{Y}) = \frac{1}{n\_samples} \sum_{i=1}^{n\_samples} 1(\hat{y}_i = y_i)$$

$$\text{Avec } 1(a) := \begin{Bmatrix} 1 \; if \; a = True \\ 0 \; if \; a = False \end{Bmatrix}$$

- $top - k\_accuracy\_score$ est une généralisation de $accuracy\_score$ pour laquelle une prédiction est considérée comme correcte si le "vrai" résultat y_i est associé à l'une des $k \geq 1$ prédictions avec les plus hauts scores. Dans ce cas, on note $\hat{Y}_i = (\hat{y}_{i,1}, \hat{y}_{i,2}, \dots, \hat{y}_{i,k})$ les k résultats de prédictions avec les plus hauts scores associés à chaque x_i.

$$top - k\_accuracy\_score(Y, \hat{Y}) = \frac{1}{n\_samples} \sum_{i=1}^{n\_samples} \sum_{j=1}^{k} 1(\hat{y}_{i,j} = y_i)$$

$$\text{Avec } 1(a) := \begin{Bmatrix} 1 \; if \; a = True \\ 0 \; if \; a = False \end{Bmatrix}$$

On remarque que $accuracy\_score(Y, \hat{Y})$ est en réalité un cas particulier de $top - k\_accuracy\_score(Y, \hat{Y})$ avec $k = 1$.

Dans certains cas, il est important de considérer les éventuelles différences de résultats entre les sous-ensembles $\hat{Y}_l$ de $\hat{Y}$ On évalue alors les résultats associés à un label l de L comme pour une fonction binaire qui ne prédit que les deux résultats possibles $f(x) = l$ ($ie.$"$True$") ou $f(x) \neq l$ ($ie.$"$False$") pour tout x. On note alors :

- TP_l l'ensemble des "True Positives" de $\hat{Y}_l$ avec $\hat{y}_i = l$ et $y_i = l$
- FP_l est l'ensemble des "False Positives" de $\hat{Y}_l$ avec $\hat{y}_i = l$ alors que $y_i \neq l$.
- FN_l est l'ensemble des "False Negatives" de $\hat{Y}_l$ avec $\hat{y}_i \neq l$ alors que $y_i = l$.
- TN_l est l'ensemble des "True Negatives" de $\hat{Y}_l$avec $\hat{y}_i \neq l$ et $y_i \neq l$.

- $precision\_score_l$ est la fraction des données correctement prédites sur l'ensemble des données prédites comme appartenant à $\hat{Y}_l$. Autrement dit, la précision mesure la capabilité du modèle à correctement identifier les données d'une classe particulière

$$precision\_score_l = \frac{TP_l}{TP_l + FP_l}$$

- $recall\_score_l$ est la fraction des données correctement prédites comme appartenant à $\hat{Y}_l$ sur l'ensemble des données de Y_l. Autrement dit, le recall mesure la capabilité du modèle à identifier toutes les données d'une classe particulière

$$recall\_score_l = \frac{TP_l}{TP_l + FN_l}$$

- $F1\_score_l$ est la moyenne harmonique des scores de précision et de recall. Sa valeur se situe entre 0 et 1:

$$F1 - score_l = 2 * \frac{(precision\_score_l * recall\_score_l)}{precision\_score_l + recall\_score_l}$$

- Pour généraliser les scores de précisons sur l'ensemble des résultats $\hat{Y}$, on étudie la métrique $average\_precision\_score$, soit la macro-moyenne des scores de précision sur l'ensemble des classes du modèle.

$$average\_precision\_score = \frac{1}{|L|}\sum_{l \in L} precision\_score_l$$

- $average\_recall\_score$ et $average\_F1\_score$ sont définis de la même manière que $average\_precision\_score$ en calculant la moyenne des valeurs $recall\_score_l$ ou $F1 - score_l$ pour chaque sous ensemble l de L.