Modèle pour la conception immersive et intuitive : application à l'industrie automobile

Pierre Martin

Université Paris-Sud
École Doctorale d'Informatique de l'Université Paris-Sud

Thèse

pour l'obtention du grade de

Docteur de l'Université Paris-Sud
Spécialité Informatique

Soutenue le 7 juillet 2014 par

Pierre MARTIN

Modèle pour la conception immersive et intuitive : application à l'industrie automobile

Composition du jury :

M.	Umberto CUGINI	Professeur Politecnico di Milano	Rapporteur
M.	Doru TALABA	Professeur Universitatea Transilvania din Brasov	Rapporteur
M.	Jean-Claude MARTIN	Professeur Université Paris-Sud	Examinateur
Mme.	Indira THOUVENIN	Enseignant-Chercheur Université de Technologie Compiègne	Examinatrice
M.	Patrick BOURDOT	Directeur de Recherche Université Paris-Sud	Directeur de thèse
M.	Stéphane MASFRAND	Responsable MSRV PSA Peugeot Citroën	Co-encadrant CIFRE

Thèse préparée au sein du
Laboratoire d'Informatique pour la Mécanique et les Sciences de l'Ingénieur
(LIMSI-CNRS)

Groupe RV&A VENISE
LIMSI-CNRS
B.P. 133
91403 Orsay Cedex, France

EDIPS - ED427
UFR Sciences - Université Paris-Sud
Bat 650 PCRI bureau 417 - Rue Noetzlin
91403 Orsay Cedex, France

Cette thèse est dédiée à
mes défunts grand-pères,
ma famille,
et Amandine.

Remerciements

Je tiens tout d'abord à remercier mon directeur de thèse M. Patrick Bourdot, Directeur de Recherche et directeur du groupe VENISE du LIMSI-CNRS. Cette thèse est l'aboutissement de six années passées dans son groupe, tout d'abord comme stagiaire, puis finalement, par une construction lente mais sûre, comme doctorant CIFRE puis docteur. Je le remercie chaleureusement pour la confiance qu'il a porté en moi, et son réel enthousiasme quant aux différents travaux que nous avons menés ensemble, et notamment ceux de cette thèse sur un sujet qui lui est cher.

Cette thèse n'aurait pu avoir lieu sans mon co-encadrant M. Stéphane Masfrand, Responsable des Moyens de Simulations de Conduite et de Réalité Virtuelle chez PSA Peugeot Citroën. Je le remercie énormément pour la confiance qu'il m'a faite ainsi que les nombreux conseils qui m'ont permis d'intégrer au mieux mes recherches dans ce contexte industriel.

Je veux ensuite exprimer toute ma gratitute à mes deux rapporteurs, M. Umberto Cugini, Professeur à Politecnico di Milano (Italie), et M. Doru Talaba, Professeur à l'Universitatea Transilvania din Brasov (Roumanie), d'avoir accepté de lire et d'évaluer mon travail. J'en suis très honoré. De même, je remercie les autres membres du jury, Mme. Indira Thouvenin, Enseignant-Chercheur à l'Université de Technologie de Compiègne, et M. Jean-Claude Martin, Professeur à l'Université Paris-Sud. Je suis flatté d'avoir pu montrer mes travaux à ce jury, et qu'il ait pu les apprécier. Les discussions autour du sujet ont été extrêmement intéressantes et j'espère que ces travaux seront continués, ici ou ailleurs.

Un groupe de recherche, c'est aussi une ambiance, une équipe, des gens sur lesquels on peut compter à chaque instant. Je remercie énormément Jean-Marc pour tous ses bons conseils, sa compréhension ; Nicolas pour toutes les discussions, de travail et autres bien sûr, ses conseils, enfin d'avoir été toujours présent et compréhensif quand il le fallait ; Damien pour m'avoir mis le pied à l'étrier de la Réalité Virtuelle en étant mon premier tuteur de stage il y a six ans maintenant.

Durant ces cinq années, j'ai côtoyé bon nombre de camarades de jeu, doctorants eux aussi. Je n'oublierai pas tous ces moments, et je remercie notamment Bob, Flavien, Guillaume, Sebastien, Tony, Mikael, Weiya, Nicolas, Alexandre. Je comprends maintenant cet état si particulier dans lequel on tombe les dernières semaines de doctorat. Avant de devenir thésard, il faut passer par la case stagiaire. Je remercie Rémi, qui je l'espère reprendra le flambeau, pour son travail de stage lors de la fin de ma thèse ; Constantin, Anthony, pour leurs travaux qui ont permis de faire avancer mes recherches.

Enfin je remercie les gens du LIMSI qui m'ont rendu le séjour dans ce laboratoire plus agréable, en étant cobaye, en testant des cobayes (Céline), en voyageant (Tom), en rendant de grands services (secrétariat, infrastructures, équipe informatique, cafétériat) et tant d'autres que j'oublie sûrement.

Le cadre industriel de cette thèse aura été le Centre de Réalité Virtuelle (CRV) de PSA Peugeot Citroën à Vélizy, et donc l'équipe de Stéphane Masfrand mon encadrant CIFRE. Dans ce cadre j'ai été amené à côtoyé un bon nombre de personne, depuis mon premier stage jusqu'à la fin de cette thèse. Je remercie donc énormément l'équipe actuelle du CRV, Thomas, Benoît, Michel, Matthieu, Jocelyn, pour leur accompagnement et leur bonne humeur. J'ai une pensée également pour tous ceux dont j'ai croisé le chemin chez PSA au CRV, Aurélie, Félicien, Clément (c'est lui le rétroviseur), les alternants (Renaud, Rémy, Franck, Laure), Fred, et tous ceux que j'ai croisés sur le gazon la nuit tombant, Laurent, les Juliens, Clément, Bruno, Christophe, Alessandro, et bien d'autres. Enfin je remercie Mihaï qui anime la communauté des doctorants chez PSA, pour sa gentillesse et le soutien qu'il nous apporte.

J'ai une pensée particulière pour mes défunts grand-pères, Jean-Paul, à qui c'est la "faute" si j'ai un ordinateur entre les mains et si j'en suis là aujourd'hui, et Bernard, dont le sourire m'accompagne tous les jours. J'embrasse très fort ma famille, et mes parents qui ont toujours cru en moi et qui m'ont toujours soutenu dans mes choix et mes projets. Enfin je n'aurais pu arriver au bout de ce travail sans le soutien sans faille de celle qui partage ma vie, Amandine. Cette thèse c'est aussi un peu la tienne, et j'espère qu'elle nous emmènera loin...

Résumé

Cette thèse traite de l'utilisation des technologies de Réalité Virtuelle (RV) dans les activités de Conception Assistée par Ordinateur (CAO). Plus précisément, les travaux de recherche portent sur une approche pour la modification directe et interactive d'objets CAO, notamment adaptée aux processus de conception en industrie.

Généralement, les logiciels de CAO requièrent des compétences (expérience et connaissance), à la fois sur les fonctionnalités même du logiciel et les représentations utilisées, ainsi que sur les objets CAO concernés (principalement sur leur historique de construction, savoir de quelle façon ils ont été construits). D'un autre coté, la RV apporte de nouveaux paradigmes d'interaction 3d, tels que l'immersion et la perception multi-sensorimotrice (stéréoscopie, audio 3d, haptique, etc.), et il apparaît nécessaire de disposer de *middleware* intelligents pour gérer les objets CAO dans ces Environnements Virtuels (EV) immersifs. De précédents travaux ont proposé un mécanisme d'édition implicite d'objets CAO permettant la modification du Graphe d'Historique de Construction (GHC) de ces objets à partir de la manipulation de la représentation visuelle 3d de ces objets. Basé sur un processus d'étiquetage des éléments de frontière (B-Rep), et couplé avec un moteur d'inférence, ce mécanisme décrit un chaînage arrière entre ces éléments de frontières et les opérateurs d'un GHC. Cependant, cette approche avait pour limite majeure de proposer un modèle particulier de GHC, ce qui l'empêchait d'être intégrée à des systèmes CAO fermés ou commerciaux tels que ceux utilisés dans l'industrie et en particulier l'industrie automobile.

Notre première contribution consiste donc en la proposition d'un modèle et d'une architecture permettant de généraliser ce mécanisme de chaînage arrière à n'importe quel système CAO basé sur les représentations de type B-Rep et GHC. Pour ce faire, nous avons spécifié plusieurs structures d'encapsulation pour la gestion des opérateurs du GHC ainsi que de leurs paramètres, et des composants de B-Rep. Deuxièmement, le précédent étiquetage, désormais attaché à ces structures d'encapsulation et non plus aux éléments de B-Rep directement, a été étendu pour permettre un chaînage arrière multiple. Certains éléments de frontières peuvent en effet être le résultat de plusieurs opérateurs du GHC, être liés à plusieurs éléments "parents", et ainsi plusieurs décisions peuvent être inférées à partir de leur manipulation. Ces avancées rendent possible la modification directe et intuitive d'objets CAO déjà existants (i.e. via le parcours et l'analyse de base de données CAO précédemment créées), en analysant leur GHC et en remplissant nos structures avec les données nécessaires. De plus, le mécanisme de chaînage arrière multiple renforce la capacité du moteur d'inférence à libérer les utilisateurs, et spécialement les non-experts, de connaissances trop complexes à propos des modèles CAO. Comme preuve de concept de notre modèle, nous présentons un exemple détaillé de notre approche sur le noyau géométrique de CATIA et montrons comment notre modèle permet d'envisager un nouveau concept d'interaction en revue de projet immersive : permettre aux participants de modifier directement les modèles CAO sans quelque interaction sur station de travail.

Mots-clefs : réalité virtuelle, CAO, graphe d'historique de conception, B-Rep.

Abstract

This thesis addresses the use of Virtual Reality (VR) technologies in the activities of Computer-Aided Design (CAD). More precisely, this research focuses on an approach for direct and interactive modifications of CAD objects, an approach which might be adapted to the conception process in industry.

Usually, CAD software requires some skills (experience and knowledge), on the software's functionalities and representations, as well as on CAD objects (principally on their design history, on the way they were built). Moreover, VR technologies bring new interactive paradigms of 3D interaction, such as immersion and multi-sensorimotor perception (stereoscopy, 3D audio, haptics, and so on), and one needs intelligent middleware to manage CAD objects in these immersive Virtual Environments (VE). Some previous work proposed a mechanism allowing implicit edition of CAD objects, from the manipulation of their 3D visual representations. Based on a technique of Boundary Representations (B-Rep) elements labelling, and coupled with an inference engine, this mechanism describes a backward chaining of B-Rep elements towards the operators of a dedicated model of Constructive History Graphs (CHG). However, this approach had a major limitation: since it was based on a specific model of CHG, its integration within commercial CAD softwares used in industry (and especially in automotive industry) was far from obvious.

Our first contribution is then to propose a data model and an architecture to generalize this backward chaining mechanism to any of CAD system based on B-Rep and CHG representations. In order to to that, we have designed several encapsulations structures, to manage CHG operators and their parameters, and the B-Rep components. Secondly, the previous labelling, now attached to these structures, has been extended to enable a multi backward chaining. Actually, some B-Rep elements may be the result of several CHG operators, and thus, several decisions may be inferred from their manipulation. These improvements make possible to have direct and interactive modifications of existing CAD objects by parsing their CHG to fill our structures with useful data. Moreover, the multi backward chaining mechanism reinforces the ability of the inference engine to free users, especially non-expert ones, from too complex understandings on CAD models. As a proof of concept of our model, we present a detailed example of our approach on the geometric kernel of CATIA and we show how one can consider new concepts of interaction during immersive project reviews: to allow participants to directly modify CAD objects without any interaction on desktop workstation.

Keywords: virtual reality, CAD, constructive history graph, B-Rep.

Table des matières

Glossaire

Notation ou expression	Signification
API	Application Programming Interface
B-Rep	Boundary Representation
CAO	Conception Assisté par Ordinateur
CAVE	Cave Automatic Virtual Environment
CSG	Constructive Solid Geometry
GHC	Graphe d'Historique de Construction
PLM	Product Lifecycle Management
RV	Réalité Virtuelle

Introduction Générale

Contexte et problématique

Cette thèse porte sur l'utilisation des technologies de **Réalité Virtuelle** (RV) dans les activités de **Conception Assistée par Ordinateur** (CAO). Plus précisemment, les travaux de recherche présentés dans ce mémoire décrivent un modèle de données et une architecture permettant de généraliser une approche pour la modification directe et interactive d'objets CAO, notamment adaptée aux processus de conception en industrie.

Les technologies de RV sont de plus en plus exploitées dans l'étude du **cycle de vie des produits** (PLM, en anglais), en particulier pour leur conception et celle de leur processus de fabrication. C'est typiquement le cas dans le secteur de l'industrie automobile où ces technologies servent, par exemple, à étudier l'ergonomie du poste de conduite, à analyser les problèmes d'assemblages, ou à concevoir des chaînes de montage. En outre, les grands dispositifs immersifs, à savoir les systèmes de type CAVE [51] permettant notamment la visualisation d'objets/de maquettes à l'échelle 1:1, sont particulièrement utiles aux revues de projet sur maquette numérique où des équipes métier/projet décident des évolutions à réaliser sur le produit en gestation. C'est sur les revues de projet, allant des activités de *design* (style, lignes, surfaces, etc.) aux activités de *conception détaillée* (architecture, pièces mécaniques, outils, etc.) que porte ce sujet de thèse, car les approches actuellement proposées en RV induisent une organisation complexe du travail, qui ne permet pas d'atteindre une efficacité optimale en termes de nombre de solutions évaluées en un temps donné.

De nos jours, on peut distinguer deux types de revue de projet, chaque type s'articulant sur un modèle de gestion particulier de la maquette numérique. Le premier, à caractère plutôt esthétique ou ergonomique, s'intéresse à passer en revue le produit pour prendre des décisions sur son style (couleur, matière, ou autre paramètre ne remettant pas en cause la "forme" du produit) ou valider ses usages en mettant le produit en situation. Le couplage RV-CAO associé à ce genre d'activité, s'appuie généralement sur le concept de modèle intermédiaire. Dans ce cas, les données issues des systèmes CAO subissent une série de conversions et de formatages à destination de la plateforme de RV utilisée (par exemple, Virtools [57]) pour obtenir une maquette numérique à la fois plus simple à gérer en RV, mais en même temps permettant des rendus réalistes du produit à l'aide de différents *shaders*. Le second type de revue de projet s'intéresse à la validation et l'évolution du produit du point de vue de ses fonctionnalités techniques ou de son ingénierie. Dans ce cas, la revue du projet doit s'opérer sur les données fournies par les concepteurs généralement au format d'un système CAO, comme CATIA [58]. Pour disposer de sessions immersives, différentes solutions de *clustering* graphique ont été publiées depuis quelques années [50], tandis que certaines autres sont commercialisées, soit comme produit à part (cf. Techviz [158]), soit directement intégrées dans les plateformes de RV (cf. Virtools). L'utilisation de telle ou telle solution dépend donc du type de session immersive, du type de données que l'on veut manipuler, ainsi que du type d'interactions que

l'on veut avoir avec ces données.

L'ambition de nombreux industriels est maintenant de gagner en efficacité pour les deux types de revue de projets identifiés ci-dessus. En effet, dans les deux cas, une séance de travail dans le dispositif immersif est organisée durant laquelle les évolutions du produit sont envisagées par les différents acteurs de la conception. Ces évolutions sont ensuite mises en œuvre en bureau d'étude par les concepteurs, puis une nouvelle revue est programmée au sein du dispositif immersif plusieurs jours après. L'une des problématiques de cette thèse est donc de **réduire le temps entre plusieurs itérations de conception**. En effet, les aller-retours peuvent facilement se multiplier et il est primordial de pouvoir se passer de conversion, et de travailler directement sur les données fournies par les concepteurs. L'amélioration de l'efficacité de ce processus permettra à terme de réduire les délais de conception du produit, ou bien, à durée de conception constante, de multiplier le nombre des simulations sur le produit.

Généralement, les logiciels de CAO requièrent des compétences — de l'expérience et de la connaissance — à la fois sur les fonctionnalités même du logiciel et les représentations utilisées, ainsi que sur les objets CAO concernés. Ce sont via des interfaces "classiques" 2d (clavier, souris, etc.) sur un poste de travail, que se font les interactions sur les maquettes CAO. L'utilisation de ces techniques "classiques" n'est clairement pas envisageable pour des revues de projet immersives, du moins pour les utilisateurs immergés. Mais l'un des intérêts réputés de la RV est de **permettre des interactions dites intuitives**. Actuellement, les revues de projet immersives visant la modification de la définition même du produit voient l'existence d'une séparation entre la visualisation stéréoscopique, assurant l'immersion au sein de l'environnement virtuel, et les interactions sur les données CAO qui se font sur une station de travail déportée.

Dans cette thèse nous nous consacrons donc à **élaborer un modèle d'interfaçage RV-CAO** tenant compte du contexte et du processus de conception industriels, afin de faire évoluer les méthodes de travail — amener les utilisateurs à pouvoir à la fois visualiser et interagir de façon immersive avec les données CAO — et d'obtenir un gain d'efficacité dans ces processus.

Organisation du manuscrit

Le **premier chapitre** présente le contexte et les domaines abordés durant ces travaux de recherche, ainsi que le positionnement de notre travail au sein de ces différents domaines. Dans un premier temps nous présentons ce qu'est le cycle de vie d'un produit (PLM), processus important dans l'industrie, son état et notamment la place que prend la CAO au sein de ce processus, ses perspectives et ses enjeux. Nous exposons ensuite une revue de l'utilisation de la RV dans ce cycle de vie du produit, en parcourant les diverses applications que l'on peut trouver à l'heure actuelle. Une fois ce contexte posé, nous présentons un état de l'art de l'interfaçage RV-CAO, ce qui permet de positionner notre approche et nos travaux.

Dans le **deuxième chapitre**, nous rappelons et décrivons les principaux concepts de la CAO abordés et exploités dans mes travaux, ainsi que leur utilisation au sein de logiciels commerciaux. Nous insistons sur la modélisation paramétrique et sur les logiciels basés sur ces concepts, ainsi que sur les problèmes majeurs liés à cette modélisation, tel que le *persistent naming*. Ensuite est décrit la place de la CAO dans le processus de conception chez PSA Peugeot Citroën, et l'état de l'utilisation des outils CAO avec les technologies de RV dont dispose ce groupe industriel. Ceci permet de préciser le champ d'application direct de mes travaux ainsi que les perspectives à long terme.

Notre modèle d'interfaçage RV-CAO est présenté dans le **troisième chapitre**. Ce modèle, reposant sur le principe de *réactivité* des objets, se veut une méthode pour l'application de ce principe à n'importe quel contexte, en particulier le contexte industriel caractérisé par des outils et des besoins spécifiques. Nous détaillons ainsi les différentes parties de ce modéle, comprenant plusieurs structures d'encapsulation de données permettant la gestion des éléments des représentations B-Rep et GHC (opérateurs, paramètres, éléments de frontière, éléments d'esquisse, contraintes), ainsi que les algorithmes et processus en jeu (récupération et mise à jour des données, etc.). Nous explicitons également l'évolution des deux processus clés pour la réactivité : le *labelling* et l'identification des cibles. Enfin nous présentons le couplage de ces structures de données au système immersif choisi.

Le **quatrième chapitre** aborde les apports directs de ce modèle d'interfaçage RV-CAO : ce qu'il est possible de faire. Nous présentons les schémas de réactivité que l'on peut établir à partir de ce modèle, à savoir les différents types et niveaux d'interaction, ainsi que les fonctionnalités qu'il est possible de mettre en place, notamment la supervision des erreurs lors de l'interaction. Pour illustrer ces apports nous détaillons un exemple de modification interactive d'objet CAO — un rétroviseur — grâce à une implémentation de cet interfaçage RV-CAO autour de CATIA et de son noyau géométrique : *cRea-RV*.

Enfin dans le **cinquième chapitre** nous présentons les perspectives de la RV-CAO : ce qu'il est permis d'imaginer. Nous considérons trois niveaux de perspectives. D'abord sont explicitées les perspectives *à court terme*. Elles comprennent toutes les évolutions de notre modèle d'interfaçage RV-CAO : extension des comportements et modifications gérés (plus d'opérateurs pris en compte, plus de paramètres, etc.), ajout d'un moteur d'inférence pour superviser les interactions de l'utilisateur, prise en compte du profil des utilisateurs pour adapter le modèle durant l'interaction, et développement de nouvelles interactions (multimodalité, guidage pseudo/haptique). Viennent ensuite les perspectives *à moyen terme*. Elles supposent une évolution ou déclinaison du modèle : pour une adaptation à la modélisation surfacique, ou dans l'optique d'une utilisation experte pouvant conduire à un conflit de connaissances : l'interprétation de l'utilisateur expert *versus* la construction même de l'objet. Enfin les perspectives *à long terme* vont concerner le développement de nouveaux modèles d'interfaçage pour étendre l'utilisation de la RV-CAO dans le processus de conception.

Chapitre 1

La gestion de cycle de vie du produit et la RV-CAO

Résumé. *Dans ce chapitre sont donnés le contexte de travail et le domaine général dans lesquels s'inscrivent nos travaux. Nous y décrivons ce qu'est le Product Lifecycle Management (PLM) ainsi que ses liens avec la Conception Assistée par Ordinateur (CAO). Nous montrons la place qu'ont prise les technologies de la Réalité Virtuelle (RV) au sein de ce PLM et leurs différents champs d'application. Un état de l'art de l'interfaçage RV-CAO est présenté afin de préciser le positionnement de nos travaux. Enfin nous exprimons la problématique et les objectifs de cette thèse.*

1.1 Introduction

Les travaux de recherche de cette thèse se placent à l'intersection de trois domaines : la gestion de cycle de vie du produit (en anglais, Product Lifecycle Management - PLM), la Conception Assistée par Ordinateur (CAO) et la Réalité Virtuelle (RV). Dans notre société, l'informatique est partie intégrante de la vie de tous les jours, tant dans le domaine privé que professionnel. L'industrie a naturellement intégré au plus vite l'informatique dans ses différents processus (conception, fabrication, gestion). La capacité d'adaptation et d'intégration des nouvelles technologies au sein du PLM représente un enjeu majeur pour l'industrie, dans une concurrence mondiale acharnée. C'est ainsi qu'au cours des dernières décennies, la CAO s'est de plus en plus introduite au sein du PLM, pour devenir aujourd'hui un outil indispensable notamment dans la phase amont de la conception du produit. Le travail autour de données et maquettes numériques est devenu incontournable. En parallèle, la RV a elle aussi fait sa place dans l'industrie, toujours dans l'optique de gagner en efficacité, en réduisant le temps de conception du produit.

1.2 Gestion de cycle de vie d'un produit (PLM)

La stratégie et la gestion globale du processus de conception sont parmi les points les plus importants que doivent maîtriser les entreprises dans le contexte concurrentiel actuel. Une organisation défaillante et non optimisée des tâches et activités de conception pourrait être la cause de déboires et de pertes énormes. Le PLM, actuellement en cours de déploiement chez PSA Peugeot Citroën, permet notamment aux industries automobile et aéronautique qui l'ont bien intégré, de réaliser des nombreux bénéfices quantitatifs et qualitatifs, et de renforcer leurs positions [2].Ainsi dans cette section nous nous intéressons au cycle de vie d'un produit et sa gestion en donnant des définitions et des exemples, et en décrivant les étapes importantes de ce cycle. Ensuite nous nous centrons sur la CAO et sur son importance au sein du PLM, avec notamment l'utilisation de maquettes numériques.

La communauté scientifique a largement balayé le domaine de la conception et nous pouvons donc en extraire quelques définitions précises. Ainsi, nous basant sur les travaux de Maître et al. [104] et de Duchamp [65], nous considérons la conception comme la *transformation d'une idée en un produit*. Plus spécifiquement, et pour comprendre les différents éléments en jeu dans le processus de conception, il nous faut rappeler ce que Pomian et al. [132] définissent comme "méthode" et "technique" :

> *Une méthode propose une démarche cherchant à définir l'ordonnancement de toutes les activités à mener à bien pour atteindre un objectif final. Une technique est un procédé permettant de réaliser une ou plusieurs de ces activités. Une méthode peut donc demander l'emploi successif de plusieurs techniques.*

Ce sont sur ces bases que Marsot [109] définit ce qu'est la "démarche de conception" :

> *La démarche de conception représente l'ensemble des activités, donc des acteurs — acteur n'est pas forcément attaché à une seule personne et peut être un service, une entreprise, un organisme représentés par une personne — et de leurs tâches*

qu'il faut organiser au mieux pour transformer un concept (abstrait) en un produit (réel). Les techniques (ou outils) désignent l'ensemble des moyens utilisés lors des différentes activités.

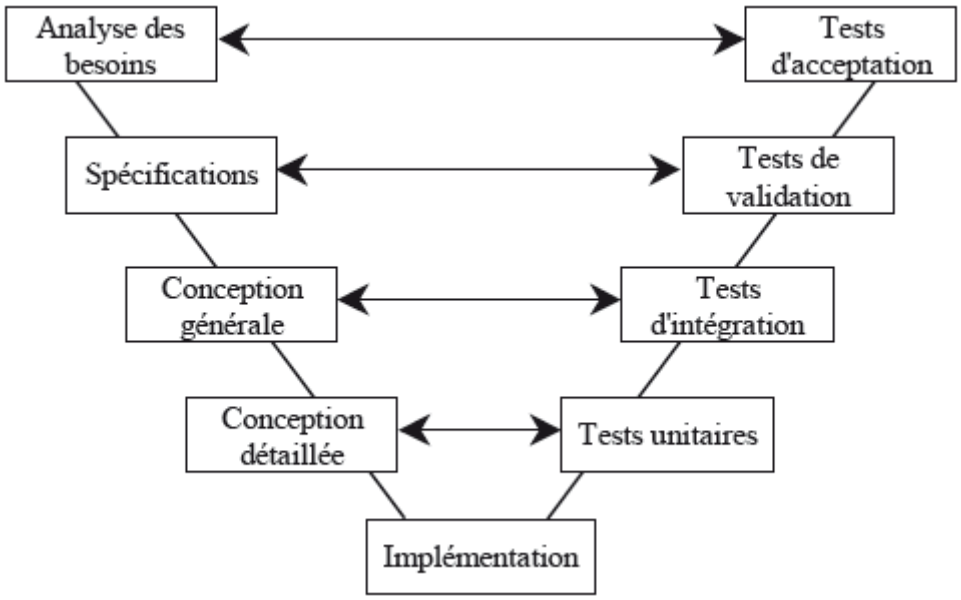


FIGURE 1.1 – Cycle de conception en "V"

L'organisation de ces tâches et la modélisation des processus en jeu sont depuis de nombreuses années un sujet d'étude et de recherche. Ainsi, plusieurs modèles ont été établis, présentant une grande base commune mais ayant des nuances. Ces différences ont notamment été mises en valeur par les travaux de Perrin-Bruneau [126] : certains se concentrent essentiellement sur la transformation de l'idée en produit [13,124,169], d'autres intègrent l'industrialisation du produit — un produit ne pouvant être commercialisable que si les processus d'industrialisation sont conçus — [6,25,35,66,79,86,89,116,127,135,161], ou enfin considérent également les notions de planning et de stratégie de lancement [13,35,89,116,161].

Parmi les représentations les plus connues et les plus utilisées pour organiser les phases du cycle de conception, on retrouve incontestablement le modèle du **cycle en V** normalisé par l'AFCIQ[1] en 1990 [73] (voir figure 1.1). Ce modèle est le résultat du besoin d'optimisation des premiers cycles classiques dit *séquentiels*, séquentialité qui tire son origine de l'approche tayloriste du travail (division du travail en tâches simples et répétitives). Mais la rigidité et la difficulté de mise en œuvre de ces cycles séquentiels — impossibilité de faire évoluer les spécifications de départ sans relancer l'ensemble du processus — ont conduit à la modélisation du cycle en "V". L'avancée principale d'un tel cycle réside dans les passerelles entre les différentes tâches qui permettent une plus grande souplesse du processus permettant d'itérer plusieurs fois sans relancer l'intégralité du processus. Cependant, les volontés d'optimiser se

1. AFCIQ : Association Française pour le Contrôle Industriel et la Qualité

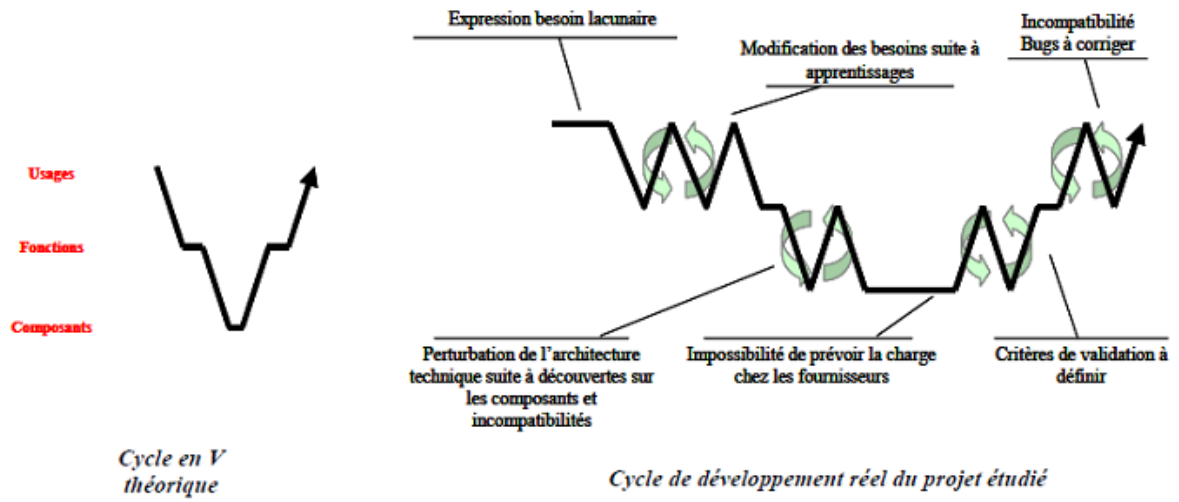


Figure 1.2 – Écart entre le cycle en "V" théorique et le déroulement réel du projet.

concentrent toujours sur la réduction accrue du nombre de retours en arrière. En effet, comme l'ont étudié Midler et Lenfle, dans certains cas d'application du cycle en "V", de trop nombreuses itérations peuvent conduire à un retard important dans l'aboutissement du projet, comme le montre la figure 1.2 [97].

C'est, bien évidemment, le contexte industriel qui dicte la volonté de changement et d'évolution du cycle de conception. Avec une forte pression économique et dans des temps où le client ne se contente plus de l'offre du constructeur, les industries n'ont d'autre choix que de faire évoluer la modélisation de ces cycles de conception. Ainsi pour surmonter certaines limitations inhérentes aux modélisations existantes, sont apparus les schémas de conception concourrante. Telle que définie par Sohlenius [151], cette approche de la conception concourrante, aussi appelée ingéniérie simultanée, consiste en l'intégration et l'exécution des activités des différents acteurs, le plus possible en parallèle, afin de réduire la durée globale du processus. La figure 1.3 montre l'effet de cette approche "simultanée" sur le cycle de conception. Et il n'est donc finalement pas étonnant de retrouver cette nécessité de paralléliser dans les différentes stratégies industrielles du moment.

Pour une analyse plus fine de ces processus de conception — plus de définitions et une description plus complète de leurs évolutions — nous nous reporterons aux travaux de Perrin-Bruneau [126], Madhjoub [103] et surtout, plus récemment, à ceux de Bennes [18].

1.2.1 Étapes du cycle de vie du produit

Bien que les différentes modélisations du cycle de vie du produit fleurissent les unes après les autres, un découpage similaire ressort. Peuvent ainsi globalement être définies 4 phases : *expression du besoin et spécifications*, *conception préliminaire*, *conception détaillée*, et *industrialisation et production*, qui elles aussi peuvent être regroupées de différentes manières. Ce découpage se retrouve ainsi chez plusieurs constructeurs automobiles, comme par exemple BMW (travaux de de Sa et al. [64]) ou PSA Peugeot Citroën avec le Schéma Opérationnel de

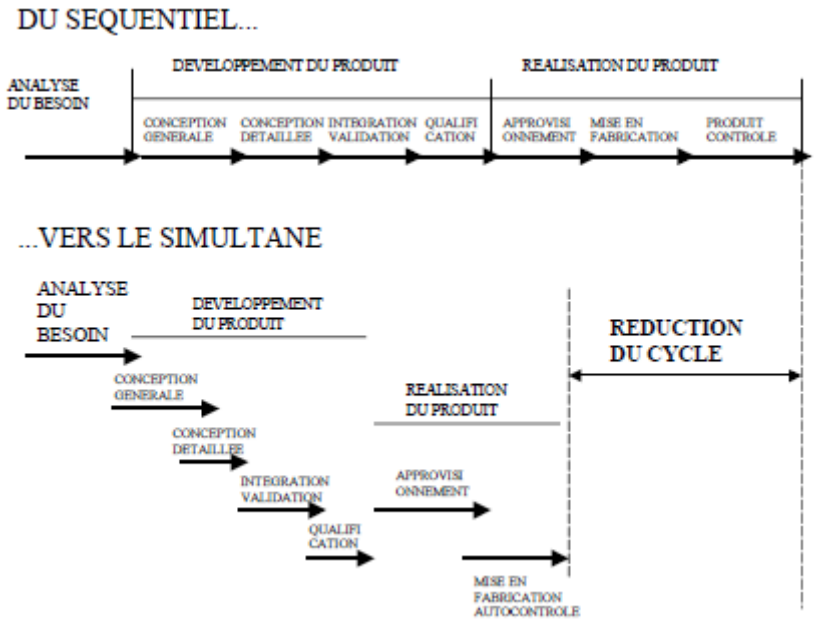


FIGURE 1.3 – Optimisation du cycle de conception : de l'organisation séquentielle à l'organisation simultanée (issue des travaux de Bourdichon [30])

Développement (SOD, voir figure 1.4). Dans ce dernier cas, représentant le contexte industriel de nos travaux, il est important de rappeler ce que comportent globalement les différentes phases.

Le Schéma Opérationnel de Développement est le référentiel décrivant la logique d'ensemble et les principales étapes de la *Conception*, du *Développement* et de l'*Industrialisation*. Ce référentiel recense finalement l'ensemble des "bonnes pratiques" du cycle de vie du produit, principalement à certains moments clés de ce cycle, qui sont mises à jour par les retours d'expérience au fur et à mesure des années. Nous retrouvons bien évidemment dans ce SOD les grandes phases "de base" déterminées plus haut.

1. **Avance de phase générique** aussi appelée **Avant-projet**
   Cette étape concerne l'analyse de nouveaux concepts automobiles et d'innovations à intégrer, et aboutit à l'*expression d'un besoin*. Selon les différentes données initiales, telles que la stratégie du groupe, les données économiques, les contraintes réglementaires et consuméristes, est identifié, exploré et construit le véhicule à développer aux travers de différents concepts. Au final, un *concept* est choisi ce qui permet l'engagement du projet.

2. **Avance de phase programmée** que l'on peut considérer comme la **Mise en projet**
   Cette étape représente la phase de spécifications : on construit et organise le projet en partant de l'*expression d'un besoin* à un *projet faisable*. À partir d'un concept, les différents acteurs de cette phase convergent vers une *silhouette*, satisfaisant notamment les nombreuses contraintes techniques, accompagnée d'un programme cadrant objectifs, ressources, et autres spécifications, et définissant le besoin client.

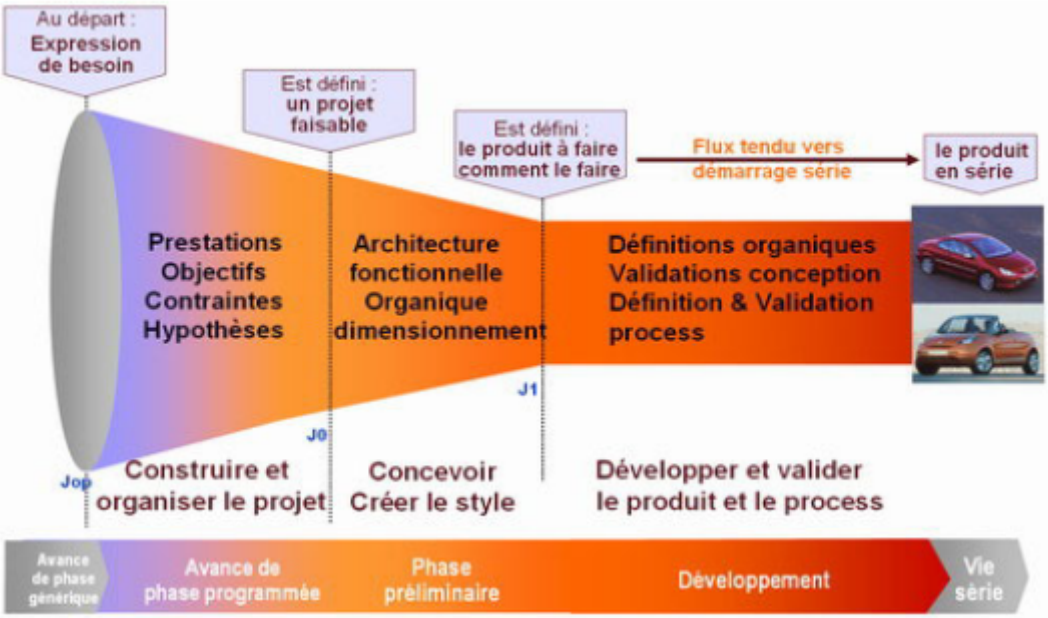


Figure 1.4 – Schéma Opérationnel de Développement (source PSA Peugeot Citroën, 2002)

3. **Phase préliminaire**
   Cette étape correspond à la phase de conception préliminaire et permet de continuer la transformation du concept en produit. Ainsi, le besoin client précédemment *défini* est désormais *spécifié* : un style véhicule est arrêté ainsi que les principaux dimensionnements, les grandes mesures de son architecture physique. La fin de cette étape marque ainsi la fin de la création de style (l'enveloppe de surfaces visibles) : le véhicule est concrètement défini au travers d'un cachier des charges validé, exprimant également *comment* doit être réalisé le véhicule et *avec qui* — les principaux fournisseurs.

4. **Conception détaillée**
   Cette phase consiste en la construction de toutes les définitions détaillées nécessaires — le plan détaillé de chaque pièce — pour l'industrialisation. Ces définitions sont obtenues après un grand nombre d'étapes (études, calculs, simulations, tests, essais) s'assurant de la robustesse de ces données. L'organisation de ces différentes étapes est elle-même spécifique, faisant intervenir différents cycles en "V".

5. **Mise au point Produit**
   Cette étape correspond aux validations et mises au point du produit et de ses processus de production. Un point fondamental de cette phase est de détecter tous les défauts de conception pour les corriger avant le lancement de la production. En parallèle de ces différentes vérifications sont mis en place les outils et le processus de fabrication des pièces du véhicule en question. La détection d'erreurs et leur résolution devant se faire le plus rapidement possible, toute une stratégie spécifique est mise en place durant cette phase, optimisant l'itération des différentes boucles et tâches en jeu. Une fois les objectifs de cette phase atteints, on passe à une *production en série*.

6. **Production et Vie série**
   Cette phase lance la production des véhicules dans le but d'atteindre le nombre préalablement défini pour le lancement du produit. Puis l'objectif principal est de veiller au bon déroulement de l'augmentation de la cadence de production. À la fin de la phase de lancement de la production, incluant le possibilité de répercuter quelques modifications de pièces, le projet véhicule se conclut, un retour d'expérience est fait. Enfin, le dernier processus est lancé, il s'agit de la supervision du produit durant sa fin de vie, sa *vie série / courante*.

1.2.2 Optimisation et enjeux

L'optimisation de l'organisation d'autant de tâches et d'activités représente un énorme challenge du fait de la complexité même du processus de conception et de développement d'un produit comme une automobile, mais aussi et surtout une nécessité dans la course à la compétitivité. On peut distinguer plusieurs niveaux de mise au point, d'amélioration de ce cycle de vie. Un premier niveau "général" consisterait à trouver le meilleur agencement et découpage des grandes étapes décrites ci-dessus, trouver la parallélisation la plus adéquate (voir figure 1.3). Un deuxième niveau, "intermédiaire" optimiserait les différents cycles en "V" qui interviennent dans chacune de ces grandes étapes, tandis qu'un troisième niveau, "particulier", s'intéresserait aux activités les plus singulières au sein de ces cycles et de ces grandes étapes.

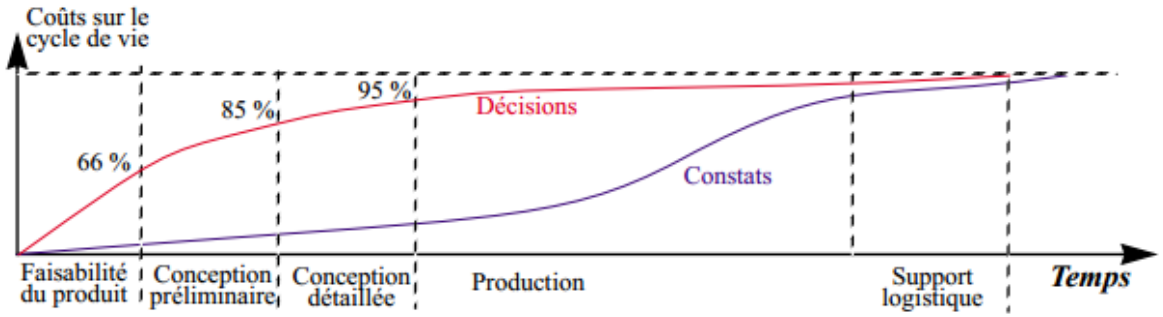


FIGURE 1.5 – Evolution des coûts dépensés (décidés) et des coûts engagés au cours du cycle de vie.

Pour réduire les coûts et les délais de conception, il a été montré qu'il fallait principalement agir sur les différentes étapes de la phase amont du cycle de vie du produit. En effet, comme on peut le voir sur la figure 1.5, les décisions prises lors des premières étapes ont une extrême importance : 80 à 90 % des dépenses totales encourues sont déjà déterminées à la fin du processus de conception, alors que les coûts réellement engagés (la courbe Constats) sont loin d'être à cette hauteur [19]. Cette importante remarque est à mettre en relation avec l'évolution des techniques et technologies, notamment informatiques, en jeu dans ces différentes phases de conception et principalement celles les plus en amont. L'apparition de la CAO, puis de la RV, va en effet amener une meilleure évaluation de la situation ainsi qu'une prise de décision plus rapide et plus sereine durant ces étapes, permettant donc de soutenir la logique de réduction des délais et de gain d'efficacité au plus tôt du cycle de vie du produit.

1.2.3 CAO et cycle de conception

À ses tout débuts, la CAO s'est d'abord intéressée au dessin. C'était dans les années 1960, avec *SketchPad*, le premier outil de dessin sur ordinateur créé par Ivan Sutherland [155]. On parlera d'ailleurs pour ces activités, de Dessin Assisté par Ordinateur (DAO). Les faibles disponibilités et utilisabilités de ces systèmes ne permettaient pas à l'époque d'atteindre une efficacité assez satisfaisante pour voir se propager ces innovations. Les premiers systèmes CAO sont véritablement apparus dans les années 1970 avec des systèmes de *Solid modeling* comme EUCLID conçu au LIMSI-CNRS et commercialisé dans les années 1980 par Datavision qui deviendra Matra Datavision. Il a fallu attendre plusieurs années, et notamment le passage aux années 1990 pour voir les technologies progresser et montrer tout leur potentiel pour une utilisation courante, principalement dans l'industrie manufacturière. C'est ainsi qu'on est passé de l'aide au dessin 2d à l'aide à la modélisation 3d, et ce par la commercialisation de nombreuses solutions comme Alias [9], 3ds Max [8], SolidWorks [59], AutoCAD [10], ProEngineer [125] ou CATIA [58].

La CAO est désormais partie intégrante et indispensable du processus de conception industriel. Elle couvre une grande partie des activités du cycle de vie du produit, dont les phases de design des objets, de conception de pièces (architecture des produits) et, de simulation et d'analyse numériques. L'industrie a en particulier rapidement adopté les logiciels proposant une modélisation paramétrique (voir section 2.2, chapitre 2). En 2011, Parametric Technology Corporation (PTC [2]), le développeur de logiciel CAO, établissait, à partir d'une étude réalisée auprès de 7000 personnes liées aux métiers de la conception et venant de 51 pays [143], qu'*un peu plus de 10 % utilisaient principalement des outils de conception 2d, [que] près de la moitié (47,3 %) utilisaient un mélange d'outils CAO 2d et 3d, et [que] 41,2 % utilisaient principalement la CAO 3d.*

Depuis son effervescence dans les années 1990 et son introduction au cœur des cycles de conception industriels, la CAO n'a pas connu de réel bouleversement majeur, mais n'a cessé d'évoluer petit à petit, fonctionnalité après fonctionnalité. Cependant, un des problèmes critiques majeurs restant encore aujourd'hui à surmonter est l'utilisabilité des logiciels dans une démarche de conception où les intervenants sont de plus en plus nombreux à agir simultanément soit pour donner leur avis, soit pour apporter des modifications. Dans ses travaux, Picon [128] a modélisé les interactions entre les principaux acteurs que l'on retrouve lors des phases amont de conception (voir figure 1.6). Les processus en jeu sont des processus itératifs, dans lesquels des changements ou des modifications impliquent des bouclages, jusqu'à ce qu'une décision soit arrêtée. Une optimisation possible de cette organisation est bien sûr l'approche de l'utilisation d'un système CAO commun aux différents métiers afin de faciliter les échanges, de permettre une conception collaborative, et d'assurer la réutilisabilité des données. L'interopérabilité entre les différents acteurs et les systèmes CAO, est un enjeu majeur pouvant se réveler un véritable obstacle dans la course à la réduction des délais lorsqu'il n'est pas maîtrisé. Il faut notamment remarquer que le plus gros désavantage de la conversion de données lors des échanges entre acteurs est la perte de l'intention de conception. En effet, ces conversions enlèvent quasiment systématiquement toute notion d'historique pour ne garder que les géométries finales. Malgré ces difficultés avérées et potentielles, le processus

2. www.ptc.com

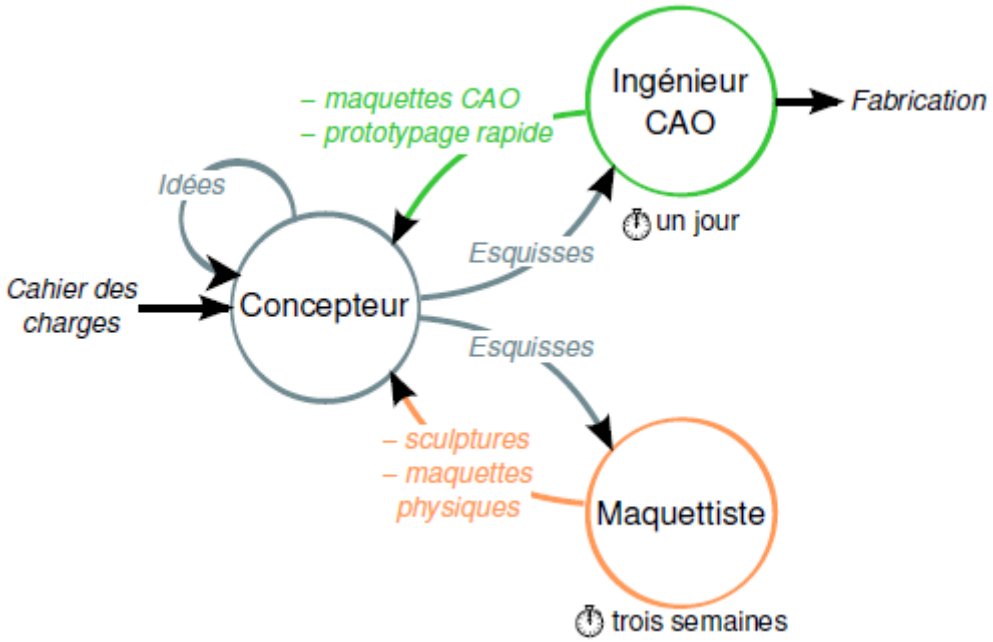


FIGURE 1.6 – Représentation des trois acteurs que l'on peut retrouver dans un cycle de conception, issue des travaux de Picon [128]).

"numérique" offre de nombreux avantages, notamment dans l'optimisation de ces boucles de conception. En particulier, il a été établi que le cycle de mise à jour numérique était plus rapide que la création d'une nouvelle maquette physique [94].

1.2.4 Maquette numérique et prototypage virtuel

Avant d'entamer la phase de conception détaillée d'un produit, il est nécessaire de le définir et de le dimensionner. Cela est fait sur la base de modèles plus ou moins précis, à ce stade, pouvant encore évoluer. Les prototypes réalisés durant cette phase permettent ainsi de répondre à bon nombre de questions durant le cycle de conception et donc de réduire le nombre d'erreurs — erreurs nécessitant parfois des retours en arrière comme nous l'avons vu précemment. Le prototypage virtuel (PV) va permettre de présenter, tester, et analyser des objets CAO avant d'en faire un prototype physique. Cette démarche a rapidement pris de l'importance dans les industries automobiles et aérospatiales afin de réduire les délais de conception et le nombre de prototypes physiques — et donc les sommes d'argent engagées. De Sa et al. [63] proposent plusieurs définitions pour le PV — des points de vue *infographie* et *ingéniérie mécanique* — et pour la maquette numérique :

Définition "infographie"

> *Le prototypage virtuel est l'utilisation de la RV pour prototyper des maquettes physiques* (physical mock-ups - PMUs)*. Le système de RV permet alors de simuler et de rendre toutes les caractéristiques voulues, de la façon la plus précise et la plus réaliste possible dans un environnement immersif.*

Définition "ingéniérie mécanique"

> *Le prototypage virtuel est l'utilisation de prototypes logiciels en lieu et place de prototypes physiques. Cela inclut toutes les simulations géométriques et fonctionnelles, impliquant ou non l'humain.*

Définition de la "maquette numérique"

> *La maquette numérique (*digital mock-up - DMU*) est une simulation informatique réaliste d'un produit, comportant toutes les fonctionnalités requises, du design à la maintenance [55].*

Ces définitions rentrent ainsi dans le schéma considérant le PV comme un sous-ensemble de la maquette virtuelle. Pour nuancer ce propos, Wang [163] propose une autre définition, légèrement différente et apportant quelques précisions :

> *Un prototype virtuel, ou une maquette numérique, est une simulation informatique d'un produit physique qui peut être visualisée, analysée, et testée selon différents aspects du cycle de vie du produit, comme la conception et l'ingénierie, la fabrication, le service, et le recyclage, comme s'il s'agissait d'un modèle physique. La construction et les tests d'un prototype virtuel forment le prototypage virtuel (PV).*

Par cette définition, Wang entend spécifier que le prototype virtuel est une maquette numérique, décrire les fonctions d'un tel prototype virtuel et s'abstraire de toute technologie particulière pour définir la démarche de prototypage virtuel.

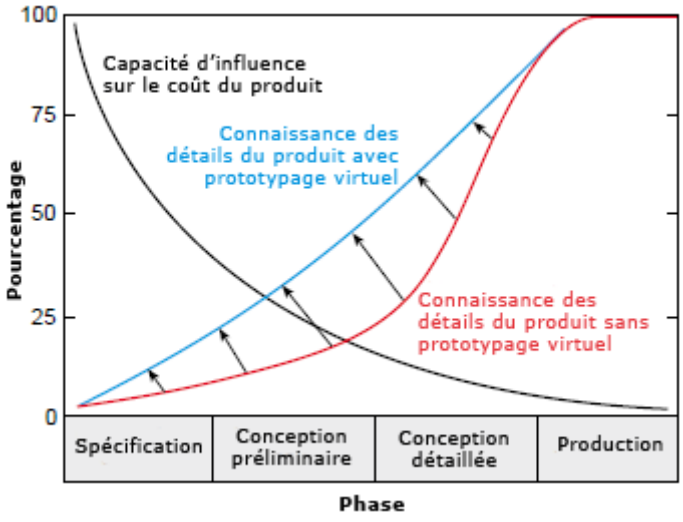


Figure 1.7 – Importance du prototypage virtuel pour la connaissance du produit et la maîtrise des coûts (issue des travaux de Lombardo [101]).

C'est ainsi que dans les industries, les maquettes numériques supplantent de plus en plus les maquettes physiques pour les prises de décision à différentes étapes du processus de conception, et principalement dans les phases amont qui comme nous l'avons vu précemment, sont

les plus sensibles. Comme le rappelle Sadoul [142], c'est le cas de PSA Peugeot Citroën qui utilisa pour la première fois une maquette numérique en 1993, durant la phase de conception préliminaire d'un projet véhicule. Puis petit à petit cette maquette numérique fut également utilisée dans la phase de développement pour devenir concrètement *un support d'étude pour l'ensemble du produit*. Le succès de l'intégration de cette maquette numérique doit notamment aux fonctions qui lui sont attachées, permettant d'optimiser le processus de conception. Ainsi comme le décrit Sadoul :

> *La maquette numérique est le support informatique d'initialisation de la conception globale, de conception simultanée du produit et du process, de canalisation, de redistribution, de mise à jour permanente et de validation des travaux de développement, de préparation de la prise de décision, de généralisation des développements simultanés à l'entreprise étendue, de "capitalisation" des acquis, bref,* ***un outil rendant possible la communication complexe requise par les développements simultanés du produit et du process****.*

On voit bien dans ces définitions combien le prototypage virtuel occupe une place centrale dans le processus de conception, permettant une adaptation aux nouvelles technologies, allant de la CAO à la RV. Cette option alternative aux prototypes physique a d'autant pris de l'importance que les avantages tirés du prototypage virtuel sont intéressants et répondent à la simultanéité accrue des différentes activités de conception. La nature "numérique" couplée aux capacités informatiques puissantes ont très nettement permis d'améliorer la communication, la productivité et l'efficacité au travers de présentations graphiques réalistes, de largement réduire les temps de dessin, et d'observer dynamiquement les modèles [173]. La figure 1.7 illustre ainsi les gains qu'apportent le prototypage virtuel dans la connaissance du produit au cours du cycle de conception, et notamment une meilleure connaissance accessible plus rapidement.

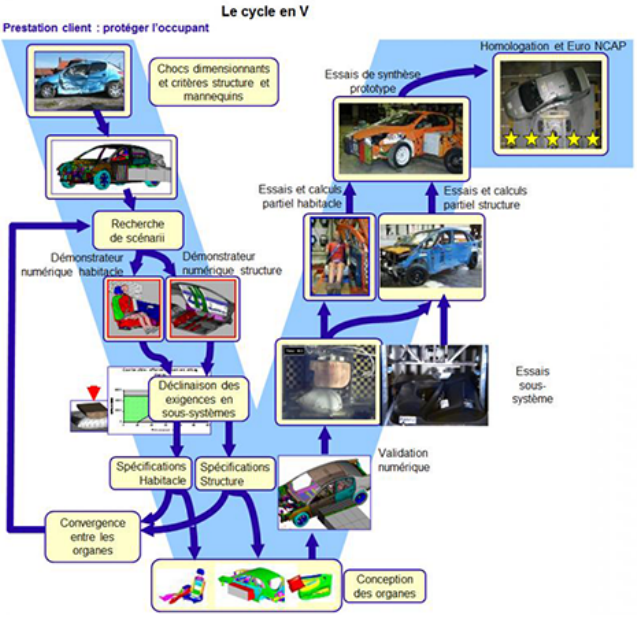


FIGURE 1.8 – Le cycle en V de la simulation dans la conception d'une voiture chez PSA.

À titre d'exemple nous citons le cas de la conception de la Peugeot 208. Dans une interview donnée au site 01Business [45], Richard Zeitouni, maître expert en sécurité passive chez PSA Peugeot Citroën, raconte comment l'investissement réalisé dans la simulation numérique — puissance de calcul multipliée par huit en cinq ans — a permis de baisser les coûts de production. Ainsi, en intégrant au cycle de conception la simulation sur des maquettes numériques (voir figure 1.8), le nombre de tests physiques et de longues et coûteuses constructions de prototypes physiques ont été réduits, ce qui a permis de gagner de 30 à 50 % du temps consacré au prototypage.

1.2.5 Conclusion

Dans un contexte concurrentiel acharné, l'industrie, et en particulier l'industrie automobile, cherche à optimiser le cycle de conception de produit. De plus en plus d'activités se font désormais en parallèle, faisant intervenir à un même instant du processus de nombreux acteurs le plus souvent aux compétences différentes. Le développement de technologies comme la CAO et l'utilisation grandissante d'outils de simulation leur permet d'optimiser ce processus de conception en accélérant et en renforçant la prise de décision ainsi qu'en réduisant la durée et le coût de certaines étapes comme la construction et les tests de maquettes physiques. Cependant, les points critiques de l'utilisation de la CAO restent l'interopérabilité et la modification des données, tous deux potentiels facteurs d'augmentation des délais de conception. Les échanges de fichiers sont nombreux, que ce soit au sein des différentes équipes travaillant sur le projet, ou que ce soit avec les fournisseurs ou clients, pouvant instaurer de nombreux délais. Les échanges qui nous intéressent plus particulièrement dans ces travaux sont ceux relatifs aux données "internes", c'est à dire les échanges qui s'opèrent entre les différents métiers / acteurs d'une même entreprise. C'est aussi dans ce cadre interne que nous restreindrons le problème des modifications de données, et en particulier à celles des maquettes numériques.

1.3 La Réalité Virtuelle dans le PLM : un état de l'art

Les dernières décennies ont vu de nombreuses avancées technologiques, notamment en informatique : puissance de calcul toujours plus grande, de nouvelles interfaces utilisateurs, de nouveaux paradigmes, etc. Ces avancées ont alors permis l'explosion de certains domaines comme l'informatique graphique, la synthèse d'image et l'interface homme-machine (IHM), aboutissant à une interactivité toujours plus grande entre le contenu numérique et l'homme. Le domaine de la Réalité Virtuelle (RV) se trouve de fait au confluent des domaines des sciences et techniques en ce qui concerne les moyens utilisés, et des sciences humaines en ce qui concerne les utilisateurs et l'usage de ces moyens.

Dans cette section nous proposons un condensé des différentes définitions et acceptions sur la RV, et présentons ses applications actuelles dans le PLM des industries.

1.3.1 Introduction à la Réalité Virtuelle

L'expression "réalité virtuelle" est une traduction, discutable, d'une expression anglaise, *virtual reality*, introduite dans les années 1980 par Jaron Lanier. La traduction française, littérale et largement utilisée maintenant, perd un peu le sens originel de l'expression anglaise, incluant notamment l'idée d'un processus amenant à percevoir comme "réaliste" un environnement virtuel donné. Cet écart de sens n'est pas négligeable puisqu'on peut le considérer comme un facteur à l'origine de confusions lorsqu'il s'agit de définir ce qu'est la RV. En

particulier les définitions que l'on trouve peuvent parfois mélanger moyens (techniques) et finalité (objectifs).

1.3.1.1 Définitions

Il n'existe pas une, et une seule, définition, mais plusieurs sont désormais acceptées par la communauté, notamment la communauté française dont nous allons extraire certaines de ces définitions. Ainsi Fuchs [74] propose une définition **fonctionnelle** de la RV :

> *La réalité virtuelle va permettre à l'homme de s'extraire de la réalité physique pour changer virtuellement de temps, de lieu et/ou de type d'interaction : interaction avec un environnement simulant la réalité ou interaction avec un monde imaginaire ou symbolique.*

D'un autre côté, Arnaldi et al. [7] proposent une définition **technique** de la RV :

> *La réalité virtuelle est un domaine scientifique et technique exploitant l'informatique (1) et des interfaces comportementales (2) en vue de simuler dans un monde virtuel (3) le comportement d'entités 3D, qui sont en interaction en temps réel (4) entre elles et avec un ou des utilisateurs en immersion pseudo-naturelle (5) par l'intermédiaire de canaux sensori-moteurs.*

Ces définitions sont importantes puisqu'elles permettent donc de préciser ce qu'est véritablement le domaine de la RV et de poser sa **finalité** [74] :

> *La finalité de la RV est de permettre à une personne (ou à plusieurs) une activité sensori-motrice et cognitive dans un monde artificiel, créé numériquement, qui peut être imaginaire, symbolique ou une simulation de certains aspects du monde réel.*

Enfin terminons par la définition de la RV telle que l'entend le groupe VENISE du LIMSI-CNRS [31] :

> *La **Réalité Virtuelle** vise à mettre au point des systèmes informatiques qui donnent à l'humain la capacité de **percevoir** et d'**interagir** de façon **multi-sensori-motrice** avec des **données numériques** ou **mondes virtuels**.*
> *Quand en plus, ces données numériques intègrent une **virtualisation d'une partie de l'univers réel** et permettent ainsi de gérer des **interactions** entre des **objets totalement virtuels** et des **objets réels**, on parle alors de **Réalité Augmentée**, voire de **Réalité Mixte**.*

Ainsi il faut mettre de côté l'idée que la RV se veut imiter le plus fidèlement, d'une façon la plus réaliste possible, le monde réel. Il y a bien davantage de buts et de manières d'approcher ce domaine. En fait, ce qui caractérise principalement une activité de RV est la conciliation entre les aspects d'immersion et d'interaction. C'est la combinaison de ces deux aspects, dans des proportions jamais nulles et à fixer selon la finalité de la simulation, qui déterminera la "qualité" du système proposé. Cependant, en ce qui concerne le contexte de mes travaux, à savoir l'utilisation de la RV pour des activités industrielles, il s'agira souvent de reproduire

un processus ou un produit, le plus fidèlement et de façon aussi réaliste que possible. Ainsi par exemple, le réalisme est requis tant pour donner à percevoir les matériaux pour les activités liées au style (qualité perçue, etc.), que pour disposer d'une simulation physique exacte (assemblage, mécanique, etc.).

1.3.1.2 Immersion et présence

La RV a donc pour finalité de permettre aux hommes d'interagir dans des environnement numériques. Parmi les notions importantes liées à la RV et aux Environnements Virtuels (EVs), il y a la **présence** et l'**immersion**. La notion d'immersion est souvent associée, à tord, à la qualité de visualisation et au degré de réalisme du monde virtuel. Or comme le montre Slater et Wilbur [150] elle ne dépend pas uniquement de ces deux facteurs. L'immersion doit permettre de ressentir le fait de quitter le monde réel et d'être "présent" dans le monde virtuel. Plus exactement, l'immersion se caractérise d'un point de vue *technique* [117] :

> *L'immersion est réalisée en supprimant le plus possible de sensations liées au monde réel et en les replaçant par des sensations correspondant à l'environnement virtuel.*

L'immersion dépend ainsi de nombreux facteurs. Les systèmes de visualisation en sont une part importante mais ils ne doivent pas occulter l'importance de la qualité de l'intégration des différentes techniques d'interaction et de rendu. Cette intégration est fondamentale pour que les actions et les perceptions de l'utilisateur au sein de l'environnement immersif soient les plus cohérentes possible. Nous entendons donc ici par, environnements immersifs, l'ensemble des technologies utilisées pour permettre l'immersion de l'utilisateur au sein d'un monde virtuel.

Pour conclure ce très bref aperçu des notions fondamentales de la RV — qui ne sont pas l'objet de mes travaux — il nous faut donner la définition de la présence. Au contraire de l'immersion, que l'on peut donc définir comme une mesure objective avec des critères définis, la présence est une notion subjective. Ainsi selon Slater [150] :

> *La présence est un état de conscience, le sentiment (psychologique) d'être dans l'environnement virtuel. L'idée fondamentale est que les participants qui se sentent fortement présents devraient éprouver l'environnement virtuel comme une réalité plus engageante que le monde physique environnant, et considérer l'environnement spécifié par les affichages comme des places visitées et non des images vues. Les comportements dans les environnements virtuels devraient être cohérents avec les comportements qui arriveraient dans la vie de tous les jours pour des situations similaires.*

La présence est une problématique largement étudiée depuis plusieurs années en RV. Pour un approfondissement de cette notion, nous nous reporterons aux travaux de Biocca [24], Lombard et Ditton [100], Slater [149, 150], et Burkhardt [37].

1.3.2 Dispositifs de RV

Dans le monde réel, l'homme évolue selon un modèle bien connu, à savoir celui de la boucle perception-action. C'est d'abord au travers de la perception de l'environnement, par l'intermédiaire des différents canaux sensorimoteurs, que l'homme récupère et analyse les informations qui vont lui servir à décider et finalement agir. Un des points clés de la RV

consiste alors à permettre à l'homme de percevoir, le mieux possible, l'environnement virtuel dans lequel il évolue, pour qu'il puisse y agir de la manière la mieux adaptée.

Dans cette sous-section nous présentons donc les différentes **interfaces comportementales** — *dispositifs qui exploitent la motricité ou les perceptions de l'homme issues de son comportement dans le monde réel* [75] — à disposition. Ces interfaces sont théoriquement classées selon trois groupes distincts : les *interfaces sensorielles* qui doivent permettre à l'homme de percevoir l'environnement virtuel, les *interfaces motrices* qui doivent permettre à l'homme d'agir dans l'environnement virtuel (et que ce dernier reçoive les informations émanant de l'homme), et les *interfaces sensori-motrices* qui permettent la communication entre l'homme et l'environnement dans les deux sens.

Nous nous intéresserons principalement ici aux dispositifs pouvant permettre une activité de conception en environnement immersif et le lecteur désirant avoir une description complète de ces interfaces comportementales pourra consulter notamment le Volume 2 du Traité de la Réalité Virtuelle [75].

1.3.2.1 Systèmes de visualisation / interfaces sensorielles

La vue est sans doute le sens principal utilisé pour percevoir le monde qui nous entoure. L'homme voit en trois dimensions et il faudrait donc idéalement lui permettre de percevoir l'environnement virtuel également en trois dimensions. Nous allons donc présenter ici les principales interfaces sensorielles, que sont les dispositifs de visualisation, utilisées pour immerger visuellement l'utilisateur. Nous ne parlerons pas ici des interfaces olfactives, très peu utilisées globalement, ni des interfaces tactiles. Notons simplement que ces interfaces tactiles, liées au toucher, peuvent être un bon moyen pour donner des informations de contact avec des objets, ou bien pour transmettre des informations permettant de mieux apprécier un objet.

Principe de la stéréoscopie

Le principe de la stéréoscopie est simple et consiste à fournir à chaque oeil une image avec un point de vue légèrement différent, ces deux images étant ensuite assimilées par le cerveau pour percevoir en relief. Plusieurs solutions existent pour fabriquer de telles images et les restituer à chaque œil. Ici, nous prendrons le cas des images calculées par ordinateur et restituées par des écrans ou des projecteurs. Un des points clés pour restituer les images aux yeux réside dans la séparation de celles-ci : on parle de séparation des images active (lunettes à occultation) ou passive (lunettes polarisées). Existe aussi la séparation colorimétrique, bien connue notamment avec ses lunettes anaglyphes cyan et rouge, mais elle n'assure pas de rendu de qualité suffisante pour être utilisée dans l'industrie ou dans le monde académique.

Le principe de la séparation passive consiste à afficher les deux images en même temps, et à les séparer par l'intérmédiaire d'un filtre polarisant. Ce filtre est placé à la fois en sortie d'écran (ou de projecteur), et sur les lunettes (une polarisation différente par œil). La séparation active consiste elle à afficher successivement les images, chaque œil ne devant voir qu'une image sur deux. Pour restituer la bonne image on va utiliser des lunettes dites actives au principe simple. Elles sont constituées d'obturateurs à cristaux liquides, un pour chaque œil, qui vont ainsi être activés successivement et de façon synchronisée avec l'écran ou le projecteur. Ainsi pour une image donnée, seul un œil voit au travers des lunettes. Les fréquences d'affichage de l'image et d'obturation des lunettes étant relativement élevées, l'utilisateur ne s'en rend absolument pas compte (certains auteurs soulignent cependant une fatigue du cerveau au delà de vingt minutes d'utilisation).

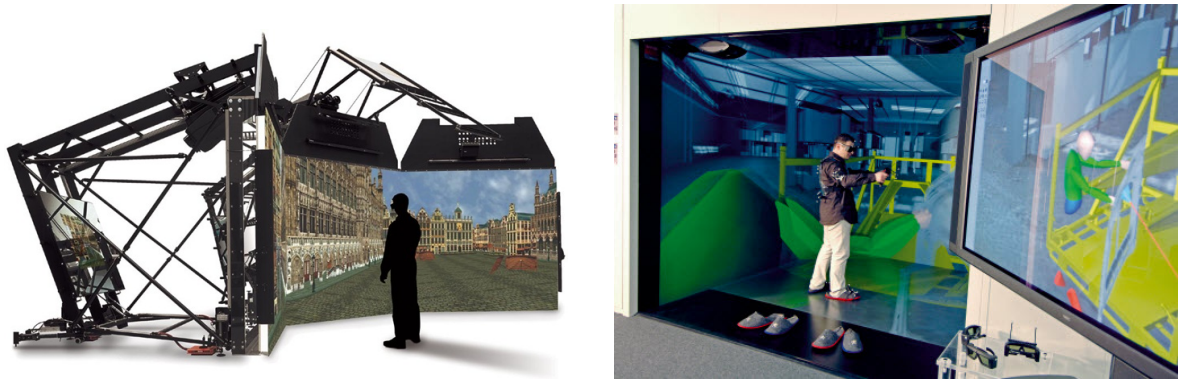


Figure 1.9 – Systèmes de visualisation de type CAVE. A gauche le MoVE (Multi-purpose virtual environments) proposé par Barco, et à droite le CAVE de PSA Peugeot Citroën.

Dispositifs à support fixe

Commençons bien évidemment par les traditionnels écrans d'ordinateurs monoscopiques. Ces matériels sont largements répandus, et sont un des outils importants pour les différents designeurs et ingénieurs travaillant sur un poste de travail. Notons simplement que ces dispositifs ne présentent pas à l'heure actuelle de résolution assez importante pour se rapprocher de l'acuité de la vision humaine. Cependant ils sont en perpétuelle évolution, permettant différents agencements par la multiplication des écrans, et proposant des tailles et résolutions toujours plus importantes. Certains proposent désormais un rendu stéréoscopique, permettant ainsi à l'utilisateur de visualiser le contenu informatique en trois dimensions.

Le champ de vision de l'utilisateur sur les écrans d'ordinateur étant relativement restreint, des techniques proposent de projeter des images sur un grand écran grâce à des projecteurs. Ce type de dispositif est particulièrement apprécié, notamment dans l'industrie, car il permet simplement de pouvoir visualiser un produit à l'échelle 1:1, comme par exemple une voiture. Ces écrans peuvent facilement atteindre des largeurs de 4 à 5 mètres, et utilisent généralement une stéréoscopie passive, dont les lunettes sont moins coûteuses et plus agréables à porter, point important lorsqu'il s'agit de présentation à de nombreuses personnes. Ces écrans peuvent être multipliés pour former des murs d'écrans afin d'augmenter la taille de l'image et de conserver une résolution acceptable.

Mais les grands dispositifs les plus immersifs sont certainement ceux de type CAVE (Cave Automatic Virtual Environement [51]), comme on peut le voir sur la figure 1.9. Ces dispositifs sont les plus coûteux et les plus lourds à mettre en place, mais permettent une très bonne immersion. La configuration de ces dispositifs varie selon les usages, mais va principalement aller d'une configuration à trois écrans — sol, face et un côté — à cinq écrans — sol, gauche, face, droite et plafond — avec une projection stéréoscopique active. Rares sont les dispositifs en cube complet. Une des principales caractéristiques de ces dispositifs est ainsi de proposer à un utilisateur, une immersion visuelle quasi totale dans l'environnement virtuel. Remarquons que les évolutions technologiques constantes des matériels de projection font qu'il devient possible d'immerger visuellement plusieurs utilisateurs avec leur propre point de vue, comme c'est le cas avec EVE[3] (Evolutive Virtuel Environments) le dispositif immersif du groupe VENISE du LIMSI-CNRS (voir figure 1.14).

3. http://www.limsi.fr/venise/EVEsystem

Dispositifs à support mobile

L'autre grande catégorie d'interfaces visuelles sont les dispositifs portables que sont les visiocasques, ou en anglais Head-Mounted Displays (HMD) (voir figure 1.10). Bien qu'ils ne connaissent pas un grand succès dans l'industrie aujourd'hui, ces dispositifs sont depuis longtemps utilisés par le milieu académique, et de plus en plus par l'univers du jeu vidéo (notons le succès récent de l'Oculus Rift tout juste racheté par Facebook [4]).

Ces dispositifs vont notamment permettre à l'utilisateur de disposer d'une vision stéréoscopique et d'un champ visuel quasi identique à celui de ses yeux. Pour la stéréoscopie, ces dispositifs sont généralement composés de deux petits écrans, proposant ainsi directement une image à chaque œil. Des capteurs sont habituellement intégrés à ces dispositifs pour que les images affichées correspondent à l'orientation de la tête de l'utilisateur.

Parmi les points négatifs de ces dispositifs qui font qu'ils ne sont quasiment pas utilisés dans l'industrie, il faut noter qu'ils coupent visuellement totalement du monde extérieur, bon point en ce qui concerne l'immersion, mais mauvais point lorsqu'il s'agit de communiquer avec d'autres personnes comme c'est souvent le cas dans les séances de travail. Aussi, la qualité de l'image proposée par ces dispositifs n'est pas encore totalement satisfaisante. Mais les derniers modèles vont vers de très haute résolutions (proches ou supérieures à la Full HD - 1920×1080 pixels), ce qui pourrait inverser la tendance. Notons cependant que la proximité avec l'écran ne permet pas encore d'exploiter au mieux l'acuité visuelle humaine. De même et compte tenu de la situation économique, leur coût, bien moindre comparé aux systèmes de projection, pourrait aussi être un facteur menant vers une utilisation accrue dans les années à venir.

FIGURE 1.10 – Systèmes de visualisation de type HMD. A gauche l'Oculus Rift (Version kit de développement 2) et à droite, le Neo^{PRO} de Light & Shadows [99].

1.3.2.2 Systèmes de capture / interfaces motrices

Les interfaces motrices vont permettre de positionner l'utilisateur et les objets qu'il manipule dans l'environnement virtuel. En effet, le déplacement est une action primordiale qui doit être permise à l'utilisateur. Ainsi, plusieurs captures de position vont être disponibles afin de repérer dans l'espace notamment le corps ou différentes parties du corps de l'utilisateur.

4. http://siliconvalley.blog.lemonde.fr/2014/03/25/facebook-rachete-les-casques-de-realite-virtuelle-oculus-rift/

Un des principaux matériels utilisés va être le dispositif de *tracking* optique, composé de caméras infra-rouge (voir figure 1.11) et de marqueurs réfléchissants (voir figure 1.12). Le principe est simple : plusieurs caméras infra-rouge envoient des flashs qui vont être réfléchis par les marqueurs cibles, permettant ainsi de localiser précisément via une triangulation ces marqueurs.

Figure 1.11 – Caméras infra-rouge ART pour la capture de mouvement. A droite un modèle spécial de caméra pour rentrer dans les coins d'un CAVE.

La capture des mouvements de la tête va être particulièrement importante dans les activités de RV, et notamment dans les activités industrielles. En effet, connaître la position et l'orientation de la tête, grâce notamment à des marqueurs placés sur les lunettes stéréoscopiques (figure 1.12 gauche), va permettre d'adapter le rendu graphique (stéréoscopique) pour qu'il corresponde au point de vue exact de l'utilisateur. Ceci donnera ainsi à l'utilisateur la sensation de réellement évoluer au sein de l'environnement virtuel.

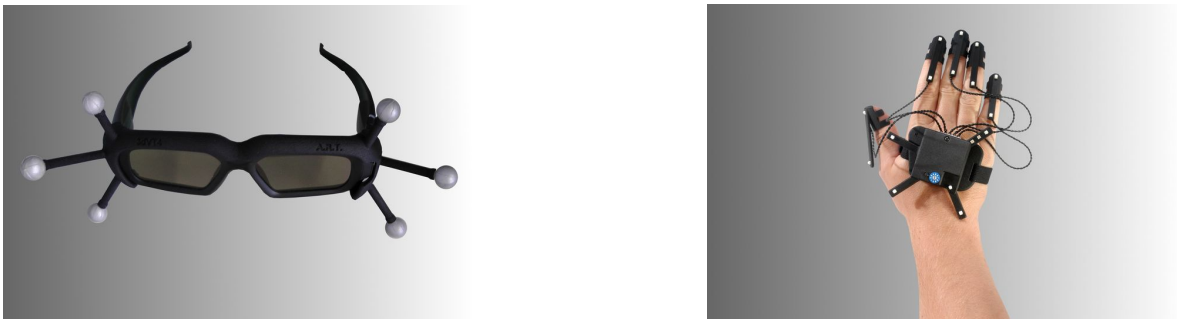


Figure 1.12 – Systèmes de capture de position et de mouvement. À gauche des lunettes pour la vision stéréoscopique avec des marqueurs passifs pour capter la position de la tête. À droite le Fingertracking de ART version 5 doigts, permettant la capture des mouvements de la main et des 5 doigts.

Ensuite, la capture des gestes, et plus précisément des gestes de la main, va permettre des intéractions très précises. Parmi les matériels à la pointe, il faut noter le Fingertracking

de ART (figure 1.12 droite) qui donne la possibilité de capter les mouvement de la main et des doigts. Ceci sera pratique lorsqu'il s'agira de simuler des manipulations d'objets comme dans le monde réel. D'autres manipulations d'objets sont aussi couramment réalisées par l'intermédiaire d'outils, dont la forme peut varier selon la nature de l'activité en RV. Ces outils vont de la simple souris 3d appelée *flystick* à des objets plus adaptés comme un outillage industriel qui seront également "trackés" pour une manipulation la plus fidèle possible.

La capture de mouvement du corps complet va aussi être utilisée dans l'industrie, notamment pour des études ergonomiques (voir section 1.3.3.3). Pour ce faire, le même dispositif de *tracking* optique sera utilisé, et l'utilisateur portera une combinaison permettant ainsi de récupérer la position et l'orientation de différentes parties du corps. Ceci permettra en particulier d'animer un personnage virtuel complet.

1.3.2.3 Systèmes haptique et interfaces sensori-motrices

Certaines interfaces vont permettre à la fois la communication de l'homme vers l'environnement virtuel, et de l'environnement virtuel vers l'homme. C'est en effet le cas des interfaces haptiques, comme notamment les bras à retour d'effort (voir figure 1.13). Ces matériels vont principalement être utilisés pour manipuler des objets et ressentir des collisions, ou d'autres informations liées à ces objets, comme dans les activités d'assemblage (voir section 1.3.3.4). Notons que ces dispositifs sont très utilisés dans le milieu médical, permettant notamment les opérations robotisées et/ou à distance.

En terme d'aide à l'édition CAO, l'interaction haptique a fait l'objet de plusieurs travaux, comme par exemple ceux de Picon [128–131].

FIGURE 1.13 – Systèmes d'interaction à retour d'effort. Ici le Virtuose 6D de Haption, permettant des interactions et un retour d'effort à 6 degrés de liberté.

1.3.2.4 Interactions et commandes naturelles

Pour améliorer l'expérience de l'utilisateur, de nombreux travaux tendent à reproduire les moyens de communication de l'homme dans le monde réel, au sein des environnements virtuels. L'idée principale est en fait d'arriver à ce que les utilisateurs agissent aussi naturellement dans le monde virtuel que dans le monde réel.

Nous venons juste de parler des interactions gestuelles, mais il faut également parler des moyens permettant aux utilisateurs de communiquer avec l'environnement informatique par la voix. La reconnaissance vocale est en effet souvent utilisée pour réaliser des commandes,

ce qui permet d'éviter l'utilisation de menus, point important lorsque nous évoluons dans un monde virtuel.

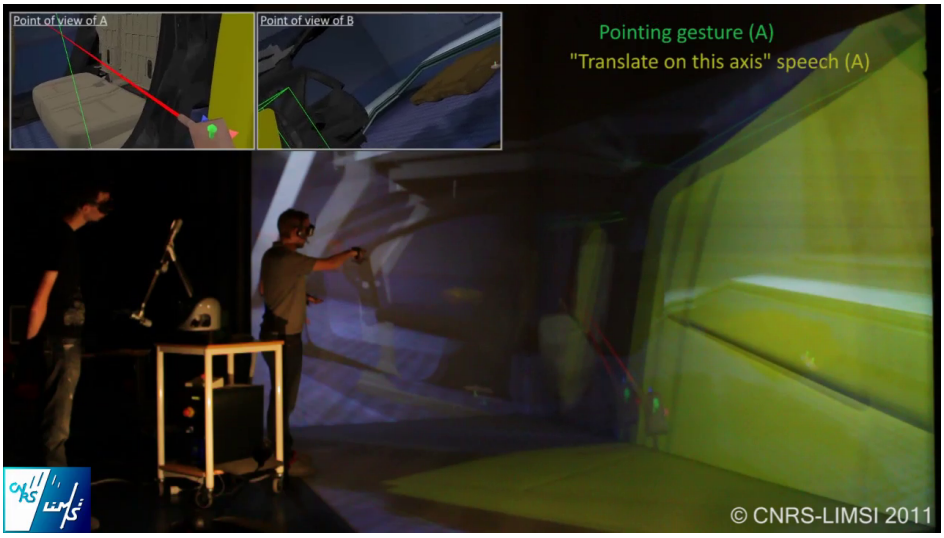


Figure 1.14 – Exemple d'interaction multimodale. Ici, dans une tâche d'introduction de siège arrière de voiture, un utilisateur designe un axe de manipulation, grâce à la combinaison de sa voix et de son geste. Dans le cadre de ce démonstrateur cet axe va servir à contraindre haptiquement la manipulation du siège pour le second utilisateur ayant lui aussi son propre point de vue grâce à une multi-stéréoscopie (voir section 1.3.2.1).

La combinaison de ces différents moyens d'interaction constitue en elle-même un champ de recherche [112,115]. On parle ainsi de **multimodalité**, consistant à proposer aux utilisateurs des interactions au travers de leurs différents canaux sensori-moteurs (voir figure 1.14). Nous mettons en évidence principalement la combinaison haptique-voix-gestes, combinaison de différents moyens déjà utilisés dans l'industrie. La combinaison voix-geste a été le sujet de travaux de recherches communs entre le groupe VENISE du LIMSI-CNRS, et PSA Peugeot Citroën, au cours du projet ANR [5] Perf-RV2 [6].

1.3.3 Applications de la RV dans l'industrie

Après avoir longuement présenté le contexte industriel de nos travaux et détaillé les processus de conception de produits (section 1.2), nous avons rappelé les définitions et concepts principaux de la RV (section 1.3.1) puis donné un aperçu des dispositifs technologiques les plus utilisés (section 1.3.2). Nous allons maintenant exposer dans cette sous-section les principales utilisations de la RV dans l'industrie dont on peut trouver un petit récapitulatif de ces activités dans les travaux de Mujber et al. [122].

1.3.3.1 Analyse et simulations

Les simulations informatiques de mécanique des fluides sont depuis de nombreuses années une étape importante de la conception d'un produit. C'est notamment le cas des industries

5. ANR : Agence Nationale de la Recherche
6. Plateforme française de Réalité Virtuelle : intégrer l'humain virtuel dans l'usine numérique pour améliorer dès la conception, l'efficacité et l'ergonomie des postes de travail. http://www.perfrv2.fr/

automobile, aéronautique, ou du bâtiment. Les techniques de RV ont rapidement été considérées comme un outil très intéressant pour visualiser le résultat de ces simulations tout en étant immergé avec le produit. Ainsi, les études thermiques d'un bâtiment par rapport au vent vont pouvoir être visualisées tout en étant complètement immergé au sein de celui-ci ou de son environnement (voir figure 1.15).

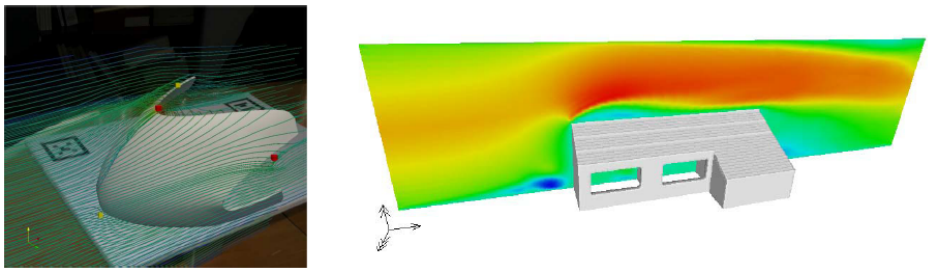


FIGURE 1.15 – Exemple de simulation de mécanique des fluides. À gauche, une simulation du flux de l'air sur un pare-brise, avant et après modification de ce dernier (Bordegoni et al. [27]). À droite, une simulation du flux de l'air selon la géométrie du bâtiment (Kühner et al. [93]).

L'industrie automobile n'est évidemment pas en reste, que ce soit pour les simulations aérodynamiques sur la partie extérieure de la voiture, ou bien les crash-tests, et les simulations des flux d'air dans l'habitacle [7]. Pouvoir visualiser en trois dimensions ces flux, en fonction de la déformation des surfaces comme dans les travaux de Bordegoni et al. [27] (voir figure 1.15), ou bien selon la position de l'utilisateur dans la voiture, permet aux ingénieurs de comprendre beaucoup plus facilement que sur station de travail, les résultats de ces simulations.

1.3.3.2 Formation et entraînement

Les technologies de RV offrent aussi la possibilité de présenter les informations sous divers formats et points de vue, et de pouvoir interagir avec celles-ci. Cette flexibilité est un atout en ce qui concerne les activités d'apprentissage, puisqu'elle permet de pouvoir s'adapter, et à l'environnement, et à l'utilisateur. Parmi les énormes avantages de la formation par le virtuel, on peut noter :

- la non-nécessité de se déplacer sur le lieu réel de l'activité, parfois difficilement accessible (l'espace) ou dangereux (une usine de produits chimiques), etc.
- la possibilité d'aborder la tâche de diverses façons : la présenter sous une forme de plus en plus complète, avec ou sans aide/assistance, etc.
- la possibilité d'aborder des événements extrêmement rares comme les incidents techniques (un accident), des situations critiques (immeuble en feu)

Ceci permet donc de réaliser des formations sans aucun danger indépendamment du risque réel, de faciliter l'initiation à une tâche quelle que soit sa complexité, de répéter autant de fois que nécessaire l'activité, et de contôler de bout en bout ce processus d'apprentissage. On retrouve donc bien évidemment cet apprentissage par le virtuel dans l'industrie.

Le domaine médical est un des pionniers dans l'utilisation des technologies de RV pour l'apprentissage, pour des raisons évidentes de criticité de ces activités : manipulation du corps

7. 3D immersive presentation and evaluation of CFD simulation results (Automotive) by ICIDO : http://youtu.be/zAZdNK7mrqM

humain. Ainsi, nombreux sont les systèmes permettant l'apprentissage d'activités médicales, comme il l'est rappelé dans les travaux de Mantovani et al. [105].

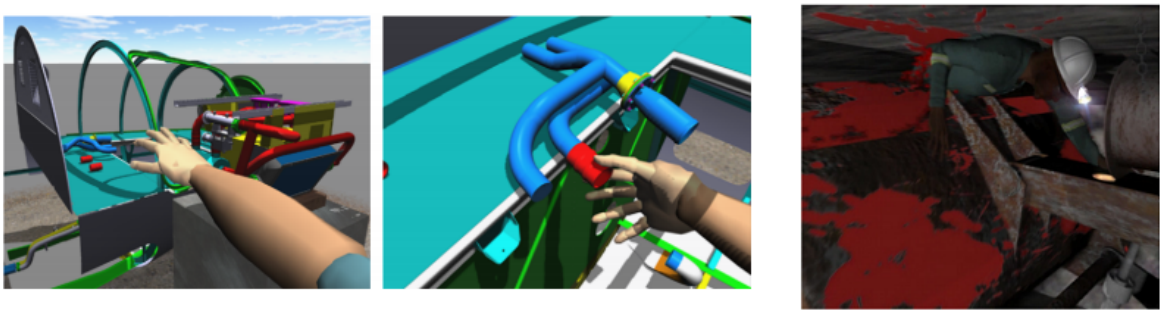


Figure 1.16 – Exemple d'apprentissage par la RV. Les deux images de gauche présentent une activité de maintenance d'avion, en extérieur (Abate et al. [1]). L'image de droite présente la formation de mineurs dans leur environnement de travail dangereux (van Wyk et al. [162]).

Van Wyk et al. [162] ont étudié la formation par la RV sur des activités dangereuses que sont celles des mineurs de fond (figure 1.16, droite). Abate et al. [1] ont eux présenté un système d'apprentissage assisté haptiquement pour une tâche de maintenance d'avion située en environnement extérieur (figure 1.16, gauche). Cet exemple illustre bien l'apport de la RV pour ce genre de tâche : d'une part, l'utilisateur n'a pas à se rendre dans le lieu réel de pratique de la tâche (ici en extérieur), d'autre part une aide peut-être proposée (ici une assistance haptique).

Parmi les activités de formation par la RV les plus répandues et les plus connues, il y a bien évidemment les différentes simulations de conduite. Les industries automobile et aéronautique se sont ainsi rapidement équipées de simulateurs, permettant notamment aux pilotes d'avion des sessions d'apprentissage sans risque.

1.3.3.3 Études ergonomiques

L'ergonomie et la RV partagent de nombreuses relations [37], des étapes de conception même des environnements virtuels, en passant par la formation et l'apprentissage, jusqu'à l'étude de l'ergonomie des postes de travail et des produits finis. De la même façon que l'apprentissage par la RV en permettant aux utilisateurs d'appréhender des situations ou activités futures, a suscité l'intérêt des industries, l'étude de ces utilisateurs sur leur poste de travail (figure 1.17) est rapidement devenu un champ d'application également très suivi par l'industrie, notamment manufacturière. Il en va de même pour l'ergonomie du produit fini qui consiste à déterminer s'il sera pratique à utiliser, à prendre en main, etc.

Concernant l'ergonomie du produit, on peut notamment citer certaines études de l'industrie automobile. De nombreux cas vont être simulés, notamment celui de l'utilisabilité des habitacles de voiture. Ainsi on va pouvoir aisément étudier les problématiques de visibilité, que ce soit celle du conducteur ou des passagers, les problématiques de fonctionnalité de la planche de bord, etc.

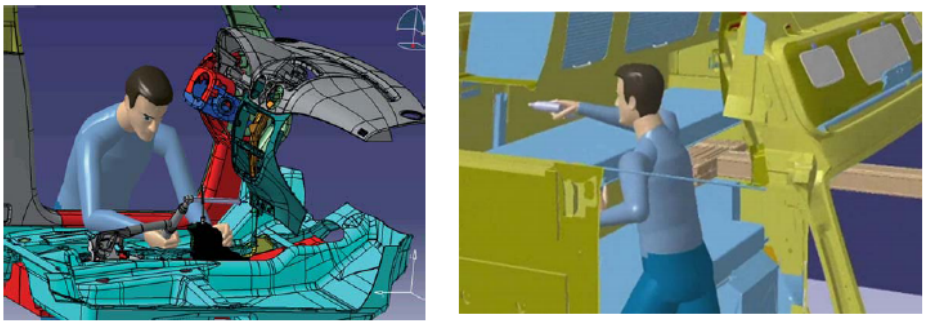


FIGURE 1.17 – Exemples d'étude ergonomique utilisant la RV. A gauche, un opérateur est étudié pendant son activité de vissage (Claudon et al. [46]). A droite, un opérateur réalise une activité d'assemblage / montage (Jayaram et al. [84]). L'idée est de determiner si les postures de ce poste de travail ne sont pas dangereuses pour la santé.

Par ailleurs, l'industrie — dont PSA Peugeot Citroën — est également active dans les études ergonomiques de l'activité d'un opérateur sur son poste de travail. Dans les travaux de Claudon et al. [46], on voit par exemple comment la RV et l'utilisation de mannequin numérique peuvent servir à analyser les postures du coprs humain et à déterminer une éventuelle dangerosité pour la santé. On va également trouver de nombreux cas d'études d'opérations de montage et d'assemblage (travaux de Jayaram et al. [84] et de Michalos et al. [120]).

1.3.3.4 Assemblage

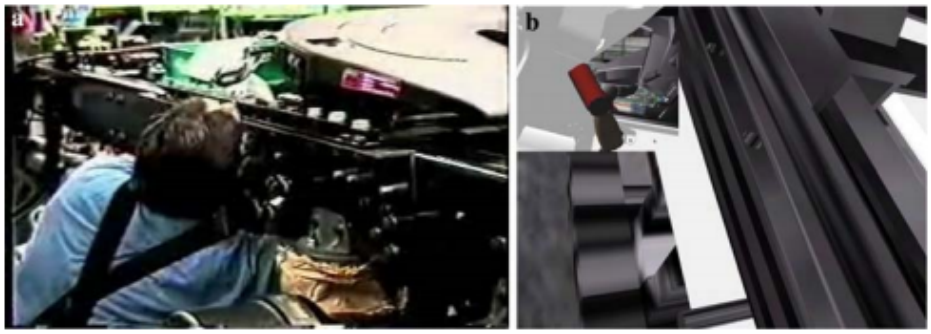


FIGURE 1.18 – Exemples d'assemblage virtuel (Jayaram et al [82]). A gauche, un opérateur place un boulon dans une usine réelle. A droite, ce même opérateur réalise l'assemblage dans l'environnement virtuel VADE.

Les activités d'assemblage et de montage ne sont pas simulées uniquement pour étudier l'opérateur et son corps. Ces étapes étant d'une importance extrême pour certaines industries, et étant sujettes à de nombreuses évolutions au cours du temps, il s'agit de les simuler aussi pour mettre au point ces processus, vérifier leur bon fonctionnement et leur bon déroulement, avant de les mettre en place physiquement et de les mettre en route pour de bon. Il existe ainsi, dans les industries et dans la littérature, de nombreux cas d'assemblages virtuels [42, 64, 87], s'intéressant parfois aux intéractions "naturelles" (Zachmann et al. [171]) ou aux intéractions haptiques (Tching [156]).

Ainsi Jayaram et al. présentent deux cas industriels d'assemblage virtuel [82] basés sur VADE, un système immersif précédemment développé [83] (figure 1.18). Dans ces travaux sont évalués la capacité à modéliser et à étudier les situations d'assemblage, ainsi que l'apport de telles simulations.

Seth et al. [145] proposent un large tour d'horizon des différentes activités d'assemblage en environnement virtuel. Ils couvrent notamment les différentes applications que l'on peut trouver dans la thématique de l'assemblage, et pointent les différents défis que sont la detection de collision, la gestion de contraintes entre éléments et la simulation de la physique des objets.

1.3.3.5 Revues de projets

Comme nous l'avons vu dans la section 1.2.4, le prototypage virtuel et par conséquent l'utilisation de la RV ont une grande importance dans la bonne gestion du projet (revoir figure 1.7). Pour la revue de projet utilisant la RV, on peut même parler de revue de projet immersive lorsque tout est mis en œuvre pour complètement plonger virtuellement les ingénieurs et concepteurs dans le prototype virtuel et dans son environnement. Comme nous l'avons vu précedemment, la RV va permettre dans de nombreux cas de pouvoir visualiser, analyser, vérifier, un produit ou le process de fabrication de ce produit, au plus tôt. La revue de projet, et plus particulièrement, l'évaluation du produit dès les premières phases de conception, devient alors une étape hautement importante, réunissant généralement un groupe de personnes (l'équipe du projet en cours), dont l'expertise peut varier.

En particulier, Antonya et Talaba [5] montrent comment l'utilisation de la RV pour l'évaluation et la modification de produit au cours de revues de projets offre une praticité et une flexibilité nouvelles. Dans ces travaux sont illustrés deux cas de revue du produit : l'analyse immersive pour spécifier des modifications, et la vérification de ces modifications lors d'une session immersive ultérieure. Le premier cas est sans doute le plus répandu, puisque le moins complexe. Les matériels à disposition aujourd'hui permettent relativement facilement de pouvoir visualiser un produit, fidèlement rendu à l'échelle 1:1.

Par contre, les approches pour la revue de projet modificative restent encore un peu balbutiantes, de par la complexité de leur mise en place. Il nous faut citer les travaux de Meyrueis [118] qui a mis en place des techniques permettant de visualiser et modifier des maquettes numériques au travers de déformations. Les grandes difficultés liées à la revue de projets modificative vont être la compatibilité des données initiales par rapport à l'environnement virtuel, ainsi que la mise en pratique même des modifications sur la maquette numérique.

Nous aborderons plus spécifiquement ces différents types de revue de projet dans le contexte de PSA Peugeot Citroën, dans le chapitre 2 (section 2.4.3).

C'est justement cette revue de projet immersive et modificative qui va être la cible principale de mes travaux de thèse.

1.4 État de l'art de la RV-CAO : de la visualisation stéréoscopique à l'édition implicite de données

Dans cette section nous présentons un état de l'art des intégrations de la Réalité Virtuelle dans les activités de Conception Assistée par Ordinateur. Par conception, nous entendons ici la démarche de création et de modélisation de produits, de formes, etc. Ainsi nous voulons montrer et exposer les différentes approches, partant de la simple visualisation de données

(revue de projet, présentation), à l'édition complexe d'objets CAO, tendant vers la revue de projet modificative.

1.4.1 Visualisation améliorée et manipulation

Les techniques et technologies de RV sont parmi les plus prometteuses et les plus efficaces pour améliorer l'interactivité des systèmes CAO et la visualisation de leurs données. Beaucoup de travaux de recherche sur l'intégration RV-CAO ont justement essayé d'améliorer la visualisation et la manipulation d'objets conçus par des systèmes CAO [15,20,47,54,63,68,85,87,137]. Le principe général, commun à toutes ces recherches, est de convertir les objets CAO provenant d'un système donné, et de les importer dans un Environnement Virtuel (EV) pour en avoir une meilleure perception. Les manipulations d'objets et/ou d'assemblages virtuels, ainsi que la navigation, sont souvent disponibles, bien qu'une des limites principales de ces approches soit l'impossibilité de modifier directement les objets CAO dans les EV.

C'est pourquoi un grand nombre de travaux se sont concentrés sur l'intégration RV-CAO permettant la création et la modification d'objets CAO : c'est ce que nous allons appeler le *modelage immersif*.

1.4.2 Esquisses et dessins en environnement virtuel

Au niveau de la création d'objet CAO, de nombreux travaux ont porté, et portent encore, sur l'esquisse immersive et sur le dessin immersif. Parmi elles, nombreuses sont les approches se concentrant sur le modelage de surfaces en environnement immersif [39,71,80,88,141]. Un résultat important issu de l'ensemble des recherches de ce domaine est que la conception de formes libres (*free-form design*) et l'esquisse en environnement immersif présentent un très fort potentiel, notamment dans les phases amont du processus de conception [60,69,153]. Ce résultat doit evidemment être rapporté au constat observable sur la figure 1.5, qui montre l'importance de ces phases pour l'engagement des coûts dans un projet.

FIGURE 1.19 – Sketching immersif en 3d (Stark et al. [153]).

Ainsi, Fiorentino et al. [69] donnent un bon exemple de système permettant la création, la visualisation et l'édition de surface, avec SpaceDesign. Ils ont notamment intégré une vé-

rification de contrainte visuelle immédiate, ce qui réduit la durée de la boucle de conception *esquisse - numérisation - vérification de contraite*. Stark et al. [153] proposent eux un environnement de modelage hybride, donnant la possibilié d'esquisser des objets en environnement immersif et de les intégrer dans un système CAO/SGDT — Système de Gestion de Données Techniques ; Product Data Management (PDM) en anglais. Ils fournissent également des indications techniques et empiriques sur l'implémentation industrielle de leur solution.

Cependant, une des limites majeures du modelage de formes libres immersif reste que la précision des interactions techniques 2d – généralement contraintes par un support comme une feuille reposant sur une table, ou une tablette graphique — ne peut pas être réellement assurée et retrouvée via les techniques de RV et des interactions 3d. Il existe néanmoins une piste dans la recherche d'interactions précises avec l'utilisation de périphériques haptiques, dont nous exposons quelques références.

Raymaekers et al. [138] vont ainsi proposer un outil pour dessiner intuitivement des esquisses. L'utilisation d'un modèle mathématique pour l'esquisse permet ainsi à l'utilisateur, sans qu'il connaisse le fonctionnement de ce modèle, de disposer d'un puissant outil d'édition pouvant particulièrement être couplé aux interfaces haptiques.

De leur côté, Sener et al. [144], ont étudié l'intégration de FreeForm, un outil de conception de formes libres par l'interaction haptique, dans un environnement industriel. Cet outil est particulièrement adapté aux phases préliminaires de la conception du produit, permettant d'explorer intuitivement et rapidement différentes idées de modelage via des déformations. Cependant, parmi les limites pointées de ce système, on trouve le manque de contrainte durant le modelage pouvant affaiblir la précision.

Bordegoni et Cugini [26] ont étudié l'introduction du modelage haptique dans les phases conceptuelles du développement du produit. Le sujet de leur étude est un système CAO comprenant des outils et des modalités pour aider le designeur dans ses interactions, avec notamment l'utilisation d'un périphérique haptique à six degrés de liberté. Parmi les résultats de cette étude, il ressort que les designeurs et ingénieurs CAO convergent vers le fait que cet outil est particulièrement adapté pour la création d'ébauche de formes ainsi que pour la vérification du modèle avant de passer à un prototype physique. On notera également les intéressants travaux d'Evans et al. [67] sur l'évaluation du retour d'information haptique pendant l'activité de conception, et dans leur cas précis, durant la conception d'un grille-pain.

Enfin Alcaide-Marzal et al. [4], présentent un état de l'art de l'utilisation de la sculpture numérique dans les premières phases de la conception du produit.

1.4.3 Modelage solide immersif

Le modelage solide est une composante utilisée par de nombreux systèmes CAO et joue un rôle important dans les industries de conception et de fabrication. Nombreux sont les auteurs qui ont proposé des solutions pour ce modelage solide immersif.

Parmi eux, Liang et al. [98] ont présenté un système de modelage 3d hautement interactif : JDCAD. Grâce à ce système, les utilisateurs peuvent créer des objets solides basés sur des primitives simples, et réaliser des opérations booléennes sur ces objets.

Dans le même esprit, Trikal et al. [160] ont développé un système de RV pour le modelage d'objets 3d, intégrant le système de modelage TWIN et permettant aux utilisateurs de créer des volumes à partir de primitives et d'effectuer des opérations de type solide. Une des caractéristiques principales de ce système était le maintient de la connaissance des cavités de l'objet avec leurs composants adjacents, et de polyèdres approximés, représentations graphiques des *design features*.

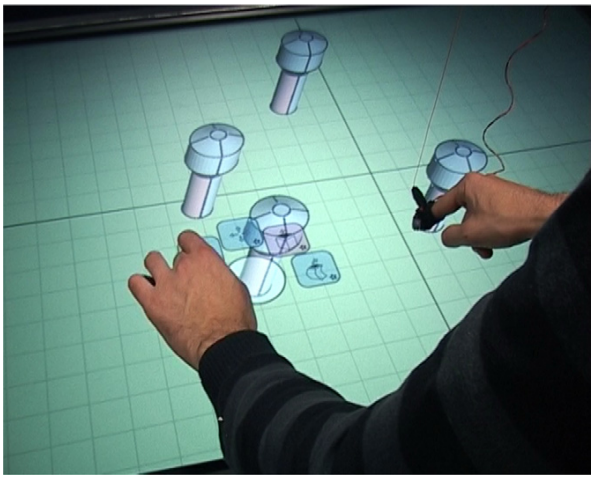


FIGURE 1.20 – Mockup Builder (de Araújo et al. [62]). L'utilisateur utilise ici sa main gauche pour interagir avec la surface tactile et accéder à des menus, pendant qu'il manipule les objets en faisant des gestes avec sa main droite.

Geo et al. [76] proposent l'extension d'un modèle solide pour le modelage solide à base de contraintes, dans environnement de RV semi-immersif. Leur approche donne la possibilité aux utilisateurs de manipuler des primitives graphiques au travers de données ajoutées spécifiquement, en l'occurrence des points de contrôle et des points de contrainte. Le comportement résultant de l'usage de ces primitives n'est cependant pas garanti en termes d'exactitude de la forme produite (par exemple, selon le degré des courbes et les conditions aux limites, plusieurs courbes sont possibles pour interpoler un ensemble de points). Ceci peut donc conduire à des interactions contre-intuitives et une certaine frustration chez les utilisateurs.

Ye et al. [170] ont travaillé sur le système LUCID qui propose de nouveaux paradigmes d'interaction aux utilisateurs, pour créer, modifier, visualiser et sentir des objets CAO dans un environnement de type poste de travail. Ainsi il était possible d'interagir avec ses deux mains, l'une tenant la souris, l'autre manipulant un périphérique haptique, tout en visualisant les données au travers d'un écran stéréoscopique.

Enfin De Araújo et al. [62] ont développé Mockup Builder, un environnement semi-immersif pour la modélisation 3d (voir figure 1.20). Cet environnement donne la possibilité de concevoir, créer, et manipuler des objets 3d issus de primitives géométriques classiques, par l'intermédiaire d'une table immersive et de gestes bi-manuels. La table immersive est dotée d'un rendu stéréoscopique permettant donc la visualisation en trois dimensions des objets, et son aspect tactile assure une partie de l'interaction, notamment la gestion de menus.

Afin d'assurer la généricité de l'intégration RV-CAO, de nombreux travaux ont été réalisés en se basant sur des noyaux géométriques communs. Parmi ces noyaux, les deux les plus utilisés sont ACIS [152] et OpenCascade [157].

ACIS kernel — Le noyau géométrique de ACIS a été la base de nombreux travaux [43, 56, 61, 154]. Dani et al. [56] puis Chu et al. [43] ont développé et présenté COVIRDS, un système de modelage. Grâce aux techniques d'interaction de RV, en particulier l'utilisation de la voix et des gestes, les utilisateurs peuvent créer et modifier des formes 3d basées sur

des primitives géométriques simples. Cependant, une des limites de cette approche se trouve dans le manque de précision des interactions 3d.

Cette précision de l'interaction est justement ce sur quoi Stork et al. [154] se sont concentrés. Ils ont en effet développé des techniques d'interaction pour supporter la création et la modification précise et rapide d'objets 3d en utilisant une souris 3d. Ce qu'il faut ressortir de leur approche est l'utilisation de grilles 3d et de techniques de *snapping* permettant justement de "coller" aux grilles, pour assurer la précision. Des règles sont également utilisées en fonction de l'élément topologique sélectionné — sommet, arête ou face — pour déterminer quelle modification appliquer. Ce système de contraintes semble malgré tout un peu trop rigide.

De Amicis et al [61] ont construit le sytème ARCADE/VT. Pour ce faire, ils ont combiné les techniques d'interaction de RV avec une *Virtual Table*, des fonctionnalités de CAO au travers du noyau ACIS, le *toolkit* 3d Open Inventor, toujours dans l'optique d'assurer un modelage précis. Ce travail a évolué pour donner SpaceDesign [69]. Les mêmes auteurs, Fiorentino et al., ont également présenté une étude sur la performance des utilisateurs dans un environnement de RV-CAO [70]. S'intéressant notamment aux tâches de sélection, de pointage et de dessin en environnement immersif, ils proposent trois outils afin d'améliorer l'interaction de l'utilisateur dans ces cas : un outil de transparence intelligente permettant une interaction non perturbée par l'occultation de certains objets ; le "3d Ortho Tool" pour aider l'utilisateur à dessiner des lignes droites horizontales, verticales ou perpendiculaires aux directions de l'écran ; et enfin le "3d Object Snap" qui est une évolution des outils de *snapping* 2d (aide à la sélection) déjà présents dans certains logiciels CAO.

OpenCascade kernel — D'autres approches pour l'intégration RV-CAO et le modelage solide immersif se sont plutôt basées sur OpenCascade, un noyau géométrique open-source. C'est notamment le cas de Ingrassia et al. [81] qui ont travaillé sur une intégration MEF-CAO [8] complète. Les objets — courbes et surfaces NURBS, et solides — peuvent être importés comme ils peuvent aussi être directement créés et une analyse MEF est disponible.

Foursa et al. [72] ont présenté un *framework* pour deux utilisateurs simultanés, leur permettant d'importer et créer des objets et de réaliser des tâches d'assemblage collaborativement. Pour aider les utilisateurs dans leurs manipulations, ils ont utilisé un système de contraintes d'assemblage basé sur une grammaire spécifique.

Citons aussi d'autres travaux, comme ceux de Weidlich et al. [167] qui présentent trois moyens pour intégrer la RV en tant qu'interface utilisateur pour le modelage géométrique immersif. En particulier dans leur méthode intitulée VRAx, le modelage immersif est combiné avec un kit de construction permettant d'assurer le design rapide de produit.

Ce qu'il faut retenir et remarquer de tous ces travaux, c'est qu'il ne s'agit jamais de lien *direct* avec le système CAO. En effet, dans chaque cas, les utilisateurs doivent convertir les géométries créées ou modifiées en sessions immersives, dans un format standard comme le format STEP ou IGES, et revenir au système CAO pour s'en servir. D'un point de vue "industriel", ces conversions et ce changement d'environnement sont synonymes de délai et donc d'augmentation du temps de conception.

Aujourd'hui, les principaux systèmes CAO utilisent un Graphe d'Historique de Conception (GHC, voir section 2.2.6) pour stocker tous les détails, l'historique de l'édition d'objets CAO [49]. La plupart du temps les informations sur l'historique de construction et de modification

8. MEF : Méthode des Élements Finis

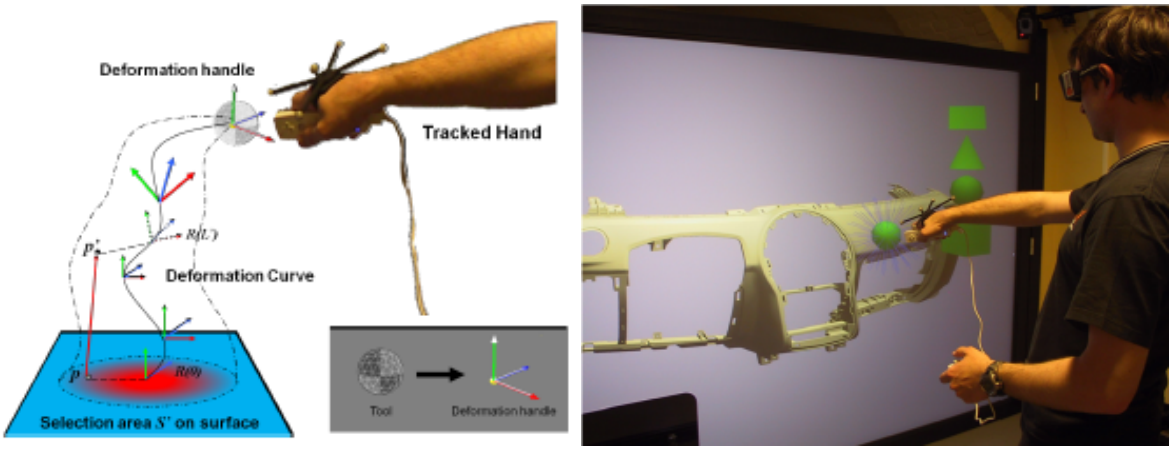


FIGURE 1.21 – Déformation 3d en environnement immersif (Meyrueis et al. [119]).

sont perdues lorsque les objets sont convertis dans d'autres formats. Des travaux ont tenté de surpasser cette limitation. C'est le cas de Meyrueis et al. [119] qui proposent une méthode, nommée D_3, qui utilise le noyau de ACIS pour déformer le maillage (*mesh*) d'objets 3d précédemment conçus dans un système CAO (voir figure 1.21). Pour ce faire, ils stockent les informations issues des déformations dans un GHC créé spécifiquement pour qu'une fois sorti de l'environnement immersif il soit possible de reconstruire les surfaces et de fusionner les modifications avec l'objet original dans le système CAO. Cependant, bien que la déformation des objets soit possible, la modification d'opérateurs existants, ainsi que leurs paramètres, ne l'est pas.

Une autre approche a été investiguée par Toma et al. [159] en se basant sur le système CAO commercial Solidworks [59] (voir figure 1.22). Grâce au module VRSolid fournissant une interface RV pour Solidworks, et grâce au système de gestion de fichier VRML [166], les utilisateurs peuvent créer et manipuler des objets CAO directement dans le système CAO cible. Ces travaux fournissent également plusieurs informations supplémentaires comme les résultats de la comparaison entre l'interaction sur poste de travail et via les environnements de RV :

- l'interface RV est plus intuitive que le poste de travail classique, et demande plus d'implication physique.
- l'interaction sur poste de travail, donc l'interaction 2d, est préférée pour la création d'objets.

Ces résultats renforcent complètement l'idée que l'un des moyens les plus prometteurs pour réduire le temps de conception, en particulier dans les processus industriels, est la modification du GHC d'objets CAO déjà construits. Pour la pratiquer en situation immersive, cela suppose l'*édition implicite* d'objets CAO que nous allons maintenant introduire.

1.4.4 Édition implicite de l'arbre de construction

Le manque d'approches, de travaux, et d'implémentations de la modification directe du GHC d'objets CAO peut facilement être expliqué par plusieurs obstacles. On peut trouver une première précision quant à ces obstacles dans les travaux de Stark et al. [153] :

– l'utilisabilité des interfaces utilisateur basées sur les techniques et technologies de RV reste limitée.
– les techniques d'interactions 2d que l'on trouve dans les systèmes CAO courants ne

peuvent pas être directement transposées aux systèmes de RV.
– la précision des opérations de modelage est difficilement assurée.

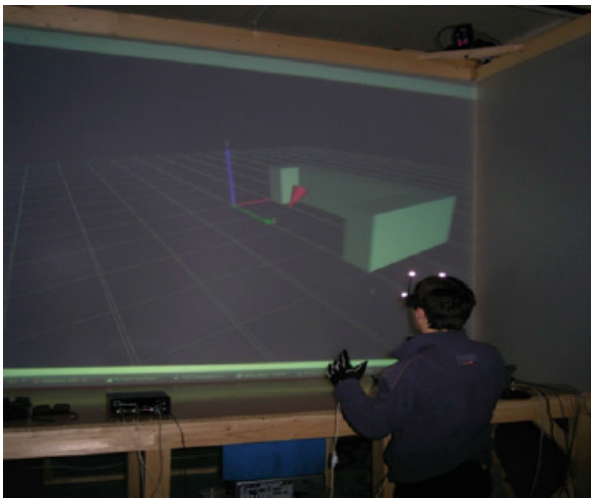


Figure 1.22 – Modelage solide immersif avec Solidworks dans l'environnement immersif HoloCave (Toma et al. [159]).

Il faut ajouter à ces problèmes la question de la complexité des objets que l'on veut modifier. En effet, parmi les limites traditionnelles des systèmes CAO utilisés couramment, on trouve le temps de mise à jour des modèles. Il arrive très régulièrement que cette mise à jour suite à une modification de l'objet puisse prendre quelques minutes, voire quelques dizaines de minutes. Ces délais pourraient très sérieusement déteriorer l'interaction temps-réel. Mais un des principaux problèmes à surmonter pour une inégration RV-CAO complète et efficace réside dans la possibilité d'utiliser et de réutiliser le *persistent naming* des systèmes CAO cibles. Il faut en effet être capable d'accéder au cœur des systèmes CAO et de leur noyau géométrique, et ce au travers de leur interface de programmation lorsqu'elle est disponible (API[9]).

Ma, Zhong, et al. [102, 172] ont conçu un modèle données basé sur des contraintes et structuré hierarchiquement pour le modelage solide en environnement de RV. Ce modèle se compose précisemment de trois couches : un modèle de contraintes pour la manipulation et la modification, des représentations hybrides "faites maison" des structures B-Rep et CSG (Constructive Solid Geometry, en fait l'ancêtre du GHC car limité aux opérations booléennes), et enfin la représentation polygonale des objets pour l'interaction et la visualisation temps-réel. De plus, les modèles B-Rep et de représentation polygonale sont liés par une référence croisée, et les opérations géométriques basiques sont réalisées par l'intermédiaire du noyau ACIS. Leurs travaux présentent une implémentation expérimentale avec le logiciel Division Reality [133].

Wang et al. [164] ont eux présenté un environnement RV lié à un environnement CAO. Ils ont conçu une intéressante structure de données hybride pour permettre la modification

9. API : Application Programming Interface

lors de séances de conception paramétrique (voir figure 1.23). Ainsi, un système de RV, plus exactement un poste de travail immersif, et un système CAO (Autodesk [10]) sont liés, lancés et tournent chacun de leur côté distinctement. Les informations sémantiques des objets CAO sont extraites lors de la création de l'environnement de RV, et chaque modification est directement répercutée dans le système CAO grâce à l'introduction du mécanisme de *persistent naming* dans cette intégration RV-CAO. L'objet CAO est alors rechargé dans l'environnement virtuel pour continuer la séance de travail.

FIGURE 1.23 – Modification paramétrique en direct dans un environnement virtuel (Wang et al. [164]).

Lors de précédent travaux, le groupe de Réalité Virtuelle et Augmentée VENISE du LIMSI-CNRS s'est lui aussi penché sur l'intégration RV-CAO [32, 33, 49]. Ces recherches ont consisté en la conception d'un modèle CAO permettant l'édition implicite du GHC d'objets CAO. Ce modèle est basé sur un mécanisme de *labelling* couplé à un *persistent naming* spécifiquement développé et assurant un lien permanent entre les représentations B-Rep et GHC. Comme preuve de faisabilité de cette approche, le démonstrateur VRAD a été développé autour du noyau géométrique d'OpenCascade. En plus de cette implémentation, des pistes ont été données pour l'application de cette approche dans un contexte industriel et notamment autour du noyau de CATIA — et ses structures B-Rep et GHC spécifiques [32]. Nous reviendrons sur ces éléments lors de la présentation de notre modèle de données au chapitre 3.

1.4.5 Conclusion

De nos jours, la conception paramétrique basée sur des *features* est utilisée par quasiment toutes les industries manufacturières. Les objets CAO sont généralement construits à partir d'opérateurs géométriques élementaires — extrusion, révolution, chanfrein, trous, etc. — mais aussi d'autres beaucoup plus complexes. L'historique de conception est stocké dans un Graphe d'Historique de Construction (GHC) aussi nommé "graphe d'historique de conception" ou "graphe de dépendance des *features*". Dans les principaux systèmes CAO, une représentation des éléments de frontière (B-Rep) est attachée à chaque nœud du GHC. Depuis de nombreuses années, les systèmes de CAO et de RV sont utilisés, de plus en plus, afin de gagner en efficacité, par la réduction du temps de conception et des coûts, et plu-

sieurs approches ont tenté de combiner ces différents systèmes. Mais il manque clairement d'approches qui pourraient trouver du succès dans les environnements contraints que sont les industries. L'intégration RV-CAO est un des principaux challenges pour les années à venir en ce qui concerne l'optimisation des phases de conception et de design.

1.5 Problématique

D'un côté, la CAO est utilisée dans un grand nombre d'applications industrielles, et de l'autre, la RV fournit de nouvelles perspectives pour les utilisateurs dans leurs interactions avec ces outils de CAO, et en particulier lors des revues de projet. La RV appliquée à la CAO offre alors de grandes perspectives d'interaction et de manipulation des données, dans l'optique d'optimiser les processus de conception (voir figure 1.24).

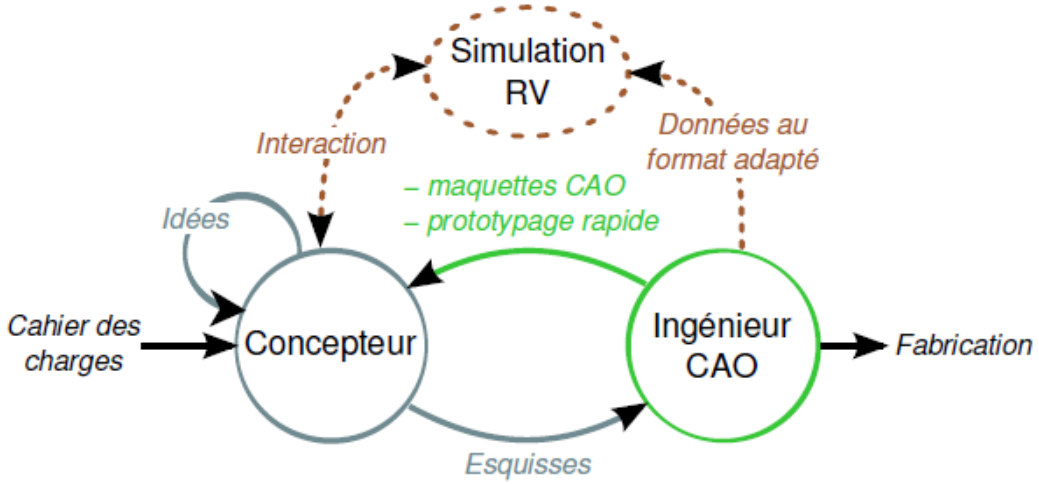


Figure 1.24 – Schématisation de l'introduction de la RV dans le cycle de conception (Picon [128]).

Ainsi, bien que la mise en place de la RV dans le cycle de conception du produit ait permis de réduire le temps imparti à certaines activités, on voit que certaines étapes ou contraintes pourraient être elles aussi optimisées :

- Compatibilité des données : bien souvent, les plateformes de CAO et de RV sont distinctes, et les données doivent être traitées et préparées pour être utilisées dans les environnements virtuels. Cette incompatibilité cause également des soucis lorsqu'il s'agit de vouloir modifier des données.
- Interaction : les logiciels de CAO courants, sont destinés à être utilisés sur des postes de travail standard, avec des interactions clavier/souris, voire souris 3d et tablettes graphiques. Cependant ces moyens d'interaction n'ont pas lieu d'être utilisés dans les environnements virtuels.

La plupart des approches immersives pour le design géométrique sont basées sur des représentations qui sont favorables aux interactions 3d, exploitant des modèles permettant la

création et la modification par des manipulations directes : grilles de voxels, représentations de fonctions, dessins de surfaces libres, primitives de polyédriques simples, déformations basées sur la physique ou maillages multirésolution complexes. Cependant, toutes ces représentations ne peuvent être utilisées pour le design que dans les premières phases du cycle de vie du produit, notamment les phases de croquis. En revanche, peu de recherches ont été menées dans le domaine de la conception de pièces, les objets de ces études ne pouvant généralement pas être modifiés en environnement immersif. De plus, de notre point de vue, il importe de ne pas perdre la richesse des informations que contiennent ces modèles CAO en termes de sémantique esthétique, géométrique, fonctionnelle, technique et métier, d'autant que dans le domaine de la création de produits industriels, l'activité de conception consiste souvent à adapter des composants existants.

La problématique principale qui nous intéresse est donc de trouver des moyens qui permettent de travailler en RV avec des données CAO, de manière optimale : sans avoir à réaliser des conversions qui contraignent et appauvrissent les données, que ce soit en entrée ou en sortie de session de travail, et en assurant la modification de ces données pendant la revue de projet immersive.

1.6 Objectifs

L'objectif de cette thèse est donc de proposer un modèle de conception immersive qui permette aux principaux acteurs de la conception de travailler ensemble, de pouvoir modifier en temps réel les modèles numériques, et ce directement au cours des séances de travail dans des dispositifs immersifs. Comme nous l'avons explicité dans la problématique, deux axes de recherche seront investigués :

- l'**accès direct** en RV aux données natives de systèmes CAO, pour atteindre les nombreuses informations que contiennent le graphe de construction et les modèles paramétriques (cf. *features*) ;
- l'**interaction intuitive** et/ou multimodale en RV, pour gérer des tâches de revues de conception en situation immersive.

Le premier axe va passer par l'élaboration de nouvelles approches dans l'interfaçage entre les modèles de données des systèmes CAO et les techniques interactives de la RV. Comme nous l'avons signalé précédemment, la solution idéale pour l'industriel est de travailler directement avec les données fournies par les concepteurs. Ceci tend donc à exclure, même pour les revues de style ou pour des séances de conception ergonomique, toute approche basée sur le concept de modèle intermédiaire. En évitant ainsi le dédoublement de la maquette numérique (RV vs. CAO), ce choix permet d'éliminer tout problème de synchronisation et de mise en cohérence des deux maquettes, ainsi que de conserver en RV toutes les informations contenues dans le modèle CAO sur l'état d'avancement du produit en gestation. Cependant, même si des solutions de *clustering* graphique permettent déjà un affichage stéréoscopique de données CAO dans des dispositifs de type CAVE, modifier en temps réel ces maquettes numériques au cours de séances immersives de revue de projet reste un problème ouvert.

Le principe de l'édition implicite du Graphe d'Historique de Construction (GHC) des objets CAO, sujet auquel s'intéresse depuis de nombreuses années le groupe VENISE du LIMSI-CNRS, est la piste qui a été privilégiée pour ces travaux de thèse. Les précédents travaux sur le sujet n'ont pas été démontrés sur des données issues de systèmes commerciaux, largement utilisés dans l'industrie manufacturière, ni sur des données massives et complexes

de situations réelles de conception de produits. C'est là-dessus que porte notre principale contribution. Ainsi nous présenterons dans le chapitre 3 un modèle de données RV-CAO permettant de gérer, visualiser et modifier, des données CAO natives, dans des environnements de RV. Puis nous montrerons dans le chapitre 4 les apports de ce modèle.

Un second axe vise à étendre les solutions d'interactions en RV sur des objets CAO. Nous montrerons dans le chapitre 5 les avancées, en termes d'interactions, que permettent un tel modèle d'interfaçage RV-CAO, en situant notamment l'articulation de notre modèle avec les différents travaux précédemment réalisés par l'équipe VENISE et PSA Peugeot Citoën sur la multimodalité et l'haptique. L'édition implicite des CHG nécessite un système d'inférence qui peut également être utilisé pour la supervision multimodale (combinaison d'interactions vocales, de commandes gestuelles, activation de guides virtuels visuels, haptiques ou pseudo-haptiques, etc.). Nous verrons ainsi également dans ce chapitre 5 quelles sont les pistes de réflexion pour faire évoluer le système d'inférence afin qu'il tienne compte non seulement des données CAO, mais aussi des interactions de l'utilisateur.

1.7 Conclusion

Dans ce chapitre nous avons donc présenté le contexte global de nos travaux de thèse. Le contexte industriel hyper concurrentiel renvoie chaque acteur vers l'optimisation de ses processus de conception et développement dans l'idée de produire plus et plus vite ainsi que de satisfaire le besoin client. Dans la conception et le développement de produit des industries manufacturières, les phases amont sont extrêmement importantes pour la poursuite du projet, comprenant en particulier les premières études de design, de forme de l'objet, etc. Les dernières décennies ont vu l'explosion de l'utilisation des nouvelles technologies dans la vie courante de notre société et notamment dans les processus industriels. Nous parlons en particulier ici de l'essor de la Conception Assistée par Ordinateur (CAO) et de la Réalité Virtuelle (RV), cette dernière étant actuellement encore en pleine ascension. Alors que désormais presque l'intégralité du processus de conception est informatisé, la RV trouve, elle, de plus en plus d'applications au sein de ce processus, allant de l'assemblage à la revue de projet et en passant par l'ergonomie des produits mais aussi de leur fabrication. Mais la RV nécessite d'être encore plus présente dans les phases amont du processus de conception au sein desquelles elle a un rôle à jouer dans le besoin d'optimiser ce processus.

Chapitre 2

La CAO : concepts de base et utilisation dans l'industrie automobile

Résumé. *Ce chapitre va nous permettre de présenter les différents concepts sur lesquels se basent les systèmes CAO, et principalement les logiciels de modélisation "volumique" couramment utilisés dans l'industrie aujourd'hui : la représentation par frontière (B-Rep), la représentation type Graphe d'Historique de Construction (GHC), les types d'opérateurs booléens et* form-features. *Parmi les concepts et problématiques inhérents à ces systèmes, on trouve le problème du* persistent naming. *Nous caractériserons également les différentes approches pour l'intégration RV-CAO, ce qui nous permettra de qualifier au mieux nos travaux. Enfin nous donnerons un aperçu de l'utilisation de la CAO et de la RV-CAO chez PSA Peugeot Citroën.*

Sommaire

2.1 Introduction

Comme nous venons de le voir dans le chaptire 1, la CAO est aujourd'hui partie intégrante, voire indispensable, du processus de conception dans l'industrie. L'explosion du nombre de systèmes CAO sur le marché est notamment intervenue dans les années 1990 avec l'apparition de nombreux logiciels. Mais quels sont les fondements de ces logiciels, et plus précisément sur quels concepts se basent-ils ? Afin d'aborder la problématique de la modification d'objets CAO, il est important de comprendre les modèles sur lesquels s'appuie la modélisation et la représentation d'objets CAO.

De nos jours, les systèmes CAO proposent essentiellement les modélisations de surfaces et de solides, appelées "modélisation surfacique" et "modélisation volumique". Alors que la modélisation de surfaces consiste en la création de formes libres, la modélisation de solides permet, elle, la création d'objets comportant des propriétés topologiques, à savoir la capacité à représenter l'existence ou l'absence de matière. Deux types de modèles existent pour cela, les approches discrètes à base de voxels, et les approches les plus répandues dans les sytèmes CAO décrivant les frontières des objets. Ce sont sur ces dernières que nous nous concentrerons. La compréhension de ces différents modèles va nous permettre d'aborder au mieux la modification immersive et intuitive des objets CAO, notamment pour la modélisation de solides qui sera au cœur de nos travaux.

2.2 Concepts de base de la CAO

Dans cette section nous allons présenter les concepts clés des systèmes CAO couramment utilisés, en particulier dans l'industrie et l'industrie automobile. La modélisation géométrique a bien evidemment profité de l'évolution des techniques informatiques et des différentes améliorations technologiques, ce qui lui permet aujourd'hui d'assurer bon nombre d'opérations complexes.

D'un côté, la modélisation surfacique va être utilisée pendant les étapes de conception où l'esthétique du produit est primordiale.

De l'autre côté, la modélisation de solides, généralement utilisée lors de la conception technique du produit (faisabilité, montage, etc.) est devenue dans sa version moderne, une modélisation paramétrique de solides, associant aux géométries et aux éléments topologiques — sommet, arête et face — la connaissance de l'ingénieur et du concepteur. À la représentation des élements de frontière sont donc venues s'ajouter les différentes évolutions de la représentation de l'historique de construction des objets CAO, qui a permis d'introduire des processus de modification et d'édition évolués. Ce sont d'ailleurs sur ces deux représentations (frontière et historique) que va se jouer dans notre approche le lien avec les systèmes de RV.

2.2.1 Modélisation surfacique

La modélisation surfacique, comme son nom l'indique, est une modélisation géométrique se basant sur les surfaces. C'est cette modélisation qui est le premier outil des concepteurs de formes puisqu'il permet la représentation et la manipulation des surfaces. La méthode la plus couramment utilisée pour la modélisation surfacique va être de modéliser et d'assembler des surfaces, appelées "carreaux" (très souvent basés sur le formalisme de Bézier). L'objet va alors être entièrement défini par de tels assemblages, permettant un contrôle précis de ses différentes courbures. Les carreaux de ces surfaces sont en fait définis par une équation bi

ou tri-paramétrique résultant du produit de matrices de points de contrôle avec des bases polynomiales. Lesdits points de contrôle permettent d'ajuster les propriétés de continuité tandis que l'approximation polyédrique (maillage) de l'équation paramétrique déterminée va permettre l'affichage de la surface.

Lorsque les courbures de l'objet vont être plus complexes, cette modélisation par carreaux ne va plus être adaptée et on préférera gérer plus globalement ces carreaux de surfaces, par des interpolations de points et courbes de passage via des formalismes paramétriques plus génériques tels que les NURBS (*Non-Uniform Rational Basis Splines*). L'avantage est en particulier de produire directement des réseaux de carreaux en garantissant des continuités paramétriques ou géométriques du second degré.

Ceci étant, le formalisme NURBS utilise aussi des points de contrôle que l'utilisateur va pouvoir modifier (déplacer). Quoi qu'il en soit, grâce à ces modèles, la définition de surfaces complexes va pouvoir être effectuée de différentes façons :

- créer des surfaces en appui sur des bords ;
- créer des surfaces par le balayage d'une courbe le long de deux courbes ;
- créer des surfaces par l'interpolation de courbes elles-mêmes issues d'une interpolation de points ;
- assembler des carreaux ou des réseaux de carreaux pour créer des surfaces plus complexes ;
- raffiner des carreaux ou des réseaux de carreaux en imposant de nouveaux points de passage et en transformant des surfaces en patchwork de carreaux ou de surfaces ;
- déformer les carreaux ou surfaces sans changer leur structure de réseau ou de patchwork en agissant sur les points de contrôles et les propriétés de continuités ;
- etc.

Il est clair que modéliser des surfaces vise principalement les étapes de conception du produit où l'esthétique et le style sont primordiaux par rapport aux aspects techniques du produit. Cette modélisation présente en effet l'avantage de permettre une meilleure visualisation, une meilleure présentation, les objets apparaissant le plus réaliste.

2.2.2 Modélisation solide

Comme nous l'avons signalé en introduction, la modélisation solide, dite aussi volumique, vise à représenter la notion de matière par rapport à l'absence de matière à partir de primitives solides (voir la description du modèle CSG en section 2.2.4). Deux modèles sont fondamentalement possibles pour cela, comme le montre le schéma de Rechiqua (figure 2.1). Le premier consiste à représenter de façon discrète la matière à l'aide d'un descriptif à unité élémentaire de volume appelé *voxel* (*volumetric pixel*), autrement dit une cellule cubique. Un *voxel* intérieur à un objet aura une valeur booléenne de 1, celui extérieur une valeur de 0. Afin d'optimiser la représentation mémoire les *voxels* connexes de même valeur (1), sont structurés dans des octrees. Même si cette représentation est par exemple très utile pour de l'édition 3d, elle présente pour inconvénient majeur de devoir être réévaluée en fonction de la précision requise. En outre, si cette représentation permet de gérer de façon triviale le résultat d'opérations booléennes (voir section 2.2.4), elle ne décrit pas de façon explicite les frontières entre matière et absence de matière, et ne permet pas une intégration facile entre modélisation surfacique et volumique.

C'est pourquoi la plupart des systèmes CAO utilise comme représentation volumique un modèle de graphe topologique appelé B-Rep (*Boundary Representation*) que nous décrivons

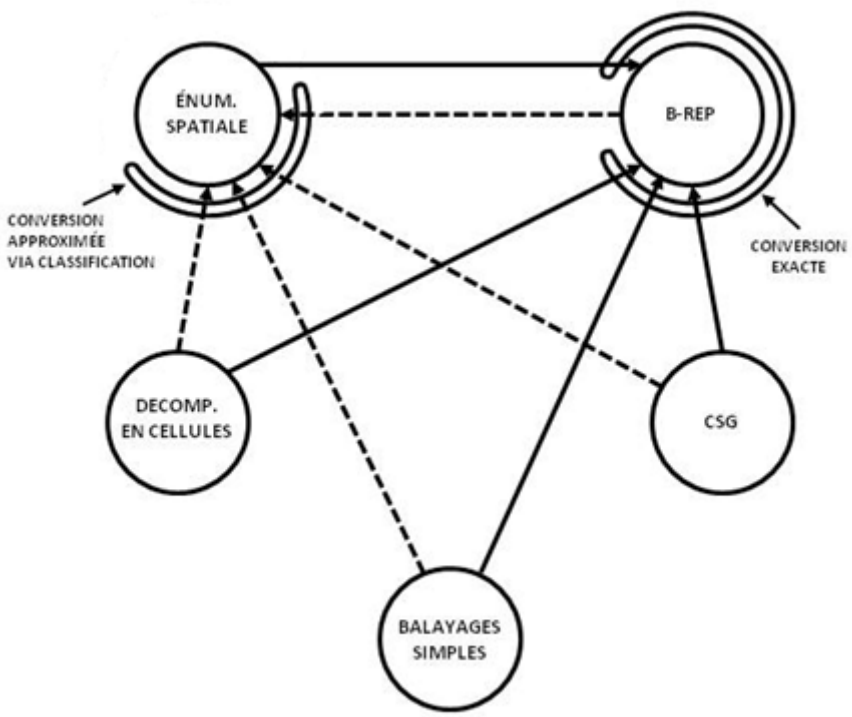


Figure 2.1 – Conversion entre les différentes représentations de solides (Requicha [140]).

dans la section 2.2.3.

De façon générale, les modèles solides créés vont servir aux études de fonctionnement du produit, servant ainsi à différentes simulations numériques comme la résistance de pièce, ou permettant la conception d'assemblages et de mécanismes. Aujourd'hui, la modélisation solide va aussi servir de base au prototypage rapide.

Nous allons dans les sections qui suivent, présenter les différents modèles et représentations utilisés dans le cadre de la conception de solides géométriques.

2.2.3 B-Rep

Le modèle de représentation de la géométrie des objets que l'on retrouve dans nombre de systèmes CAO utilisés pour la modélisation paramétrique est la représentation des éléments de frontière, ou en anglais *Boundary Representation* (B-Rep) [17,34]. Ce modèle permet ainsi de caractériser un solide en 3d par les éléments qui composent ses bords : sommets, arêtes et faces (voir figure 2.2). Il associe les informations topologiques de connexité entre les éléments décrits à la métrique des objets (chaque sommet, arête, ou face pointe vers une définition numérique, allant de simples coordonnées de points à des courbes ou surfaces paramétriques) et donne ainsi la possibilité de pouvoir facilement réaliser des requêtes pour retrouver certains de ces éléments.

La première structure de données pour un B-Rep est sans doute la *Winged-Edge Data Structure* de Baumgart [16, 17]. Cette structure permet de décrire des polygones, sans trous, par l'intermédiaire de leurs arêtes auxquelles les informations suivantes sont attachées : les sommets de départ et d'arrivée, les faces droite et gauche, les arêtes précédente et suivante dans la face de gauche, et les arêtes précédente et suivante dans la face de droite. Les préde-

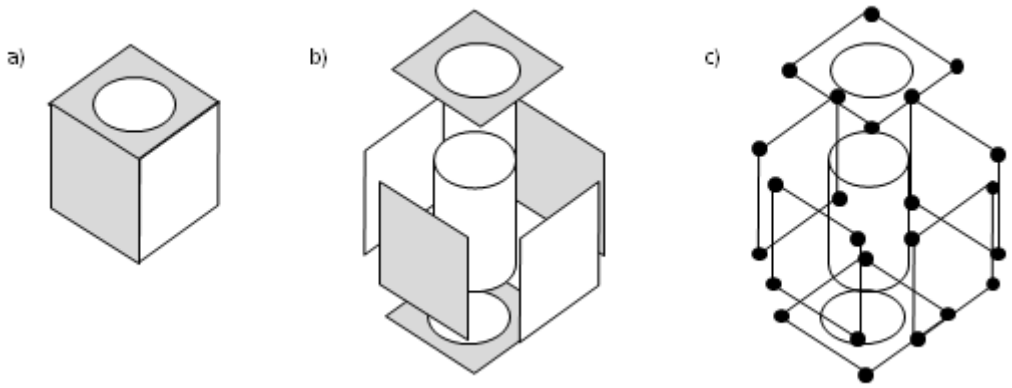


Figure 2.2 – Définition d'un modèle B-Rep. À gauche (a), l'objet solide complet ; au milieu (b), sa représentation en termes de surfaces topologiques ; à droite (c), sa représentation en termes d'arêtes et de sommets topologiques.

cesseurs et successeurs sont déterminés pour chaque face incidente selon le sens inverse des aiguilles d'une montre. Un exemple de cette structure est représenté dans la figure 2.3.

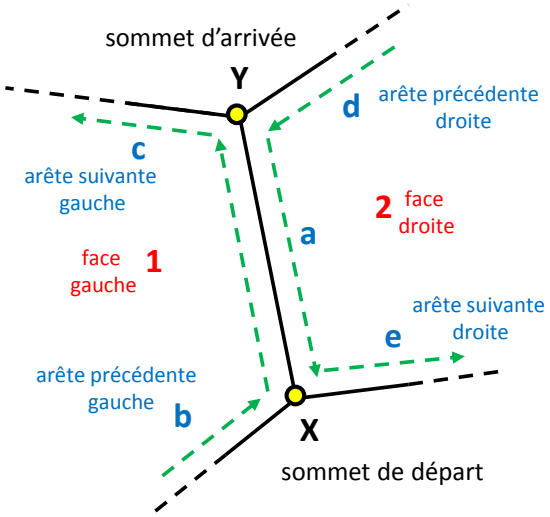


Figure 2.3 – Exemple de structure *Winged Edge*. L'arête principale est la a), partant du sommet X et arrivant au sommet Y. Ainsi sont définies les faces gauche (1) et droite (2) puis les prédecesseurs (b et d) et les successeurs (c et e).

Mais que se passe-t-il si les faces du solide contiennent des trous ? Dans ces cas là, le modèle décrit ci-dessus ne fonctionne plus. Cependant quelques pistes ont rapidement été proposées :

– Les boucles "internes", donc les parcours des arêtes définissant les trous, sont parcourus

dans le sens contraire des boucles "externes".
- Une arête est rajoutée, liant les boucles externe et interne pour qu'il n'y ait au final qu'une seule et même boucle, englobant le trou de la face.

Mais il existe également une autre structure de données pour les BRep, mieux adaptée aux opérations d'Euler : on parle de *Half Edge*, mise au point par Mãntylã [107]. Rappelons que les operateurs d'Euler sont des opérateurs de base permettant de modifier des objets *2d-manifold* (variété affine de dimension 2). Ces opérateurs permettent notamment de modifier les objets en gardant un contrôle sur la topologie, et Mãntylã a montré que ces opérateurs, formant un ensemble complet, peuvent être utilisés pour construire n'importe quel polyèdre [106]. Sa particularité est de couper en deux les arêtes dans le sens de la longueur — une par sens de parcours de l'arête initiale —, pour que chaque demi-arête soit utilisée par la seule face à sa gauche. Chaque demi-arête possède toujours les liens vers les sommets de départ et d'arrivée, et possède également les liens vers la demi-arête qui la précède, et celle qui la suit, au sein de la boucle.

Cette représentation B-Rep est la base du noyau ACIS [152], qui a été un important vecteur de propagation de son utilisation au sein du domaine de la CAO.

2.2.4 Géométrie de construction de solides et opérations booléennes

La représentation CSG a été initialement la représentation la plus utilisée par les systèmes CAO commerciaux du fait de sa simplicité d'implémentation. Dans cette représentation, issue des travaux de Requicha et al. [139, 140], un solide est décrit comme un ensemble d'objets solides simples basés sur des primitives géométriques (sphère, cube, cylindre, etc.) combinés à l'aide d'opérateurs booléens "réguliers" [1] (voir figure 2.4) :
- Union : cette opération additionne, fusionne, les deux objets complets.
- Intersection : cette opération garde la partie commune aux deux objets.
- Différence : cette opération soustrait un objet à un autre objet.

Cette représentation a rapidement connu un large succès par son aspect pratique et nombreux sont les logiciels qui se sont basés dessus. Cette approche CSG a comme principaux aspects attrayants le fait de pouvoir facilement construire des objets complexes à partir de formes simples, tout en conservant une précision mathématique importante. Il faut aussi noter que ces objets se prêtent aisément à la détection de collisions et plus généralement à toute sorte de calculs se basant sur les volumes.

Bien qu'ayant donc connu un grand succès dans les premiers logiciels de modélisation 3d, cette représentation CSG n'est plus utilisée dans sa forme initiale, principalement du fait des limitations inhérentes à ce modèle : la modélisation à partir de primitives géométriques simples reste limitée, ne permet pas de réaliser des formes moins "standard" à base de surfaces de formes libres, et la réalisation d'objets complexes peut rapidement devenir extrêmement lourde.

Ainsi, une première évolution a été de combiner la représentation CSG avec la représentation B-Rep, association qui permettait d'envisager des opérations topologiques entre objets plus complexes que de simples primitives. Dans ce cas, chaque nœud du CSG est porteur

1. Un opérateur booléen est dit "régulier" lorsqu'il garantit que l'opération est interne à l'ensemble des solides.

d'un B-Rep, B-Rep dont la métrique peut donc être constituée, entre autres, de courbes et surfaces paramétriques. Ce faisant, les logiciels CAO ont vu leur représentation CSG évoluer vers des graphes d'historiques (voir section 2.2.6) pour intégrer les opérateurs de définition de ces métriques de forme libre, mais aussi différents opérateurs de balayage. En outre, les opérateurs booléens ont été revisités pour les encapsuler dans un formalisme de plus haut niveau, les *form-features*.

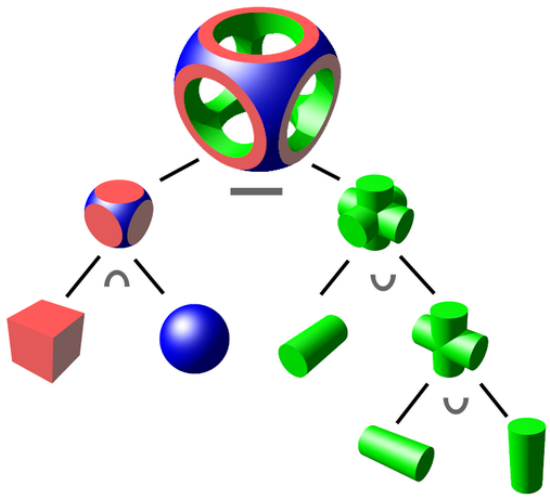


FIGURE 2.4 – Exemple d'arbre CSG. L'objet solide final est en fait le résultat d'une soustraction entre deux objets : le premier objet, à gauche, est l'intersection entre une sphère et un cube ; le deuxième solide, à droite est le résultat de l'union de trois cylindres.

2.2.5 Caractéristiques de formes (*form-features*)

Le concept de *feature*, en français d'*entité* ou de *caractéristique*, est apparu pour parer aux limitations propres aux opérateurs géométriques simples, en particulier l'insuffisance d'informations liées aux objets modélisés. Comme le rappelle Shah [146], il existe plusieurs points de vue sur le concept de *feature*.

D'un point de vue **manufacturier**,

> *"les* features *représentent des attributs de formes et des technologies, associés à des opérations et des outils"*.

Mais du point de vue de la **conception géométrique**,

> *"les* features *sont des groupements d'entités géométriques et topologiques qui doivent être référencées ensemble"*.

Ainsi, bien que des nuances puissent apparaître dans la définition de ce qu'est une *feature*, la notion commune qui se dégage est la volonté de représenter pour une partie (ou la totalité) de l'objet modélisé, la signification de sa conception en termes d'ingénierie.

Plusieurs classifications ont été proposées pour ces *features* [38, 53, 147], notamment celle différenciant leur type :

– *Features* de forme (*form-features*) : informations liées à la géométrie nominale ;

- *Features* de précision : informations sur la mesure, telles des écarts, des tolérances, etc. ;
- *Features* technologiques : informations telles les paramètres de performance, etc. ;
- *Features* de matériau : informations sur le matériau, telles sa composition, son traitement, sa condition, etc. ;
- *Features* d'assemblage : informations pour l'assemblage d'éléments, telles l'orientation des objets, les relations cinématiques, etc.

Dans notre cas, seules les caractéristiques de formes, les *form-features*, nous intéresseront. Ce sont en effet celles-ci que l'on retrouve dans les logiciels de CAO utilisés de nos jours. De nombreuses études ont été réalisées sur ce type de *feature*, notamment celles de Shah et Mãntylã [148], et des classifications ont été proposées, comme celle de Cunningham et al. [53] établissant deux groupes — statique et cinétique. Ainsi, on peut distinguer les *form-features* selon le rôle qu'elles jouent dans la conception, et notamment classer les caractéristiques "statiques" selon qu'elles ont trait à la forme générale, à des modifications locales, à des combinaisons de primitives, etc.

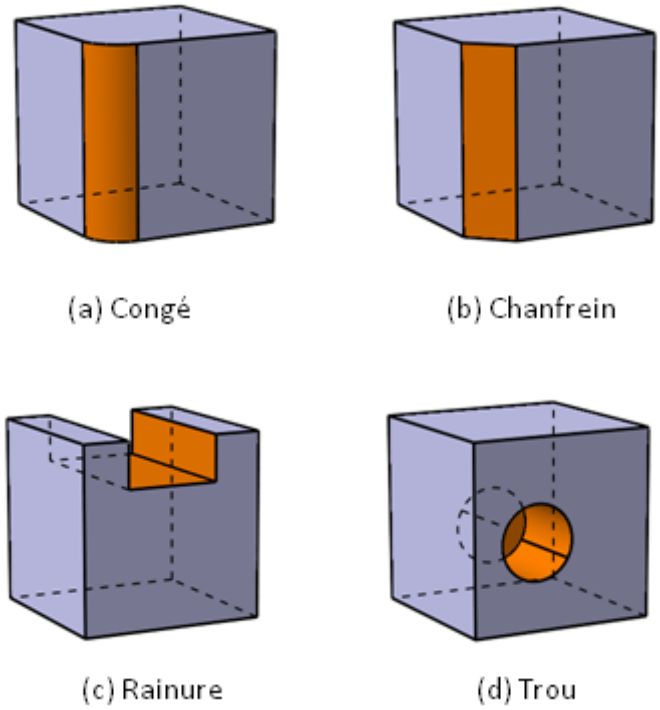


Figure 2.5 – Exemple de *form-features*. Dans les 4 cas, la *form-feature* en question, représentée en orange, est appliquée sur une partie d'un solide extrudé, en violet.

Avec les *form-features*, les ingénieurs disposent ainsi de puissants outils pour concevoir différents types de pièces, et répliquer facilement des opérations spécifiques (voir figure 2.5). Ces caractéristiques de formes agrégeant et embarquant plusieurs informations, dépendantes du domaine d'application dans lequel s'insère l'activité de conception, sont paramétrées numériquement (dimensions, etc.), symboliquement (type de formes : type de trou, type de

limite pour une extrusion, etc.) et/ou topologiquement (élément sur lequel s'applique la *feature*). Il devient alors aisé de modifier les formes, sans recommencer la conception depuis le début. De même, le fait de pouvoir accéder directement à ces informations permet de se concentrer localement sur l'objet, sans connaître forcément toute l'intention de conception, ni la géométrie de l'objet entier.

2.2.6 Graphe d'historique de construction (GHC)

Comme nous l'avons vu dans la section 2.2.4, les logiciels CAO se sont d'abord basés sur l'approche CSG, représentant alors un objet sous forme d'un arbre, où les feuilles sont des primitives et les nœuds sont des opérateurs booléens. Mais cette représentation, limitée, devait être dépassée. C'est ainsi que les graphes d'historique de construction (GHC) [50] — appelés aussi *feature dependency graph* [78], *design history* [168], ou *design feature history* [44] — sont apparus, étendant et généralisant les CSG.

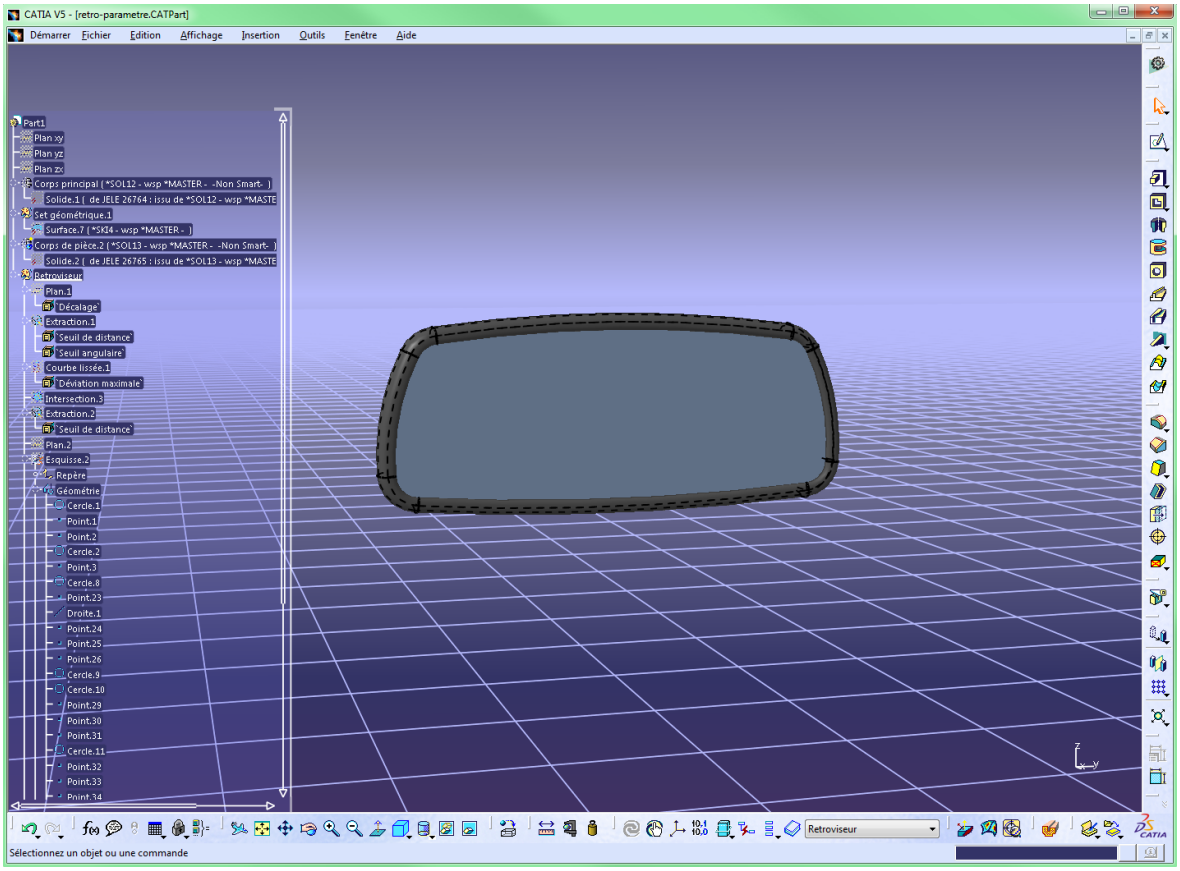


FIGURE 2.6 – Exemple de GHC dans CATIA V5. A droite l'objet, un rétroviseur ; à gauche, le graphe d'historique de construction de ce rétroviseur.

Alors que le CSG opérait uniquement sur des solides, les GHC vont maintenant permettre d'opérer aussi bien sur des géométries 2d et 3d, surfaces et solides. De plus, les opérateurs ne sont plus uniquement booléens, mais peuvent être, et c'est là l'une des forces de ce modèle,

des *form-features*, permettant une combinatoire plus riche (en particulier d'encapsuler des opérations topologiques dans des concepts de formes métier paramétrables).

Comme on peut le voir sur la figure 2.6, les GHC des logiciels CAO d'aujourd'hui sont présentés sous la forme de hiérarchie, ou graphe orienté acyclique. Toute les étapes de la conception — création d'éléments géométriques, opérations, etc. — sont ainsi répertoriées, permettant aux utilisateurs de pouvoir modifier l'objet final en modifiant uniquement certaines parties du graphe et en le rejouant. Enfin remarquons qu'à chaque opérateur (ou presque) d'un GHC est associé un B-Rep, permettant de visualiser le résultat à différents niveaux de l'historique, selon la place et les liens de l'opérateur en question.

2.2.7 Persistent Naming

Le problème du *persistent naming* [77,90,91], ou en français "nomination persistante", est un problème propre aux systèmes CAO basés sur un historique de conception et propre aux B-Rep. Les opérateurs, on l'a vu, agissent sur une partie ou la totalité des élements topologiques d'un solide donné. Or si l'on veut pouvoir modifier l'historique à n'importe quel niveau, le système doit pouvoir retrouver en toute circonstance les éléments sur lesquels l'opérateur en question agit. En effet, au fur et à mesure de l'historique et de l'application d'opérateurs, le B-Rep, résultat final du graphe, évolue. Des différences apparaissent alors entre l'ancien B-Rep et le nouveau tout juste généré : des éléments topologiques peuvent apparaître tandis que d'autres disparaîtront, certains resteront intacts pendant que d'autres seront modifiés.

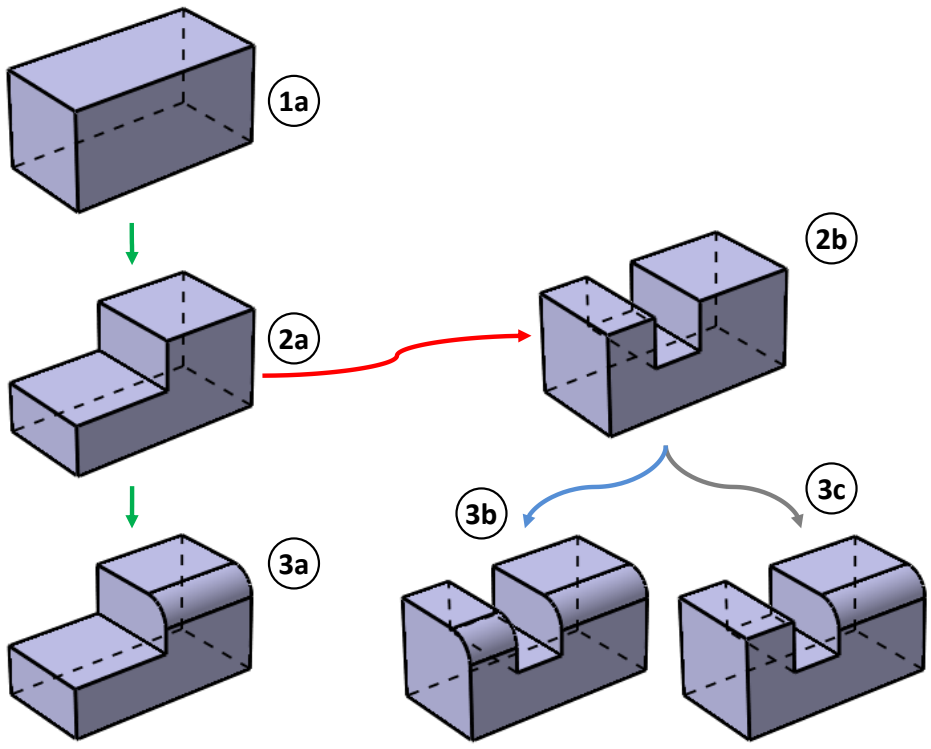


Figure 2.7 – Exemple du problème du *persistent naming*. Selon les systèmes CAO et leur mécanisme de *persistent naming*, la réévaluation du graphe d'historique après une modification (2a-2b) peut aboutir à des résultats différents. Ici CATIA donne le résultat (3b) tandis que PTC Creo donne le (3c).

Pour retrouver ces éléments topologiques à tout moment, les systèmes CAO possèdent un mécanisme, le *naming*. Ce mécanisme a pour but de fournir un identifiant unique pour chaque élément. Cet identifiant ne peut en aucun cas être une simple référence ou un pointeur (informatiquement parlant) sur cet élément, puisque l'existence de l'élément au cours de l'historique n'est pas garantie et ces références ou pointeurs pourraient être nuls. Généralement, ce mécanisme de *naming* va consister à identifier de façon unique les éléments topologiques selon leur connexité avec d'autres éléments, en particulier les faces. Parmi les principes fondamentaux de tels mécanismes, il y a l'idée qu'un élément n'ayant subi aucune modification et ayant gardé les mêmes liens de connexité conservera son identifiant. C'est ainsi qu'on parle de *persistent naming*. La littérature compte bon nombre d'approches et de tentatives de résolution de ce problème : Hoffmann et al. [40,41,77], Kripac [90–92], Cicirello [44], Raghothama et al. [136], Agbodan et al. [3], Marcheix et al. [108], Bidarra et al. [21–23], Wu et al. [168], Convard et al. [48,49], Mun et al. [123], Wang et al. [165], Baba-Ali et al. [11,12].

Ce problème du *persistent naming* est illustré par la figure 2.7. La première intention de construction, le chemin (a), part d'un solide extrudé (1a) auquel on applique une rainure (2a). Ensuite un congé est appliqué sur une des arêtes (3a). Mais que se passe-t-il si l'on modifie la définition de la rainure (2a) ? En guise de modification, le profil de la rainure est réduit — on passe d'un rectangle à un carré — ce qui donne un nouveau résultat (2b). La partie supérieure du solide initial est donc "coupée en deux" : la surface sur le dessus est désormais coupée en deux surfaces distinctes. Un problème va alors se poser lors de la réévaluation du graphe, et notamment lors de la réévaluation de l'opération de congé. En effet, le congé avait été appliqué sur une arrête (2a), mais dans le nouveau résultat (2b) on trouve deux arêtes distinctes, dont celle sur laquelle le congé a été initialement appliqué, ayant des liens de parenté et de connexité. Le résultat de la réévaluation complète peut différer d'un mécanisme de *naming* à un autre. En particulier, alors que le mécanisme présent dans CATIA V5 donnera comme résultat final le solide (3b), SolidWorks et Pro/Engineer — de même que sa version actuelle PTC Creo — donneront le solide (3c).

Dans ces travaux de thèse nous proposons une approche pour intégrer les fonctionnalités CAO existantes proposées par les logiciels du moment, notamment commerciaux, dans un environnement immersif. Ainsi nous ne proposerons pas de pistes pour mieux répondre à ce problème du *persistent naming*, mais plutôt pour utiliser ces mécanismes existants (voir section 3.2.1). Nous n'exposerons donc pas ici de comparaison entre les différentes approches, et nous invitons ainsi le lecteur intéressé à aller voir les travaux de Marcheix et Pierra [108], Convard [48], et Baba-Ali [11] qui proposent notamment une large vue sur cette question, ainsi que des pistes de reflexion quant à une meilleure résolution de ce problème.

2.2.8 Modification paramétrique et réactivité

Comme nous venons de le voir, la modélisation de solides s'appuie sur un historique de conception répertoriant les différentes étapes de conception. L'utilisateur a ainsi la possibilité d'éditer l'objet conçu en modifiant par exemple la valeur des paramètres de certaines caractéristiques de forme.

Les logiciels de modélisation sont destinés à un usage sur poste de travail, utilisant des interactions 2d classiques comme la souris et le clavier. La modification du graphe d'historique de l'objet peut très rapidement être fastidieuse lorsque l'arbre de conception devient complexe, et c'est très généralement le cas dans l'industrie. Il faut d'une part pouvoir retrouver l'opérateur ou le paramètre auquel on veut accéder puis passer par une suite de menus afin d'accéder à l'élément que l'on veut modifier.

La RV permet l'interaction 3d, naturelle. Avec les logiciels CAO, il apparait évident que cette modification paramétrique, se faisant via le parcours d'un arbre et des interactions sur menus, n'est pas adaptée à ces environnements virtuels. C'est pourquoi l'équipe VENISE du LIMSI-CNRS, au travers des travaux de Convard et Bourdot [32,50] a proposé le concept de réactivité des objets CAO (voir figure 2.8).

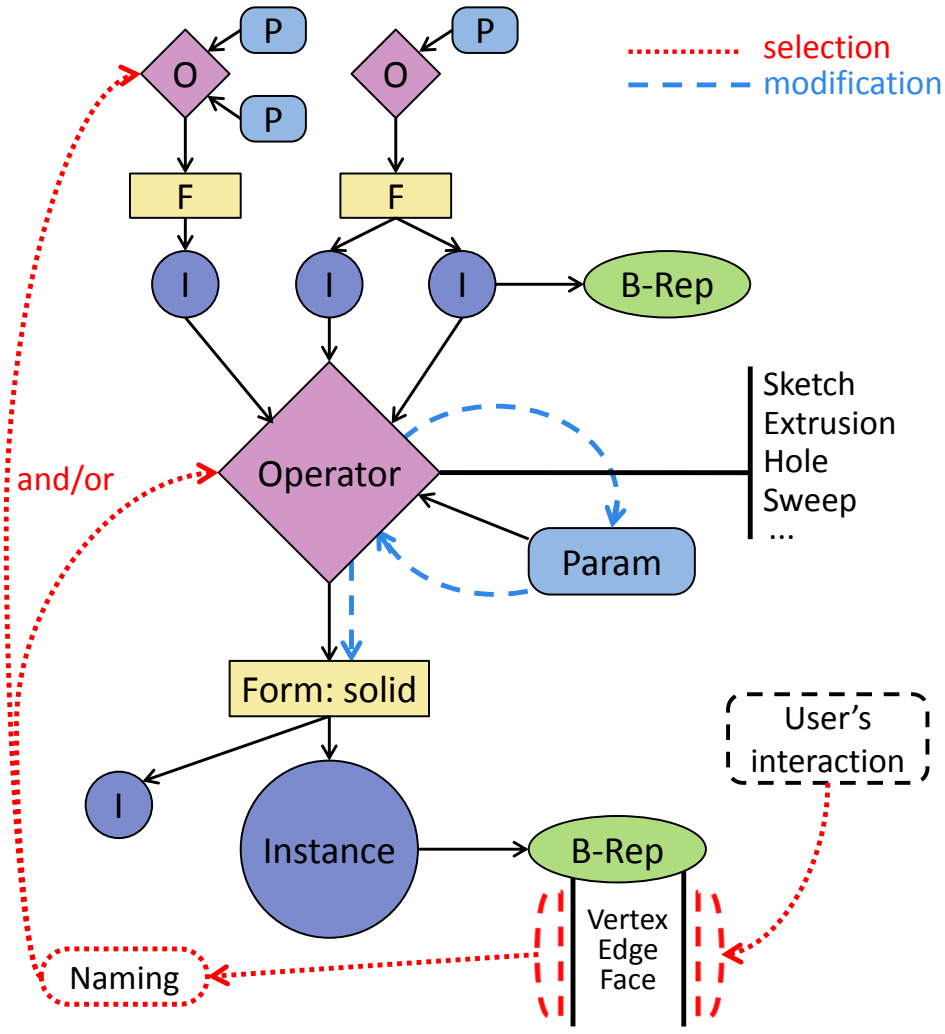


Figure 2.8 – Principe de la modification du GHC pour des objets CAO réactifs (Bourdot et al. [32]).

Ce concept de réactivité repose sur la gestion de modèles de représentation GHC et B-Rep, spécifiquement conçus à l'origine autour du noyau géométrique d'OpenCascade. Un mécanisme d'étiquetage (*labelling*) des éléments de B-Rep permet le lien direct entre ces éléments topologiques et l'historique de construction. Il faut donc bien voir la différence avec

le *persistent naming* qui, lui, constitue un chaînage entre les éléments de B-Rep successifs au sein d'un graphe d'historique.

Ainsi, l'utilisateur en accédant directement aux éléments de B-Rep, à travers leur représentation visuelle, peut modifier directement des éléments de l'historique. Cette approche sera concrétisée par le développement du démonstrateur VRAD (*Virtual Reality Aided Design*) présenté par la figure 2.9. Ce concept va se retrouver au cœur de nos travaux, et une description approfondie y sera consacrée au chapitre 3.

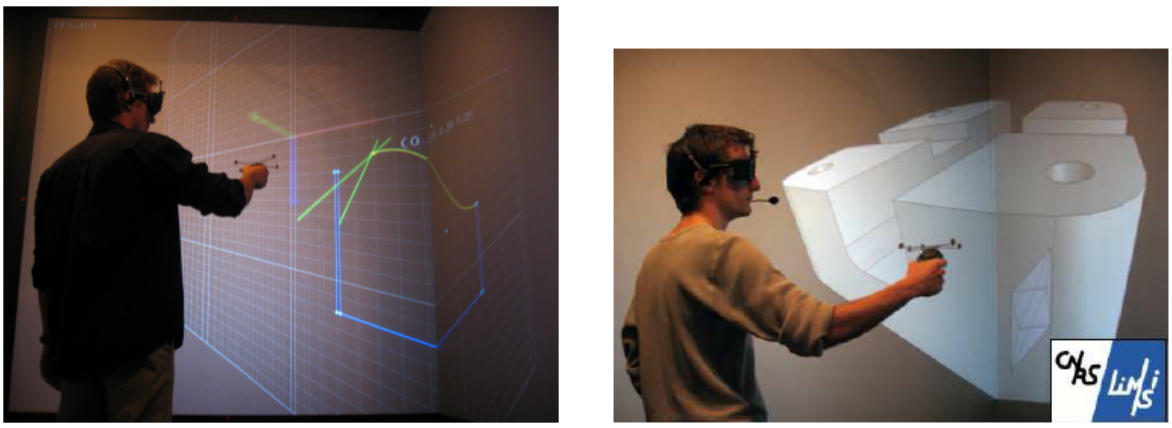


Figure 2.9 – Démonstrateur VRAD. Exemple d'édition implicite du graphe de conception d'objets CAO construit avec OpenCascade (Convard [48]).

2.3 Concepts de l'intégration RV-CAO

Nous l'avons vu dans le chapitre 1, plusieurs approches d'interfaçage RV-CAO ont été réalisées au cours des dernières années. Toutes ces approches présentent des caractéristiques soit différentes, soit communes, qu'il importe d'expliciter afin de préciser davantage le positionnement de mes travaux. Quels sont les liens entre les environnements de CAO et de RV ? Quels sont les liens avec le noyau géométrique du système CAO ? Nous allons ainsi dans cette section, décrire les différents couplages et liens RV-CAO existants.

2.3.1 Couplage indirect vs. couplage direct

L'interfaçage RV-CAO va permettre de visualiser, voire modifier, une maquette CAO dans un environnement virtuel. Dans l'environnement virtuel va être affichée ce que l'on appelle la **maquette virtuelle**, le logiciel CAO possédant lui la **maquette CAO**. Une des premières caractérisations d'un interfaçage RV-CAO est le lien entre ces deux maquettes : on parle de **couplage**, et on peut en distinguer deux types, direct et indirect (figure 2.10).

Couplage RV-CAO indirect

Dans un couplage indirect, il n'est possible que de simplement visualiser la maquette CAO par l'intermédiaire de la maquette virtuelle. L'utilisateur n'a pas la possibilité de modifier cette maquette CAO depuis la maquette virtuelle, d'où l'appellation de couplage indirect. Ainsi, lorsqu'aucune modification sur la maquette virtuelle ne sera possible, il s'agira de

spécifier des modifications à réaliser directement sur la maquette CAO puis de mettre à jour la maquette virtuelle pour les visualiser. Ceci étant, il est parfois possible de modifier la maquette virtuelle mais il sera nécessaire dans tous les cas de transférer ces modifications au logiciel CAO. Ce transfert se fait sous la forme de commandes ou de fichiers au format adapté. C'est le cas de l'approche de Meyrueis, Jezernik [85, 118].

Couplage RV-CAO direct

Lorsque les modifications de la maquette virtuelle permettent de modifier directement la maquette CAO, on parlera de couplage direct. Ces modifications directes sont rendues possibles par un lien entre les fonctionnalités RV et les fonctionnalités CAO.

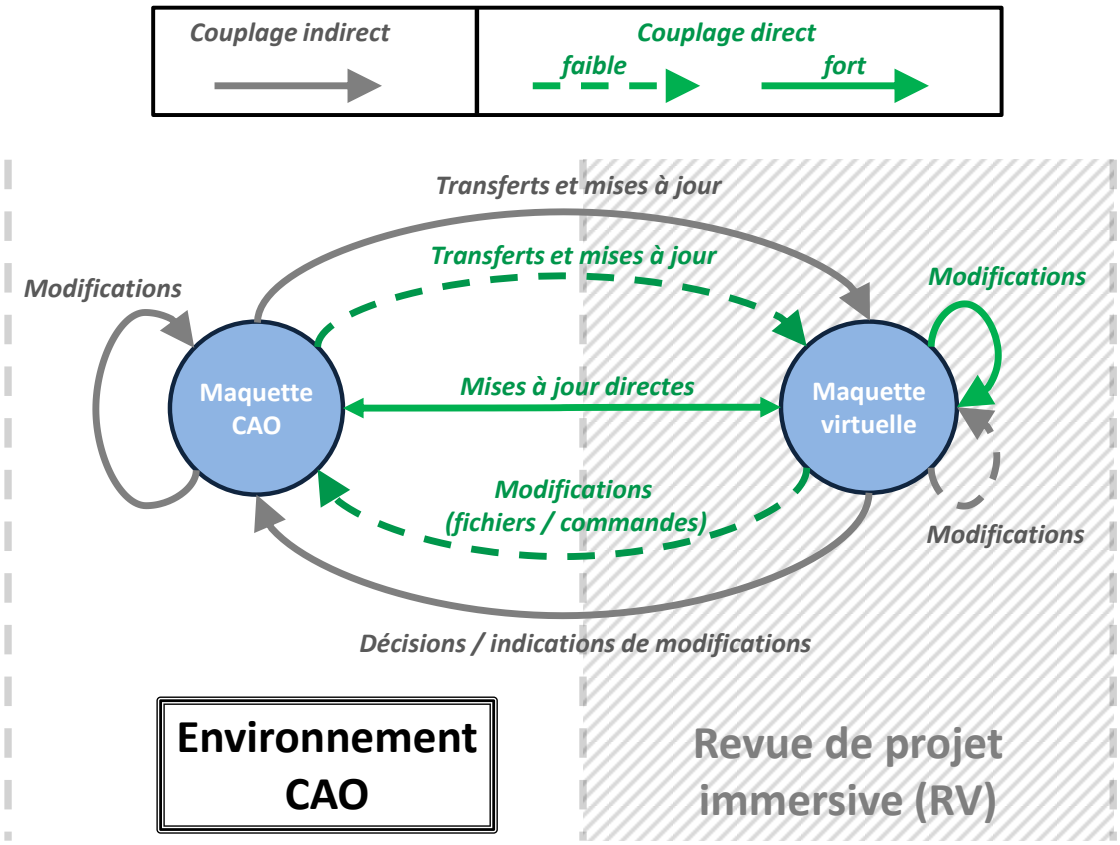


Figure 2.10 – Couplages RV-CAO. En gris le couplage indirect : généralement les modifications sont uniquement disponibles sur la maquette CAO, et nécessitent d'être transferées pour une mise à jour de la maquette virtuelle. Parfois des modifications de la maquette virtuelles sont possibles, mais donneront uniquement des indications des modifications à réaliser sur la maquette CAO. En vert le couplage direct : les modifications se font sur la maquette virtuelle et sont répercutées sur la maquette CAO. Soit directement lorsque le lien RV-CAO est fort (communication avec le noyau géométrique), soit indirectement par transfert de fichier ou de commande, lorsque le lien est faible.

2.3.2 Lien faible vs. lien fort

La possibilité de modifier la maquette CAO par la modification directe de la maquette virtuelle va elle aussi prendre plusieurs formes. Ces différences vont résider dans le type de lien entre l'environnement virtuel et l'environnement CAO. Comme le rappelle Meyrueis [118], Weidlich et al. [167] distinguent deux types de liens RV-CAO : lien avec un noyau ou lien avec un logiciel. Meyrueis va lui appeler ces liens respectivement "fort" ou "faible".

Lien RV-CAO fort

Ainsi un lien RV-CAO fort consiste à intégrer les fonctionnalités du noyau géométrique ciblé directement dans l'environnement de RV. Cela conduit alors à ce que certaines fonctionnalités RV appellent directement les fonctionnalités CAO. De plus, l'accès aux fonctionnalités CAO va permettre à la maquette virtuelle d'être une représentation directe de la maquette CAO, via la récupération des éléments géométriques et topologiques de cette dernière. Les environnements RV et CAO ont la possibilité de ne faire qu'un. Le modèle d'interfaçage RV-CAO présenté dans ce document se caractérise notamment par un lien RV-CAO fort.

Lien RV-CAO faible

D'un autre côté, un lien RV-CAO faible va consister à faire communiquer les environnements RV et CAO, mais de manière moins profonde et moins complète. Ces environnements vont rester distincts l'un de l'autre mais des moyens de communication, dans les deux sens, vont être mis en place. D'une part les interactions dans l'environnement RV vont devoir être traduites par des commandes à envoyer à l'environnement CAO. De l'autre, l'environnement CAO doit transférer la représentation visuelle à l'environnement RV.

Ce transfert d'éléments visuels va pouvoir se faire selon le principe de l'interception et du transfert de flux graphique de l'environnement CAO vers l'environnement RV. C'est sur ce principe que se basent des logiciels comme TechViz [158]. Mais ce transfert peut aussi s'effectuer par échange de fichier, plusieurs standards ayant été mis au point au cours de ces dernières années (3DXML, VRML, etc.), ce qui peut causer malgré tout des soucis dus à la conversion des données qui entraîne souvent une perte d'information sur la maquette.

2.4 Utilisation de la CAO dans le processus de conception et développement de l'industrie automobile

L'industrie automobile est parmi les industries les plus en pointe dans l'utilisation des technologies de RV et de CAO. Alors que la CAO est depuis nombre d'années utilisée dans le processus de conception et de développement, la RV a elle aussi pris une part très importante dans ces processus ces dernières années.

Comme nous l'avons vu, ce sont les premières phases de la conception qui présentent la plus grande criticité dans la bonne conduite de la suite du processus. En l'occurrence, dans le contexte automobile de PSA Peugeot Citroën, ce sont les étapes de l'**Avant projet** à la **Conception détaillée** qui vont nous intéresser. Nous allons donc dans cette section décrire certains métiers impliqués dans ces phases amont de la conception chez PSA Peugeot Citroën, avec leurs outils et leurs éventuels liens avec la RV.

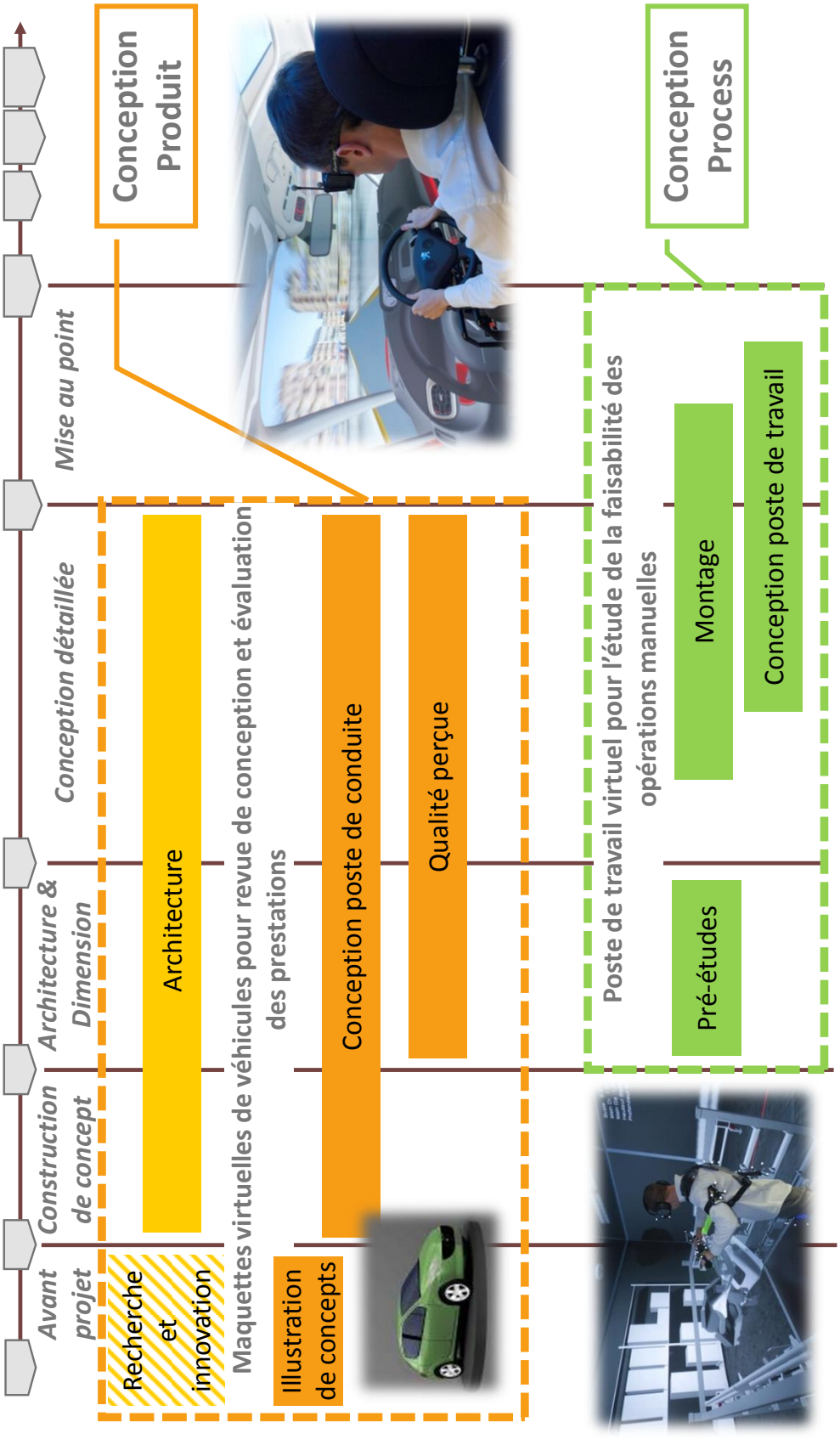


Figure 2.11 – Utilisation de la RV au cours du cycle de conception chez PSA Peugeot Citroën. L'utilisation de la RV dans la conception produit va permettre de se mettre à la place du client et d'évaluer l'esthétique, l'habitabilité, l'accessibilité, etc. La RV pour la conception process va, elle, consister à se mettre à la place de l'opérateur sur une chaine de montage.

2.4.1 Les métiers

Dans cette sous-section, nous présentons brièvement quelques métiers de la conception qui sont des cibles plus ou moins directes de nos travaux en RV-CAO.

Modeleur numérique

L'acteur appelé **modeleur numérique** va développer les différents modèles CAO, les concepts et styles validés, dans le respect des contraintes fournies. Il travaille en collaboration étroite avec le designer sur les travaux duquel il se base, afin que toutes les intentions initiales se retrouvent dans la modélisation 3d. En effet, le modeleur numérique est en charge de traduire en trois dimensions et dans le respect du cahier des charges technique, les intentions du designer que l'on retrouve sur les sketchs 2d.

Concepteur plan de forme

Le **concepteur plan de forme** est chargé de modéliser les surfaces visibles du véhicule. Il est l'interface privilégiée entre les designers, et les métiers de la technique. En partant des choix de design, puis en s'appuyant sur un cahier des charges et en tenant compte de contraintes données, il va fournir les données numériques du véhicule nécessaires pour la suite du projet et pour la fabrication des outillages.

Architecte véhicule

L'**architecte véhicule** s'occupe de coordonner les études relatives à la conception et au développement des véhicules. Son travail consiste à coordonner les travaux des différents métiers qui concourent à construire le véhicule dans son ensemble (extérieur et intérieur) et à assurer les prestations d'architecture (visibilité, accessibilité, habitabilité, poste de conduite, coffre) correspondant au contrat de prestations convenu. Concrètement, le métier d'architecture véhicule est en charge d'animer différentes séances de travail dont les objectifs vont être : 1) orienter une synthèse technique/style/prestation lorsqu'une situation problématique est détectée, ou 2) faire converger les métiers du style vers une définition compatible d'une synthèse technique/prestation du véhicule qui fonctionne.

En phase amont des projets, les architectes véhicules vont être amenés à construire des modèles génériques d'architecture afin d'évaluer au plus tôt différentes pistes. Lorsque le projet est véritablement lancé, les données issues des autres métiers vont être utilisées et analysées afin de détecter les éventuels problèmes.

2.4.2 Les outils CAO

Dans l'optique d'optimiser le processus de conception, il existe actuellement dans l'industrie une volonté de limiter le plus possible l'utilisation de logiciels différents pour tendre vers une seule et même suite logicielle tout au long du cycle de vie du produit. Ceci étant, la situation est telle qu'au cours des phases amont, plusieurs logiciels différents sont utilisés, chacun étant adapté au type d'activité (style, modélisation 3d, conception mécanique, etc.). Les différences entre les plateformes logicielles surfacique et paramétrique tendent à se réduire mais il existe encore une réelle séparation entre les deux.

Logiciels surfaciques

Les concepteurs plan de forme et les modeleurs numériques vont principalement utiliser les logiciels de conception surfacique. On va par exemple avoir les outils de la gamme Alias [2] comme **Alias Surface** ou **Alias Autostudio**. Ce sont des logiciels de design industriel et de modélisation de surfaces de classe A (surfaces de forme libre avec continuité des courbures et des tangentes permettant d'obtenir une esthétique de qualité).

Parmi les logiciels les plus utilisés on va aussi trouver **ICEM Surf** [3]. Ce logiciel permet également la modélisation de surface explicite, de classe A. De très bons effets pour le style sont disponibles de même que des outils de manipulation des surfaces.

Logiciels paramétriques

Parmi les logiciels de CAO paramétrique, un va particulièrement retenir notre attention, à savoir **CATIA**. Ce logiciel est une référence pour la conception de produit, permettant notamment de s'intégrer dans une solution PLM. L'industrie manufacturière, dont l'industrie automobile, est un gros client de cette solution logicielle, les fonctionnalités proposées permettant de couvrir nombre d'étapes du cycle de vie du produit. En particulier, c'est un outil très puissant pour la conception mécanique, permettant de modéliser des assemblages complexes. Ainsi, dans le processus de conception de PSA Peugeot Citroën, l'utilisation de maquette numérique CATIA apparaît rapidement, notamment pour les étapes de conception de l'architecture véhicule, ou pour la réalisation de simulations physiques et de calculs.

2.4.3 Activités de RV-CAO chez PSA

La RV trouve bon nombre d'applications dans l'industrie (revoir section 1.3.3) et PSA Peugeot Citroën est parmi les industries les plus avancées dans son utilisation. Globalement on va trouver deux types d'utilisation de la RV (figure 2.11) :

- pour la conception dite **produit** où le but est de se mettre à la place du client
- pour la conception dite **process** où l'on va se mettre à la place de l'opérateur sur la chaîne de montage.

La RV est ainsi devenue assez rapidement une extension des activités CAO et notamment un très bon complément aux maquettes physiques. Nous présentons ici quelques exemples d'utilisation de la RV en conception produit puisque c'est au sein de ces activités que la maquette CAO va être régulièrement modifiée.

Style

Les principales utilisations de la RV par les métiers du style vont être la comparaison de silhouettes pour la visualisation de volumes, l'évaluation de différentes versions d'agencement de planche de bord, ou bien le test de maquette d'habitabilité. Les maquettes numériques sont issues de logiciels surfaciques comme ICEM Surf, Alias ou DeltaGen. La principale motivation

2. http://www.autodesk.fr/products/autodesk-alias-products/overview
3. http://www.3ds.com/products-services/catia/capabilities/design/advanced-surface-modeling/icem-surf/

des ces différents métiers pour la RV est la possibilité de visualiser leurs différents travaux à l'échelle 1:1.

Architecture

La visualisation à l'échelle 1:1 est également une des motivations premières de l'utilisation de la RV pour les métiers de l'architecture véhicule. On va notamment trouver des revues de projet pour :

- l'agencement de planche de bord,
- les tests de ciel ouvert ou toit ouvrant,
- la comparaison de silhouette ou de volume extérieur,
- les tests d'habitabilité,
- les tests de visibilité (conducteur et passagers).

Les données utilisées lors des séances de revue de projet architecture sont issues de CATIA. Cependant, la revue de projet ne se fait pas forcément avec ce logiciel. En effet, selon certains critères comme la qualité de rendu dans l'environnement immersif ou la performance des interactions (notamment le cacher/afficher), d'autres logiciels comme Virtools peuvent être choisis pour la séance immersive. Ceci implique en particulier une conversion des données, parfois problématique, mais qui dans tous les cas rend impossible la modification des données natives pendant la séance.

Pour tous les types de séances évoqués (style et architecture), le cycle principal pour la revue de projet immersive est le suivant :

1. préparation de la maquette numérique pour la séance de travail,
2. envoi des données aux équipes de RV pour la préparation de la séance,
3. revue de projet immersive sans modification des données,
4. retour sur poste et application des modifications,
5. retour à 1.

Ainsi au vu de l'impossibilité de modifier les données de la maquette numérique dans l'environnement immersif, la préparation des données consiste à concevoir différentes hypothèses de la maquette — hypothèses correspondant à des versions aux paramétrages différents. Ainsi les séances de revue de projet immersives consistent généralement à cacher/afficher les différentes hypothèses pour les comparer.

2.5 Conclusion

Nous avons présenté dans ce chapitre les différents concepts et outils de CAO que l'on retrouve dans le contexte industriel, en l'occurrence celui de l'industrie automobile avec PSA Peugeot Citroën, et qui vont se retrouver dans mes travaux de recherche. Bien qu'il existe une volonté d'optimiser le processus de conception en limitant le plus possible l'utilisation de logiciels différents, et en tendant vers une seule et même suite logicielle, on voit malgré tout que quelques métiers conservent leurs propres outils. Les différences entre les plateformes

logicielles surfacique et paramétrique tendent à se réduire mais il existe encore une réelle séparation entre les deux. Ceci étant, les différents métiers de la conception sont concernés par l'utilisation de la RV, qui leur offre par exemple la possibilité de visualiser les formes et les volumes à l'échelle 1:1.

Pour permettre à ces différents logiciels d'être utilisés en environnement de RV, différents couplages RV-CAO sont utilisés, le plus souvent de type indirect : l'environnement RV permet une visualisation immersive, et le poste de travail à côté permet parfois la modification du modèle. Ces séances nécessitent une préparation des données afin d'optimiser les différentes mises à jour entre l'environnement RV et l'environnement CAO, mais aussi s'inscrivent dans une organisation du travail aux cycles plus lents.

Nos travaux se placent donc dans le cadre de l'optimisation de l'utilisation de ces environnements en proposant une approche de **couplage RV-CAO direct**, permettant la visualisation et la modification de la maquette numérique dans l'environnement virtuel, indépendamment de l'environnement CAO, grâce à un **lien fort** (taxinomie issue de plusieurs auteurs dont nous avons rappelé les définitions à la section 2.3 de ce chapitre).

Chapitre 3

Modèle d'interfaçage RV-CAO pour la revue de projet modificative

Résumé. *Nous décrivons dans ce chapitre, le modèle de données que nous avons mis au point dans le cadre de notre interfaçage RV-CAO pour la revue de projet modificative. Nous expliquons l'architecture de ce modèle ainsi que les différents processus en jeu dans la gestion des données. Le maniement des modèles de représentation de type B-Rep et GHC sera explicité, de même que notre exploitation du mécanisme de* persistent naming *des systèmes CAO cibles. Nous présentons également comment le principe de réactivité est articulé à ce modèle de données ainsi que les différentes évolutions que nous avons été amenés à réaliser. Enfin nous terminons par la visualisation et le rendu graphique de ces données dans les environnements virtuels.*

Sommaire

3.1 Introduction

Comme nous l'avons vu dans le chapitre 1, la combinaison des domaines de la CAO et de la RV offre de forts potentiels pour optimiser les processus de conception de produit. En particulier, les activités d'édition d'objets CAO en environnement immersif prennent tout leur sens dans les phases préliminaires de conception. Le monde industriel s'est fortement investi dans l'utilisation de ces technologies, et les revues de projet immersives sont devenues extrêmement courante, notamment chez PSA Peugeot Citroën.

Cependant, ces revues de projet immersives proposent une interaction encore trop limitée avec les objets CAO. Alors que de nombreux travaux ont porté sur la modification immersive d'objets CAO, et certains en particuliers avec des logiciels utilisés dans l'industrie, aucun n'a proposé un interfaçage optimal permettant l'édition.

Mes travaux de thèse vont principalement se baser sur l'approche de Convard et Bourdot [49] sur l'édition implicite du graphe d'historique de construction des objets CAO (voir chapitre 2, section 2.2.8). Alors que la faisabilité de cette approche avait été démontrée sur des représentations B-Rep et GHC créées pour l'occasion autour du noyau géométrique d'OpenCascade, le principal objectif de ma thèse a été de généraliser le modèle sous-jacent pour prendre en compte les B-Rep et GHC que l'on peut déjà trouver dans les logiciels de CAO paramétriques couramment utilisés dans l'industrie. Dans ce but, il nous fallait satisfaire plusieurs contraintes :

- l'utilisation optimale des données CAO entre les systèmes de RV et les systèmes CAO : pas de conversion de données entre les différents environnements
- la modification immersive des objets via un mécanisme d'édition implicite du GHC des sytèmes CAO cibles
- l'usage maximisé des fonctionnalités offertes par les logiciels CAO cibles, en particulier les mécanismes de *persistent naming*, ou ceux de mise à jours des différents modèles de représentation, etc.

Dans ce chapitre nous décrivons ainsi l'architecture que nous avons mise en place pour la modification immersive d'objets CAO. Cette architecture, illustrée par la figure 3.1, se base sur plusieurs composants, dont le modèle de données générique pour l'interfaçage de logiciels CAO avec les technologies de RV. Ce modèle permet notamment la gestion des modèles de représentation B-Rep et GHC. Notre architecture comprend également une composante de visualisation, permettant l'affichage des données CAO en environnement immersif, un moteur d'inférence permettant le lien direct entre les éléments topologiques et le GHC des objets, et un moteur d'interaction permettant de gérer les différents périphériques d'interaction.

3.2 Gestion des modèles GHC et B-Rep

Un des points clé de notre approche est de proposer un couplage direct entre l'environnement de RV et le système CAO. Parmi les obstacles les plus connus et importants dans l'utilisation de la CAO avec des technologies de RV au cours du processus de conception, on trouve la conversion de données. Cette conversion est problématique à plusieurs niveaux : la conversion peut parfois aboutir à de mauvais résultats (par exemple, les normales inversées) voire même être avortée du fait d'erreurs majeures. De plus, la transcription des données d'un format à l'autre n'est quasiment jamais bijective. La plupart du temps les informations sur la construction de l'objet CAO sont perdues, seuls les résultats géométriques sont conservés. Il

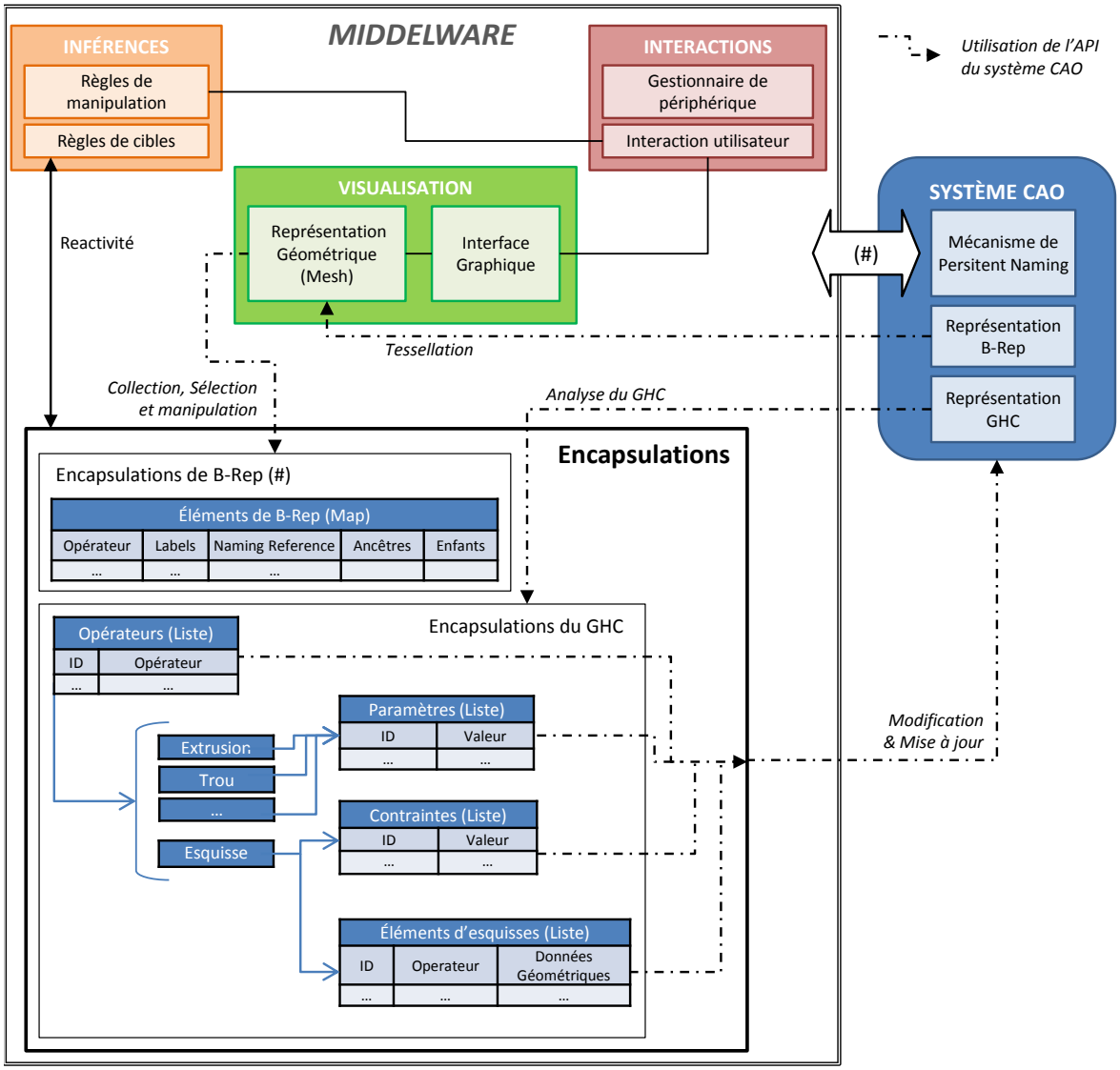


FIGURE 3.1 – Modèle de données pour la RV-CAO. Ce modèle comprend 4 composantes. La partie **inférence** en orange permet les liens, configurables, entre les éléments de B-Rep et des nœuds du GHC. La partie **interaction** en rouge, gère les actions de l'utilisateur au sein du monde virtuel. La partie **visualisation** en vert gère la représentation graphique de l'objet CAO, à partir de laquelle l'utilisateur va interagir. Enfin, la partie **encapsulation** en blanc, permet la gestion des données CAO (GHC + B-Rep) de l'objet que l'on manipule.

nous fallait donc pour nos travaux, absolument trouver un moyen de se passer de conversion de données, et donc de pouvoir gérer les données natives des logiciels cibles.

Nous avons ainsi conçu un modèle de données permettant la gestion des modèles de représentation B-Rep et GHC que l'on peut trouver dans les logiciels de CAO paramétriques. Ce modèle va s'appuyer sur différentes structures de données, que l'on décrit dans cette section, permettant d'encaspuler tout ce qui sera nécessaire à la gestion de l'objet CAO en environnement immersif.

3.2.1 Point clé : le *Persistent Naming*

Comme nous l'avons vu précédemment (chapitre 2, section 2.2.7), le mécanisme de *persistent naming* est un point clé des systèmes CAO basés sur les représentations B-Rep et GHC. Son intégration dans notre modèle de données constitue un point important : ce mécanisme permet en effet d'accéder à n'importe quel élément de B-Rep à n'importe quel moment. Cependant, le niveau de maîtrise de ce mécanisme repris pour notre approche doit être précisé. En effet, autant le système CAO cible doit pouvoir nous fournir des moyens d'y accéder pour que nous puissions l'utiliser, autant notre approche ne requiert aucune connaissance quant au fonctionnement exact de ce mécanisme au sein du système CAO (que ce soit en termes de grammaire de nommage ou que ce soit en termes de comportements lors de la réévaluation du GHC).

En fait, dans la précédente approche d'édition implicite du GHC de Convard et Bourdot [49,50], le mécanisme de *persistent naming* était indispensable, et ce notamment pour la mise à jour du GHC dont certains opérateurs font référence à des éléments de B-Rep pouvant ne plus exister dans le résultat final. Dans notre cas, ces mises à jour des B-Rep et du GHC sont réalisées par le système CAO et nous ne faisons que les activer. Nous n'avons donc pas les mêmes dépendances sur ce point là, en particulier parce que l'une de nos contraintes est de respecter le format et les principales fonctionnalités du système CAO cible. Néanmoins, pouvoir utiliser les informations produites par ce mécanisme va nous servir à accéder à tout instant à n'importe quel élément de B-Rep. En particulier, lorsqu'un opérateur et donc le GHC sont mis à jour, même si aucun élémént de B-Rep n'est apparu ni n'a disparu, et que donc le nommage des éléments n'a pas changé, l'accès aux éléments de B-Rep ne peut se passer des informations produites par le *persistent naming*. Nous y reviendrons dans la section 3.2.3.3.

3.2.2 Structures d'encapsulation

Dans cette sous-section nous décrivons les différentes structures de données qui vont servir à l'encapsulation et à la gestion des éléments de B-Rep et du GHC. Nous avons ici conçu cinq types d'encapsulation afin de gérer les éléments de B-Rep, les opérateurs du GHC, les paramètres des opérateurs, les contraintes des esquisses, et les éléments géométriques 2d d'esquisse.

3.2.2.1 Éléments de B-Rep

En ayant en mémoire le problème du *persistent naming*, il est facile de comprendre que l'encaspulation des éléments de B-Rep ne peut pas reposer sur l'encapsulation d'une référence ou d'un pointeur — informatiquement parlant — sur cet élément. Il faut en effet bien voir que suite à une opération, de création ou de modification, certains éléments topologiques peuvent disparaître, et leur référence/pointeur devenir **NULL**. Ainsi, l'idée principale est de

stocker les informations de *persistent naming* des éléments de B-Rep dans leurs encapsulations respectives.

Comme nous l'avons dit, notre modèle de données ne requiert aucune connaissance sur le fonctionnement même du mécanisme de *persistent naming* en question, et la démarche consiste "simplement" à intégrer ce type d'information comme une boîte noire au sein de nos structures d'encapsulation. Bien évidemement, une des conditions nécessaire au bon fonctionnement de notre démarche d'interfaçage RV-CAO, est de pouvoir accéder à ces différentes informations. Ceci est notamment rendu possible par certains systèmes CAO disposant d'une *Application Programming Interface*, communément appelée API, et ouvrant leurs codes sources aux développeurs. Procédant ainsi, il nous suffit alors d'utiliser ces informations de *persistent naming*, nous permettant notamment d'accéder à tout élément de B-Rep à n'importe quel moment, sans avoir à s'occuper de son fonctionnement et de sa gestion internes. Il est certain que l'utilisation de ces données de *persistent naming* variera d'un système CAO à l'autre.

Listing 3.1 – Structure d'encapsulation des éléments de B-Rep

```
class myBRepElement{
  // Données
  int m_Type; //Vertex, Edge or Face
  BRepElementObject *m_BRepElement; //Le type BRepElementObject dépend du
      système CAO utilisé
  myOperator * m_Operator;
  std::list<myTopologyLabel*> m_Labels;
  std::list<myBRepElement*> m_Ancestors;
  std::list<myBRepElement*> m_Children;
  NamingInformation* m_NamingInformation;  //Le type NamingInformation
      dépend du système CAO utilisé
  GeometricalElement * m_GeometricalElement;  //Le type GeometricalElement
      dépend du système CAO utilisé

  // Fonctions
  bool label(); // Etiquetage de l'élément
  void findAncestors(); // Recherche des ancêtres
  void dispatchLabels(); // Transmission de l'étiquetage au éléments fils
  void graphicalRepresentation(); // Création de la représentation
      graphique au format adapté au système de visualisation

}
class myBRepVertexElement: public myBRepElement {...}
class myBRepEdgeElement: public myBRepElement {...}
class myBRepFaceElement: public myBRepElement {...}
```

Au final, la structure d'encapsulation des éléments de B-Rep, décrite par le Listing 3.1, comprend :

- le type de l'élément de B-Rep
- un pointeur sur l'interface d'accès à l'élément de B-Rep
- un pointeur sur l'opérateur générateur de l'élément
- une liste de *labels* (étiquettes)
- deux listes de pointeurs sur les structures d'encapsulation des éléments de B-Rep respectivement ancêtres et fils

- un pointeur sur la structure contenant les informations du *persistent naming*
- un pointeur sur le B-Rep complet
- les différentes fonctions permettant d'étiqueter l'élément, de rechercher son historique, de construire sa représentation graphique

Les listes d'ancêtres et de fils vont servir à notre mécanisme de *labelling* (étiquetage) des éléments de B-Reps (voir section 3.3.1), qu'il ne faut pas confondre avec le mécanisme de *persistent naming* précédemment décrit. La classe de base `myBRepElement` va être dérivée en trois classes `myBRepVertexElement`, `myBRepEdgeElement`, `myBRepFaceElement`. Ces trois classes vont nous permettent comme leur nom l'indique, de gérer respectivement les sommets, arêtes et faces des objets CAO. En particulier, le comportement de certaines fonctions va varier selon le type de l'élément topologique, comme justement le mécanisme d'étiquetage.

Autant nous avont attiré l'attention sur le fait que les pointeurs/références des éléments topologiques ne pouvaient pas être utilisés comme base de notre encapsulation, autant il faut préciser que les pointeurs/références des opérateurs ne posent strictement aucun problème. Par définition, les éléments topologiques vont être liés à l'opérateur qui est à l'origine de leur génération. L'existence des ces éléments est donc liée à l'exsistence même de l'opérateur en question. De fait, lorsqu'un élément de B-Rep existe — ou est en mémoire lorsqu'il n'est plus à jour — le pointeur vers l'opérateur générateur de cet élément est toujours valide. Si une modification du GHC amenait à la suppression d'un opérateur, alors tous les éléments de B-Rep associés seraient eux aussi supprimés.

3.2.2.2 Opérateurs

La gestion des opérateurs du GHC est clairement plus facile que la gestion des éléments de B-Rep. Leur encapsulation en devient naturellement moins complexe, comme on peut le voir sur le Listing 3.2. Cette facilité tient au fait qu'il n'y a pas de problème de persistence des éléments comme explicité ci-dessus. Un opérateur, tant qu'il existe est toujours accessible. Ainsi, la structure d'encapsulation d'un opérateur, est *a minima*, la suivante :

- un identifiant unique
- un type définissant la nature de l'opérateur
- un pointeur sur l'opérateur
- une liste de pointeurs sur les structures d'encapsulations des paramètres de l'opérateur
- les fonctions permettant les mises à jour des différents paramètres et de l'opérateur lui-même

Cette structure, comme nous le voyons, est plutôt simple. L'identifiant unique, qui est dans notre cas un entier, est tiré d'un compteur incrémenté à chaque création de nouvel opérateur. Ainsi l'opérateur le plus ancien aura l'identifiant le plus petit, et le dernier opérateur créé aura l'identifiant le plus grand. La classe de base `myOperator`, va être dérivée selon le type de l'opérateur. Ainsi, par exemple, les encapsulations d'une extrusion et d'un trou seront réalisées respectivement au travers des classes `myOperatorPad` et `myOperatorHole`. Chaque type d'opérateur aura ses paramètres encapsulés (`myParameter`, section 3.2.2.3) et regroupés dans une liste qui lui est propre.

De la même façon, l'opérateur d'esquisse, sera encapsulé via la classe `myOperatorSketch`. Cette classe va légèrement différer des autres dans sa structure puisqu'elle contient des données propres à l'esquisse, que les autres n'ont pas. En l'occurrence, cette classe comprend

deux listes : une liste de pointeurs sur les encapsulations des éléments géométriques 2d de l'esquisse (`mySketchElement`, section 3.2.2.5), et une liste de pointeurs sur les encapsulations des contraintes de l'esquisse (`myConstraint`, section 3.2.2.4).

```
class myOperator{
  // Données
  int m_ID;
  int m_Type; //Extrusion, Trou, Esquisse, Balayage, etc..
  OperatorObject * m_Operator; //Le type OperatorObject dépend du système
     CAO utilisé
  std::list<myParameter*> m_Parameters;
  // Ajouté à cela d'autres données dépendantes du type de l'opérateur (
     voir exemple de l'esquisse ci-dessous)

  // Fonctions
  void update(int paramID, Value v); // Mise à jour du paramètre paramID
     avec la Value v
  void globalUpdate(); // Mise à jour global de l'opérateur et de ses
     paramètres
}
class myOperatorSketch: public myOperator{
  // Données
  std::list<mySketchElement*> m_Elements;
  std::list<myConstraint*> m_Constraints;

  // Fonctions
  void updateProfile(mySketchElement el); // Mise à jour de l'esquisse en
     fonction de l'élément modifi "el".
  void updateConstraint(int constraintID, Value v); // Mise à jour de la
     contrainte constraintID avec la Value v
}
class myOperatorPad: public myOperator {...}
class myOperatorHole: public myOperator {...}
...
```

Listing 3.2 – Structure d'encapsulation des opérateurs. Exemple de la structure propre aux opérateurs d'esquisse et autres.

Comme pour les différentes classes dérivées pour l'encapsulation des éléments de B-Rep, les différentes classes dérivées pour l'encapsulation des opérateurs vont présenter des nuances dans le comportement de certaines fonctions, notamment lors des procédures de mise à jour. Ainsi les mises à jour et la gestion des paramètres et contraintes, est directement contrôlée et assurée par les encapsulations concernées.

3.2.2.3 Paramètres

Les paramètres représentent les principaux éléments du GHC auxquels les utilisateurs veulent accéder lors de la modification d'objets CAO. Ces paramètres vont être par exemple la profondeur d'un trou, la hauteur d'une extrusion, l'angle d'une révolution, ou l'épaisseur intérieure d'une coque. L'approche de l'édition implicite du GHC va précisément établir des liens entre les éléments topologiques que sont les entités de B-Rep, visualisés par leur représentation *mesh*, et ces paramètres des opérateurs du GHC. Ceci est indispensable puisque

rappelons le, le GHC ne sera pas (et ne doit pas) être visible lors des sessions de travail immersives, ce pour des raisons pratiques : le GHC peut être énorme, et sa modification via son parcours peut vite être pénible (ouverture de widgets) et fastidieuse (profondeur du graphe).

La structure d'encapsulation des paramètres (`myParameter`) n'est pas beaucoup plus compliquée, comme on peut le voir dans le Listing 3.3 :

- un identifiant unique
- un type définissant la nature du paramètre (un entier, un double, un vecteur, etc.)
- le nom du paramètre
- la valeur du paramètre (dont le type dépend de la nature du paramètre)
- les fonctions pour récupérer et mettre à jour sa valeur

Cette classe de base va aussi être dérivée selon le type de paramètre, permettant ainsi un comportement spécifique pour chacun.

```
class myParameter{
  // Données
  int m_ID;
  int m_Type; //Le type de paramètre (entier, double, vecteur, etc.)
  char *name;
  myValue m_Value; //Le type myValue varie selon le type de paramètre
      auquel on a affaire

  // Fonctions
  myValue getValue(); // Récupération de la valeur du paramètre
  void setValue(myValue v); // Mise à jour de la valeur du paramètre
}
class myParameterDouble{ double m_Value; //myValue }
class myParameterVector{ vector m_Value; //myValue }
```

Listing 3.3 – Structure d'encapsulation des paramètres

3.2.2.4 Contraintes

Les contraintes peuvent être considérées comme des paramètres du GHC, à la nuance près qu'elles sont uniquement associées aux esquisses. Nous avons ainsi conçu une structure d'encapsulation spécifique (`myConstraint`), différente de la structure d'encapsulation des paramètres, et qui, comme le montre le Listing 3.4, contient :

- un identifiant unique
- un type définissant la nature de la contrainte (un angle, un rayon, une distance, etc.)
- un pointeur sur la contrainte
- le nom de la contrainte
- la valeur de la contrainte
- la liste d'éléments d'esquisses liés à cette contrainte
- les fonctions pour récupérer et mettre à jour sa valeur

```
class myConstraint{
  // Données
  int m_ID;
  int m_Type; // Type de contrainte (angle, rayon, distance, etc.)
  ConstraintObject *m_Constraint; // Le type ConstraintObject dépend du
      système CAO utilisé
  char *name;
  double m_Value;
  vector<char*> m_Elements; // Liste de noms des éléments liés à cette
      contrainte

  // Fonction
  bool isModifiable(); // La contrainte est-elle modifiable ?
  void setValue(double v); // Mise à jour de la contrainte
  double getValue(); // Récupération de la valeur de la contrainte
}
```

Listing 3.4 – Structure d'encapsulation des contraintes

On voit ainsi que cette encaspulation ressemble à celle des paramètres. Mais c'est notamment ses fonctions et son comportement qui va différer. Dans son état actuel, seules les contraintes dites "numériques" — un angle, un rayon, une distance, etc. — sont prises en compte, étant les principales contraintes devant être modifiées. Ainsi ne sont pas gérées les contraintes "symboliques", à savoir les relations géométriques comme la perpendicularité, le parallélisme ou le contact. Il faut cependant voir que la modification de telles contraintes est un peu plus complexe que la modification des contraintes numériques qui consiste simplement à changer leur valeur. Mais notre modèle, tel qu'il est conçu, pourrait donc s'étendre à terme aux contraintes "symboliques" aussi. De la même façon que les paramètres (leur modification et leur mise à jour) est contrôlée par les opérateurs, les contraintes sont entièrement gérées par les esquisses.

3.2.2.5 Éléments d'esquisse

Les éléments géométriques 2d représentent tous les éléments qui composent l'esquisse comme les points, les plans, les lignes et courbes, ou les cercles. Les esquisses sont massivement utilisées dans les systèmes de modélisation paramétriques basés sur des caractéristiques (*feature-based*), et en particulier dans l'industrie. Il est de fait évident qu'un système de modification d'objets CAO puisse permettre de modifier ces composants d'esquisse. Comme on peut le voir dans le Listing 3.5, nous avons ainsi conçu une structure d'encapsulation pour ces éléments d'esquisse 2d (`mySketchElement`) contenant :

- un identifiant unique
- un type définissant la nature de l'élément géométrique (point, ligne, cercle, etc.)
- une liste de noms des contraintes associées à cet élémént
- un pointeur sur l'élément d'esquisse
- un pointeur sur l'encapsulation de l'esquisse contenant cet élément
- les fonctions de mise à jour des données géométriques

```
class mySketchElement{
  // Données
  int m_ID;
  int m_Type; //Point, Line, Circle, etc.
  list<char *> m_NConstraints;
  Element *m_SketchElement; //Le type Element dépend du système CAO utilisé
  myOperator *m_Sketch;

  // Fonctions
  void update(); // Mise à jour des données géométriques (coordonnées)
}
class mySketchPoint: public mySketchElement{
  double *m_Coords; //3D coordinates
  double *m_2Coords; //2D coordinates
}
class mySketchCircle: public mySketchElement{
  myPoint m_Origine;
  myVector m_Normal;
  double m_Radius;
}
class mySketchLine: public mySketchElement {...}
class mySketchCurve: public mySketchElement {...}
class mySketchPlane: public mySketchElement {...}
...
```

Listing 3.5 – Structure d'encapsulation des composants d'esquisse. Exemple de l'encapsulation des Points et des Cercles.

Seuls les noms des contraintes sont stockés dans cette structure pour éviter la redondance et donc l'alourdissement du modèle, les encapsulation des contraintes étant déjà stockées dans l'opérateur d'esquisse. La classe de base `mySketchElement` va être dérivée selon le type de l'élément 2d. Une ligne sera encapsulée par la classe `mySketchLine`, un cercle par la classe `mySketchCircle`, et ainsi de suite. Chaque classe va alors contenir des informations spécifiques, les éléments géométriques n'étant pas caractérisés par les mêmes données. Ainsi par exemple, un point va être défini par des coordonnées 2d (dans le plan de l'esquisse) et 3d (dans le répère monde) ; un cercle par son centre (un point), un vecteur normal et un rayon.

3.2.3 Parcours et analyse du GHC

Il a été observé, notamment par Toma et al. [159], que les designers et ingénieurs préfèrent créer les objets CAO dans un environnement de type "poste de travail" classique, plutôt que par l'intermédiaires des technologies de RV dans des environnements virtuels. Plus exactement, les utilisateurs bien qu'ayant mis à peu près le même temps pour compléter leur tâche dans les deux environnements, l'interaction naturelle (gestes) faisait parcourir davantage de distance et stressait d'autant plus les muscles. Rajoutons également que la précision des interactions classiques 2d est difficilement atteignables avec les interactions 3d mains libres.

C'est en se basant sur cette observation, à savoir que c'est plus la modification de formes que leur pure création qui doit pouvoir être menée en environnement immersif, que nous avons considéré le parcours et l'analyse des Graphes d'Historique de Construction (GHC) des objets CAO comme une étape nécessairement indispensable à notre approche. Ceci est rendu possible, comme nous l'avons vu plus haut, lorsque les fonctionnalités du système CAO ciblé sont accessibles, via son API. Le but de ce processus de parcours et d'analyse du GHC est

alors de récupérer et stocker, au sein de nos structures d'encapsulation, toutes les données de l'objet CAO qui vont être utiles à sa modification.

3.2.3.1 Récupération des données

La récupération des données au cours du processus de parcours et d'analyse du GHC de l'objet CAO (voir Algorithme 1) se décompose en trois étapes. La première consiste à parcourir le GHC et à récupérer tous les opérateurs (les esquisses étant mises de côté) et leurs paramètres. Durant la deuxième étape, ce sont les esquisses et leurs éléments qui sont récupérés. Enfin la troisème et dernière étape va collecter et analyser les éléments topologiques du B-Rep. Ce processus de parcours et d'analyse du GHC est réalisé au début de la session de travail, puis est relancé chaque fois que le GHC de l'objet CAO est modifié. Nous détaillons le processus de mise à jour dans la section 3.2.3.3.

Étape 1 : Opérateurs et paramètres

La première étape pour la récupération des données du CHG est celle visant des opérateurs, nœuds importants puisque contenant des paramètres, et qui sont associés (la plupart du temps) à un B-Rep. D'une manière générale, tous les opérateurs n'ont pas à être analysés. Tous ne sont pas "intéressants" pour l'utilisateur. Lors de séances de travail classiques, seuls certains paramètres vont être mis en avant et modifiés. Cette définition de ce qui est important ou pas va être réalisée au travers d'une configuration — dont la forme actuelle est un fichier XML — déterminée par les utilisateurs en fonction de leurs besoins. Ainsi, pendant le processus de parcours et d'analyse du GHC, pour chaque opérateur va être vérifié s'il a été préalablement désigné comme "important", et si oui, il sera analysé, ses données seront récupérées puis finalement stockées dans nos structures d'encapsulation (voir section 3.2.2.2).

Parmi les données incontournables des opérateurs, on trouve bien sûr leurs paramètres. Les paramètres qui vont être définis par l'utilisateur comme l'objet de la séance de travail vont ainsi être encapsulés.

Cette étape aboutit à la constitution de plusieurs listes d'encapsulation. La première, qui appartient à la structure de données globale (voir Listing 3.6), est la liste des encapsulations des opérateurs : `list<myOperator*>`. Les autres listes sont les listes d'encapsulations des paramètres de chaque opérateurs — une liste par opérateur — faisant donc parti de la structure de données propre à chaque opérateur : `list<myParameter*>`.

Étape 2 : Esquisses, éléments géométriques 2d et contraintes

Cette étape consiste à récupérer et analyser les esquisses de l'objet CAO. Les objets CAO conçus dans des systèmes CAO paramétriques le sont souvent, voire principalement, à partir d'esquisses que l'on peut considérer comme des extensions aux primitives géométriques simples. Selon le système CAO, les éléments géométriques 2d qui composent une esquisse vont être soit agrégés et contenus au sein d'un opérateur esquisse, soit complètement décrits dans l'arbre en dehors de toute entité structurante. Ainsi, alors que CATIA va disposer d'opérateurs *Sketches* comprenant tous les éléments liés aux esquisses (éléments géométriques, contraintes, etc.), OpenCascade lui ne dispose pas de telles structures. Le traitement de ces éléments d'esquisse va donc naturellement différer d'un système à l'autre.

Dans les cas où les esquisses sont représentées par des opérateurs, le principe consiste donc à récupérer d'abord ces entités, à les encapsuler, puis à parcourir leurs données et

Algorithme 1 Parcours et analyse du GHC et encapsulation des données

```
Collecter les opérateurs
Pour chaque Opérateur Faire
    ▷ Configuré = préalablement défini/requis par les utilisateurs
    Si Configuré Alors
        Fonction EncaspulateOperator
            Récupérér et stocker les données de l'opérateur
            Collecter les paramètres de l'opérateur
            Pour chaque Paramètre Faire
                Si Configuré Alors
                    Fonction ParameterEncapsulation
                        Récupérer et stocker les données du paramètre
                    Fin Fonction
                Fin Si
            Fin Pour
        Fin Fonction
    Fin Si
Fin Pour

Collecter les esquisses
Pour chaque Esquisse Faire
    Fonction SketchEncapsulation
        Récupérer les données de l'esquisse
        Collecter les éléments géométriques 2d
        Pour chaque Élément 2d Faire
            Fonction 2DElementEncaspulation
                Récupérer et stocker les données de l'élément 2d
            Fin Fonction
        Fin Pour
        Collecter les contraintes de l'esquisse
        Pour chaque Contrainte Faire
            Si Configurée Alors
                Fonction ConstraintEncapsulation
                    Récupérer et stocker les données de la contrainte
                Fin Fonction
            Fin Si
        Fin Pour
    Fin Fonction
Fin Pour

Collecter les éléments topologiques
Pour chaque Élément topologique Faire
    Fonction EncapsulateBRepElement
        Récupérer les données de l'élément de B-Rep
        Étiqueter l'élément
        Collecter les éléments ancêtres et récupérer leurs étiquettes
    Fin Fonction
Fin Pour
```

traiter les éléments géométriques, de la même façon que l'on traite les opérateurs et leurs paramètres. Lorsque ces éléments d'esquisses ne sont pas intégrés à une entité structurante de type opérateur, il faut aller les récupérer directement dans le GHC.

Cette étape aboutit alors à la constitution d'une liste de pointeurs sur les structures d'encapsulation des éléments 2d : `list<mySketchElement*>`.

Étape 3 : Éléments de B-Rep

La dernière étape du processus de parcours et d'analyse du GHC va concerner les éléments de B-Rep. Le traitement de ces données topologiques va être plus sophistiqué que le reste des autres traitements du GHC. En effet, bien que l'encapsulation des éléments de B-Rep est plutôt simple dans sa structure (voir section 3.2.2.1), les processus de récupération des données et de remplissage des structures sont complexes. Le traitement d'un élémént de B-Rep va se décomposer en trois phases :

- Récupération des informations sur le *persistent naming* de l'élément
- Étiquetage de l'élément
- Analyse des ancêtres

La première phase de ce processus consiste donc à récupérer les informations sur le *persistent naming* de l'élément, point clé de notre approche, et de les intégrer à la structure d'encapsulation. Ici la complexité réside dans l'accès même à ces informations, à savoir trouver le bon moyen, la/les bonne(s) fonction(s) à utiliser parmi les fonctionnalités offertes par l'API du système CAO cible.

La deuxième phase va ensuite être le processus d'étiquetage de l'élément de B-Rep, dit *labelling*, que l'on décrira dans la section 3.3.1. Rappelons que ce *labelling* permet de mettre en relation directe (chaînage) les éléments de B-Rep avec des opérateurs du GHC, contrairement au *persistent naming* qui lui met en relation différentes versions de B-Rep.

Enfin durant la troisième et dernière phase, ce sont les informations des ancêtres de l'élément qui vont être récupérées. L'idée est dans un premier temps accéder à ces différents parents impliqués dans la construction de l'élément de B-Rep en cours, puis de récupérer les informations qui nous sont nécessaires, en l'occurrence, leurs informations de *labelling*. Ce traitement va nous permettre de faire descendre les différentes informations de l'historique de l'élément de B-Rep dans son encapsulation. Il sera alors possible, depuis un seul élément visible, d'accéder à différents endroits du GHC. C'est ce que nous appelons le *multi-labelling* (voir section 3.3.1.2).

Le résultat final de ce traitement des éléments de B-Rep est la constitution de trois structures de données de type `map` — une map permet d'associer une clé (identifiant) à une donnée (table associative) — contenant respectivement les pointeurs sur les encapsulations des faces, arêtes et sommets des B-Rep (voir les structures _BRepFaceElements, _BRepEdgeElements et _BRepVertexElements dans le Listing 3.6).

```
class myApp{
  // ...

  // Gestion des éléments de B-Rep
  map<char*, map<char*,myBRepElement*>> _BRepFaceElements;
  map<char*, map<char*,myBRepElement*>> _BRepEdgeElements;
  map<char*, map<char*,myBRepElement*>> _BRepVertexElements;
  // Le premier char* représente le "nom générique" (unique) de l'opérateur
  // Le deuxième char* représente la clé des éléments de B-Rep, dépendante
      du système CAO utilisé.

  // Gestion des opérateurs
  list<myOperator*> _operators;

  // Gestion des éléments d'esquisse
  list<mySketchElement*> _sketchelements;

  // Gestion de la visualisation
  map<BGMesh*,myBRepElement*> _visuBRepMesh;
  map<BGPolyLine*,myBRepElement*> _visuBRepPolyLine;
  map<BGPointCloud*,myBRepElement*> _visuBRepPointCloud;

  // Composants
  inferenceEngine *_inferenceEngine; // Moteur d'inférence
  interactionEngine *_interactionEngine; // Moteur d'intéraction
  graphicDrs * _graphicDrs; // Système de visualisation

  // ...
}
```

Listing 3.6 – Structure globale de données pour la gestion des encapsulations

3.2.3.2 Gestion des données

Le choix des structures permettant de gérer les données des éléments de B-Rep, ces `map` (voir Listing 3.6), n'est pas dû au hasard mais au contraire guidé par un souci d'optimisation de ce processus de gestion. L'idée est de pouvoir accéder le plus rapidement possible aux données, et donc de limiter le temps de parcours des structures. Nous avons ainsi choisi, dans un premier temps, de séparer les éléments topologiques selon leur dimension, d'où les trois structures distinctes.

Dans un deuxième temps, nous avons voulu optimiser l'accès aux éléments au sein de chaque structure. Nous avons alors opté pour un découpage selon l'historique, et plus exactement selon l'opérateur générateur de chaque élément. C'est ainsi que l'on retrouve, pour chaque type d'élément topologique, une première structure

```
map<char*,map<char*,myBRepElement*> >
```

où l'on va associer à chaque opérateur, ici désigné par son nom sous forme de `char*`, l'ensemble des éléments de B-Rep ayant été générés par celui-ci. Cet ensemble est désigné par la structure

```
map<char*,myBRepElement*>
```

au sein de laquelle chaque encapsulation d'élément de B-Rep est associée à une "clé" (ici

un `char*`), clé qui pourra changer selon le système CAO cible.

Ce choix de structure des données nous permet ainsi de pouvoir gérer des ensembles d'éléments selon leur opérateur générateur, ce qui optimisera les processus de gestion et de mise à jour des données, et nous aidera pour la visualisation de ces mêmes données (voir section 3.4). On peut se demander pourquoi on ne retrouve pas ces choix pour la gestion des opérateurs et des éléments d'esquisse, contenus respectivement dans une structure de type `list`. La réponse est simple et réside dans le fait que pour ces éléments, il n'y a pas de problème de *persistent naming*, et ils peuvent simplement être retrouvés par un identifiant.

Au demeurant, la phase sans doute la plus complexe de ce processus de gestion des données, va être celle des mises à jour du GHC de l'objet CAO.

3.2.3.3 Mise à jour des données

Le principe de notre approche d'interfaçage RV-CAO est de proposer la meilleurs utilisation possible des fonctionnalités des systèmes CAO existants, dans des environnements virtuels. Ainsi, parmi les fonctionnalités importantes que l'on va retrouver ici, il y a la gestion des modèles de représentations B-Rep et CHG, et notammant leur mise à jour. Nos structures de données permettent de gérer toutes ces données par la création d'une surcouche logicielle établissant un lien direct entre l'environnement virtuel et les données natives du système CAO cible. Mais contrairement à l'approche précédente de Convard et Bourdot [49], nous n'intervenons pas dans le processus de mise à jour des données : nous ne faisons que déclencher ces mises à jour, et en proposons une supervision (voir section 4.5, chapitre 4).

Ainsi, ne contrôlant pas le mécanisme de mise à jour, nous nous retrouvons dans la situation où nous devons gérer toutes les nouvelles données issues de cette mise à jour. Certaines données auront été supprimées, créées, modifiées, ou seront restées intactes. La quantité de données à traiter pouvant rapidement devenir énorme, il nous faut avoir une stratégie de traitement.

Si la modification d'un opérateur conduit à la suppression ou à la création d'éléments topologiques (voir figure 2.7), alors le GHC doit être parcouru et analysé depuis l'opérateur dont dépend la plus "ancienne" suppression/création d'élément. En effet, la création/suppression d'élément va conduire à une modification du graphe à partir duquel le mécanisme de *persistent naming* va fonctionner. Les données précédemment récupérées peuvent donc être devenues obsolètes et une mise à jour de toutes ces données est nécessaire. Par contre, si la modification de l'opérateur ne conduit qu'à une modification de valeurs paramétriques, sans aucune suppression ni création d'élément topologique, alors une mise à jour n'est pas nécessaire. Les pointeurs vers les éléments topologiques de nos encapsulations ne seront sans doute plus valides, mais les données de *persistent naming* contenues elles-aussi dans ces encapsulations seront toujours valables et nous permettront de ré-accéder aux éléments topologiques.

3.2.3.4 Discussion

Les sessions de travail en environnement immersif requièrent une interaction que l'on qualifiera de "temps-réel". Nous entendons par là que l'interaction en immersif doit se vouloir aussi "fluide" voire plus rapide (puisqu'il faut rafraichir parfois plusieurs stéréoscopies) qu'une interaction sur poste de travail, dans l'enchaînement des tâches. Nous le savons bien, parmi les points durs de l'interaction avec des systèmes CAO se trouve le temps des mises à

jour : plus l'objet CAO va être complexe et plus la modification remonte loin dans l'arbre de construction, plus la mise à jour sera elle-aussi complexe et chronophage. Dans cette limite-là, notre approche doit être aussi réactive que l'interaction sur poste. Mais plus l'objet va être complexe, plus nous aurons de données à analyser, notamment après les mises à jours du GHC, ce qui pourrait amener des temps de latence durant la session et éventuellement causer de la frustration chez l'utilisateur. Nous avons alors réfléchi quant aux différentes façons d'aborder le parcours et l'anaylse du GHC des objets CAO, pour répondre au mieux à cette problématique.

En fait, nous avons envisagé deux procédures différentes :

- analyser l'ensemble des données de l'arbre en suivant la chronologie de la construction, ou
- analyser uniquement les éléments sur lesquels nous pouvons agir en début de session, à savoir les seuls éléments de B-Rep visibles

Rappelons que lors de l'analyse et de la récupération des données des éléments de B-Rep, il y a une phase de d'analyse des éléments ancêtres. C'est principalement cette phase qui va être au centre de la discussion relative à nos deux procédures.

En effet, dans le premier cas, celui où toutes les données seraient récupérées dans un ordre chronologique, les différentes analyses d'ancêtres et donc de dépendances, ne poseront aucun problème, les éléments étant déjà présents lorsqu'ils seront recherchés. Néanmoins, plus l'objet sera complexe, topologiquement, plus il y aura de données à analyser, que ce soit lors de la création de la session, où après les mises à jours nécessitant une nouvelle analyse.

Dans le deuxième cas, nous commençons uniquement par analyser et encapsuler les données topologiques visibles. Lors de ces phases d'encapsulations, la recherche d'ancêtres va se révéler plus complexe que pour le premier cas : les ancêtres potentiels devront être à leur tour analysés et encapsulés, ne l'ayant pas été avant puisqu'invisible (ils n'existent pas dans le résultat final), ce qui nous amènera pour eux aussi à éventuellement analyser les données de leurs ancêtres. Et ainsi de suite. Le comportement récursif de cette recherche d'ancêtres, on le voit, peut elle aussi amener une lourdeur de traitement et une certaine latence, proportionnelle à la complexité topologique de l'objet CAO.

Dans le souci de pouvoir contrôler et appréhender les traitements de données et leur durée, nous avons choisi de suivre la première procédure, à savoir le parcours et l'analyse de tout le GHC (et de tous les B-Rep) selon la chronologie de la construction de l'objet CAO. Nous avons alors décidé d'apporter quelques contraintes à ces traitements, au travers d'une configuration de la sesssion de travail, afin de les restreindre uniquement aux parties de l'objet CAO auxquelles les utilisateurs voudront accéder pour cause de modification.

3.3 Réactivité des objets CAO

Comme nous l'avons décrit dans le chapitre 2 (section 2.8), le principe de réactivité des objets CAO repose sur le mécanisme de *labelling* des éléments de B-Rep. Cette réactivité peut être schématisée comme sur la figure 3.2 qui se lit en partant du coin haut gauche (B-Rep), en allant vers la droite (Moteur d'inférence), puis en finissant dans le coin bas gauche (Graphe d'Historique de Construction) :

1. L'utilisateur intéragit sur l'approximation polyédrique (*mesh*) de l'objet CAO, pour sélectionner un élément de B-Rep.

2. L'étiquetage de cet élément topologique va être envoyé au moteur d'inférence qui selon un jeu de règle va sortir comme résultat un ensemble de cibles potentielles.
3. Cette liste de cibles va être fournie à la deuxième partie du moteur d'inférence qui va sélectionner une ou des cibles parmi cette liste.
4. Selon la sélection finale, des contraintes de manipulation sont activées.
5. L'interaction de l'utilisateur modifie les cibles (valeurs des paramètres des nœuds du GHC)

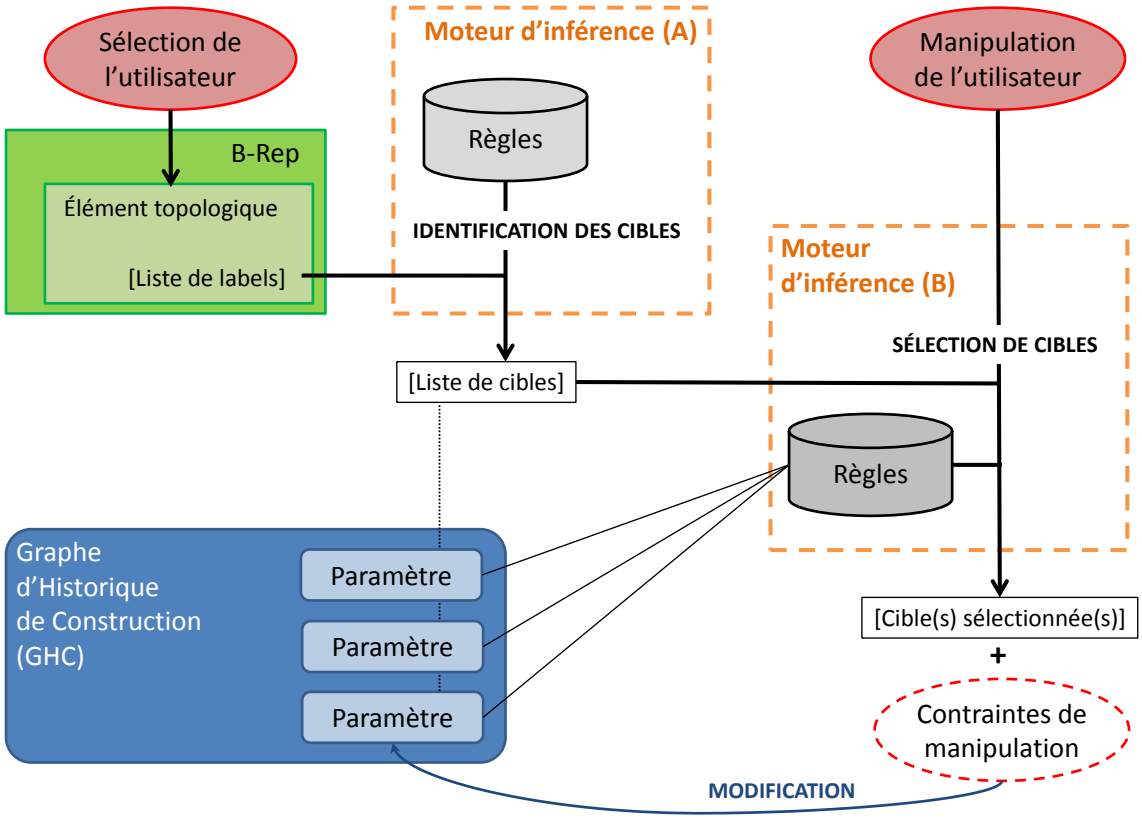


FIGURE 3.2 – Principe de l'édition implicite du GHC

Cette schématisation tient compte des différentes évolutions que nous avons apportées au concept de réactivité initialement présenté par Convard et Bourdot [49], évolutions que nous décrivons dans cette section.

3.3.1 Point clé : le *Labelling*

Le concept de réactivité des objets CAO repose sur un processus d'étiquetage des éléments de B-Rep. Ce mécanisme de *labelling* est une des grandes originalités de cette approche l'interfaçage RV-CAO qu'est l'édition implicite du GHC. Dans cette sous-section nous décrivons

donc dans un premier temps ce mécanisme, initialement créé lors des précédents travaux de Convard et Bourdot [49], puis nous présentons les évolutions apportées afin d'intégrer ce processus dans notre structure de données RV-CAO.

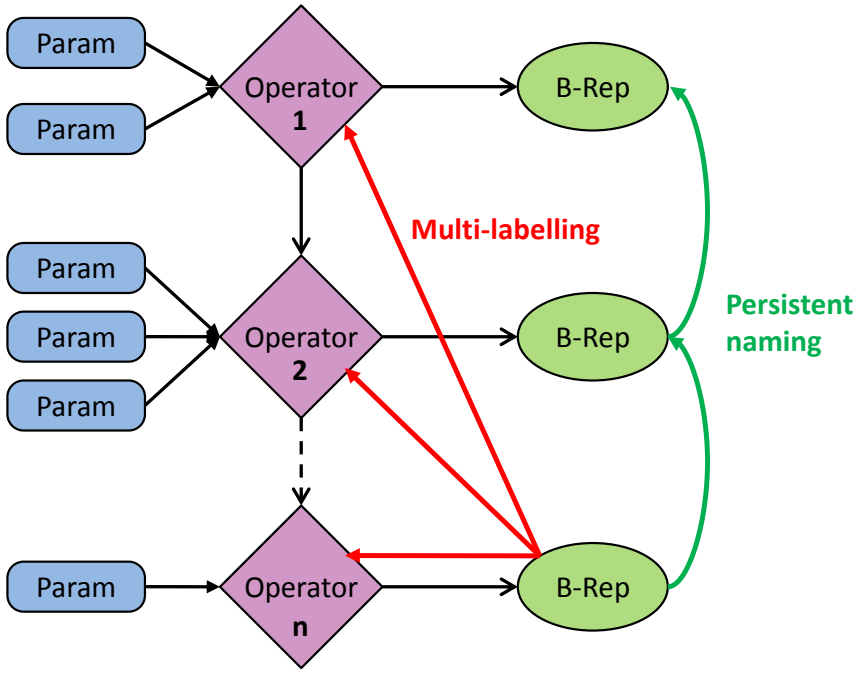


Figure 3.3 – Différence entre le *multi-labelling* et le *persistent naming*. Le *labelling*, ici dans sa version améliorée (*multi-labelling*) permet de lier les éléments de B-Rep aux nœuds du GHC tandique le *persistent naming* constitue un chaînage entre les éléments de B-Rep successifs.

3.3.1.1 Définition

Un *label*, une étiquette, est attaché à une élément de B-Rep et peut être vu comme un lien direct entre cet élément et un nœud du GHC. Ce *label* va dépendre de la nature de l'opérateur générateur de l'élément de B-Rep. Il en va de même pour les cibles (nœuds du graphe) accessibles depuis ce *label*. Ces cibles vont par exemple pouvoir être un des paramètres de l'opérateur générateur, ou bien un élément géométrique d'esquisse à l'origine de cet élément de B-Rep. Répétons ici une nouvelle fois qu'il ne faut pas confondre ce *labelling* des éléments de B-Rep, avec le *persistent naming* décrit plus haut (figure 3.3). En particulier, il est tout à fait possible que plusieurs éléments soient étiquetés, à un premier niveau, de la même façon, pointant donc vers le même nœud du graphe. Au contraire du *persistent naming* qui est unique pour chaque élément.

La forme standard d'une étiquette va être :

$$\texttt{NOM(ID}_{skel}\texttt{, ID}_{op}\texttt{)} \text{ ou}$$
$$\texttt{NOM(ID}_{op}\texttt{).}$$

Ici `NOM` va représenter le nom de l'étiquette. Nous aurons par exemple `VSK` pour le point d'une esquisse, `VPT` pour un vertex appartenant à la face "top" d'une extrusion, etc.. Deux formes existent donc pour ces étiquettes, la différence étant la présence de un identifiant ($\texttt{ID}_{op}$) ou de deux identifiants ($\texttt{ID}_{skel}$ et $\texttt{ID}_{op}$) dans cette étiquette. L'identifiant $\texttt{ID}_{op}$ sera toujours présent, et réfère à l'identifiant de l'opérateur générateur de l'élément de B-Rep. L'identifiant $\texttt{ID}_{skel}$, va lui désigner l'identifiant d'un élément d'esquisse, identifiant que l'on le retrouvera sous une forme plus spécifique dans les étiquettes : $\texttt{ID}_{pnt}$ pour désigner un point, $\texttt{ID}_{crv}$ une courbe, etc.

#	Label	Sémantique	Cibles
1	VSK(ID_{pnt}, ID_{op})	Vertex issu d'un point d'esquisse	• point ID_{pnt} de l'esquisse ID_{op}
2	ESK(ID_{crv}, ID_{op})	Arête issue d'une courbe d'esquisse	• courbe (ligne, courbe, cercle, spline) ID_{crv} de l'esquisse ID_{op}
3	VPT(ID_{op})	Vertex de la face "top" d'une extrusion	• direction de l'extrusion ID_{op}
4	EPB(ID_{op})	Arête de la face "bottom" d'une extrusion	• limite basse de l'extrusion ID_{op}
5	FPT(ID_{op})	Face "top" d'une extrusion	• limite haute de l'extrusion ID_{op}
6	FPB(ID_{op})	Face "bottom" d'une extrusion	• limite basse de l'extrusion ID_{op}
7	EH(ID_{op})	Arête d'un trou	• position du centre du trou ID_{op} • diamètre du trou ID_{op}
8	FH(ID_{op})	Face d'un trou	• profondeur du trou ID_{op}

TABLE 3.1 – Exemple d'étiquettes et de cibles. V : Vertex. E : Edge. F : Face. SK : Sketch. PT : Pad Top. PB : Pad Bottom. H : Hole. Certaines cibles correspondent à des paramètres d'opérateurs. D'autres sont des objets (ex : un élément d'esquisse) parce qu'on ne connait pas à l'avance les cibles accessibles depuis ces objets (ex : une contrainte d'esquisse).

3.3.1.2 Évolution du processus d'étiquetage

Mon travail de généralisation de l'approche de l'édition implicite a conduit à faire évoluer quelque peu ce processus d'étiquetage des éléments de B-Rep. Dans la précédente approche de Convard et Bourdot [49], il était possible de paramétrer le processus de *labelling* ainsi que les règles de logique du moteur d'inférence. Ceci est bien évidemment toujours d'actualité. Cependant, le champ d'action de ces *labels* était limité : une ou plusieurs cibles (nœuds du GHC) pouvaient être définis comme résultats potentiels d'un *label* sélectionné, mais à un élément de B-Rep n'était associé qu'un unique *label* — ce *label* n'étant dépendant que de l'opérateur générateur. Ainsi, les utilisateurs disposaient d'un choix limité de modification du graphe d'historique de construction, un *label* ne corresponsant alors qu'à un certain niveau du GHC, alors qu'une forme peut résulter de l'application de plusieurs opérateurs.

Afin de dépasser cette limite, j'ai étendu ce mécanisme de *labelling* en m'appuyant sur le modèle de données RV-CAO que j'ai mis en place.

L'idée était d'abord de multiplier le nombre de *labels* et de cibles, afin de couvrir les principaux opérateurs de CAO, de sorte à mieux couvrir les besoins des utilisateurs. On retrouve certaines de ces étiquettes dans le tableau 3.1. Cette extension du nombre d'étiquettes disponibles ne peut être vue comme une réelle évolution du processus, puisque ne consistant uniquement en l'adaptation de la forme existante à de nouveaux opérateurs. Cette étape n'en restait pas moins indispensable. En fait, la véritable évolution du processus d'étiquetage concerne trois aspects fondamentaux :

1. modification du lien entre *label* et élément de B-Rep
2. modification de la forme existante des *labels*
3. introduction d'un *multi-labelling*

La première modification dans le processus de *labelling* va être une modification structurelle, tenant compte de l'architecture que nous avons mise en place. Initialement, un *label* est associé directement à un élément de B-Rep. Notre modèle de données proposant des encapsulations pour ces éléments de B-Rep, il nous a fallu naturellement attacher ce *label* directement à notre structure d'encapsulation.

La deuxième modification, celle de la forme des *labels* peut être illustrée par la règle 4 du Tableau 3.1 concernant l'étiquetage d'un vertex appartenant à la face "top" d'une extrusion. Par face "top", il faut entendre la face caractérisée par la limite haute (hauteur) de l'extrusion. Ainsi dans la précédente approche, ce vertex était associé à un *label* ayant deux identifiants, celui de l'opérateur et celui de l'élément d'esquisse. Notre modèle permet désormais d'étiqueter ce vertex par un *label* ayant un unique identifiant, en l'occurrence $\texttt{ID}_{op}$) correspondant à celui de l'opérateur d'extrusion en question. Ce qui nous donne l'étiquette suivante : $\texttt{VPT(ID}_{op}\texttt{)}$. Cette modification de la forme des étiquettes est en fait liée à notre troisième modification, celle de l'introduction d'un *multi-labelling*.

Ce *multi-labelling* est en fait la traduction de notre prise en compte de l'historique complèt de construction des éléments de B-Rep, ceci grâce notamment à l'étape d'analyse des ancêtres. Structurellement, les encapsulations de B-Rep ne vont plus être associées à un unique *label*, mais à une liste de *labels* — liste pouvant ne comporter qu'un unique élément. Dans notre exemple du vertex de la face "top" d'une extrusion, son étiquetage devient alors, dans un premier temps, une liste d'un unique *label* dépendant de l'opérateur extrusion ($\texttt{VPT(ID}_{op}\texttt{)}$). Mais ce vertex tire son origine d'un point d'une esquisse, esquisse qui sera le profile de base

de l'extrusion. Cette esquisse, bien qu'en deux dimensions, possède elle aussi une représentation de type B-Rep en trois dimensions. Il existe ainsi, un élément de B-Rep représentant le point de l'esquisse. Cet élément sera étiqueté, comme on le voit dans la règle 1 du Tableau 3.1, par le *label* `VSK(`$\texttt{ID}_{pnt}$`, `$\texttt{ID}_{op}$`)` où $\texttt{ID}_{op}$ représente l'identifiant de l'opérateur esquisse, et $\texttt{ID}_{pnt}$ représente l'identifiant du point de l'esquisse en question. Lorsque l'on cherchera les ancêtres de notre vertex de la face "top" de l'extrusion, nous retomberons sur l'élément de B-Rep associé au point de l'esquisse. Le mécanisme de *multi-labelling* va alors consister à redescendre toutes les étiquettes associées à cet élément, ici un seul *label*, dans la structure d'encapsulation de notre vertex de la face "top". La liste d'étiquettes devient alors :

$$\texttt{[VPT(ID}_{op}\texttt{=2) , VSK(ID}_{pnt}\texttt{, ID}_{op}\texttt{=1)]}$$

où l'extrusion a l'identifiant 2, l'esquisse l'identifiant 1, et le point de l'esquisse l'identifiant $\texttt{ID}_{pnt}$.

En sélectionnant ce vertex de la face "top" de l'extrusion, l'utilisateur aura ainsi le choix entre une interaction liée à l'opérateur extrusion, ou bien une liée à l'opérateur esquisse. Ce découpage, ce *multi-labelling*, va permettre, au delà de cet exemple, d'accéder potentiellement à un plus grand nombre d'endroits dans le GHC. Il rend également possible une gestion de ces différentes options d'accès.

3.3.2 Moteur d'inférence : identification de cibles

Nous venons de présenter les premières composantes de notre modèle de données pour l'interfaçage RV-CAO que sont les structures d'encapsulation et le mécanisme de *labelling* des éléments de B-Rep. Notre modèle est également composé d'un moteur d'inférence, composant "intelligent", dont la fonction est relativement simple : aider l'utilisateur dans le choix des paramètres à modifier à partir de l'élément topologique sélectionné. Nous allons dans cette section, décrire le principe de fonctionnement de ce moteur d'inférence et expliciter là aussi les différentes évolutions qu'il a subies.

3.3.2.1 Principe

Le moteur d'inférence peut être décrit simplement comme un processus permettant de choisir un/des nœud(s) du GHC à modifier, à partir de la sélection d'un élémént topologique, et plus exactement, à partir du résultat de son étiquetage (une liste d'étiquettes). C'est ce que l'on va appeler ici l'étape d'**identification des cibles** (partie A sur la figure 3.2).

Cette identification est réalisée par un système à base de règles logiques, qui selon les données d'entrées (un étiquetage composé d'une liste plus ou moins fournie de *labels*), donne comme résultat une liste de cibles. Dans notre cas, le système intelligent est basé sur des règles Prolog [1], dont des exemples sont donnés dans le Tableau 3.2. Le point d'entrée de ce moteur d'inférence est dons le résultat de l'étiquetage d'un élément de B-Rep, à savoir une liste de *labels*. En fait, une règle Prolog va associer à un *label* donné, un ou plusieurs nœuds du GHC. La forme générique d'une telle règle sera :

```
targ_([list of parameters], label).
```

Ces règles logiques peuvent et doivent être établies selon les besoins des utilisateurs, leur définition se faisant simplement via l'édition d'un fichier texte.

1. SWI-Prolog : http://www.swi-prolog.org/

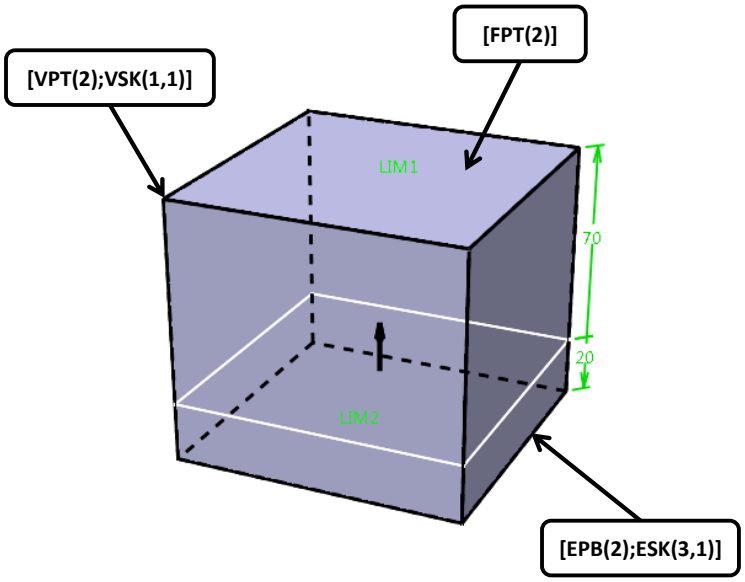


Figure 3.4 – Exemple de *labelling* et de cibles potentielles. L'objet final est le résultat de l'extrusion ($\mathrm{ID}_{op} = 2$) d'une esquisse ($\mathrm{ID}_{op} = 1$) représentée en blanc. Le résultat de l'étiquetage de trois éléments topologiques différents est présenté. La face est étiquetée par le *label* `FPT(2)` signifiant que cette face est la face "top" de l'opérateur extrusion ($\mathrm{ID}_{op} = 2$). L'arête est étiquetée par deux *labels* : `ETB(2)` et `ESK(3,1)` signifiant respectivement qu'elle appartient à la face "bottom" de l'extrusion ($\mathrm{ID}_{op} = 2$), et qu'elle tire son orginie de l'élément ($\mathrm{ID}_{crv} = 3$, une ligne) de l'esquisse ($\mathrm{ID}_{op} = 1$). Enfin le vertex est étiqueté par deux *labels* : `VPT(2)` et `VSK(1,1)` signifiant respectivement qu'il appartient à la face "top" de l'extrusion ($\mathrm{ID}_{op} = 2$) et qu'il tire son origine de l'élément ($\mathrm{ID}_{pnt} = 1$, un point) de l'esquisse ($\mathrm{ID}_{op} = 1$). Selon le tableau 3.1, on voit que la face est associée à la limite haute de l'extrusion de l'extrusion dont la valeur est 70. De même l'arête va donc être associée à la fois à l'élément d'esquisse ($\mathrm{ID}_{crv} = 3$) qui est une ligne d'esquisse, et à la limite basse de l'extrusion dont la valeur est 20.

Par exemple, la règle (1) du Tableau 3.2, associe le paramètre `param(`$\mathtt{ID}_{op}$`,1)` de l'extrusion $\mathtt{ID}_{op}$, au *label* `FPT(`$\mathtt{ID}_{op}$`)`. Concrètement, on va associer le paramètre 1 de l'extrusion $\mathtt{ID}_{op}$ — qui correspondra ici à la limite "haute" d'une extrusion — à la face "top" de l'extrusion $\mathtt{ID}_{op}$ (voir figure 3.4). Ceci étant, lorsque l'utilisateur sélectionnera cette face, il pourra directement modifier la limite "haute" de l'extrusion.

La règle (5) va, elle, lier la courbe d'esquisse `curve(`$\mathtt{ID}_{crv}$`)` avec le *label* `ESK(`$\mathtt{ID}_{crv}$`,`$\mathtt{ID}_{op}$`)`, utilisé pour étiqueter une arête de B-Rep d'une esquisse. Ainsi, lorsque l'utilisateur sélectionnera soit cette arête, soit une autre dépendante de celle-ci — comme par exemple une des arêtes de la face "top" d'une extrusion tirant son origine d'une ligne de l'esquisse —, alors il pourra directement agir sur l'élément d'esquisse en question, à savoir la courbe $\mathtt{ID}_{crv}$ de l'esquisse `curve(`$\mathtt{ID}_{op}$`)`.

# Extrusion : 1 = limite haute, 2 = limite basse, 3 = direction extrusion	
`targets_( [param(ID`$_{op}$`,1)], FPT(ID`$_{op}$`) ).`	(1)
`targets_( [param(ID`$_{op}$`,2)], EPB(ID`$_{op}$`) ).`	(2)
`targets_( [param(ID`$_{op}$`, 3)], VPT(ID`$_{op}$`) ).`	(3)
# Esquisse	
`targets_( [point(ID`$_{pnt}$`)], VSK(ID`$_{pnt}$`,ID`$_{op}$`) ).`	(4)
`targets_( [curve(ID`$_{crv}$`)], ESK(ID`$_{crv}$`,ID`$_{op}$`) ).`	(5)
# Trou : 1 = position, 2 = profondeur, 3 = diamètre	
`targets_( [param(ID`$_{op}$`,2)], FH(ID`$_{op}$`) ).`	(6)
`targets_( [param(ID`$_{op}$`,1),param(ID`$_{op}$`,3)], EH(ID`$_{op}$`) ).`	(7)

TABLE 3.2 – Exemples de règles Prolog pour l'identification de cibles.

3.3.2.2 Processus d'identification des cibles

Le modèle de données pour l'interfaçage RV-CAO que j'ai mis en place permet, comme nous venons de le voir, de prendre véritablement en compte tout l'historique de construction. Cette exhaustivité des informations issues du parcours et de l'analyse du GHC de l'objet CAO n'est pas une fin en soi. Elle permet de multiplicier les choix d'interaction ou plus fondamentalement d'avoir plus de degrés de liberté pour configurer l'interaction aux besoins des utilisateurs, et rend possible d'ailleurs plusieurs schémas d'interaction que nous présentons dans la section 4.2.3. Du reste, cette augmentation des possibilités rend d'autant plus complexe leur gestion, leur détermination. Il faut donc penser aux moyens qui nous permettent de prendre la meilleure décision face à toutes ces possibilités d'interaction.

C'est en ce sens que notre modèle de données et notre architecture ont été pensés. Comme on peut le voir sur la figure 3.2, le moteur d'inférence est en deux parties (A et B), chacune représentant une étape dans le processus de détermination des nœuds du GHC à modifier. La première étape, est celle décrite ci-dessus, permettant à partir du resultat de l'étiquetage — une étiquette ou d'une liste d'étiquettes — d'un élément de B-Rep, de pointer vers des cibles potentielles. La deuxième partie va alors choisir parmi ces cibles potentielles, la ou les cibles actives.

En l'état actuel des choses, la première étape consiste à envoyer entièrement la liste d'étiquettes attachée à l'élément de B-Rep sélectionné, afin de récupérer la liste complète des cibles atteignables depuis cet élément. En sortie de cette première étape, nous nous retrouvons donc avec une seule et même liste ordonnée contenant les différentes cibles accessibles en fonction des étiquettes traitées.

Cependant, le choix entre les différentes possibilités ne se fait pas au travers d'un "composant intelligent" de même niveau que le moteur d'inférence de la première étape. Toutes les cibles identifiées et regroupées dans cette liste, vont être traitées relativement simplement.

Ainsi la gestion de ces différentes possibilités se fait par un accès séquentiel à chaque entité de la liste : l'utilisateur navigue au sein de la liste des possibilités, en passant de l'une à l'autre grâce à une commande simple.

Nous retrouverons les possibles évolutions de cette étape de choix de cibles parmi une liste de cibles potentielles dans le chapitre 5 à la section 5.4.

3.4 Visualisation immersive

La volonté de modifier voire d'éditer des objets CAO en environnement immersif, et notamment dans les environnements de type CAVE, requiert un système de rendu graphique adapté. Rappelons que le principe de l'édition implicite consiste à modifier le GHC à partir de la manipulation d'élément visuels. Il faut donc que le système de visualisation permette un lien entre la représentation graphique de l'objet CAO et les structures de données en jeu dans la gestion de cet objet.

Au cours de précédents travaux (ceux de Convard et al. [50]), a été développé au sein du groupe VENISE du LIMSI-CRNS un outil de *clustering* graphique appelé *Distributed Rendering System (DRS)*. Cet outil de *clustering* graphique est conçu pour permettre la distribution de données au sein d'une grappe de PC, sur le principe de client/serveur (figure 3.5). L'un des avantages de cette librairie graphique est d'être relativement facile à utiliser avec n'importe quelle application gérant des représentations 3d.

Dans cette section nous allons ainsi décrire l'interfaçage que nous avons réalisé afin de permettre un rendu graphique de notre modèle de données et la gestion des éléments visuels.

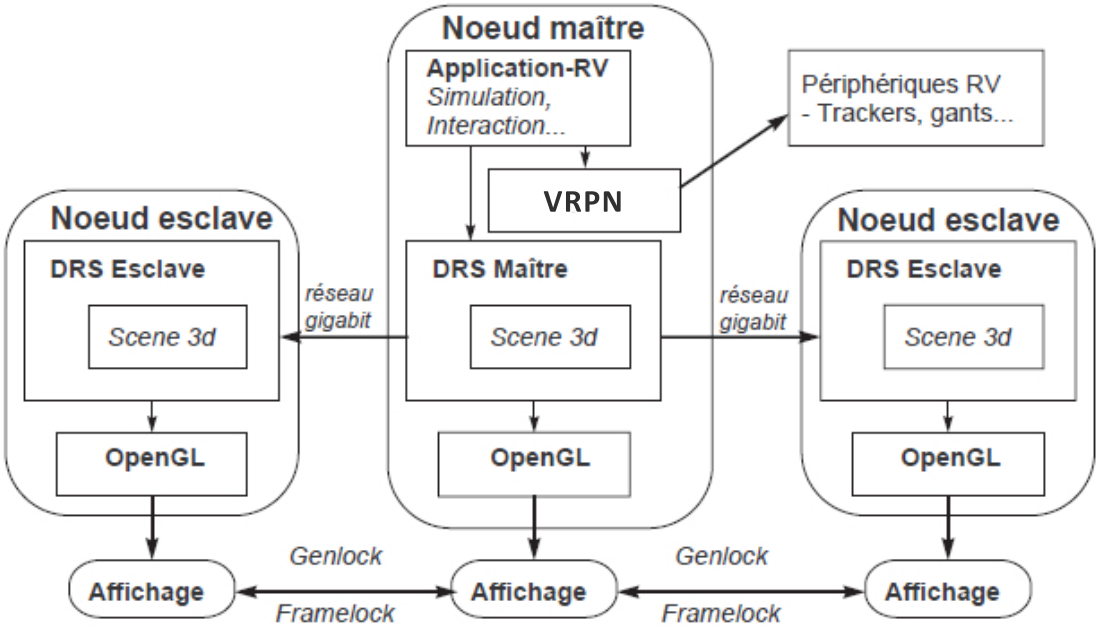


Figure 3.5 – Architecture du composant de visualisation DRS. Dans notre version, VRPN a remplacé VR-Toolkit initialement utilisé. (Convard [48])

3.4.1 Interfaçage avec DRS

DRS est composée entre autre, d'une librairie appelée BG, permettant la gestion d'objets graphiques, comme les *mesh*, les textures, et autres données. Cette librairie se base sur une structure hierarchique globale, incluant plusieurs branches spécifiques à différents types d'objets (voir figure 3.6). Ainsi, nous voyons que la structure `MetaObject` chapeaute trois branches, en l'occurence celle des `Objects`, des `Wires` et des `PointSets`, permettant respectivement la gestion des *mesh* (représentation polygonale nous permettant de visualiser les éléments topologiques de type Face), des arêtes, et des vertices.

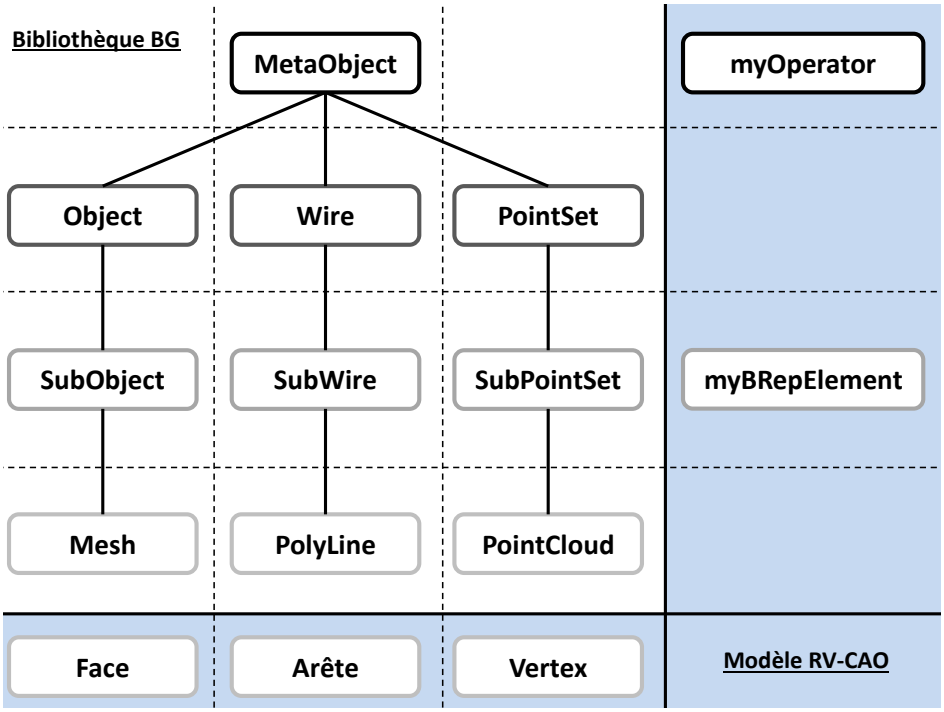


Figure 3.6 – Dans la partie blanche est représentée la structure hierarchique de la librairie graphique BG, et dans la partie bleu sa correspondance avec les différents éléments du modèle de données RV-CAO.

Sans rentrer dans les détails concernant le fonctionnament de la librairie BG, nous présentons comment nous avons interfacé notre modèle de données avec cette librairie. En effet, les fonctionnalités permettant la manipulation des objets graphiques, et notamment ce qui va nous permettre un retour d'information visuel adapté, vont être déterminantes dans la conception de cet interfaçage.

Dans les logiciels de CAO classiques, les nœuds du GHC (du moins ceux qui induisent des transformations topologiques) sont généralement associés à un résultat sous forme de B-Rep. Ce B-Rep est lui même composé de différents sommets, arêtes et faces. Lors de l'interaction en environnement immersif, l'utilisateur va justement jouer avec ces différents éléments topologiques, dont le rendu doit donc être distinctement géré. Lorsque un élément va être sélectionné (ou désigné, ou quelque interaction que ce soit propre à cet élément), un rendu particulier doit être réalisé afin de procurer à l'utilisateur un retour d'information (*feedback*)

lui permettant de le conforter dans son interaction. Or il faut noter ici que les principales interactions sur les objets graphiques vont se réaliser au niveau des structures `SubObject`, `SubWire` et `SubPointSet`. Ce sont ces structures de données qui vont être porteuses des informations de rendu, comme les matériaux. Sachant en plus que chaque élément de B-Rep est géré par une encapsulation qui lui est spécifique, il est apparu naturel d'associer ces différentes structures ensembles. Ainsi, selon que l'élément de B-Rep est un sommet, une arête, ou une face, son encapsulation gérera respectivement une structure de type `SubPointSet`, `SubWire`, ou `SubObject`. Aussi, un B-Rep complet étant attaché à un opérateur particulier, nous avons associé un `MetaObject` à chaque structure d'encapsulation des opérateurs (`myOperator`). Ces associations peuvent donc être résumées comme ceci :

- un **MetaObject** pour la gestion complète de l'approximation des métriques d'un B-Rep. Un `MetaObject` est composé d'`Objects`, de `Wires` ou de `PointSets` et géré par une encapsulation d'opérateur (`myOperator`, section 3.2.2.2).
- un **Object** pour la gestion des éléments topologiques de type "face". Ces faces sont représentées au travers de *meshs* (approximations polyédriques de la métrique réelle des faces topologiques) composés de triangles (classiques, strips, fans).
- un **Wire** pour la gestion des éléments topologiques de type "arête". Cette structure est composée de lignes (classiques, strips).
- un **PointSet** pour la gestion des éléments topologiques de type "sommet". Cette structure est composée de points.

Ainsi un `MetaObject` représentant un B-Rep complet, est composé d'autant d'`Objects`, de `Wires` et de `PointSets` qu'il existe de faces, arêtes et sommets dans ce B-Rep. Chaque `Object` / `Wire` / `PointSet` est lui composé d'un unique `SubObject` / `SubWire` / `SubPointSet`. En fait la librairie BG prévoit que les `Objects` / `Wires` / `PointSets` puissent être composés de plusieurs `SubObjects` / `SubWires` / `SubPointSets` dans le cadre d'une gestion du niveau de détails (LOD). Ainsi par exemple, plusieurs `SubObjects` contenu dans un `Object` représenteraient chacun un niveau de détail différent. Dans le cadre de nos travaux nous ne prendrons pas en compte cette problématique, d'où le fait que les `Objects` / `Wires` / `PointSets` ne contiennent chacun qu'un seul `SubObject` / `SubWire` / `SubPointSet`. Enfin les `SubObjects` / `SubWiress` / `SubPointSet` contiendront des `Meshs` / `PolyLines` / `PointClouds`, réelles représentations graphiques des différents éléments géométriques.

3.4.2 Gestion des éléments de visualisation

Maintenant que nous avons présenté notre interfaçage entre le modèle de RV-CAO et le système de visualisation immersif, nous décrivons dans cette sous-section, les différents processus en jeu permettant la gestion des éléments de visualisation.

Techniquement, pendant la première analyse du GHC, nous créons un `MetaObject` pour chaque opérateur. Lors de l'analyse des éléments de B-Rep de chaque opérateur de l'objet CAO, nous créons dans chaque encapsulation, la structure graphique qui doit lui correspondre : un `SubObject` pour une face, un `SubWire` pour une arête, et un `SubPointSet` pour un sommet. Lorsque l'élément de B-Rep a fini d'être analysé, les données géométriques associées sont envoyées dans les structures adaptées : `Mesh` (face), `PolyLine` (arête), et `PointCloud` (sommet). Puis la hierarchie est reconstituée, étant au final contenue par le `MetaObject` de l'opérateur correspondant. Enfin, les liens entre éléments visuels et éléments de B-Rep, nécessaires à l'interaction, seront réalisées grâce à trois structures de données associant chaque représentation géométrique — `Mesh` / `PolyLine` / `PointCloud` — à une encapsulation d'élé-

ment de B-Rep (`myBRepElement`) :

```
map<BGMesh*,myBRepElement*>
map<BGPolyLine*,myBRepElement*>
map<BGPointCloud*,myBRepElement*>
```

que l'on retrouve dans le Listing 3.6. Une fois que cette table d'association entre les éléments graphiques et les encapsulations est créée, DRS va alors distribuer toutes ces données géométriques (éléments de la librairie BG) aux différents nœuds du cluster qui mettront à jour leur affichage.

L'analyse des éléments de visualisation d'un B-Rep est réalisée dès lors qu'une mise à jour de l'objet CAO a été effectuée. Les différentes structures seront alors mises à jour elles-aussi lorsqu'il sera nécessaire. Chaque fois que ces structures de données graphiques seront modifiées, elles seront recommuniquées aux différents nœuds du cluster.

3.5 Moteur d'interaction

Cette section présente le dernier composant de notre architecture qui est un moteur d'interaction. Il a pour but de faire l'interface entre les périphériques de RV utilisés pendant la sesssion de travail, et l'environnement virtuel. Nous expliquons donc dans un premier temps l'architecture permettant d'intégrer et gérer les periphériques de RV puis dans un second temps comment ce composant est interfacé, notamment avec le système de visualisation.

3.5.1 Architecture

Le moteur d'interaction se base sur la librairie open-source VRPN [2] qui offre une interface transparente entre les périphériques de RV et notre application. Cette librairie va donc être embarquée dans notre composant représenté par la classe `interactionEngine` (voir Listing 3.7). En l'occurrence, la récupération des données des différents périphériques par VRPN va se faire au travers de la boucle `mainloop`.

Ainsi les structures attachées à nos périphériques de RV, comme `myFlystick` pour l'interaction via le *flystick*, et `myTracking` pour la position de l'utilisateur, vont être régulièrement mises à jour avec les données récupérées par VRPN.

Cette approche permet de rajouter autant de périphériques que nécessaires, et permet également d'ajouter ou de changer le système de communication avec les périphériques de RV (remplacer VRPN par un autre composant par exemple). Ceci étant, nous verrons dans le chapitre 5 (section 5.3), comment pourrait évoluer ce composant d'interaction.

2. VRPN : *Virtual Reality Peripheral Network*. http://www.cs.unc.edu/Research/vrpn/

```
class interactionEngine {

     interactionEngine( void );
    ~interactionEngine( void ) { }
    void Init();
    void run();
    static void *mainloop(void *); //Boucle VRPN
    void getAllFlystickData(int &button, int &state, double *position, double *
        orientation);
    void getFlystickButtonData(int &button, int &state);
    void getKeyboardButtonData(int &button, int &state);
    void getFlystickPosition(double &x, double &y, double &z);
    void getFlystickOrientation(double &x, double &y, double &z, double &w);
    void getHeadPosition(double &x, double &y, double &z);
    void getHeadOrientation(double &x, double &y, double &z, double &w);

    myFlystick *_myFlystick;
    myTracking *_myTracking;
};

class myApp{
    //Composant d'interaction
    interactionEngine *_interactionEngine;

    // Fonctions
    void updateUserLocation(); // Communique avec DRS
    void updateUserFlystick(); // Communique avec DRS
    void flystickEvent(int key, int state);
};
```

Listing 3.7 – Structure générale du composant d'interaction et liens avec le système de visualisation

3.5.2 Liens avec le système de visualisation

Les liens entre le composant de visualisation (DRS) et le composant d'interaction incluant VRPN sont très forts et très importants.

La visualisation stéréoscopique dans l'environnement immersif nécessite effectivement les données de position et d'orientation de l'utilisateur afin que le point de vue 3d proposé soit le bon. Ceci est rendu possible par la communication entre ces deux composants, au travers de la fonction `updateUserLocation`. Celle-ci récupère alors les données de l'utilisateur depuis le composant d'interaction et les envoie au composant de visualisation qui se met à jour. L'utilisateur peut donc évoluer dans l'environnement de RV avec un point de vue stéréoscopique adaptatif lui proposant l'exacte visualisation de l'objet CAO à chaque instant.

De la même façon, en session immersive, les interactions se font sur la maquette virtuelle, qui sera dans notre cas la "simple" approximation polyédrique produite par le système CAO cible pour un B-Rep donné, du fait du couplage RV-CAO direct et du lien fort avec l'environnement CAO. Les interactions de l'utilisateur vont donc se faire sur la représentation polyédrique générée par le composant de visualisation. Pour le cas de l'interaction au moyen d'un *flystick*, qui lui aussi aura une représentation visuelle, une communication entre les données du périphérique et le composant de visualisation est nécessaire. Elle estd'ailleurs assurée par la fonction `updateUserFlystick` qui récupère les données du *flystick* du composant d'interaction et les envoie au composant de visualisation qui se mettra à jour. Les interactions entre les différentes représentations d'objets virtuels, notamment les collisions, peuvent donc

être exactement calculées. De plus, la fonction `updateUserLocation` permet la communication des données de *tracking* de l'utilisateur au composant de visualisation afin d'assurer une visualisation de la maquette selon son point de vue exact.

3.6 Conclusion

Dans ce chapitre nous avons présenté l'architecture mise en place dans le cadre d'un interfaçage RV-CAO. Notre approche est un couplage RV-CAO direct dans lequel le lien avec le l'environnement CAO est un lien fort, qui permet ainsi l'édition implicite du graphe d'historique de construction (GHC) des objets CAO. L'architecture de ce couplage repose principalement sur un modèle de données RV-CAO composé de différentes structures d'encapsulation permettant de gérer les différents modèles de représentation que l'on trouve dans les systèmes de CAO paramétriques couramment utilisés dans l'industrie (section 3.2). Ce modèle de données RV-CAO est la pièce centrale de l'architecture, est ainsi découpé en cinq structures différentes permettant respectivement de gérer les éléments de B-Rep, les opérateurs du GHC, les paramètres des opérateurs du GHC, les contraintes des esquisses, et les éléments géométriques 2d composants les esquisses.

Ce modèle de données est interfaçé avec un composant d'interaction (section 3.5), un composant de visualisation (section 3.4), et un moteur d'inférence (section 3.3.2). Le composant de visualisation gère les représentations virtuelles des différents éléménts : objets CAO, interacteurs, utilisateur. Le composant d'interaction fait lui l'interface avec les périphériques de RV et ce composant de visualisation, permettant de mettre à jour les différentes représentations virtuelles en fonction des interactions de l'utilisateur. Enfin le moteur d'inférence permet d'aider l'utilisateur dans ses décisions selon ses interactions avec l'objet CAO.

Chapitre 4

Apports de ce modèle d'interfaçage RV-CAO

Résumé. *Nous décrivons dans ce chapitre, les apports de notre interfaçage RV-CAO et du modèle de données pour la revue de projet modificative. Nous expliquons ce qu'il est possible de faire grâce à cette architecture en décrivant les différents schémas d'interaction envisageables. Nous présentons également différentes supervisions rendues possibles lors des sessions de travail, comme la détection d'erreur lors des modifications de la maquette virtuelle. Nous explicitons enfin tous ces apports au travers d'une preuve de faisabilité basée sur CATIA V5 et son noyau géométrique.*

Sommaire

4.1 Introduction

Nous venons de décrire dans le chapitre précédent, le modèle de données que nous avons conçu et autour duquel nous avons bâti une architecture pour un interfaçage RV-CAO. Cet interfaçage se caractérise par un couplage RV-CAO direct et fort, en ce sens qu'il permet la modification immersive d'objets CAO préalablement contruits avec un logiciel paramétrique basé sur les caractéristiques de formes (*feature-based*), et ce sans aucune conversion ni transfert de données. Notre modèle permet en effet, lorsque tous les moyens requis sont fournis par l'API du logiciels CAO cible, d'utiliser toutes les données de l'objet CAO nécessaires à sa modification directement en environnement immersif. L'environnement CAO et l'environnement RV ne font finalement qu'un. Nous allons ainsi décrire dans ce chapitre les différents apports de notre interfaçage pour la modification d'objets CAO, en termes d'interaction, de gestion des connaissances, et nous les illustrerons en décrivant la preuve de faisabilité réalisée autour de CATIA V5 et de son noyau géométrique.

4.2 Modification intuitive du GHC

Dans sa forme actuelle, notre approche permet donc l'édition implicite du GHC depuis la sélection d'éléments de B-Rep (revoir figure 3.2). Cette approche est fondamentale pour la modification d'objets CAO en environnement immersif, en ce sens où il est impensable d'afficher le graphe d'historique complet dans l'environnement immersif, ni de gérer les interactions graphiques avec cet arbre. C'est dans cette optique que l'édition implicite, qui se traduit par un lien direct entre les éléments visuels (approximations polyédriques du B-Rep) et les éléments du GHC, prend toute son efficacité. Pour ce faire, et pour le bon fonctionnement de cette approche dans un contexte industrie, le modèle de données a été pensé pour intégrer et utiliser au maximum les fonctionnalités proposées par le logiciel CAO cible. Cette intégration se base sur les pré-requis suivant :

- le logiciel CAO est un logiciel paramétrique basé sur les caractéristiques de forme
- le logiciel CAO propose une API permettant d'accéder à son noyau géométrique et à ses fonctionnalités
- les fonctionnalités identifiées comme indispensable pour notre modèle de données, comme le mécanisme de *persistent naming*, sont disponibles et utilisables.

Nous allons décrire dans cette section les différents apports de ce modèle de données RV-CAO pour l'édition implicite dans le cadre industriel, et les différentes évolutions que j'ai été amené à réaliser.

4.2.1 Édition d'objets préalablement conçus

Le premier apport fondamental de mon approche est l'application du principe de réactivité à des bases de données CAO **existantes**.

Dans les précédents travaux du groupe VENISE [32, 48], l'édition implicite était possible sur des objets CAO créé entièrement depuis le début dans l'environnement immersif. Aussi, le noyau géométrique utilisé était un noyau open-source à partir duquel des représentations GHC et B-Rep spécifiques ont été créées. Mais le contexte industriel est tout autre. Des logiciels CAO commerciaux sont utilisés, intégrés dans le processus de conception, et il est hors de question de choisir d'autres plateformes CAO. En effet, la conversion des données est

un obstacle bien connu dans le monde industriel. Convertir les données pour passer d'une plateforme à l'autre est généralement la cause de soucis, dont le principal, outre les erreurs de conversion, est la perte des intentions de conception sur le modèle (on se retrouve généralement avec une maquette "morte" dont seule la géométrie est présente et où les différents paramétrages ont été perdus). Apporter une nouvelle solution qui se base sur une autre plateforme CAO que celles déjà utilisées était donc tout simplement contre-productif.

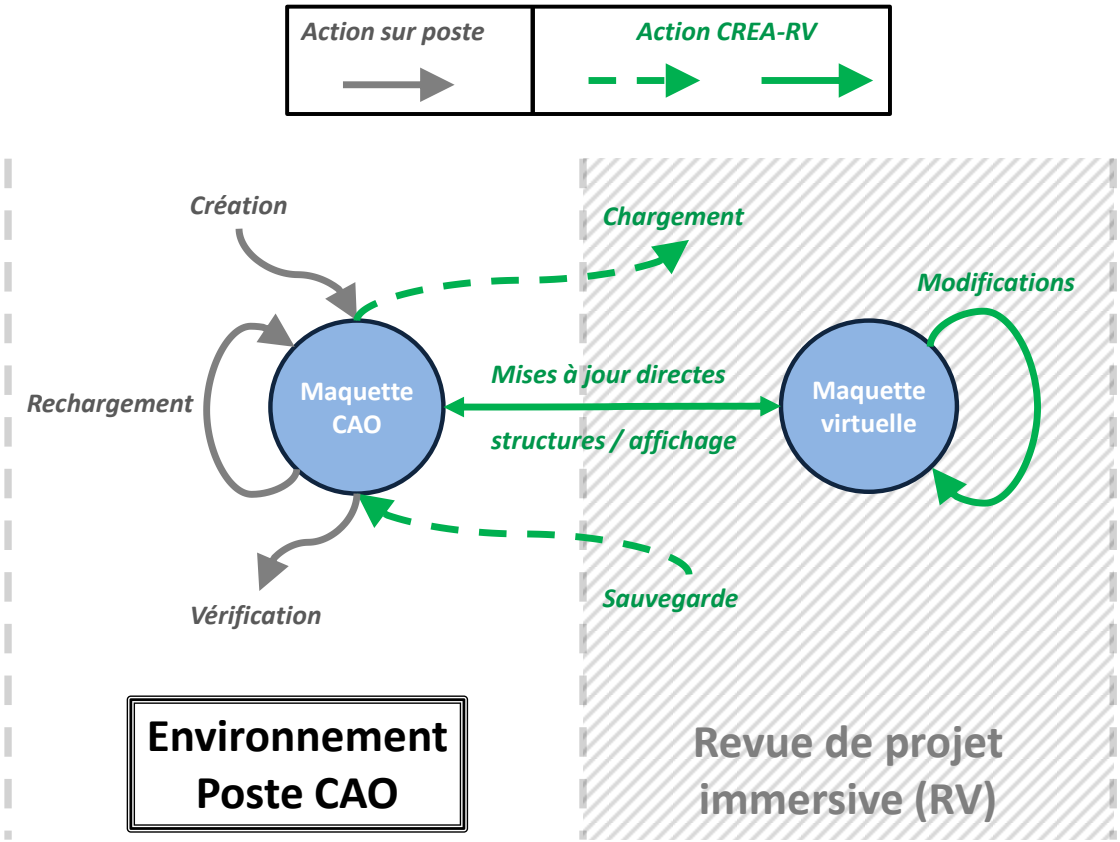


FIGURE 4.1 – Modification immersive d'un objet CAO sans conversion de données. La maquette CAO est préalablement créée sur poste de travail. CREA-RV permet son chargement et sa modification en environnement immersif, ainsi que la sauvegarde dans le format initial. La maquette CAO peut ensuite être rechargée directement sur poste de travail pour la vérification et la poursuite du travail.

La RV est utilisée pour bon nombre de séances de travail dont l'objet d'étude est l'architecture de la voiture (section 2.4.3, chapitre 2). Ces séances ont pour but d'explorer différents paramétrages, différentes hypothèses d'architecture afin d'évaluer l'impact au plus tôt d'éventuelles modifications. De telles séances immersives, nécessitent de travailler sur la maquette numérique originale, possédant encore toutes les intentions de conception, dont les différents paramétrages. Mais travailler sur la maquette numérique originale suppose qu'il existe un moyen de visualiser cette maquette telle quelle en environnement immersif, et qu'il

est possible d'accéder aux paramétrages. Généralement, lorsque la maquette est visualisable, l'interaction va être limitée, l'interfaçage RV-CAO n'étant qu'indirect. C'est par exemple le cas pour la visualisation de maquettes CATIA grâce à TechViz : la maquette est visualisable en environnement immersif, mais l'interaction sur la maquette virtuelle directement est très limitée.

Pour toutes ces raisons, il nous fallait penser un modèle permettant d'éviter toute conversion des données pour leurs utilisations en environnement immersif. C'est ce qui a été fait. Le modèle de données RV-CAO, tel que décrit dans le chapitre 3 permet, lorsque le logiciel CAO ciblé en offre la possibilité, d'embarquer les fonctionnalités CAO en environnement immersif. Il devient ainsi possible de charger des objets CAO, de parcourir et d'analyser leur GHC afin de les visualiser et de les modifier directement en environnement immersif. Ainsi, lorsque la session de travail est terminée, la maquette modifiée peut être sauvegardée, toujours dans le format natif du système CAO cible. Le principe de modification d'un objet CAO en environnement immersif est alors optimisé : **aucune conversion de données n'est nécessaire, que ce soit en entrée, ou en sortie de session de travail** (figure 4.1).

4.2.2 Accès simple et direct aux paramètres de la maquette

Le principe de l'édition implicite est simple : sélection d'un élément topologique, récupération d'une liste de cibles potentielles à partir de l'étiquetage de cet élément (inférence), choix des cibles (inférence), modification des cibles, et mise à jour de l'objet (revoir figure 3.2). Dans le cadre de la généralisation de cette approche aux représentations B-Rep et GHC que l'on trouve dans le contexte industriel, des évolutions ont été apportées à ce principe et sont décrites dans cette sous-section

4.2.2.1 Étiquetage multiple des éléments

Une des premières évolutions de ce principe d'édition porte sur le processus d'étiquetage des éléments. Dans la précédente approche, on associait un élément topologique à une étiquette (un *label*) qui servait à faire un lien direct entre cet élément et des éléments cibles du GHC, et qui était aussi utilisée par le mécanisme de *persistent naming* conçu pour l'occasion. Notre approche consiste à utiliser toute les fonctionnalités du système CAO cible et de fait, ne propose pas de mécanisme de *persistent naming* spécifique : on utilise celui du système CAO. L'étiquetage, ou *labelling*, va pour sa part se concentrer sur les liens entre les éléments de B-Rep et les nœuds du GHC. Au delà de cette clarification conceptuelle (revoir figure 3.3), l'évolution majeure apportée à ce mécanisme d'étiquetage consiste à considérer non plus une mais plusieurs étiquettes : un *multi-labelling*. Cet étiquetage multiple est le résultat de la prise en compte de l'historique complet des éléments topologiques. Chaque élément traité voit son historique retracé — ceci est rendu possible grâce au mécanisme interne de *persistent naming* du système CAO cible —, et reçoit les étiquetages de ses parents lorsqu'il en a (revoir section 3.3.1.2, chapitre 3). Ainsi, le moteur d'inférence qui permettait d'établir une liste de cibles potentielle à partir d'une étiquette donnée, va désormais agir sur une liste contenant une ou plusieurs étiquettes. Le résultat restera une liste de cibles, mais potentiellement plus exhaustive par rapport aux nœuds du GHC.

Notre modèle de données, par son *multi-labelling*, permet donc d'**accéder à un plus grand nombre de données du GHC à partir d'un seul et même élément de B-Rep visible** : les éléments accessibles au travers du *label* qui lui est directement attribué en fonction de l'opérateur générateur, et les éléments accessibles du fait de son historique de création (à partir des *labels* récupérés de ses parents).

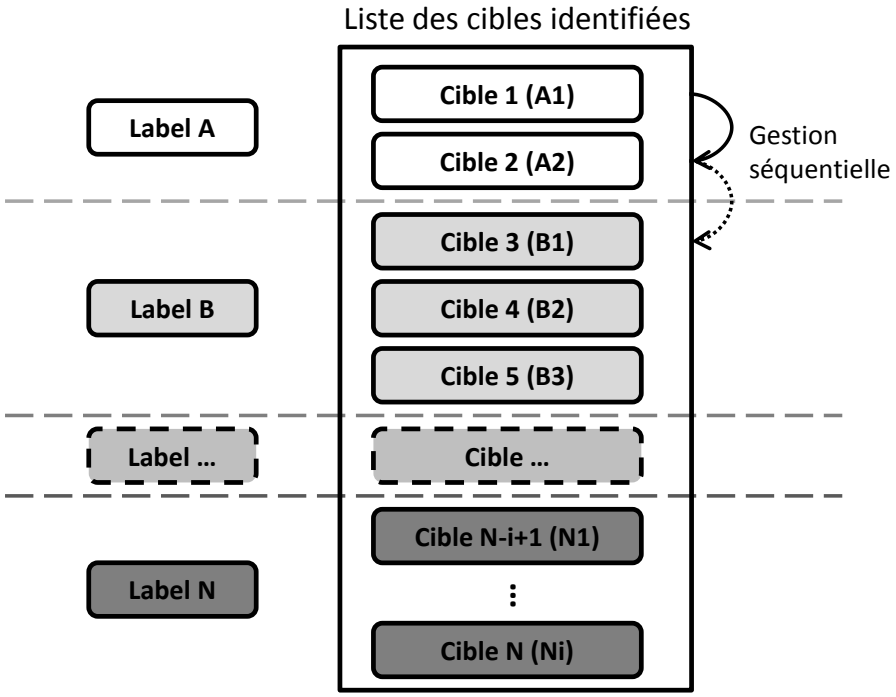


FIGURE 4.2 – Liste des cibles identifiées par le moteur d'inférence (niveau A). Les cibles sont organisées dans l'ordre d'appartenance aux différents *labels*. Les cibles identifées à partir du *label* A se retrouvent donc en tête de liste, suivies par celles identifiées depuis le *label* B, et ainsi de suite. Le *label* B est issu du premier parent pris en compte, et ainsi de suite. Ainsi plus on descend dans la liste de cibles, plus on remonte dans le graphe d'historique.

4.2.2.2 Gestion des cibles

Le moteur d'inférence présent dans l'architecture que nous proposons est un composant indispensable du principe d'édition implicite. Initialement, une sélection topologique pouvait amener à l'identification d'une ou deux cibles, et donc la modification d'un voire deux paramètres de l'objet CAO. Il n'y a avait donc pas réellement de gestion des cibles à proprement dite : les cibles identifiées étaient modifiées en même temps.

Le fait de potentiellement disposer d'un plus grand nombre de cibles à partir d'une seule sélection topologique, nous a amené à mettre en place un système de gestion de ces cibles. L'architecture proposée, prévoit non plus une étape d'inférence, mais deux étapes (revoir figure 3.2). La première étape (moteur d'inférence niveau A) consiste à idenfifier les cibles en fonction de la liste d'étiquettes de l'élément, la seconde (moteur d'inférence niveau B) à déterminer (selon la nature de l'interaction : geste ou commande de l'utilisateur), les cibles à activer pour la modification. Il est effectivement bien évident que s'il est possible d'accèder à trois, quatre, voire cinq et plus, paramètres en même temps, il n'est sans doute pas opportun de les modifier simultanément. En l'état actuel, la seconde étape de l'inférence reste à l'état de principe, elle n'est pas concrètement appliquée. Cependant, toutes les cibles identifiées ne sont pas toutes modifiées en même temps, leur activation n'est pas automatique.

En fait, le résultat de la première étape de l'inférence est une unique liste de toutes les cibles accessibles selon l'étiquetage complet de l'élément de B-Rep sélectionné. Comme on peut le voir sur la figure 4.2, les cibles sont ordonnées selon leur *label* d'origine (qui lui

n'apparait pas dans la liste). Il faut noter que comme la liste de *labels* d'un élément de B-Rep est constituée au fur et à mesure que l'on remonte dans son historique, d'ancêtre en ancêtre, la liste de cibles peut se lire ainsi : plus on la parcourt en descendant, plus on remonte potentiellement dans le graphe de conception.

La gestion de cette liste de cibles, rassemblées et ordonnées, est en l'état très simple à utiliser puisqu'elle consiste à les parcourir séquentiellement l'une après l'autre pour les modifier. Ce passage de l'une à l'autre se fait simplement par une commande (actuellement un bouton, mais on peut facilement imaginer une commande vocale, ou autre). Dans l'environnement virtuel, cette liste de cibles est affichée sur un "panneau" visible de l'utilisateur (voir partie *Targets* sur la figure 4.10). Plutôt que de parcourir séquentiellement la liste, on imagine très bien, et l'architecture le permet, d'interagir graphiquement avec la liste et d'aller accéder directement à la cible que l'on veut modifier.

Nous présentons dans le chapitre 5 (section 5.4), les évolutions envisageables pour la gestion des cibles au sein du moteur d'inférence intégrant de nouvelles données comme l'expertise de l'utilisateur, ou la nature de son interaction (vocale, gestuelle). Dans la section qui suit, nous allons voir comment notre modèle permet déjà une spécification des cibles privilégiées pour une session de travail.

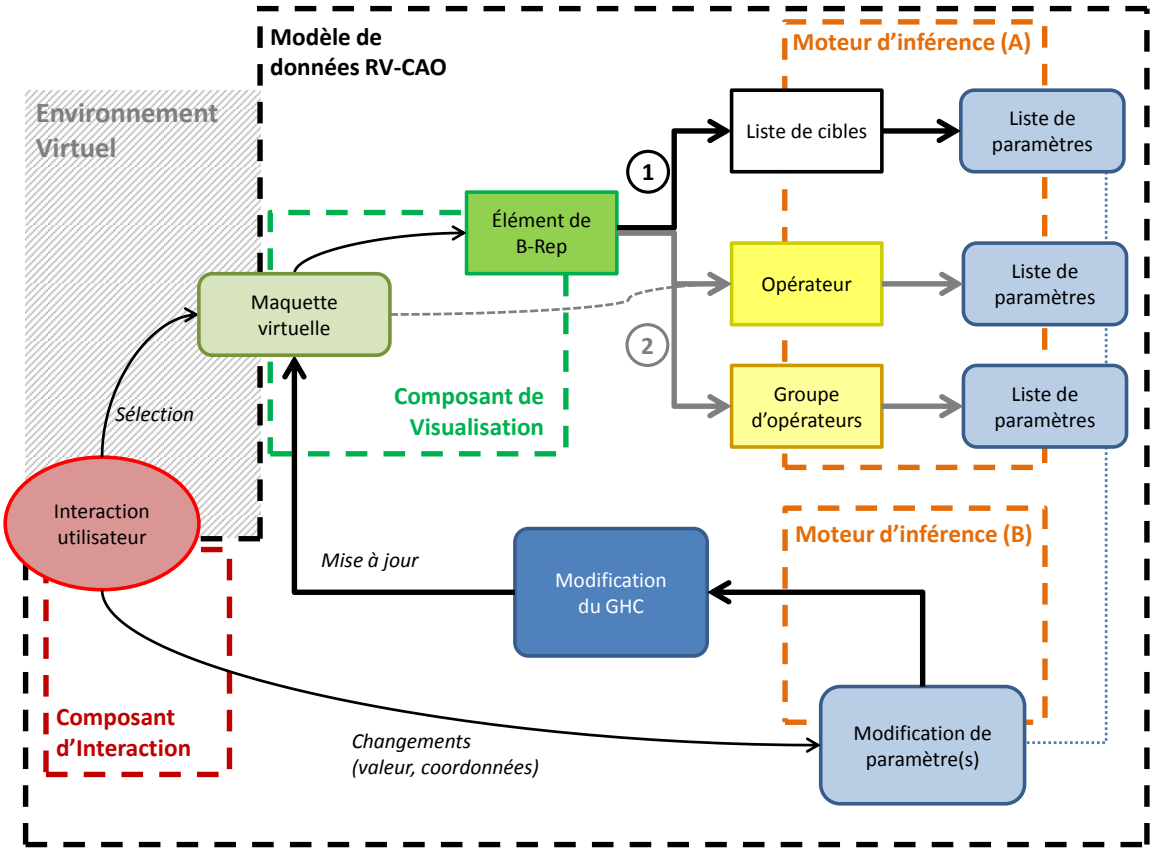


Figure 4.3 – Schématisation des différentes possibilités d'édition implicite du GHC.

4.2.3 Plusieurs schémas d'interaction

L'architecure et le modèle de données proposés dans le cadre de mes travaux permettent l'édition implicite du graphe d'historique d'objets déjà créés sur station de travail. Cette édition implicite est réalisée à partir de la manipulation d'éléments de B-Rep (sommet, arête, ou face). Ce premier schéma d'interaction d'un élément de B-Rep au GHC est illustré dans la figure 4.3 :

1. l'utilisateur agissant dans l'environnement virtuel manipule la maquette numérique pour choisir un élément de B-Rep
2. de cet élément est déterminée une liste de cibles — circuit (1) — permettant de modifier une liste de paramètres
3. l'interaction de l'utilisateur modifie ces paramètres
4. le GHC est mis à jour ainsi que la maquette numérique

Ce schéma d'interaction peut-être considéré comme le schéma *classique* de l'édition implicite. Il est particulièrement adapté lorsque la manipulation de l'objet CAO doit être fine, lorsqu'il y a un réel besoin d'accéder de façon précise aux différents éléments topologiques. C'est notamment le cas lors de la modification d'objet dont la complexité est relativement limitée, comme par exemple le rétroviseur présenté dans la section 4.4. Nous appelons complexité limitée, un objet dont le nombre d'éléments de B-Rep va être assez restreint et/ou dont le graphe de construction, et donc le nombre d'opérateurs et de paramètres, vont être eux aussi limités.

Mais il est rare que les maquettes numériques manipulées en contexte industriel soient d'une complexité limitée. C'est même le contraire. Les maquettes CAO présentent généralement un graphe d'historique de construction extrêmement vaste et fourni, ce qui se traduit par une représentation B-Rep complexe avec un très grand nombre d'éléments topologiques différents. Ces maquettes sont généralement une composition de plusieurs objets, parfois dépendant les uns des autres. Lors des séances de revue de projet immersive, certains paramètres de la maquette sont généralement mis en valeur (configurés) et seront manipulés. En fait, tous les paramètres de la maquette ne sont pas d'actualité, seuls certains le sont.

L'interaction fine sur les différents éléments de B-Rep n'est donc pas forcément adaptée dans le cadre de la modification de maquettes CAO complexes, dont seuls quelques paramètres déterminés vont intéresser l'utilisateur. D'autres schémas d'interaction doivent ainsi être envisagés, et notre modèle de données le permet. Comme on peut le voir sur la la figure 4.3, il est possible d'avoir une interaction moins fine que celle sur les éléments de B-Rep, en accédant plutôt aux B-Rep complets (qui sont associés aux opérateurs), voire à un ensemble de B-Rep (un ensemble d'opérateurs) : c'est le circuit d'interaction (2) sur ladite figure. Ces différents types de sélection sur le B-Rep sont illustrés dans la figure 4.4.

Ces nouveaux schémas d'interaction vont donc reposer sur une petite étape de configuration. Il n'est pas souhaitable d'accéder à tous les paramètres des opérateurs ou ensemble d'opérateurs — les différents logiciels CAO basés sur les caractéristiques de forme contiennent parfois une entité au-dessus de l'opérateur, agrégeant plusieurs opérateurs — et un filtrage doit donc être réalisé. Ce sont aux utilisateurs de préciser quels liens ils souhaitent voir ressortir (il peut s'agir de tous les liens). Ainsi, en précisant dans un fichier de configuration les paramètres sur lesquels ils veulent agir, ces paramètres seront identifiés comme "important"

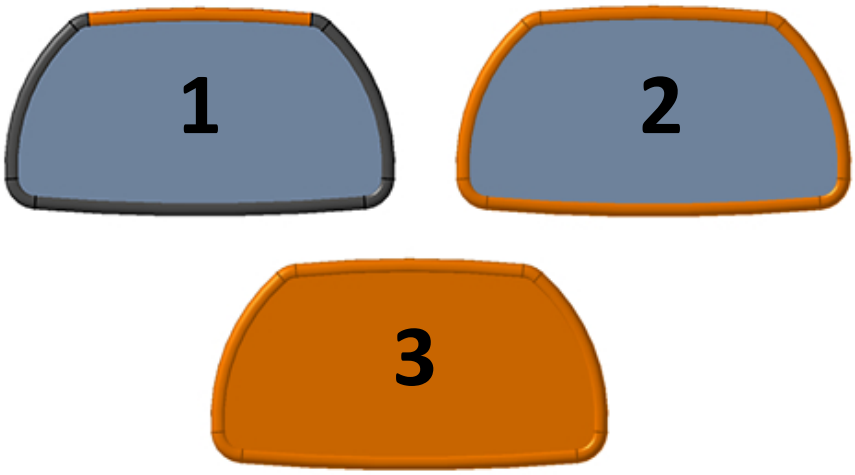


Figure 4.4 – Exemple des différents niveaux de sélection sur la maquette virtuelle. (1) Élément de B-Rep distinct. (2) B-Rep associé à un opérateur (ici un balayage). (3) B-Rep associé à un ensemble d'opérateurs (ici un balayage et un remplissage).

lors de la préparation de la session immersive. Lorsque l'utilisateur agira sur la maquette virtuelle, plutôt que de sélectionner un élément de B-Rep, il remontera au niveau du dessus pour accéder directement à un opérateur et/ou à un ensemble d'opérateurs. Cette sélection lui fournira la liste des paramètres "importants" de cet opérateur (ou de cet ensemble). Remarquons enfin que le fait de pouvoir remonter directement aux paramètres associés à un opérateur ou à un ensemble d'opérateur peut permettre de résoudre le problème des "paramètres cachés". En effet, l'accès au GHC se faisant à partir des éléments visibles du B-Rep, il se peut que parfois, certains nœuds ne soient pas acessibles. Le fait que l'utilisateur puisse spécifier à l'avance qu'il veut absolument accéder à tel ou tel paramètre durant la séance, peut ainsi résoudre partiellement ce problème des paramètres cachés.

Enfin notre modèle permet également de spécifier pour chaque paramètre, la façon dont il doit être modifié. En fait, il est courant que les ingénieurs veuillent évaluer différents paramétrages bien définis de la maquette CAO. Ainsi, notre approche permet de configurer les paramètres qui seront joués dans l'environnement immersif pour savoir si leur modification doit se faire via un changement continu de valeur (en suivant par exemple le geste de l'utilisateur), ou via un changement progressif (définition d'un pas de modification ou choix d'un jeu de valeurs à tester).

4.3 Supervision des mises à jour du GHC

Une caractéristique fondamentale qui se doit d'être assurée lors d'une séance immersive est l'interaction en temps-réel. En particulier, l'utilisateur se mouvant dans l'environnement virtuel provoque des mises à jour régulières de la visualisation (adapatation du point de vue, mise à jour des différentes représentations graphiques). Cependant, un obstacle va venir se dresser face à cette volonté d'assurer l'interaction en temps-réel : la mise à jour de l'objet CAO.

La mise à jour des objets CAO présente en effet des limites bien connues des différents concepteurs : ces mises à jour peuvent durer de quelques secondes à quelques minutes voire

plus, en fonction de la complexité de l'objet CAO. En fait la mise à jour de l'objet est fonction de la profondeur du graphe d'historique et du niveau auquel ce graphe est initialement modifié : plus le nœud du graphe que l'on modifie est "ancien" (c'est à dire "profond" dans l'arbre d'historique), plus la mise à jour de l'objet CAO a de chance de se répercuter sur d'autres nœuds et ainsi de prendre du temps. De plus, cette mise à jour, parfois longue, peut aboutir sur une erreur et un objet qui n'est plus valide.

Le type d'interfaçage RV-CAO que l'on propose ici, permet de visualiser en direct les différentes modifications apportées sur l'objet CAO. Nous exposons ainsi dans cette section les stratégies de mise à jour possibles qui vont permettre à l'utilisateur de percevoir (vérifier) au mieux — c'est à dire au plus proche du temps-réel — les modifications apportées à la maquette (figure 4.5).

4.3.1 Stratégies pour la visualisation directe des modifications

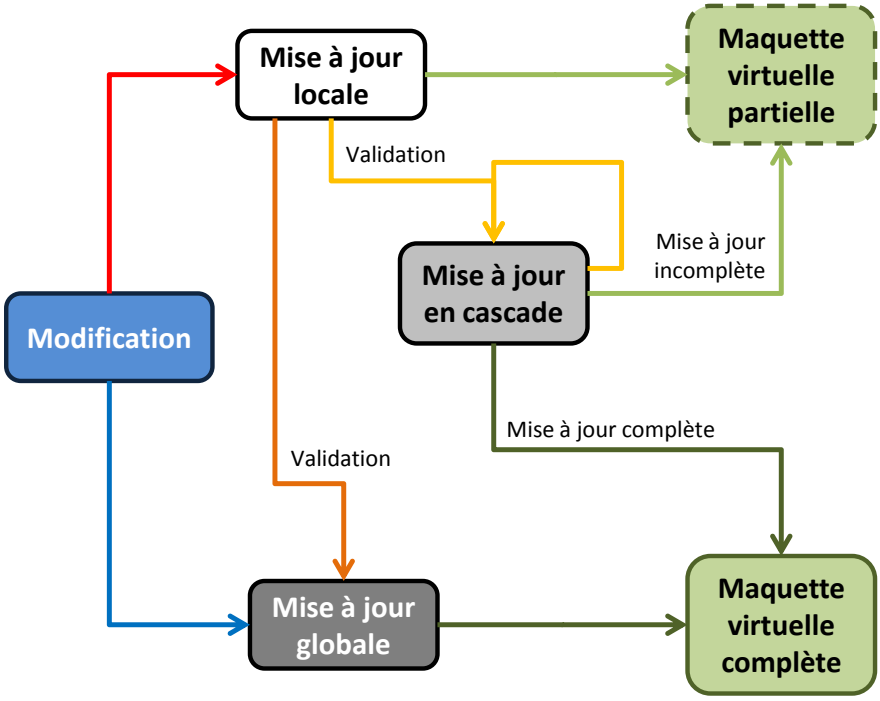


FIGURE 4.5 – Différentes stratégies pour la mise à jour de l'objet CAO. En bleu, rouge et orange, les trois stratégies de mise à jour de l'objet CAO, et en vert (clair et foncé) les différents niveaux de visualisation de la maquette. En bleu, la stratégie de mise à jour globale : à chaque modification, l'objet entier est mis à jour ce qui permet une visualisation de la maquette virtuelle complètement actualisée. En rouge, la stratégie consistant à mettre d'abord à jour l'opérateur modifié : on récupère une maquette virtuelle partiellement actualisée. Ensuite, en orange, deux possibilités pour la validation de la modification : une validation globale (orange foncé) ou une validation en cascade (orange clair).

La première possibilité, la plus simple, est de mettre à jour l'ensemble du graphe d'historique à la moindre modification de celui-ci : une mise à jour *globale* (en bleu sur la figure 4.5).

Lorsque ni le graphe d'historique ni la représentation B-Rep de l'objet ne sont complexes, cette mise à jour globale ne pose aucun problème. En effet, lorsque la réévaluation du graphe d'historique et des B-Rep est instantanée, la maquette virtuelle peut être immédiatement actualisée, et l'utilisateur a donc un aperçu direct et complet des modifications qu'il a apportées en temps-réel.

Cependant, lorsque l'objet CAO va devenir complexe et que sa mise à jour globale n'est pas instantanée, l'actualisation immédiate et en temps-réel de la maquette virtuelle complète n'est pas possible (du fait des délais créés par les divers calculs de réévaluation). Une première piste consiste à mettre à jour d'abord localement le graphe d'historique et la partie de la maquette virtuelle concernée. Plus exactement, l'idée est de mettre à jour l'opérateur sur lequel se porte la modification uniquement, et de récupérer le nouveau B-Rep associé à cet opérateur : une mise à jour *locale* (en rouge sur la figure 4.5). La partie changeante de la maquette virtuelle est actualisée ce qui donne un aperçu direct à l'utilisateur de sa modification en temps-réel, sans calculer l'impact de cette-ci sur le reste de l'objet CAO. Ensuite, une fois que l'utilisateur est satisfait de sa modification, il peut valider la modification pour évaluer les impacts complets de ses modifications (en orange sur la figure 4.5). Soit cette validation se fait par une mise à jour globale de l'objet CAO, dont l'instantaneïté n'est plus indispensable, soit par une mise à jour en cascade.

Enfin, il n'est parfois pas possible d'assurer la moindre visualisation en temps-réel de la modification apportée. Il va se trouver des cas où la moindre mise à jour, même à un niveau locale, induit des réévaluations trop complexes pour être visualisées instantanément. Dans ce cas, la modification ne doit pas se faire par une interaction progressive en temps-réel, mais par la détermination d'une valeur (possiblement un choix parmi une liste de valeur). Ainsi l'interaction de l'utilisateur conserve un caractère temps-réel jusqu'au déclenchement de la mise à jour de l'objet CAO, et aucune visualisation instantanée de la modification n'est disponible.

Tout ceci est rendu possible par le fait que sans nous occuper du fonctionnement interne des mises à jour, il est possible de contrôler la façon dont nous les déclenchons (par l'intérmédiaire de l'API du système CAO cible). Le choix de stratégie peut se faire pendant la session de travail, changeant simplement le mode de mise à jour.

4.3.2 Détection d'erreur

Le contrôle des mises à jour ne va pas seulement concerner la visualisation des modifications. En fait, l'accès à l'API du système CAO cible va permettre une supervision plus complète de ces mises à jour, et notamment de tracer les possibles erreurs lors des réévaluations du graphe d'historique. En fonction de la stratégie choisie pour le déclenchement des mises à jour (locale vs. globale), il va ainsi donc être possible de guider l'utilisateur dans son interaction, de façon plus ou moins fine, pour qu'il ne propose que des modifications valides.

Plusieurs cas de détection sont possibles selon la stratégie de mise à jour choisie (figure 4.5) :

- lors des mises à jour globales, seul un problème global de mise à jour sera détecté sans connaître réellement à quel niveau il y a un souci
- lors des mises à jour locales, les erreurs directement liées à l'opérateur peuvent être détectées et donc transmises à l'utilisateur
- une mise à jour en cascade (un opérateur après l'autre) permet de tracer précisément à quel niveau le souci intervient.

Dans tous les cas, cette supervision permet de conserver un état *valide* pour l'objet CAO modifié. Notre modèle de données permet en effet de conserver les différentes valeurs manipulées, et en cas de détection d'erreur de mise à jour, de revenir à la dernière valeur correcte. Ce traçage donne ainsi des indications à l'utilisateurs, lui spécifiant s'il doit par exemple donner une valeur supérieure ou inférieure à celle qu'il vient de donner et qui a causé l'erreur.

En fait, la différence entre les différents niveaux de mises à jour ne va pas résider dans le résultat (maintient d'un état valide), mais plutôt dans la finesse du message d'erreur communiqué à l'utilisateur. La mise à jour globale ne permettra pas forcément de détecter où l'erreur est intervenue alors que la mise à jour locale permet de le savoir précisément.

4.4 *CREA-RV* : édition implicite d'objets CATIA V5

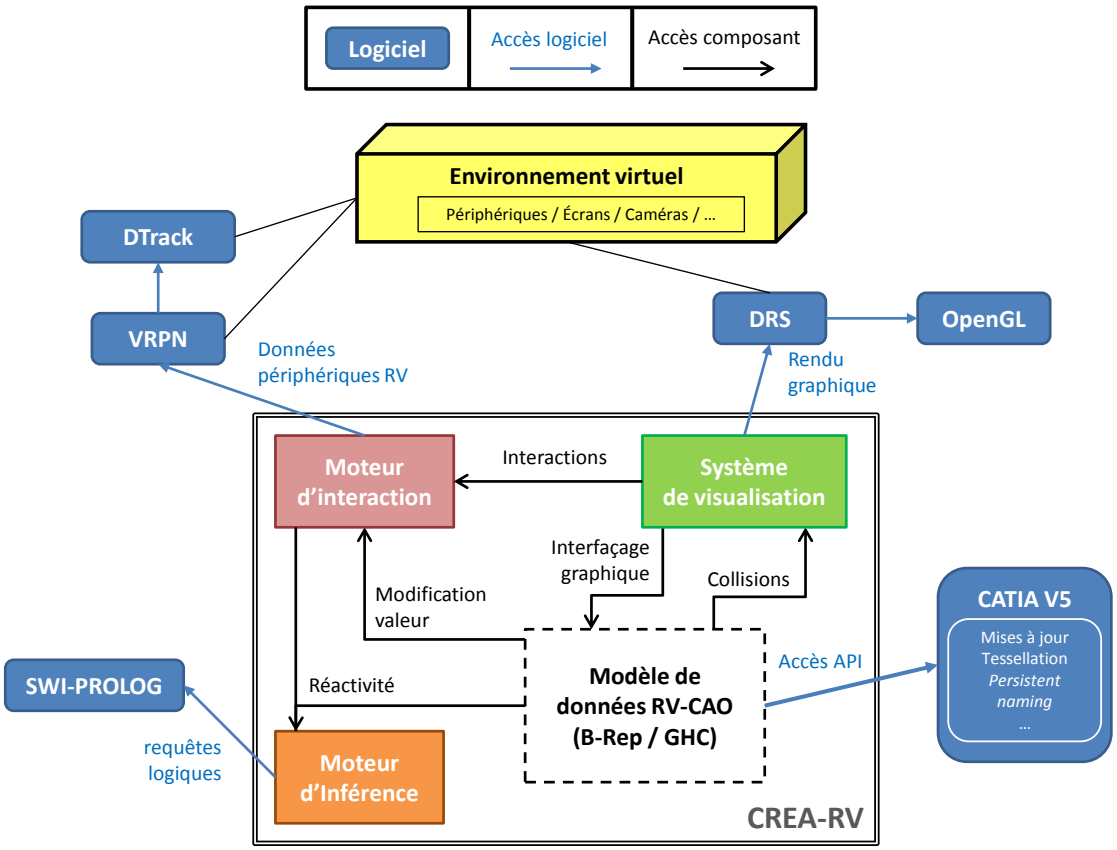


FIGURE 4.6 – Architecture logicielle de CREA-RV. Liens composants / logiciels, logiciels / matériels, composant / composant.

Un des buts principaux de cette thèse est de permettre l'application du principe de l'édition implicite aux systèmes CAO paramétriques utilisés dans l'industrie. Pour m'insérer au mieux dans le contexte industriel de PSA Peugeot Citroën, et poursuivre les travaux entrepris depuis plusieurs années par le groupe VENISE du LIMSI-CNRS, le logiciel CATIA V5 a été choisi comme base pour l'implémentation de la preuve de concept de mon approche. Ce logiciel est en effet utilisé dans une bonne partie du cycle de conception, et il offre également

un accès à ses fonctionnalités et à son noyau géométrique via une API (le CAA - *Component Application Architecture*) qui était à ma disposition.

Dans cette section nous présentons ainsi l'implémentation de notre architecture et de notre modèle de données sous CATIA V5 : **CREA-RV**, un CATIA réactif en environnement immersif.

4.4.1 Principe général

Tout d'abord, présentons les deux environnements principaux dans lesquels s'intègrent mon implémentation : l'environnement immersif dans lequel évolue l'utilisateur — les principaux utilisateurs visés par ces travaux sont les concepteurs, les ingénieur CAO, les architectes véhicules, est un système de type CAVE dont dispose et PSA et le groupe VENISE. Le logiciel CATIA V5 constitue lui l'environnement CAO.

CREA-RV est un ainsi ce que l'on peut considérer comme un middleware, une architecture pour l'interfaçage RV-CAO de type CAVE-CATIA V5. Cette architecture est conçue autour de quatre composants, déjà évoqués dans le chapitre 3, et dont nous évoquons ici la caractéristique logicielle (figure 4.6) :

- le **modèle de données** : ce modèle de données est l'encapsulation des différentes structures de données de CATIA V5, utiles à la gestion et à la modification. Les détails d'implémentations sont donnés en section 4.4.2.1.
- le **moteur d'interaction** : ce composant se base sur VRPN, bibliothèque libre permettant l'interfaçage entre notre application et les dispositifs de RV (*tracking* de la tête, et *flystick*), qui va lui communiquer entre autre avec DTrack [1] le logiciel de *tracking* de ART. Ce moteur d'interaction va ainsi être une interface entre l'homme (se mouvant au sein de l'environnement immersif) et l'environnement virtuel.
- le **moteur d'inférence** : ce composant se base sur SWI-Prolog, une implémentation open-source du langage Prolog.
- le **système de visualisation** : ce composant repose sur DRS, une bibliothèque de rendu graphique pour grappe de PC, développée lors de précédents travaux du groupe VENISE. DRS gère ainsi l'affichage de l'environnement virtuel sur les écrans de l'environnement immersif.

En fait, le composant qui va être le cœur de l'implémentation est le modèle de données. C'est en effet ce composant, décrit de façon générique dans le chapitre 3, qui va connaître une spécialisation particulière autour de CATIA V5. Les autres composants sont déjà conçus de telle sorte qu'ils sont complètement indépendant du système CAO visé.

4.4.2 Description détaillé

Dans cette sous-section, nous détaillons les spécificités de l'implémentation du modèle de données autour de CATIA V5.

4.4.2.1 Implémentation

L'encapsulation des opérateurs va se faire à partir de l'interface `CATISpecObject`qui remplace donc le type générique OperatorObject (revoir section 3.2.2.2). En fait, lors de la récupération des opérateurs du GHC, on va parcourir l'arbre, en récupérant les opérateurs sous

1. http://www.ar-tracking.com/products/software/dtrack2/

la forme de cette interface qui permet de manipuler les différentes *features* et notamment de déclencher les mécanismes de mise à jour. Elle nous permettra ensuite d'accéder à l'objet réel de l'opérateur en réalisant un *cast* avec le bon type :

Chanfrein — le type CATIA d'un chanfrein est `CATIChamfer`. Pour cet opérateur, nous allons récupérer par défaut deux paramètres pouvant être de la forme longueur/longueur ou longueur/angle. Dans les deux cas nous créons deux (`myParameterDouble`). L'objet à chanfreiner n'est pas récupéré ici puisqu'il est retrouvé lors de la recherche d'ancêtres de la surface créée par le chanfrein.

Congé — le type CATIA d'un congé d'arête est `CATIEdgeFillet`. Les paramètres qui vont nous intéresser pour un congé sont le rayon qui le définit (`myParameterDouble`), ainsi que le *Conic ratio* (`myParameterDouble`) valeur prise en compte selon l'activation ou non du mode conique, *ConicMode* géré par un paramètre booléen (`myParameterBool`).

Extrusion — le type CATIA d'une extrusion est `CATIPad`. Une extrusion est définie par un profil de base (`CATISketch`, qui est lui aussi pris en compte par le *multi-labelling*), une direction d'extrusion (`myParameterVector`) et deux limites "start" et "end" (`myParameterDouble`).

Rainure — le type CATIA d'une rainure est `CATISlot`. Une rainure repose sur un profil et une courbe guide (deux `CATISketch`), et deux épaisseurs (`myParameterDouble`) prise en compte selon l'activation de l'option associée (`myParameterBool`).

Trou — le type CATIA d'un trou est `CATIHole`. Un trou simple (de type "borgne") va être principalement défini par une position et un vecteur directeur (deux `myParameterVector`), une profondeur et un diamètre (deux `myParameterDouble`). D'autres modes de trou existent dans CATIA mais ne sont pas pris en compte actuellement.

Balayage et remplissage — le type CATIA d'un balayage est `CATIGSMSweep` et celui d'un remplissage est `CATIGSMFill`. Actuellement aucun paramètre n'est pris en compte pour ces deux opérateurs. Ils se basent sur des esquisses, qui seront finalement prises en compte par le*multi-labelling*. Le trigramme "GSM" nous indique que ces opérateurs sont des opérateurs surfaciques.

Esquisse — le type CATIA d'une esquisse est `CATISketch`. Les paramètres des esquisses sont en fait ses contraintes. Ainsi pour chaque esquisse, on récupère la liste de toutes les contraintes de type distance, longueur, angle, rayon, sous forme de `myConstraint` (revoir section 3.2.2.4). Le type générique `ConstrainteObject` devient l'interface `CATISpecObject` à partir de laquelle on accède, lorsque nécessaire, au réel type de la contrainte : `CATCst`. Le type de la valeur de ces différentes contraintes est à chaque fois `double`.

Lorsqu'une esquisse est traitée, l'encapsulation correspondante est créée puis tous les éléments d'esquisses sont récupérés. Le type générique `Element` va devenir comme pour les opérateurs l'interface `CATISpecObject` (revoir section 3.2.2.5). A partir de cette interface on peut alors accéder au réel objet en faisant un *cast* sur le bon type :

– un **cercle** (`CATI2DCircle`) est défini par un rayon (`double`) et une origine (`double*`).

– une **courbe** (`CATI2DCurve`), est définie par un point de départ et un point d'arrivée (deux `double*`). Le type "courbe" englobe notamment les "lignes" et les "splines".
– une **ligne** (`CATI2DLine`) est définie par un point de départ et une direction (deux `double*`).
– un **point**, dont le type CATIA est `CATI2DPoint`, est défini par ses coordonnées 2d (`double*`).
– une **spline** (`CATI2DSplineCurve`) est définir par un ensemble de point de contrôle (`CATI2DCstPoint`).

4.4.2.2 Éléments de B-Rep et rôle du *persistent naming*

Un point clé de cette implémentation est bien sûr l'encapsulation des éléments de B-Rep (revoir section 3.2.2.1) et l'utilisation du mécanisme de *persistent naming*. Dans CATIA V5, on accède aux éléments de B-Rep via l'interface `CATIBRepAccess`. Cette interface fournit un pointeur *temporaire* sur l'élément. Dès qu'une modification sur l'objet CAO intervient, ce pointeur n'est plus valide, même si l'élément de B-Rep n'a subi aucune modification (du moins visuellement). Comme nous l'avons dit précédemment, le mécanisme de *persistent naming* ne va pas avoir exactement le même rôle que pour les précédents travaux autour d'OpenCascade. En fait dans le cadre de ces travaux autour de CATIA, nous pouvons distinguer deux types d'utilisation de ce *persistent naming* : directe et indirecte.

Lors des différentes phases de traitement et préparation des données (analyse et parcours du GHC), le *persistent naming* va en fait être utilisé de façon indirecte. **Indirecte** parce qu'en réalité nous accédons à des fonctionnalités CAO qui, elles, utilisent les noms persistents produit par ce mécanisme. En particulier la recherche de l'historique de chaque élément de B-Rep via ces fonctionnalités n'est possible que parce que chacun de ces éléments possède un nom produit par le mécanisme de *persistent naming* qui traduit l'existent (le maintien) d'un même élément dans différentes versions de B-Rep. Nous n'avons strictement aucune idée du fonctionnement interne de ce mécanisme au sein de CATIA et nous n'avons pas besoin de le connaître puisque les fonctionnalités CAO évoquées nous permettent de connaître les informations de filiations entre éléments de B-Rep requises pour nos traitements.

Algorithme 2 Utilisation des informations de *persistent naming*

Mise à jour effectuée
Si Ni disparition ni apparition d'élément de B-Rep **Alors**
    Utilisation des informations de *persistent naming*
    Accès direct à n'importe quelle interface `CATIBRepAccess` valide
Sinon
    Re-parcours du GHC
    Récupération des nouvelles informations de *persistent naming*
    Récupération des interfaces `CATIBRepAccess` valides
Fin Si
Manipulations et modifications

Lorsque l'on sélectionne graphiquement un élément de B-Rep, nous allons accéder à son interface `CATIBRepAccess`, stockée dans notre encapsulation. Cependant, comme nous venons de le rappeler, la moindre modification va rendre obsolète le pointeur obtenu lors du traitement des données. Les informations du *persistent naming* de l'élément vont nous permettre d'accèder à une interface valide : on a ici une utilisation **directe** du *persistent naming*. Lorsqu'une modification intervient, deux cas se présentent (voir Algorithme 2) : soit la modification résulte dans l'apparition ou la disparition d'éléments de B-Rep (appartenant au B-Rep de notre élément sélectionné), soit il n'y a aucune modification du graphe de connexité topologique dans le B-Rep (seules les métriques de certains éléments ont changé).

Dans le premier cas, l'apparition ou la dispartion d'élément signifie la modification du graphe liant les éléments de B-Rep entre eux — graphe de connexité et de parenté que traduit le mécanisme de *persistent naming* — et les informations jusque là obtenues seront elles aussi devenues obsolètes et ne nous permettent pas de retrouver un accès valide à l'élément. La seule solution est alors de traiter à nouveaux tous les éléments du B-Rep concernés en reparcourant le GHC. Ainsi, on remet à jour les différentes informations au sein de nos encapsulations, notamment celles sur le *persistent naming*, et par la même occasion le pointeur sur l'interface `CATIBRepAccess`. Dans le deuxième cas, lorsqu'aucune disparition ni apparition d'élément n'est détectée, les informations jusque là obtenues nous permettent de ré-accéder directement à une interface `CATIBRepAccess` valide et de réaliser nos différents traitements.

Enfin il faut remarquer que notre approche permettrait de ne jamais utiliser directement les informations de *persistent naming*. En fait, si à chaque modification l'ensemble des éléments impactés est retraité, alors les données récupérées (dont les interfaces `CATIBRepAccess`) sont forcément valides. Cette stratégie de traitement est la plus simple à mettre en place puisqu'elle ne nécessite pas de traitement spécifique. Ceci étant, plus l'objet modifié sera complexe (constitué de nombreux éléments topologiques) plus le traitement des données sera chronophage et pourra entraîner des délais / des latences au cours de la session de travail. D'où l'intérêt de disposer de cette double stratégie dont l'une basée sur l'utilisation directe du *persistent naming*.

4.4.3 Cas pratique : modification interactive et intuitive d'un rétroviseur

Dans cette sous-section nous déroulons un exemple de modification d'un objet CAO, un rétroviseur, conçu sous CATIA V5 (figure 4.7) : préparation de la session de travail, création de la session immersive, revue de projet modificative et retour sur poste de travail. Le déroulement de la session complète est présenté en Annexe A.

4.4.3.1 Préparation de la session immersive

Avant de créer la session de RV-CAO immersive, une petite préparation est nécessaire pour gérer au mieux l'objet. Comme nous l'avons déjà dit, notre modèle permet de récupérer toutes les données de l'objet nécessaires à sa modification. Cependant, il n'est généralement pas utile de récupérer toutes les données du GHC complet de l'objet. D'une part l'objet peut être complexe et la gestion de toutes ces données créerait une lourdeur inutile, et d'autre part, la portée des modifications attendues lors d'une revue de projet est généralement spécifiée à l'avance.

Cette préparation consiste à spécifier les différentes parties de l'objet qui vont nous intéresser pour la revue de projet à venir. Pour chaque objet (des CATPart), l'utilisateur va

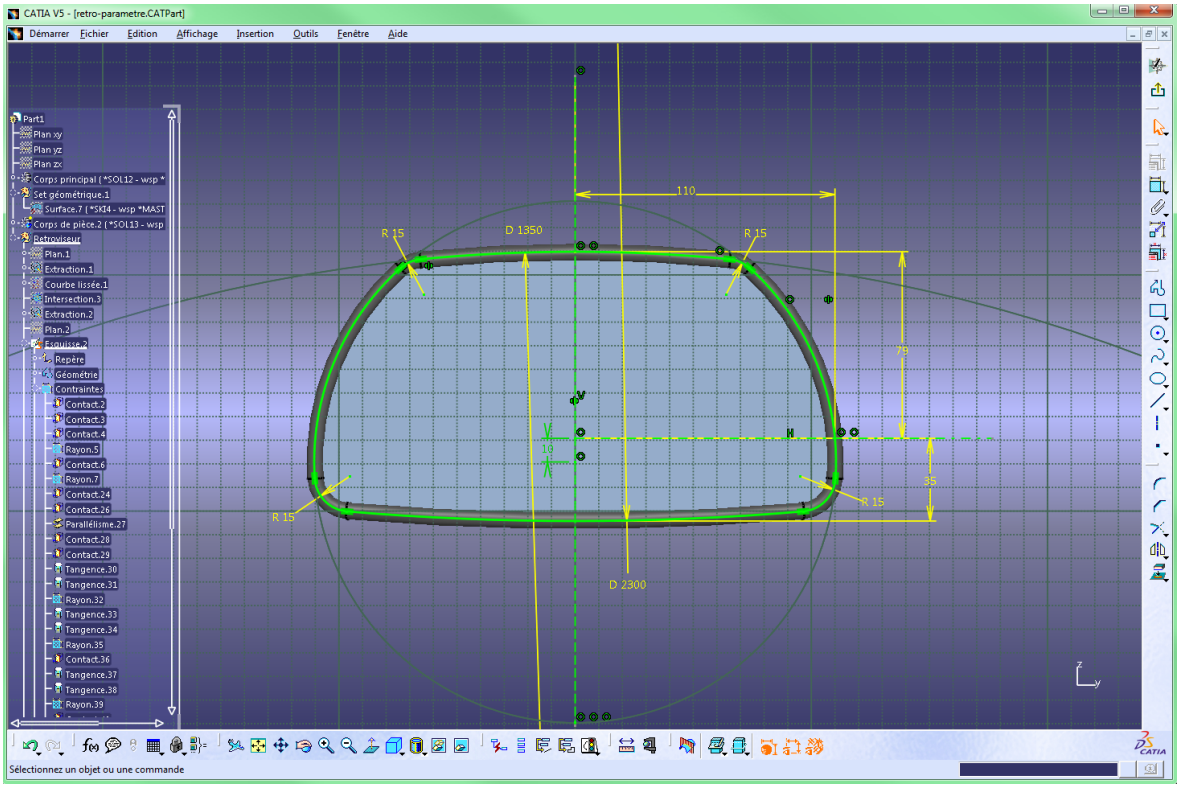


FIGURE 4.7 – Un rétroviseur conçu sous CATIA V5. Cet objet, contenu dans un fichier de type CATPart, va être modifié lors d'une revue de projet immersive grâce à CREA-RV.

donc spécifier les différents ensembles qui vont être la cible des modifications. Dans le cadre du rétroviseur, le CATPart de l'objet contient plusieurs *Sets* : un Corps Principal, un premier Set Géométrique, un Corps de pièce, et un deuxième Set Géométrique (figure 4.7). Les trois premiers *Sets* ont servi de base à la construction du dernier, qui est finalement la seule partie qui nous intéresse pour la revue de projet modificative. La configuration de la session va donc consister à spécifier ce *Set*, intitulé ici "Retroviseur" (Listing 4.1).

```
<myConfig>
        <parts>
                // Configuration d'un CATPart
                <part>
                        <name>retro-parametre</name>
                        <interesting_sets>
                                <set>Retroviseur</set>
                        </interesting_sets>
                        <interesting_parameters>
                                // Aucun paramètre spécifié
                        </interesting_parameters>
                </part>
        </parts>
</myConfig>
```

Listing 4.1 – Configuration de l'objet CAO

```
[server] // Configuration globale du serveur
// Port sur lequel va écouter le serveur
port = 2705

[stereo] // Configuration de la stéréoscopie
// Mode = "none" / "anaglyph" / "passive_left" / "passive_right" / "active"
mode = "active"
// Distance inter-occulaire (donnée dans la scène réelle)
eye_separation = 0.064
// Facteur d'échelle entre la scène réelle et la scène virtuelle
virtual_scale = 1.0

// Géométrie de l'écran
[screen]
corner_BL_x = -2.4        //          y
corner_BL_y = 0.0         //          |__ z
corner_BL_z = 1.35        //          /
                          //         x
corner_BR_x = -2.4
corner_BR_y = 0.0         // TL ------------ TR
corner_BR_z = -1.35       //    |              |
                          //    |              |
corner_TR_x = -2.4        //    |              |
corner_TR_y = 2.7         // BL ------------ BR
corner_TR_z = -1.35

corner_TL_x = -2.4
corner_TL_y = 2.7
corner_TL_z = 1.35

[window] // Configuration de la fenêtre d'affichage (taille et position)
width = 1080
height = 1080
posx = 0
posy = 0
```

Listing 4.2 – Configuration d'un serveur DRS

Ainsi, lors de la création de la session immersive, seules les données du GHC des ensembles spécifiés seront récupérées. Dans le cas où aucune configuration n'est fournie, le GHC de l'objet sera entièrement parcouru et analysé pour récupérer toutes les données, ce qui n'est pas forcément souhaitable.

En ce qui concerne l'affichage, la seule préparation va consister à paramétrer le serveur DRS lancé sur chaque machine gérant l'affichage sur un écran. Chaque serveur a sa configuration propre, dépendante de la géométrie de l'écran : position et orientation dans l'environnement virtuel et dimensions des bords. Le Listing 4.2 montre les principaux paramètres à régler : le port par lequel la communication avec le client principal va se faire, les paramètres pour la stéréoscopie, les coordonnées des coins de l'écran, les paramètres de la fenêtre d'affichage. Les paramètres de ce listing sont ceux d'un écran carré de 2.7m de côté, dont la résolution est de 1080 x 1080 pixels et qui correspond à l'écran gauche de EVE, le CAVE du LIMSI-CNRS.

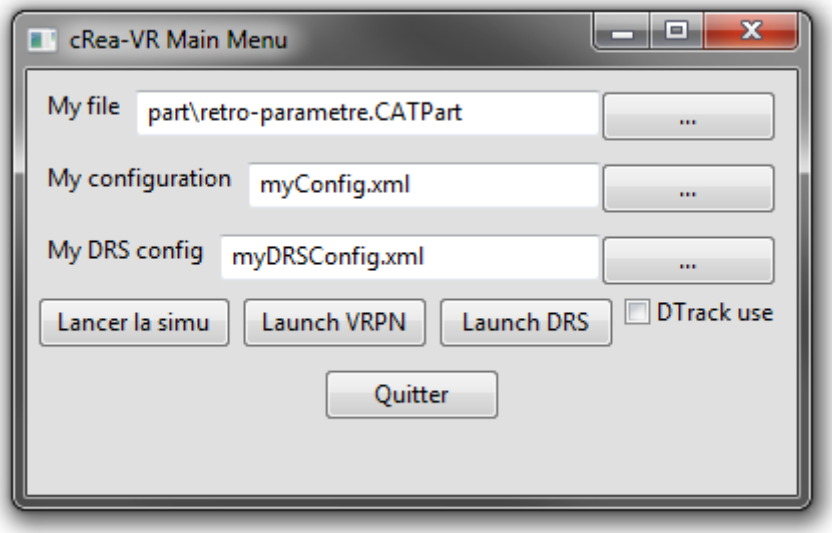


FIGURE 4.8 – Menu principal pour le lancement de CREA-RV : choix du fichier d'objet à charger (*My file*), choix de la configuration associée à cet objet (*My configuration*), choix de la configuration DRS (*My DRS config*), choix de l'utilisation de DTrack, et boutons de lancements.

4.4.3.2 Création de la session immersive

La première étape de la création de la session immersive est le lancement du système de visualisation sur les différentes machines du cluster qui vont servir à l'affichage. L'affichage sur chaque écran est géré par un serveur DRS qu'il faut donc lancer avant le démarrage de la session (figure 4.6). Chaque serveur attendra que le client principal géré par le PC maître soit lancé et commence la distribution des données à afficher.

Lorsque le composant d'affichage est prêt, CREA-RV peut être lancé. Ce lancement se fait via une première interface qui permet de sélectionner le fichier CAO à charger, de spécifier la configuration qui l'accompagne, de lancer le serveur VRPN et enfin de lancer la session immersive (figure 4.8). Le lancement du serveur VRPN, permettant de récupérer les données des différents périphériques, ne se fait qu'une seule fois (indépendemment du nombre de sessions). Tant qu'il n'est pas lancé, l'utilisateur a la possibilité de le faire depuis l'interface principale, sinon le bouton est grisé. Lorsque tous les fichiers (CAO et configuration) sont spécifiés, la session immersive peut être créée et lancée.

La création de la session immersive commence par la création d'une session CATIA (`CATSession`) et l'ouverture du document (`CATDocument`) correspondant à notre objet CAO. Ensuite le GHC de l'objet est parcouru et analysé selon l'algorithme 1 décrit en section 3.2.3. On récupère ainsi les opérateurs (sous le type générique `CATISpecObject`, comprenant les opérateur volumique de type `CATIShape` et surfacique de type `CATIGSM*`) et leurs paramètres. Les données sont ainsi récupérées et stockées dans nos structures d'encapsulation. Ensuite pour chaque opérateur, nous allons récupérer la liste des éléments qui constituent le B-Rep qui leur est associé. Cette liste d'éléments de B-Rep, que l'on manipulera sous forme de `CATIBRepAccess` est en suite traitée pour récupérer toutes les données nécessaires et pour construire la représentation graphique assurée par DRS. Cette conversion de B-Rep en représentation visuelle (voir Algorithme 3) se fait en récupérant la tesselation de chaque élément

Algorithme 3 Conversion d'un élément de B-Rep CATIA en élément visuel DRS

```
Si Face Alors
    Initialisation du BGMesh et récupération des cellules CATFace
    Pour chaque Cellule Faire
        Fonction TesselationFace
            Récupération des sommets, des triangles simples / strips / fans
            Pour chaque Sommet Faire
                Ajout du sommet au BGMesh
            Fin Pour
            Pour chaque Triangle Faire
                Ajout du triangle au BGMesh
            Fin Pour
            Pour chaque TriangleStrip Faire
                Ajout du strip au BGMesh
            Fin Pour
            Pour chaque TriangleFan Faire
                Ajout du fan au BGMesh
            Fin Pour
        Fin Fonction
    Fin Pour
    Création du BGSubObject contenant le BGMesh
    Ajout du BGSubObject au BGMetaObject de l'opérateur concerné
Fin Si
Si Arête Alors
    Initialisation du BGPolyLine et récupération des cellules CATEdge
    Pour chaque Cellule Faire
        Fonction TesselationEdge
            Récupération des sommets
            Pour chaque Sommet Faire
                Ajout du sommet au BGPolyLine
            Fin Pour
        Fin Fonction
    Fin Pour
    Création du BGSubWire contenant le BGPolyLine
    Ajout du BGSubWire au BGMetaObject de l'opérateur concerné
Fin Si
Si Sommet Alors
    Initialisation du BGPointCloud et récupération des cellules CATVertex
    Pour chaque Cellule Faire
        Fonction TesselationVertex
            Récupération des sommets
            Pour chaque Sommet Faire
                Ajout du sommet au BGPointCloud
            Fin Pour
        Fin Fonction
    Fin Pour
    Création du BGSubPointSet contenant le BGPointCloud
    Ajout du BGSubPointSet au BGMetaObject de l'opérateur concerné
Fin Si
```

topologique permettant d'obtenir les sommets, lignes et triangles que l'on transpose objets DRS selon l'association illustrée par la figure 3.6.

Lorsque toutes les données sont récupérées, et que la visualisation est prête, le client DRS principal contenant toutes les données graphiques commence à distribuer l'affichage aux serveurs en attente. Le composant d'interaction est ensuite lancé, activant l'instance VRPN qui récupère en boucle les informations des périphériques de RV (*flystick*, *tracking*). L'utilisateur peut alors commencer à évoluer dans l'environnement immersif.

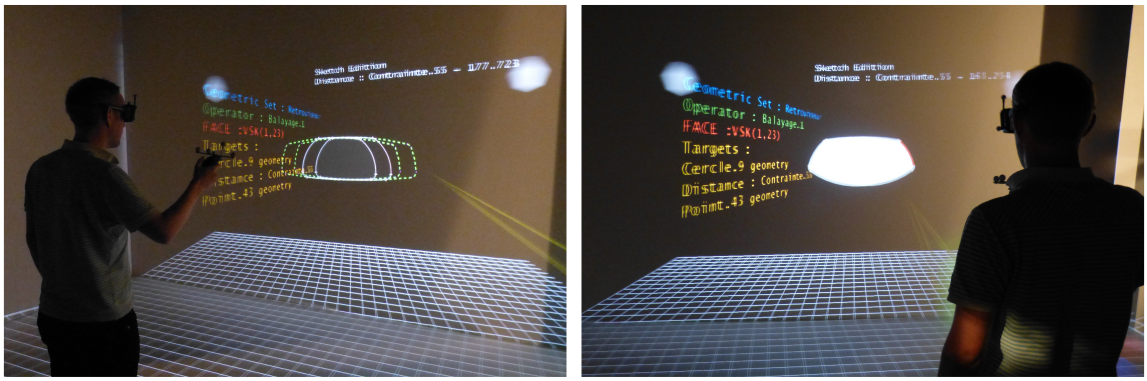


FIGURE 4.9 – Session avec CREA-RV : l'utilisateur modifie un rétroviseur conçu sous CATIA dans un dispositif immersif de type CAVE.

4.4.3.3 Interaction en environnement immersif

Dans l'environnement immersif, l'utilisateur muni de ses lunettes stéréoscopiques actives et de son *flystick*, va pouvoir se mouvoir autour de la maquette virtuelle — ici le rétroviseur — et interagir avec celle-ci (Figure 4.9). Le composant d'interaction met à jour continuellement les informations de *tracking* de la tête de l'utilisateur et du *flystick*. Ces informations sont envoyées au composant visuel DRS qui met à jour la visualisation pour assurer le point de vue exact de l'utilisateur à chaque instant. De même, la représentation visuelle du *flystick* (un interacteur ayant la forme d'un *laser beam*) est mis à jour avec les informations correspondantes. Tout ceci permet à l'utilisateur de se situer complètement dans l'environnement immersif. Concernant l'interaction, on distingue quatre phases :

- l'**exploration** : la phase d'exploration permet à l'utilisateur de pointer les différents éléments de la maquette virtuelle et de visualiser les informations de conception liées à ces éléments.
- la **sélection** : en phase de sélection, l'utilisateur peut justement choisir un élément de B-Rep (sommet, arête ou face) et consulter la liste des paramètres qu'il peut modifier.
- la **modification** : lorsqu'un élément de B-Rep est sélectionné, l'utilisateur dispose donc de la liste des paramètres modifiables. Il peut alors en choisir un et déclencher une interaction permettant de modifier la valeur du paramètre en question.
- la **validation** : une fois que l'utilisateur est satisfait de sa modification, il lance la validation de sa modification ce qui permet de mettre à jour l'objet — on ne choisit pas ici les différentes stratégies de mise à jour — et de visualiser l'impact (complet) de sa modification.

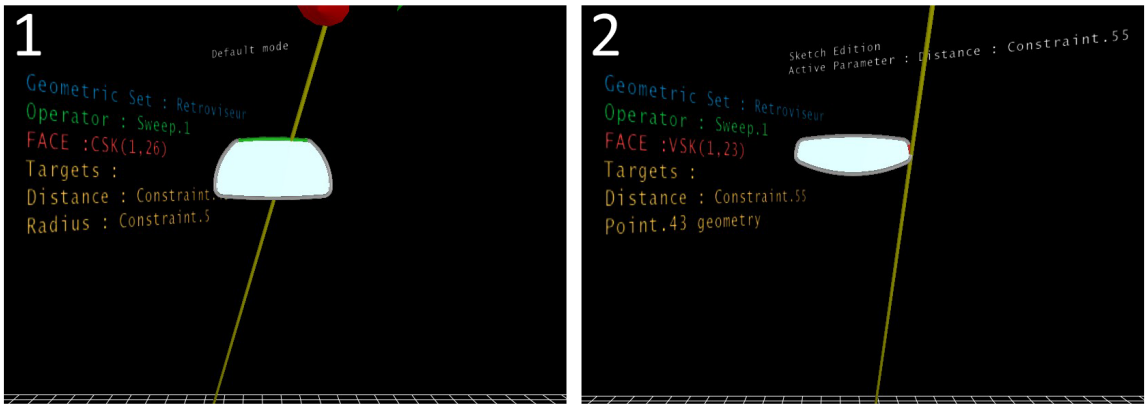


FIGURE 4.10 – Exemple de la modification d'un rétroviseur en utilisant CREA-RV. (1) Phase d'exploration de l'objet. (2) Phase de sélection d'un élément topologique (ici une face).

Exploration

Ainsi la figure 4.10 présente la **phase d'exploration** de la maquette virtuelle (image gauche). Grâce au laser virtuel piloté par le *flystick* — représenté en jaune et dont l'origine est représentée par trois axes — l'utilisateur peut pointer les différents éléments topologiques, alors signalés par la couleur verte. Un panneau d'information est affiché sur la gauche de l'objet. La première ligne de ce panneau, en bleu, indique de quel "Set Géométrique" est issu le B-Rep auquel appartient l'élément pointé (en l'occurrence ici le Set Géométrique intitulé *Retroviseur*). La deuxième ligne, en vert, indique le nom de l'opérateur du GHC générateur de l'élément de B-Rep pointé (ici l'opérateur *Sweep.1*). La troisième ligne, en rouge, indique le type et la liste d'étiquettes issue du *multi-labelling* de l'élément. Ici l'élément pointé est une surface topologique, une face, générée par le balayage et étiquetée par le *label* `CSK(1,26)`, ce qui correspond à l'élément 26, un cercle, de l'opérateur esquisse 1 (cet élément est précisément un des ancêtre de la surface). Enfin la dernière partie du panneau, en orange, indique les cibles accessibles depuis l'étiquetage. Ici le *label* `CSK(1,26)` permet d'agir sur les deux contraintes d'esquisses que sont la distance "Contrainte.46" et le rayon "Contrainte.5".

Sélection

Sur la figure 4.10, on voit une **phase de sélection** (image droite). A la différence de la phase d'exploration, la sélection d'un élément va être repérée par la couleur rouge. Le panneau (sur la gauche) affiche le même type d'information sur l'élément de B-Rep. Par ailleurs, un deuxième panneau d'information (ici au dessus du rétroviseur) affiche la fonctionnalité en cours. Ici suite à la sélection de la surface topologique désormais rouge, nous sommes rentrés dans le mode d'édition d'esquisse (*Sketch Edition*). En effet, la cible accessible depuis l'élément sélectionné est une distance ("Contrainte.55"), contrainte de l'esquisse 1. Ainsi en plus d'afficher le mode, le panneau affiche aussi lorsqu'il s'agit d'une modification, quel est le paramètre actuellement activé.

Modification

La **phase de modification** est illustrée par la figure 4.11. Suite à la précédente sélection, une modification d'esquisse est en cours, et plus spécifiquement la contrainte de distance

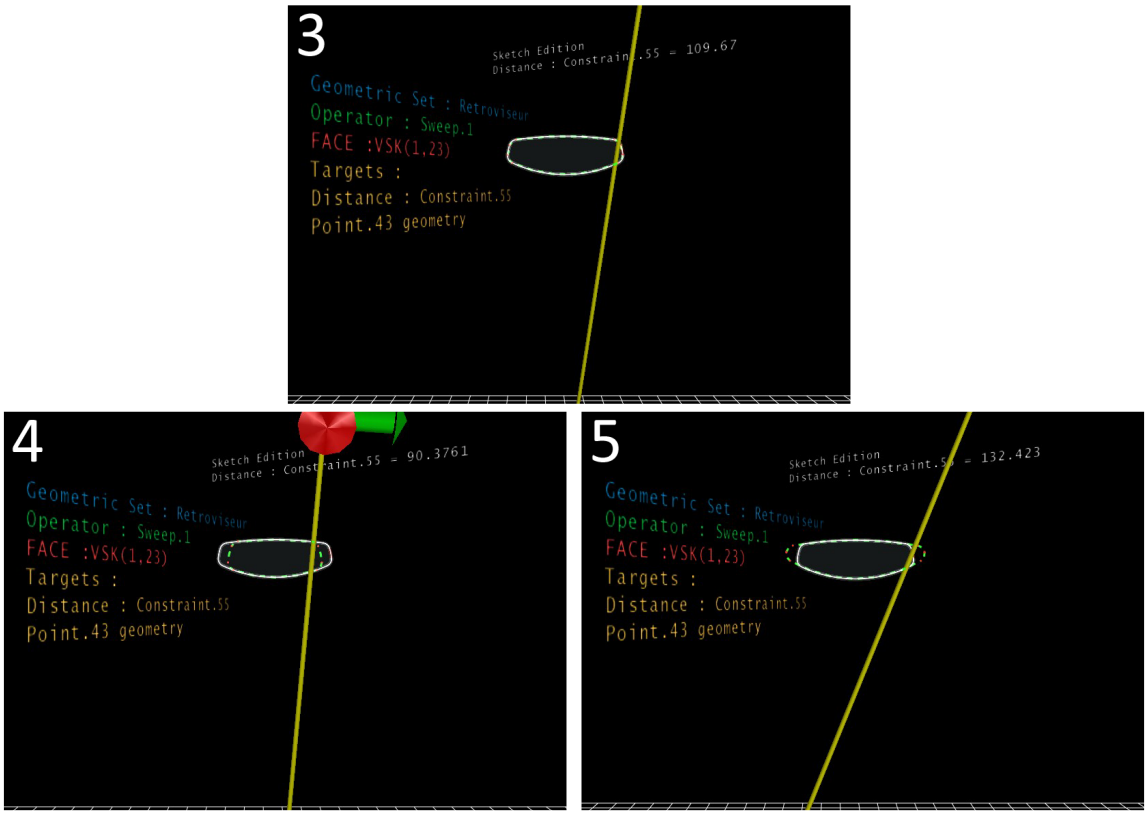


FIGURE 4.11 – Exemple de la modification d'un rétroviseur en utilisant CREA-RV. (3) Activation de la modification. (4) Modification en cours. (5) Fin de la modification.

"Contrainte.55". Ainsi par son interaction, l'utilisateur va pouvoir ajuster en temps-réel la valeur de la contrainte. La mise à jour de l'esquisse se fait instantanément et l'utilisateur peut donc apprécier en direct l'impact, sur l'esquisse, de son interaction (image 4 et 5). Le système de visualisation tel qu'il est interfacé avec le modèle de données permet en effet d'afficher un rendu spéficique lors de la modification. Ici, la stratégie de mise à jour est celle schématisé en rouge et orange foncé dans la figure 4.5 (première mise à jour locale puis mise à jour globale). Ainsi seul l'opérateur directement concerné par la modification est mis à jour, ses éléments de B-Rep associés sont mis en valeur (ici l'esquisse en pointillé vert), alors que le reste de l'objet devient transparent. Le panneau d'information situé au dessus de l'objet affiche lui aussi en temps réel la valeur courante du paramètre modifié : 110 (109.67) au début de la modification (image 3), 90.37 en cours de modification (image 4), puis 132.42 à la fin de la modification (image 5). Ici la valeur de la modification est directement impactée selon le déplacement dans l'espace du *flystick* (modification en continu).

Validation

Lorsque l'utilisateur est satisfait de sa modification, il va alors passer à la **phase de validation** de sa modification (figure 4.12). En fait il va déclencher la mise à jour globale (fonctionnalité CATIA) ce qui réévalue le GHC complètement. Ici la mise à jour globale va en fait constister à lancer un `Update` du Set Géométrique "Retroviseur". L'objet CAO dans sa version entièrement actualisée est alors fourni à DRS et l'utilisateur peut vérifier son aspect (image 6). Enfin lorsque l'utilisateur veut sauvegarder l'état actuel de l'objet CAO, soit en

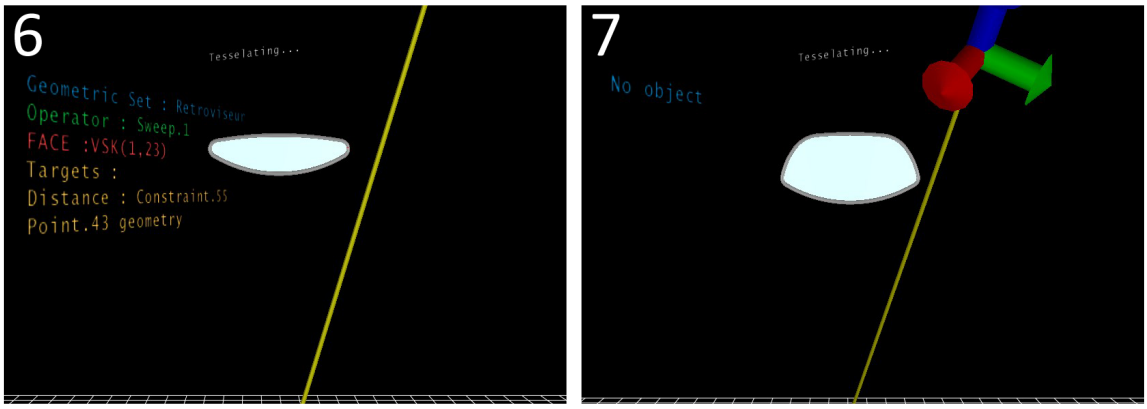


FIGURE 4.12 – Exemple de la modification d'un rétroviseur en utilisant CREA-RV. (6) Validation de la modification. (7) Sauvegarde de l'objet CAO et fin de la session immersive.

cours soit à la fin de la session de travail, il le peut. L'objet (final) est alors sauvegardé dans son format initial (image 7). Revenu à son poste de travail l'utilisateur peut alors recharger l'objet CAO pour apprécier les modifications et continuer son travail (figure 4.13).

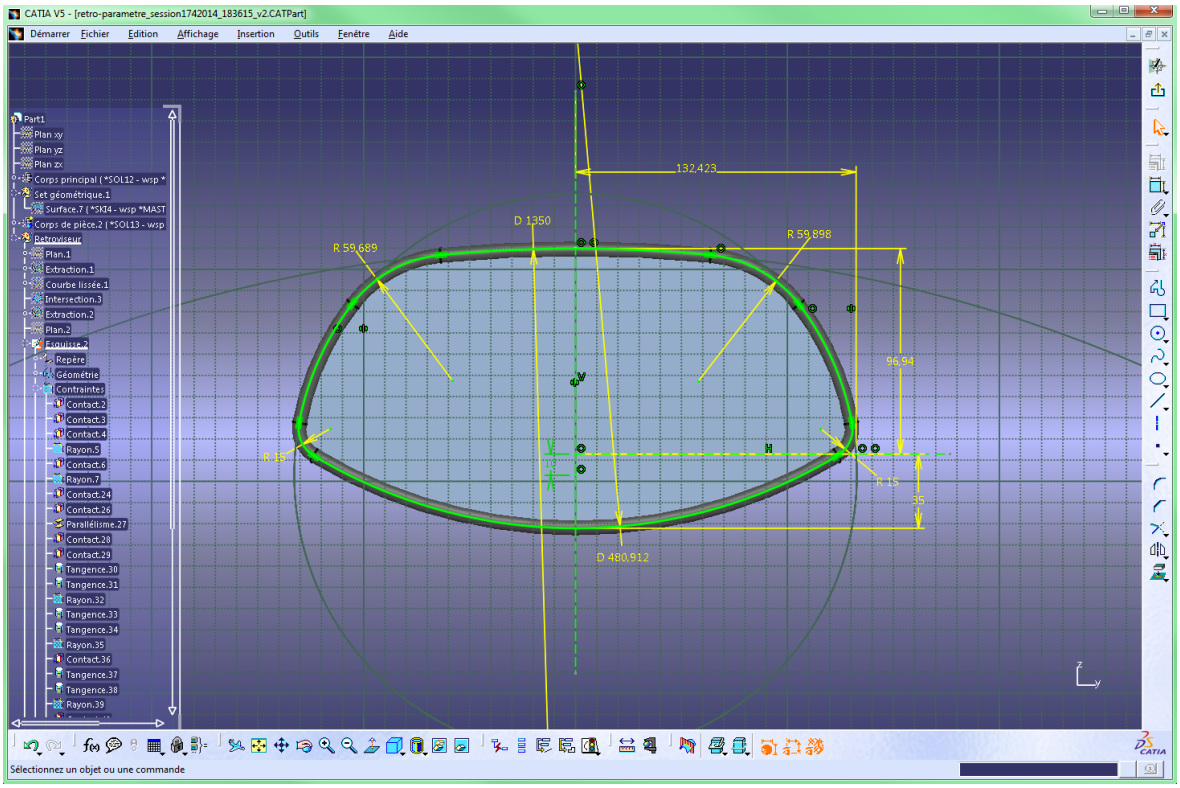


FIGURE 4.13 – Rétroviseur modifié avec CREA-VR. L'utilisateur peut directement recharger la maquette CAO sur son poste de travail (ici dans CATIA V5) et vérifier les modifications apportées par CREA-RV

4.4.4 Optimisations

En plus des stratégies de mise à jour de la maquette CAO, nous avons pu mettre en place pour cette implémentation d'autres optimisations qui concernent notamment le système de visualisation. Comme nous l'avons dit plus haut, nous créons une représentation visuelle de la maquette CAO en récupérant son approximation polyédrique par l'intermédiaires des fonctionnalités du système CAO cible. Ces traitements de données peuvent être optimisés par le paramétrage de certaines données. En particulier, les fonctionnalités de tessellation des objets, qui permettent de récupérer une discrétisation géométrique des éléments (triangles, lignes, points), est paramétrable via :

- le ***sag*** qui définit la distance maximum entre un segment (de ligne droite) et l'objet à tesseller
- le **pas** qui définit la taille maximum d'un segment
- l'**angle** qui définit l'angle maximum entre les normales à chaque extrémité des segements

Agir sur ces paramètres nous permet de gérer la quantité d'éléments graphiques que l'on va traiter à chaque mise à jour. Il est ainsi possible pendant la session immersive, via un menu pour le moment déporté sous forme de widget 2d, de spécifier la valeur de ces paramètres. Lorsque la modification de l'objet ne nécessite pas une esthétique particulière, il peut ainsi être intéressant de définir des valeurs grossières pour le *sag* et le pas. Au contraire, lorsque l'on voudra visualiser et évaluer précisément la forme de l'objet, des valeurs très faibles de ces paramètres de *sag* et de pas, permettront d'avoir une représentation visuelle extrêmement fine et précise. La figure 4.14 illustre par exemple les différences de rendu en fonction du *sag*.

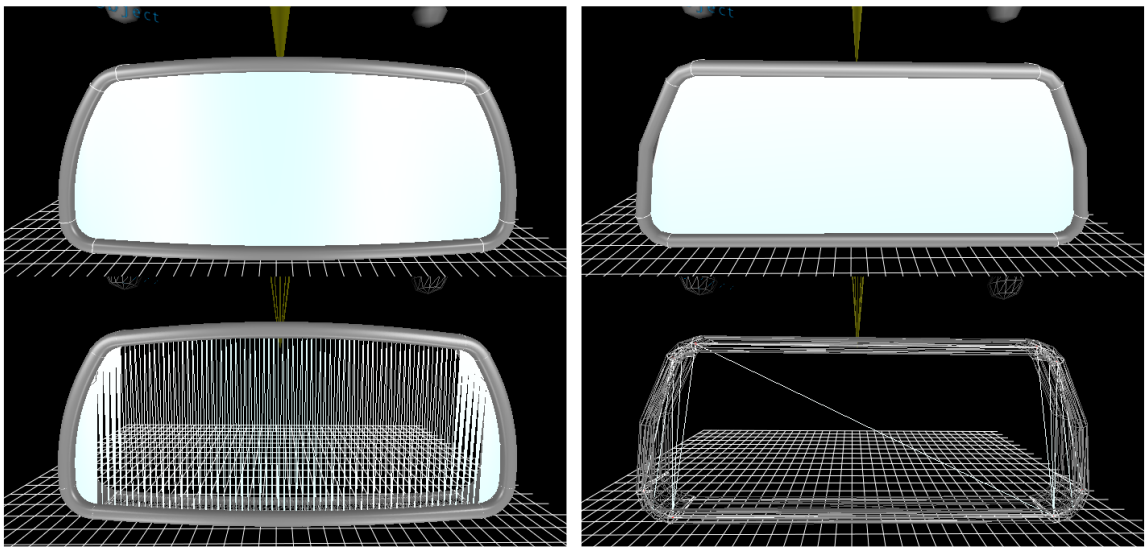


FIGURE 4.14 – Exemple d'optimisation du rendu graphique. Plus la valeur du *sag* est faible (image de gauche, sag = 0.001) plus la représentation visuelle de l'objet sera fine. Au contraire, une valeur élevée (image de droite, sag = 10) générera moins d'éléments graphiques pour une représentation plus grossière de l'objet. Les images du dessous sont la représentation filaire (*wireframe*) des objets.

4.4.5 Retours des utilisateurs

CREA-RV a été présenté aux différents métiers de la conception, qu'ils soient directement ou indirectement concernés par les technologies en jeu. Ainsi, des stylistes aux architectes véhicules, les utilisateurs ont pu approcher et tester (lorsqu'ils le désiraient) le démonstrateur. J'ai établi, en concertation avec l'entité de PSA **Moyens de Simulation et de Réalité Virtuelle (MSRV)** à laquelle j'étais rattaché, un questionnaire nous permettant de récupérer les différents avis et besoins des utilisateurs en ce qui concerne la conception en Réalité Virtuelle et plus précisément les activités autour de mon démonstrateur.

Utilisation de la RV pour la conception

En ce qui concerne l'utilisation de la RV dans les activités de conception, on note que les utilisateurs sont très demandeurs lorsqu'il s'agit de visualiser à l'échelle 1:1, et d'appréhender les formes ou le produit avec une vue subjective. L'architecture véhicule est la principale thématique où l'on va évaluer différentes paramétries du véhicule en immersif. Enfin, que les utilisateurs aient ou non déjà eu recours à la RV, l'intérêt de la visualisation en 3d à la première personne, et la possibilité de pouvoir rapidement prendre une décision et d'éviter la fabrication d'une maquette physique sont pointés comme particulièrement pertinent. Ce sont ici des confirmations de besoins et d'intérêts maintenant bien exprimés.

Impressions sur CREA-RV

En ce qui concerne le démonstrateur — utilisation de CREA-RV pour modifier en immersif un rétroviseur conçu sous CATIA V5 — les utilisateurs notent comme principaux points forts le fait de visualiser en temps-réel les modifications apportées à la maquette et le fait de pouvoir accéder en profondeur à CATIA V5. Cependant, les avis convergent sur le manque d'effet "waouh". En fait c'était un avis que j'avais tout à fait anticipé : le démonstrateur en l'état fait plutôt office de preuve de faisabilité (preuve que l'on peut accéder à tout paramétrage de CATIA V5), et non pas de "démonstrateur final" peaufiné. Le visuel (manque d'environnement, objet trop seul) et l'interaction (manipulation *flystick*) sont à juste titre pointés comme "à améliorer". Nous verrons d'ailleurs dans le chapitre 5 (section 5.3) les différentes perspectives quant à l'amélioration de l'interaction. Enfin les utilisateurs sont demandeurs d'une démonstration autour d'un cas d'utilisation plus complexe (un habitacle complet par exemple), afin d'observer la modification d'un modèle ressemblant à ce qu'ils visualisent actuellement en session de travail immersive.

4.5 Conclusion

Nous avons présenté dans ce chapitre les différents apports de notre architecture et de notre modèle de données pour l'interfaçage RV-CAO. Le premier point concerne la possibilité d'appliquer le principe d'édition implicite à des objets CAO déjà conçus dans des logiciels CAO paramétriques basés sur les caractéristiques de forme. L'accès à l'API de ces logiciels permet ainsi de remplir les différentes structures de données de notre modèle pour gérer et modifier l'objet CAO en environnement immersif.

L'introduction du *multi-labelling*, mécanisme qui attache une ou plusieurs étiquettes aux éléments de B-Rep selon leur opérateur générateur et selon leur historique (leurs parents), permet d'accéder à différents éléments et différents niveaux du GHC des objets CAO à partir

d'une seule sélection topologique. L'utilisateur peut parcourir simplement la liste d'éléments accessibles depuis sa sélection de façon séquentielle. Cependant, l'architecture en place permet une gestion plus évoluée, notamment au traver du moteur d'inférence. Plutôt que d'interagir en sélectionnant finement des éléments topologiques, l'édition implicite peut également se faire au travers de différents schémas d'interaction : à partir des opérateurs (via leur B-Rep complet), ou de groupes d'opérateurs (via un ensemble de B-Rep).

Une supervision des mises à jour est également possible ce qui permet d'assurer le caractère temps-réel de l'interaction dans l'environnement immersif. Le déclenchement des mises à jour peut en effet suivre différentes stratégie selon la complexité de l'objet CAO, allant de la mise à jour globale de l'objet, à une mise à jour locale et progressive. Ces stratégies s'accompagnent de la possibilité de tracer les erreurs lors des modifications et de guider l'utilisateur afin d'assurer la validité de la maquette CAO en tout temps.

Ces différents apports ont finalement été montrés au travers d'une preuve de concept qui a consisté en l'implémentation de notre architecture autour de CATIA V5 grâce à un accès au CAA CATIA (son API).

Chapitre 5

Perspectives de la RV-CAO

Résumé. *L'architecture et le modèle de données que l'on propose dans ces travaux de thèse pour l'interfaçage RV-CAO permettent de franchir, de par leurs apports, un pas de plus dans l'intégration des logiciels commerciaux de CAO dans les environnements immersifs. Les travaux sur le sujet sont cependant loin d'être terminés et de nombreuses pistes s'ouvrent quant à de possibles évolutions de notre modèle, voire à la conception de nouveaux modèles plus adaptés à d'autres étapes de la conception, notamment surfacique. Dans ce chapitre nous donnons donc des perspectives d'évolution de notre modèle, allant de la simple extension (nouveaux opérateurs pris en compte) au rajout de fonctionnalités (nouvelles interactions, moteur d'inférence amélioré). Nous évoquons ensuite une problématique qui pourrait intervenir, à savoir la confrontation de la connaissance de l'utilisateur expert et de la machine. Enfin nous évoquons quelques perspectives pour l'intégration RV-CAO en contexte industriel.*

Sommaire

5.1 Introduction

L'architecture et le modèle de données qui nous proposons dans ces travaux pour l'interfaçage RV-CAO permet d'appliquer le principe de l'édition implicite sur les logiciels de CAO paramétriques basés sur les caractéristiques de forme. Ces logiciels sont principalement utilisés par les industries dans le cadre de leurs activités de conception. Le démonstrateur CREA-RV autour de CATIA V5 est une preuve de faisabilité de cette approche en contexte industriel et permet de franchir un pas dans l'intégration des systèmes CAO commerciaux dans les environnements immersifs. Ceci étant, ces travaux d'interfaçage sont loin d'être terminés et au contraire, de nombreuses possibilités s'ouvrent.

Nous allons donc dans ce chapitre décrire les différentes perspectives pour la RV-CAO. Nous allons dans un premier temps présenter les différentes évolutions directement possibles à partir de nos travaux. Puis nous nous en éloignons progressivement pour évoquer une problématique plus générale à propos de différences d'appréciations du graphe d'historique entre l'homme et la machine. Enfin nous donnerons des pistes pour la RV-CAO en contexte industriel.

5.2 Extension du modèle de données

L'architecture et le modèle de données proposés pour l'interfaçage RV-CAO (revoir chapitre 3) sont fonctionnels, preuve en est l'implémentation autour de CATIA V5, CREA-RV (revoir section 4.4, chapitre 4). Le principe de l'édition implicite peut donc s'appliquer aux systèmes CAO lorsqu'ils offrent un accès à leurs fonctionnalités.

Le modèle d'interfaçage RV-CAO est générique dans sa forme. Ainsi j'ai spécifié différentes structures pour encapsuler les données nécessaires pour la gestion des données. En fonction des besoins des utilisateurs, il est possible de spécifier de nouvelles structures afin d'enrichir notre modèle et d'assurer un plus grand nombre de modifications. De plus, nous avons évoqué dans le chapitre précédent, les apports offerts par ces travaux, en particulier la gestion et la supervision des mises à jour.

Nous présentons dans cette sous-section certaines extensions du modèle de données, à savoir la spécification de nouvelles structures de données pour la gestion de nouveaux opérateurs, l'étiquetage correspondant à ces nouvelles données, et enfin l'évaluation et la mise en place des différentes stratégies de gestion des mises à jour des données CAO.

5.2.1 Opérateurs et étiquetages

Notre modèle comprend les opérateurs classiques : chanfrein, congé, extrusion, rainure, trou pour les volumiques, balayage et remplissage pour les surfaciques, et bien sûr les esquisses. La validitié de notre approche a pu être démontrée à partir de ces différents opérateurs, mais il est évident qu'ils ne sont pas suffisants pour un véritable passage à l'échelle industrielle. Il n'est pas possible d'anticiper tous les besoins des utilisateurs, ni toutes leurs façons de modéliser, mais il faut néanmoins couvrir le maximum d'opérateurs classiques, qu'ils soient volumiques ou surfaciques.

5.2.1.1 Opérateurs volumiques

Voici les quelques opérateurs classiques qui se doivent d'être absolument intégrés à notre modèle :

- **Nervure** : une nervure est l'équivalent d'une extrusion en termes d'ajout de matière, mais en lieu et place d'un axe directeur, la matière est créée à partir d'une esquisse selon une courbe (comme l'est une rainure).
- **Poche** : alors qu'une extrusion se base sur une esquisse et ajoute de la matière suivant un axe directeur, la poche, définie de la même manière, enlève de la matière.
- **Révolution** : la révolution est la création de matière par la rotation d'une esquisse autour d'un axe.
- **Gorge** : la gorge est définie comme une révolution mais enlève de la matière.
- **Coque** : la coque permet de creuser dans la matière ou d'en créer, selon des paramètres d'épaisseurs.

Ces opérateurs doivent ainsi être intégrés au sein de notre modèle par la création de leurs structures d'encapsulation respectives. Dans leurs compositions, ces structures ne diffèreront pas tellement les unes par rapport aux autres (dont celles déja créées), la variation résidant principalement dans le comportement des fonctions qui leur sont associées.

Enfin, il faut remarquer que plusieurs de ces opérateurs, dont certains qui existent déjà dans notre modèle, peuvent connaître des variations dans leur type et leur définition. C'est le cas, en particulier, pour l'extrusion, la poche et le trou. A l'heure actuelle, seule la version "simple" de ces opérateurs est prise en compte dans notre modèle, à savoir une définition uniquement selon un axe et une distance. Cependant, les limites de l'extrusion et de la poche peuvent être définies selon d'autres modes, notamment par rapport à d'autres éléments topologiques (par exemple jusqu'à la surface suivante), et ainsi la valeur même de la limite n'est plus d'actualité. Il faut donc également compléter la gestion de ces opérateurs afin que le maximum de possibilités (facilement éditables) soient disponibles pour l'utilisateur.

5.2.1.2 Opérateurs surfaciques et esquisses

La plupart des opérateurs que l'on vient de citer sont basés sur des esquisses. Et effectivement, ces esquisses sont une composante primordiale des systèmes CAO basés sur les caractéristiques de forme. Notre modèle de données comprend déjà la gestion des opérateurs d'esquisse et de certaines contraintes. Cependant, seuls certains éléments géométriques d'esquisses sont actuellement pris en compte — point, ligne, courbe, spline, cercle — et pour des modifications simples (changement de position). Afin d'offrir un outil plus complet aux utilisateurs, nous nous devons de proposer une modification plus complète de ces éléments, et notamment des splines en permettant l'accès et la modification des points de passage/-contrôle. De même toutes les différentes caractéristiques géométriques des primitives, comme par exemple les rayons des cercles et des ellipses, doivent être intégrées.

Cette gestion améliorée des esquisses et de leurs éléments permettra de mieux intégrer les opérateurs surfaciques que l'on peut également trouver dans ces différentes systèmes CAO. Notre modèle comprend déjà la spécification des opérateurs de remplissage et de balayage, sans proposer leur modification, mais qui permet d'accéder aux éléments générés par ces opérateurs. En fait les opérateurs surfaciques sont principalement définis à partir de courbes et de splines, et on voit donc que nous ne pourrons proposer une modification de type surfacique qu'en intégrant complètement la gestion des courbes. Ceci étant, ces opérateurs surfaciques contiennent généralement des paramètres et des options qu'il convient de prendre en compte dans notre modèle. Il serait par exemple opportun d'intégrer la création de surface sphérique, de surface par révolution et de surface par extrusion. Ces opérateurs ressemblent particulière-

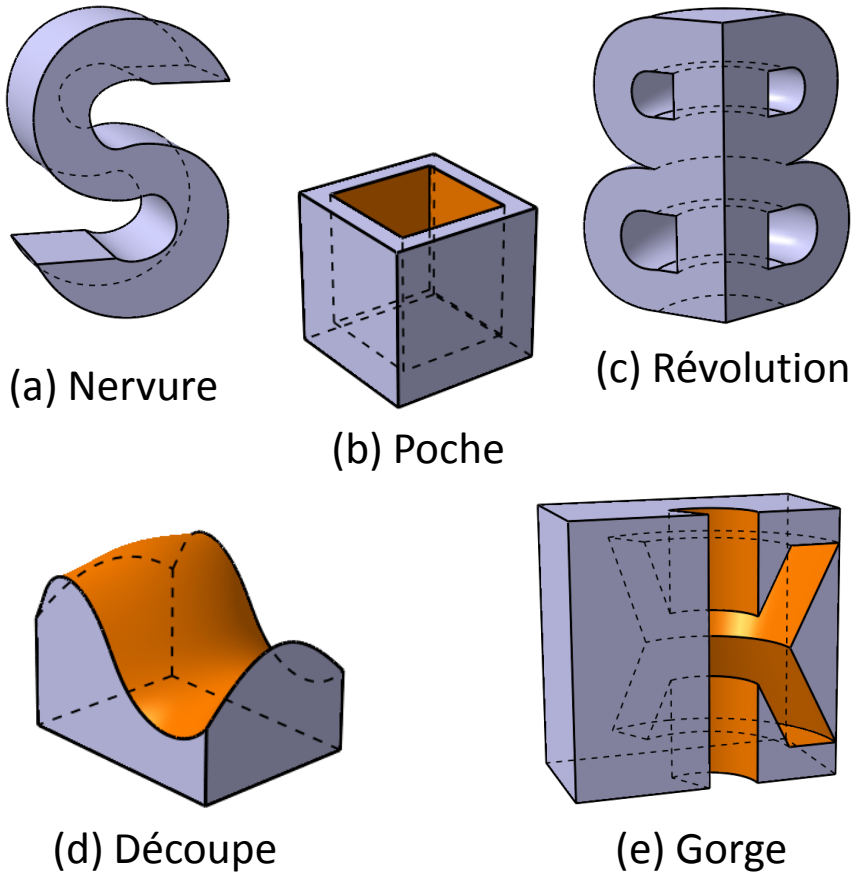


Figure 5.1 – Exemple d'opérateurs volumiques et surfaciques à intégrer au modèle de données.

ment à certains opérateurs volumiques (notamment la révolution et l'extrusion) à la différence près que l'esquisse de base n'est pas une esquisse fermée mais plutôt une courbe.

5.2.1.3 Opérateurs booléens

Dans la précédente approche basée sur OpenCascade, le démonstrateur VRAD intégrait les opérateurs booléens : union, intersection, soustraction. Dans les systèmes paramétriques basés sur les caractéristiques de forme, ces opérateurs vont par exemple permettre d'assembler les différents composants paramétrés afin de créer un ensemble complexe. Par ailleurs, ils sont parfois "intégrés" dans certaines *features* comme par exemple les trous aveugles ou débouchants. En fait, notre démarche est axée sur la modification paramétrique d'objets existants et je me suis principalement axé sur les opérateurs paramétrés standards. Ces opérateurs booléens ne présentent pas de paramètre mais vont agir sur les objets directement. Notre approche nécessite néanmoins de les prendre en compte dans le cas où l'on voudrait accéder à ou sélectionner des éléments de B-Rep générés directement par ces opérateurs. Dans le cas contraire, nous ne pourrions remonter dans le graphe d'historique depuis ces éléments. Enfin dans l'optique d'assurer la création d'objets CAO en plus de la modification (voir section 5.2.4), l'intégration de ces opérateurs booléens serait un plus.

5.2.1.4 Transformations

Les opérations de transformation des objets avaient été évoquées dès le début de ces travaux de thèse. Elles ne sont actuellement pas prises en compte par notre modèle de données qui a été principalement établi autour de cas d'usages précis. Cependant, dans l'optique de proposer toujours plus d'interaction et de modification avec les objets CAO, ces opérateurs doivent être intégrés, et notamment les opérateurs de translation, rotation et changement d'échelle qui sont paramétrés et donc sujets à modification en environnement immersif.

5.2.1.5 Étiquetage et inférence

L'originalité de l'approche d'édition implicite, base de nos travaux, est le mécanisme d'étiquetage (*labelling*) des éléments de B-Rep (revoir section 3.3.1, chapitre 3). Il est bien évident que les les spécifications d'encapsulation pour les nouveaux opérateurs doivent s'accompagner de l'extension de ce mécanisme. Cette extension passe dans un premier temps par la création des différentes étiquettes qui seront attachées aux éléments de B-Rep générés par ces opérateurs, ainsi que par la spécification de nouvelles règles Prolog qui permettront au moteur d'inférence de fonctionner avec ces nouveaux éléments (revoir section 3.3.2, chapitre 3).

Les nouvelles étiquettes devront simplement respecter la règle d'unicité, à savoir qu'une étiquette propre à un opérateur ne peut pas avoir le même nom que celle d'un autre opérateur. Il faudra par contre concevoir les nouvelles règles logiques en suivant, par défaut, une certaine intuitivité (de la même façon qu'une face *top* d'une extrusion est associée à la limite *top* de cette même extrusion). Ce "bon sens" sera confirmé ou infirmé par les besoins des utilisateurs lors des phases de test et de validation.

5.2.2 Stratégies de mises à jour

Au delà de la structure même de notre modèle, la gestion des données représente un défi technique qu'il convient d'apprivoiser. À la différence des précédents travaux sur le principe de l'édition implicite, nous ne maîtrisons pas le fonctionnement du GHC du sytème CAO cible : ni sa conception, ni sa gestion et ses mises à jour. Ainsi notre approche consiste à parcourir et analyser ce graphe dès qu'il a été modifié. Plusieurs stratégies sont possibles pour déclencher ces mises à jour (revoir section 4.3 et figure 4.5), mais dans tous les cas, nos traitements n'interviennent que dans un second temps — alors que dans la précédente approche ces traitements pouvaient être inclus dans le processus de mise à jour.

Il est donc nécessaire de mettre en place toutes les différentes stratégies de mises à jour possibles afin de les évaluer. Quelle stratégie (locale / globale / cascade) est la plus performante en termes de temps de calculs et de bonne gestion des données ? Ces différentes stratégies sont-elles contraignantes (on pourrait avoir, dans le cadre de mises à jour locales et/ou en cascade, des interactions impossibles à l'instant t car certaines données ne sont pas à jour) ?

Actuellement, le choix de stratégie de mise à jour se fait de façon statique, une fois pour toute avant la session de travail. Cette problématique d'optimisation des mises à jour est en réalité quasiment uniquement fonction de la complexité de l'objet CAO modifié : plus l'objet est complexe (de même que sa représentation B-Rep) plus il peut y avoir de traitements et donc de potentielles latences. Il faudrait ainsi mettre en place des indicateurs sur la complexité des différents B-Rep. On peut facilement imaginer que chaque structure d'encapsulation des opérateurs contienne le nombre d'éléments de B-Rep ainsi que le nombre de sommets/lignes/triangles qui composent la représentation visuelle de ces éléments. Ceci

donnerait une indication du nombre d'éléments qu'il faudra traiter après la mise à jour du GHC, ce qui permettrait de choisir "dynamiquement" la stratégie de mise à jour à appliquer. Plus l'objet serait compliqué, plus on tendrait vers une mise à jour locale en ce qui concerne l'application directe de la modification, la mise à jour globale intervenant uniquement lors de la validation de cette mise à jour pour évaluer l'impact global. Au contraire, si la complexité le permet, la mise à jour globale serait directement lancée pour appliquer la modification (avec donc visualisation instantanée de l'impact global).

5.2.3 Montée en charge / complexité

La problématique des mises à jour est, comme nous l'avons vu, liée principalement à la complexité des objets CAO que l'on veut modifier. Un des retours des utilisateurs suite à la communication de ces travaux et du démonstrateur CREA-RV au sein de PSA Peugeot Citroën était la demande de voir cette approche de l'édition implicite sur des objets plus complexes que celui présenté (un rétroviseur). Ainsi, afin de répondre à ces attentes et donc aux besoins industriels, il conviendrait de monter en charge progressivement.

Augmenter la complexité des objets modifiés permettra d'aborder deux aspects : les mises à jour (que l'on vient d'évoquer) et la visualisation des données. En effet, en plus du traitement des données du GHC, il faut également compter les traitements de données pour la représentation visuelle avec DRS. Fonctionnant sur le principe client/serveur, avec envoi/réception des données géométriques dès qu'elles ont changé, il va falloir évaluer les performances de cet outil de *clustering* graphique avec des maquettes CAO complexes. Si d'éventuels problèmes apparaissaient, des optimisations du fonctionnement même de DRS, et de son interfaçage avec le modèle de données devront être pensées.

5.2.4 Vers la création

Mes travaux ont porté sur une approche de modification d'objets CAO existants. De fait, lors des revues de projet immersives réalisées dans l'industrie, et en particulier chez PSA Peugeot Citroën, les maquettes virtuelles sont déjà créées, puis évaluées en immersif. Plus exactement, les opérations de modification sur lesquelles nous nous sommes penchés, sont les modifications d'éléments existants : paramètres d'opérateurs, contraintes d'esquisse, coordonnées d'éléments géométriques. Nous n'avons pas pris en compte les modifications consistant à créer, rajouter des éléments, comme par exemple la création de points de contrôle pour la modification d'une spline.

A partir du moment où notre approche permet d'intégrer les fonctionnalités des systèmes CAO, et d'accéder à tout paramétrage de ces systèmes, il serait bien évidemment possible de s'intéresser également à ces fonctionnalités de création d'éléments, qu'elles concernent la création d'éléments géométriques, ou l'instanciation de nouveaux opérateurs. Bien que les premières demandes des utilisateurs concernés par nos travaux n'aillent pas dans ce sens — la modification immersive de paramètres existants étant déjà suffisamment novatrice dans leur façon de travailler — cette perspective permettrait d'élaborer et de compléter notre modèle dans l'optique d'offrir toujours plus de possibilités pour la conception en environnement immersif.

5.3 Interaction intuitive

L'architecture que je propose dans ces travaux intègre une composante d'interaction, permettant aux utilisateurs d'interagir avec la maquette virtuelle. Dans sa spécification, le

moteur d'interaction permet de gérer tout genre de périphérique, qu'ils soient de type "poste de travail" comme la souris et le clavier, ou de type "RV" comme le *flystick* et le *tracking*. L'interaction qui est proposée actuellement par notre architecture, et son implémentation CREA-RV, est élémentaire puisqu'elle consiste à manipuler la maquette virtuelle au travers d'un *flystick* offrant six degrés de liberté (position et orientation) ainsi que quelques boutons de commande. Du fait du défi qu'a représenté l'implémentation de cette approche d'édition implicite sur les logiciels CAO commerciaux, ce composant d'interaction n'a pas pu être développé dans une version plus élaborée.

Pourtant, de par l'expertise du groupe VENISE sur le sujet, de nombreuses pistes avaient été évoquées pour ces travaux, comme la commande naturelle et l'interaction multimodale, ou le guidage haptique. Nous décrivons donc dans cette section les aspects d'interaction dont nous considérons l'intégration comme indispensable pour la suite de ces travaux.

5.3.1 Commandes naturelles

Le groupe VENISE du LIMSI-CNRS s'intéresse depuis de nombreuses années à la mise en place d'interactions dites "naturelles" pour la RV. Les environnements immersifs ont cette spécificité que les moyens d'interaction standards de type "poste de travail", comme le clavier et la souris, n'ont aucun intérêt à y être utilisés. Par interaction "naturelle", nous entendons par exemple la reconnaissance gestuelle et la reconnaissance vocale. Ces moyens d'interactions permettent aux utilisateurs d'interagir en environnement immersif d'une façon plus semblable au réel, à la vie de tous les jours, que les interfaces 2d classiques.

En l'occurrence, le groupe VENISE a depuis plusieurs années développé un système de reconnaissance gestuelle générique, permettant les interactions et les manipulations bi-gestuelles, avec les objets virtuels [28, 29]. Ce système permet de facilement s'interfacer avec les applications de RV, et notamment avec les périphériques de RV de type Fingertracking. Le lecteur voulant en savoir plus sur la reconnaissance gestuelle pourra notamment aller voir l'état de l'art de Mitra et Acharya [121].

Enfin la reconnaissance vocale est reconnue depuis nombreuses années comme permettant d'optimiser l'interaction en environnement virtuel [110]. La commande vocale permet en effet de se passer de clavier et de souris, par la spécification de commandes, par mot clé ou langage naturel. Ces commandes directes sont idéales pour accéder à des éléments traditionnellement accessibles au travers de menus.

5.3.2 Multimodalité

Comme le définissent Bourdot et al. (chapitre 13 de [75]) :

> *Une modalité est une forme de représentation d'information à travers un média ou un canal sensoriel. On parle de multimodalité lorsque plusieurs modalités sont utilisées dans une application et sont combinées non indépendamment les unes des autres.*

C'est justement sur cette thématique de la multimodalité que s'est particulièrement spécialisé le groupe VENISE. Les précédents travaux sur l'édition implicite offraient plusieurs modalités d'interaction (gestes via un *flystick*, voix) qui n'étaient pas combinées et restaient indépendantes les unes par rapport aux autres [32]. Au contraire, certains travaux du groupe, et notamment en partenariat avec PSA Peugeot Citroën (au sein du projet PerfRV-2), ont

porté sur la fusion de ces différentes modalités d'interaction : les modalités en entrée de l'application sont combinées, on parle de **fusion multimodale** [1] [112]. Ces précédents travaux, bien que n'ayant pu être intégrés à ceux de ma de thèse, ont été menés et continués en parallèle [115].

Ainsi, l'intégration de plusieurs modalités d'interaction (geste, voix, haptique) ainsi que l'intégration de l'approche multimodale à notre modèle d'interfaçage RV-CAO permettraient d'offrir aux utilisateurs un panel d'interactions plus intuitives et naturelles.

5.3.3 Guidage haptique / pseudo-haptique

L'interaction haptique permet d'aider l'utilisateur dans sa démarche de conception, en lui transmettant différentes informations comme par exemple le ressenti de contraintes de manipulation lors de l'édition CAO. Plusieurs travaux ont été menés en ce sens au sein du groupe VENISE s'intéressant notamment à l'édition d'objets CAO à l'aide de périphériques haptiques [128,131]. Ces approches présentent de nombreux intérêts, notamment dans les différentes phases d'interaction avec la maquette virtuelle comme la sélection et la modification.

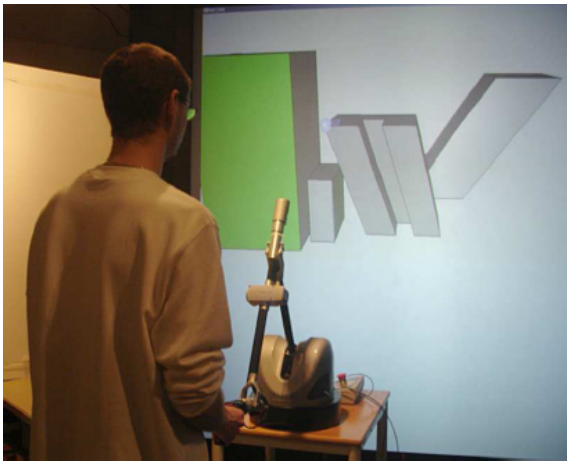


Figure 5.2 – Édition CAO à l'aide d'un périphérique haptique. Ici un utilisateur réalise une tâche d'extrusion, en étant guidé haptiquement selon l'axe de l'extrusion (Picon [128])

Bien paramétrés ces outils haptiques vont permettre de plus facilement sélectionner les différents éléments de B-Rep (sommets, arêtes et faces) par l'intérmédiaire de méthodes d'attraction par exemple. De même lors des modifications, lorsque des contraintes de manipulation sont identifiées (une extrusion suivant un axe, une modification d'esquisse dans un plan), ces périphériques vont permettre un guidage offrant à l'utilisateur une interaction aidée et optimisée (figure 5.2). De plus, les périphériques haptiques ne cessent d'évoluer, et il devient possible d'avoir une interaction haptique dans un environnement immersif complet, avec notamment l'arrivée de Scale 1 [2], dispositif à grande échelle de Haption, que vient récemment d'acquérir le groupe VENISE.

1. Voir en particulier le démonstrateur MalCoMIICs : http://youtu.be/AIMJa4gKaaA
2. http://www.haption.com/site/index.php/fr/products-menu-fr/hardware-menu-fr/scale1-menu-fr

L'interaction guidée peut néanmoins se faire sans disposer de dispositif haptique, par l'intermédiaire de méthodes dites "pseudo-haptiques" [14,95,96,134]. Alors que les dispositifs haptiques vont notamment restituer un retour d'effort (contacts, collisions), les méthodes pseudo-haptiques vont consister à fournir à l'utilisateur le même type de perception mais au travers de canaux sensori-moteurs différents comme le canal visuel.

La comparaison de ces deux approches pour l'interaction directe avec des données CAO natives en environnement immersif est une perspective de recherche très riche.

5.3.4 Retours d'information et menu déportés

Tout système d'interaction se doit de fournir aux utilisateurs des retours d'informations, des *feedbacks*. Cette nécessité est encore plus grande lorsqu'il s'agit d'interactions en environnements virtuels où il s'agit de fournir à l'utilisateur toutes les données nécessaires pour qu'il perçoive au mieux l'état courant de la scène ainsi que l'impact des ses actions. Dans la preuve de faisabilité conceptuelle de notre modèle (i.e. CREA-RV), ces retours d'informations se font au sein de petits panneaux, manipulables dans l'espace, qui se mettent à jour selon les différentes interactions sur la maquette virtuelle (revoir section 4.4).

La mise en place et la gestion de ces retours d'informations sont loin d'être des activités triviales puisqu'il faut trouver le compromis entre une quantité suffisante d'information, et un affichage / un retour qui ne doit pas être trop envahissant ou dérangeant par rapport à la scène virtuelle. Nous distinguons ici deux types de retour d'information : ceux qui ont trait à la scène virtuelle de façon globale, et ceux qui ont trait à l'interaction directe de l'utilisateur.

En ce qui concerne les *feedbacks* sur l'interaction directe de l'utilisateur, il conviendrait pour nos activités d'édition CAO, de fournir différentes indications sur ce qu'il est possible de faire. Plus exactement, lorsqu'un utilisateur sélectionne un objet à manipuler, il faudrait lui donner des informations visuelles (autres qu'une liste de possibilités) sur les interactions possibles, comme par exemple des incitations à tirer l'objet dans tel ou tel sens en lui montrant différentes directions. On pourrait également penser à de petites icônes affichées juste à côté de l'élément désigné, indiquant le type de paramètre disponible.

Enfin, les informations propres à l'état courant de la scène doivent aussi être rendues de façon la plus optimale possible. En guise de perspective, nous pointons le fait que les interfaces portables telles les tablettes ou les smartphones sont de plus en plus utilisées, et notamment dans l'industrie. Ces interfaces seraient des cibles idéales pour restituer des informations et même pour réaliser des commandes. C'est notamment une piste qui a été suivie par Brincin et al. [36] pour VR4D (*Virtual Reality for Design*). De plus, il faut noter qu'outre ces dispositifs portables, les industriels utilisent pour leurs revues de projets en RV des écrans déportés, permettant aux différentes personnes présentes lors de la session, de suivre ce qu'il se passe dans l'environnement immersif.

5.3.5 Poste de travail immersif

Ces dernières années voient l'explosion de périphériques qui existent pourtant depuis de très nombreuses années en RV : les *Head-Mounted Displays* (HMD) (revoir section 1.3.2.1). Par ailleurs, au cours de discussions autour de mes travaux de thèse avec certaines personnes comme les stylistes et les modeleurs CAO, est sorti l'idée, l'envie, de travailler un jour sur un poste de travail immersif. En fait, ces idées de poste de travail immersif se nourrissent de l'éloignement qu'il peut y avoir entre les différents métiers avec les systèmes de RV, voire l'absence même de système de RV dans certaines structures industrielles.

Le modèle de données que j'ai mis en place est générique dans sa forme, intégrant notamment un composant de visualisation basé sur DRS (revoir section 3.4). La visualisation, et l'interaction, peuvent bien sûr se faire sur poste de travail. D'ailleurs, les premiers tests de ces travaux ont été réalisés sur poste de travail standard, l'interaction allant du simple couple clavier/souris à la souris 3d. L'implémentation de notre modèle pourrait également se faire au sein même de logiciels CAO sous forme de *plugin*.

Ainsi, des efforts particuliers pourraient être menés afin de proposer une interface immersive et intuitive sur poste de travail CAO (sytlistes, ingénieurs, etc.). La visualisation pourrait se faire soit via un casque de réalité virtuelle, offrant une immersion visuelle totale, ou bien via un écran d'ordinateur stéréoscopique. Les interactions pourraient se faire via une souris 3d (SpaceMouse) ou via des interfaces gestuelles à main libre facilement utilisables sur un bureau comme le Leap Motion [3] ou un Kinect [4].

5.3.6 Conclusion

L'architecture et le modèle de données pour l'interfaçage RV-CAO proposés dans ces travaux de thèse posent des bases pour l'intégration des systèmes CAO paramétriques avec les technologies de RV. Les principaux apports de cette approche ayant été démontrés dans un environnement immersif de type CAVE avec des techniques d'interaction 3d relativement basiques, les travaux futurs doivent porter sur l'amélioration des différents composants que sont le moteur d'interaction et le système de visualisation.

Alors que les interactions doivent se diversifier pour offrir le plus de possibilités à l'utilisateur tout en assurant une forte intuitivité (reconnaissance gestuelle, reconnaissance vocale, haptique / pseudo-haptique, multimodalité) , la visualisation immersive doit être complètement testée (montée en charge). D'autres solutions que l'interaction dans un système CAVE pourraient également tout à fait être envisagées.

5.4 Inférence et paramétrage

L'édition implicite du GHC des objets CAO repose sur une étape d'inférence permettant de déterminer des cibles à modifier à partir de l'étiquetage d'un élément donné. Le modèle d'inférence proposé dans ces travaux est un moteur en deux parties (revoir figure 3.2). Actuellement, seule la première partie du moteur (A) se déroule selon des règles logiques, la partie (B) ne consistant qu'en une sélection manuelle de cibles parmi une liste. Les règles logiques, valables pour tous les utilisateurs sans distinction, sont des règles de "bon sens" définies selon les besoins des utilisateurs. Les évolutions que l'on pourrait envisager pour ce composant d'inférence seraient ainsi la prise en compte de l'action de l'utilisateur une fois la sélection réalisée, ainsi que le profil et l'expertise de l'utilisateur.

5.4.1 Moteur d'inférence pour superviser l'interaction

Dans l'optique de proposer un comportement plus intuitif à l'objet CAO lors des modifications, il faudrait pouvoir aider l'utilisateur en fonction de ses interactions (commandes gestuelles ou vocales par exemple) et non plus seulement en fonction de sa première sélection (figure 5.3).

3. https://www.leapmotion.com/product
4. http://www.xbox.com/fr-FR/Kinect

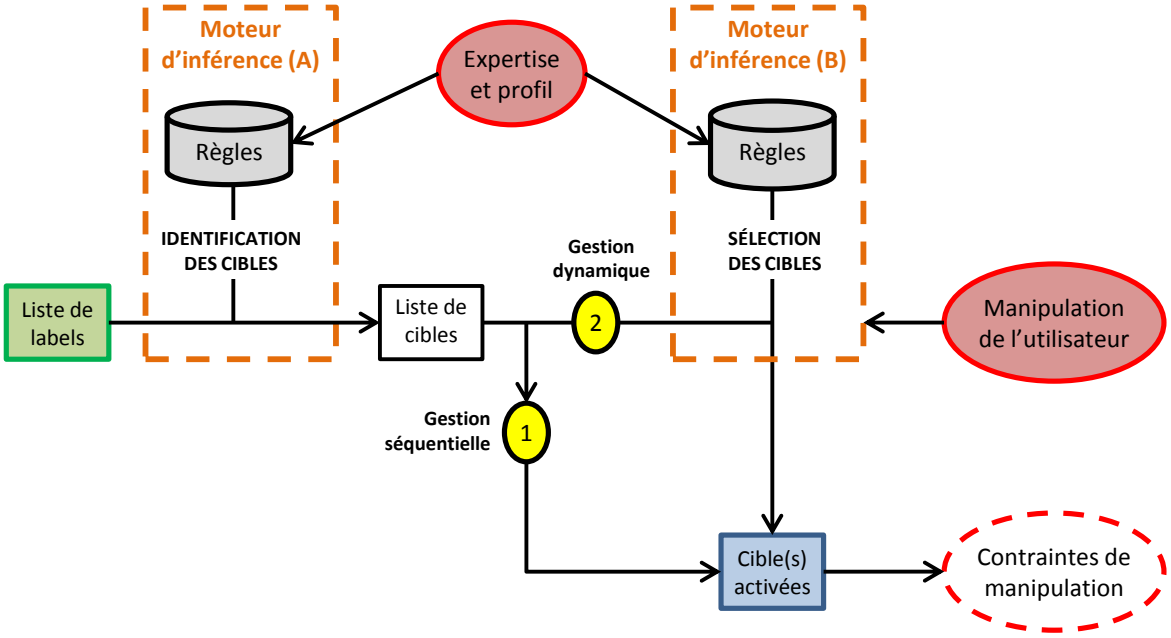


FIGURE 5.3 – Moteur d'inférence évolué. Cette version du moteur d'inférence prend en compte les interactions, l'expertise et le profil de l'utilisateur (partie B). Deux stratégies d'inférences peuvent être choisises : (1) la sélection des cibles se fait de façon séquentielle (comportement actuel par défaut), (2) la manipulation de l'utilisateur permet, en tant qu'argument supplémentaire, de sélectionner dynamiquement parmi la liste de cibles.

Actuellement, l'utilisateur sélectionne un élément, et lorsqu'une ou plusieurs cibles sont accessibles, la première est activée par défaut et l'utilisateur peut choisir manuellement d'en activer une autre. Dans une version dont un mode de modification tiendrait compte des interactions de l'utilisateur pour l'activation, ce processus serait légèrement différent. La première phase consisterait toujours à sélectionner un élément de B-Rep et à récupérer une liste de cibles accessibles depuis cet élément (phase A du moteur d'inférence). Mais dans ce mode il n'y aurait plus d'activation par défaut d'une cible : le choix de la cible à activer se ferait selon une nouvelle phase d'inférences permettant à partir d'une liste de cible et d'une commande utilisateur de déterminer quelle cible activer.

Prenons l'exemple simple de la modification d'une extrusion, illustrée par la figure 5.4. L'utilisateur a sélectionné l'arête orange (élément topologique) étiquetée avec deux *labels* : `EPB(2)` et `ESK(3,1)`. Le premier *label* `EPB(2)` est associé à la règle 2 de la table 3.2 ce qui lui permet de cibler le paramètre 2 de l'extrusion 2 : la limite basse. Le deuxième *label* `ESK(3,1)` a lui été récupéré d'un ancêtre, en l'occurence la ligne d'esquisse (ligne.3) qui sera donc la cible visée (règle 4 de la table 3.2). Nous avons donc ici une situation où deux cibles sont accessibles depuis un même élément. L'idée est donc d'activer l'une ou l'autre selon l'interaction de l'utilisateur. En l'occurrence, la limite basse de l'extrusion (LIM2) se modifie intuitivement en suivant la direction de l'extrusion, représentée par l'axe en pointillés rouge. De la même façon, la modification géométrique de la ligne d'esquisse se fera dans le plan de l'esquisse, représenté en bleu. La sélection dynamique de cibles consisterait donc, suivant que l'interaction se fait suivant l'axe ou dans le plan, à activer la cible correspondante. Pour ce

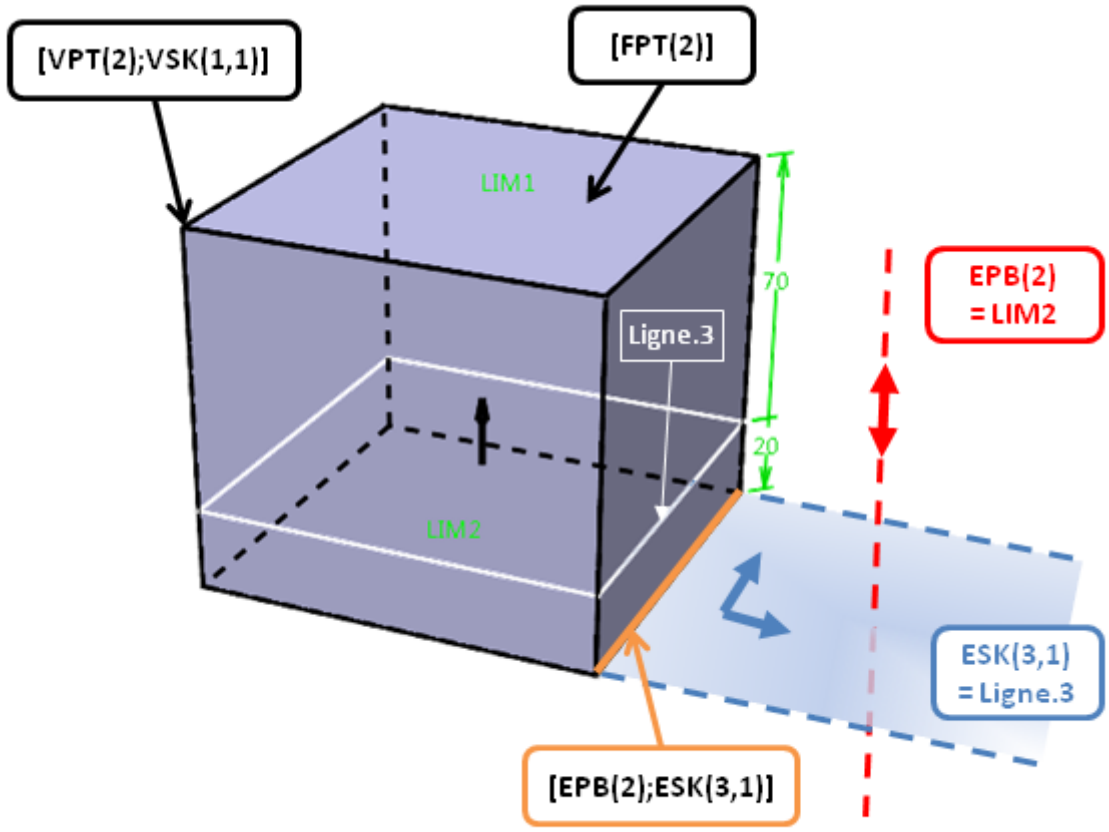


Figure 5.4 – Exemple d'activation dynamique de cibles. L'utilisateur a sélectionné l'arête coloriée en orange. Cette arête est étiquetée par deux *labels* : `EPB(2)` et `ESK(3,1)`. Selon les règles logiques définies (Table 3.2), le premier *label* permet d'atteindre le paramètre 2 (Limite 2) de l'extrusion (opérateur 2) alors que le deuxième cible l'élément 3 (ligne.3) de l'esquisse 1. Ainsi, en fonction de l'interaction de l'utilisateur, à savoir soit suivant l'axe en rouge, soit dans le plan en bleu, la cible activée sera différente.

faire, il serait possible d'associer une étiquette à l'interaction de l'utilisateur, la caractérisant par exemple suivant un axe, un plan, ou autre. Des règles associant ces étiquettes d'interaction aux cibles identifiées permettraient alors cette sélection dynamique de cibles lors de la phase B du moteur d'interaction.

5.4.2 Prise en compte du profil utilisateur

Le moteur d'interaction entier pourrait également avoir un comportement différent selon le profil de l'utilisateur. De par sa nature, ce composant "intelligent" est là pour aider l'utilisateur dans son interaction. Il permet de s'affranchir de la connaissance exacte de la construction de l'objet CAO en proposant des paramètres à modifier en fonction de la sélection topologique. Ceci étant, parmi les acteurs des revues de projet immersives, on trouve bien entendu les experts CAO, connaissant parfaitement la maquette qu'ils évaluent. On peut

se douter qu'un utilisateur expert n'aura pas la même utilisation, la même expérience immersive qu'un utilisateur novice. Il serait ainsi opportun de pouvoir adapter le comportement de l'édition implicite en fonction du profil et/ou de l'expertise des utilisateurs.

Voici quelques pistes qui nous semblent intéressantes à explorer :

- adapter les cibles proposées selon l'expertise. Exemple : une modification dite "complexe" ne sera pas proposée à un utilisateur novice.
- adapter le comportement du moteur d'inférence selon l'utilisateur. Chaque utilisateur pourrait configurer un comportement à sa convenance. Exemple : choix entre une sélection dynamique ou une sélection manuelle.
- avoir un système de règles logiques propre à chaque utilisateur. Les utilisateurs pourraient avoir des préférences différentes à propos des liens directs avec le GHC, qu'ils pourraient alors configurer selon leur affinité.

De plus, on pourrait s'interroger sur la configuration automatique des règles du moteur d'inférence (et de ses niveaux) par rapport à l'usage (répété ou non) du sytème par plusieurs utilisateurs. L'apprentissage automatique du composant d'inférence pourrait permettre de coller de mieux en mieux aux usages et aux besoins des utilisateurs.

5.5 Confrontation des connaissances et redéfinition partielle du GHC

Les différentes perspectives sur l'interaction et l'inférence pour notre approche nous conduisent à des réflexions plus théoriques sur l'édition implicite du GHC. A court terme, les utilisateurs devraient être capables d'interagir avec la maquette virtuelle de différentes manières (gestes, voix, haptique, multimodalité). L'expertise et le profil pourraient également être pris en compte afin d'assurer une intuitivité d'interaction la plus grande possible pour chacun d'entre eux. Plus on offre de possibilités d'interaction naturelle et intuitive avec la maquette virtuelle, plus on se dirige vers des décalages possibles entre la volonté de l'utilisateur et ce qu'il est réellement possible de faire.

5.5.1 La connaissance de l'expert *versus* la connaissance de la machine

Le grand principe d'interaction avec la maquette virtuelle dans notre approche de l'édition implicite consiste à proposer des modifications de la maquette à partir d'une sélection d'éléments sur celle-ci. Cependant, nous pourrions très bien imaginer, si les techniques d'interaction disponibles le permettent, qu'un utilisateur, plutôt expert, puisse spécifier plus ou moins finement par une interaction naturelle et intuitive, la modification qu'il veut effectuer sur la maquette. Le fait de spécifier une modification à apporter à la maquette pourrait aboutir au problème suivant : la spécification demandée par l'utilisateur n'est pas possible en l'état au vu du graphe d'historique de construction de l'objet. La figure 5.5 illustre très clairement ce problème.

Dans cet exemple, l'objet CAO que l'on veut modifier est l'objet 1, qui est visuellement parfaitement identique à l'objet 3. On le voit, cet objet peut être construit de différentes façons : simple extrusion d'un profil particulier (objet 1), ou extrusion d'un profil simple à

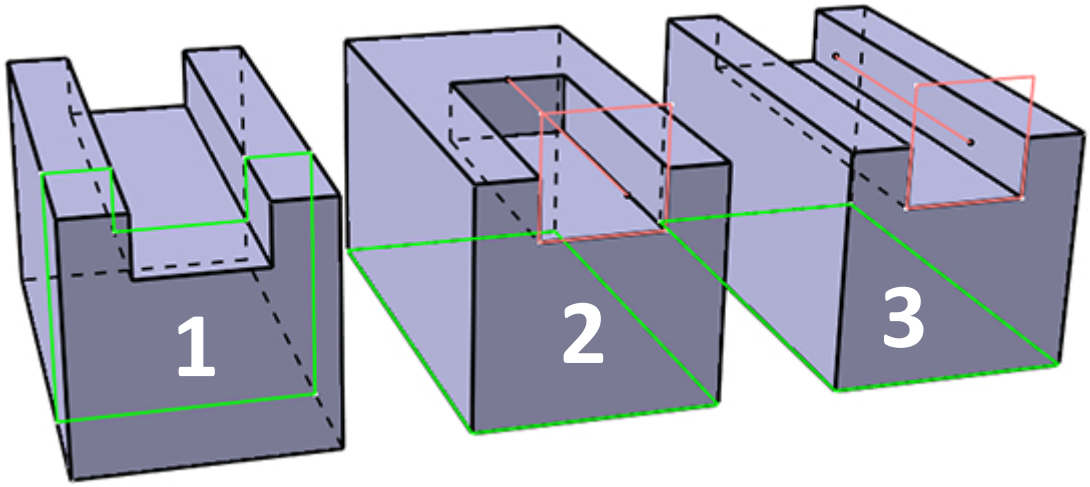


Figure 5.5 – Exemple d'un objet CAO pouvant être interprété, depuis sa représentation visuelle, de différentes façons. L'objet 1 est simplement le résultat d'une extrusion dont le profil est représenté en vert. L'objet 3 est le résultat d'une extrusion (profil en vert) à laquelle on a appliqué une rainure (profil carré rose et courbe guide rose). La rainure peut être paramétrée de différentes manières et donner par exemple l'objet 2.

laquelle on applique une rainure pouvant être paramétrée (objets 2 et 3). Lorsque l'utilisateur visualise l'objet 1, sauf si les esquisses sont affichées, il ne peut détérminer exactement quel est l'historique de construction. Dans le cas où il est possible de spécifier une modification, comme par exemple expliciter à la voix une commande de type "modifier la rainure", un problème interviendrait pour l'objet 1 n'étant pas construit à partir d'une rainure.

On va donc parler ici de confrontation des connaissances : le système CAO connaît la construction exacte de l'objet, alors que l'utilisateur a sa propre interprétation de cette construction. Est-il possible d'assurer malgré tout la modification de l'objet ?

5.5.2 Pistes de résolution de ce conflit

Lorsque l'utilisateur interprète l'historique de construction de l'objet CAO d'une manière différente du véritable historique, plusieurs situations vont intervenir. Nous décrivons ici très brièvement les problèmes et des pistes de résolution.

Tout d'abord, il se peut que la différence d'interprétation ne soit pas si grande que ça. Dans le cas où l'utilisateur spécifierait par exemple la volonté de modifier la hauteur de l'extrusion sur le cube issu de la nervure, il serait possible, suivant un traitement à déterminer, de jouer avec la définition de la courbe (ici un segment de droite) en lieu et place de cette hauteur d'extrusion. En agrandissant le segment par la modification de son point d'arrivée, le résultat serait ce que veut l'utilisateur.

Cependant, il se peut aussi que la requête de l'utilisateur ne puisse pas être résolue via le graphe d'historique existant. Dans ce cas là, l'idée serait de proposer une redéfinition partielle du graphe d'historique afin de réaliser la modification de l'utilisateur, puis par un processus à déterminer, de répercuter ces modifications sur le graphe d'origine.

Finalement, ce problème théorique repose sur l'analyse et la comparaison de graphes d'historique de constructions existants, mais aussi peut s'inspirer des travaux menés sur la problématique de la reconnaissance de *features*.

5.6 La RV-CAO dans le processus de conception industriel

Mes travaux de thèse se sont déroulés dans un contexte industriel, celui de l'industrie automobile chez PSA Peugeot Citroën. Nous l'avons vu dans ce manuscrit, la particularité de l'industrie automobile, comme l'aéronautique, est de s'être très rapidement adaptée aux nouvelles technologies. Que ce soit la CAO ou depuis quelques années maintenant la RV, ces industries font évoluer, optimisent leurs processus de conception en y intégrant ces technologies au travers d'usages et de procédés bien définis.

La conception en environnement immersif est une thématique en vogue, qui mérite que des moyens soient alloués afin de poursuivre les différents travaux qui existent. À l'heure où la concurrence est acharnée et où chacun tente de renforcer ses positions, les industries qui investiront pour pousser l'acquisition et le développement de savoir faire en interne, le plus indépendamment possible de l'extérieur, en tireront certainement une grande valeur ajoutée. Ce manuscrit montre, je l'espère, à quel point la conception en immersif, et plus spécifiquement la modification de maquette virtuelle en environnement immersif, est un domaine dont la composante "recherche" est encore extrêmement présente.

Dans cette section nous présentons donc, en nous basant sur ces travaux de thèse, quelques réflexions sur les perspectives de la RV-CAO dans l'industrie.

5.6.1 Maturation, consolidation et développement de l'approche paramétrique

La conception paramétrique et la conception surfacique sont toutes les deux abordées dans le processus de conception industriel. La conception paramétrique est notamment utilisée avec la RV afin d'évaluer le produit selon différentes hypothèses (configurations, paramétrages), évitant ainsi la construction de maquettes physiques.

Mes travaux de thèse proposent donc un modèle générique pour l'intégration de systèmes CAO commerciaux aux environnements immersifs pour la modification de maquettes virtuelles. Cette approche pour la modification paramétrique devrait être continuée et supportée dans ce contexte industriel afin d'étendre son application au plus grand nombre d'activités possible. Pour ce faire, il faudrait dans un premier temps réaliser un effort de développement pour, par exemple, étayer, consolider, rendre complètement mature la preuve de faisabilité CREA-RV, décrite dans ce manuscrit.

Dans un deuxième temps, il faudrait réaliser un véritable passage aux conditions de revues de projets immersives réelles en permettant de manipuler et modifier des maquettes CAO telles que celles utilisées au cours de projets industriels actuels (pour l'industrie automobile des maquettes de véhicules partiels voire complets). Puisque la revue de projet immersive et modificative est un concept nouveau rendu possible par notre modèle, il est important non seulement de valider le passage à l'échelle, mais plus globalement d'analyser l'activité induite pour améliorer tous les aspects évoqués ci-dessus (interface homme-machine naturelle et intuitive, moteur d'inférence, adaptation au profil/à l'expertise, etc.).

5.6.2 Nouvelles approches interactives pour l'édition surfacique

Plus on remonte tôt dans le cycle de conception, et plus on trouve des activités basées sur la conception surfacique, notamment toutes les activités de design, d'esthétique du produit. Cette conception surfacique est de plus en plus disponible au sein d'outils de conception paramétrique. On peut notamment citer l'intégration progressive de ICEM Surf au sein de CATIA. Suite à la communication de mes travaux au sein de PSA Peugeot Citroën, les

métiers concernés principalement par la conception surfacique (stylistes, modeleurs), bien qu'indirectement concernés par mes travaux sur la conception paramétrique, ont montré un grand intérêt pour la modification de maquettes CAO en environnement immersif.

Notre approche se base sur les outils de conception basés sur les graphes d'historiques de conception, outils principalement destinés à la conception paramétrique. Mais un système CAO permettant une conception surfacique tout en proposant un graphe d'historique serait intégrable à notre approche. Les éléments toplogiques seraient alors également étiquetés et leur modification permettrait un rejeu du graphe tout comme c'est le cas actuellement. La différence principale qui interviendra sera la façon de modifier les différents éléments de construction. La conception surfacique repose principalement sur la manipulation de courbes, splines et autre surfaces. On se retrouve ainsi quasiment uniquement dans des phases de modification de la géométrie dans l'espace, au contraire de l'approche paramétrique qui consiste à modifier des valeurs alphanumériques.

Pour bien aborder l'édition surfacique immersive, il faudra soigneusement choisir entre les techniques de modification des paramètres de construction de ces surfaces, ou bien des méthodes de déformation (libre ou contrainte). Dans le premier cas, un des points critiques réside bien sûr dans l'accès aux différents paramètres et coordonnées géométriques des éléments définissant les courbes et les surfaces. Nous rappelons au passage l'utilité des périphériques haptiques pour l'interaction avec ce genre d'éléménts "surfaciques" permettant de plus facilement les sélectionner et les modifier [26, 52, 129–131].

Enfin des outils comme Imagine & Shape [5] de CATIA sont à surveiller pour l'édition surfacique. En effet, ces outils qui visent à proposer des interactions intuitives avec les esquisses, et des fonctionnalités puissantes pour l'édition de surfaces, pourraient être de parfaits candidats pour une expérience immersive.

5.7 Conclusion

Dans ce chapitre, au delà des apports propres à notre modèle qui ont été détaillés au chapitre 4, nous avons voulu présenter les nombreuses perspectives de la conception immersive. Ainsi, mes travaux de thèse offrent, tout d'abord, plusieurs perspectives quant à l'interfaçage de systèmes CAO commerciaux utilisés dans l'industrie avec les environnements immersifs. Le modèle de données proposé doit être étendu afin de prendre en compte le maximum de fonctionnalités utiles pour la conception paramétrique. La gestion des données de ce modèle, et notamment lors des différentes mises à jour de l'objet CAO doit être évaluée et optimisée dans l'optique de proposer une interaction la plus robuste possible.

L'architecture qui se base sur ce modèle de données RV-CAO permet l'intégration de différentes techniques d'interaction, comme les interactions naturelles et leur combinaison (multimodalité) qu'il convient de mettre en place progressivement pour que les utilisateurs aient à disposition des outils aussi intuitifs que le procédé d'accès et de modification des données CAO. Ces moyens d'interaction permettront de faire évoluer le moteur d'inférence (outil d'aide à la décision), en spécifiant des comportements spécifiques en fonction des actions de l'utilisateur.

Ensuite, nous avons brièvement exposé le problème lié à la différence d'interprétation du graphe d'historique de construction de l'utilisateur par rapport à la machine. Dans le cas où l'utilisateur aurait la possiblité de spécifier (de façon naturelle et intuitive) la modification qu'il veut apporter à l'objet CAO, des écarts pourraient survenir entre sa demande et ce

5. http://www.3ds.com/products-services/catia/portfolio/catia-version-6/v6-portfolio/d/0/s/catia-design/p/imagine-and-shape/?cHash=4a9130458b6e0d23c5266dfe5aca02ec

qu'il est possible de faire en fonction de la description effective des objets dans le GHC. Des techniques d'adaptation de la commande de l'utilisateur au graphe, voire la redéfinition partielle du graphe sont donc à prévoir et à étudier.

Enfin, ces techniques de conception en immersif sont un domaine qui doit être soutenu dans l'industrie. Les approches, comme celle de l'édition implicite sur des logiciels CAO commerciaux qui est la nôtre, offrent de nouvelles perspectives pour l'interaction avec les maquettes numériques dans l'industrie. Ces nouveaux concepts demanderont des efforts de développement pour être réellement valables à l'échelle industrielle, ainsi que certains bouleversements dans l'organisation du travail autour du PLM. Il est donc évident qu'un passage à l'échelle avec des analyses fines de l'activité sera requis.

Conclusion générale

Cette thèse a porté sur l'interfaçage entre les systèmes de Conception Assistée par Ordinateur (CAO) et les technologies de Réalité Virtuelle. Mes recherches ont poursuivi les différents travaux du groupe VENISE du LIMSI-CRNS sur cette thématique de la RV-CAO, en trouvant un cadre applicatif qui est celui de l'industrie automobile et plus spécifiquement PSA Peugeot Citroën. La motivation principale de ces travaux aura donc été le manque d'approche permettant la modification immersive de données CAO natives, issues des logiciels commerciaux.

Nous avons ainsi dans ce manuscrit commencé par exposer le contexte de ces travaux. Du fait de la concurrence acharnée, les industries cherchent à toujours plus optimiser leurs cycles de conception et y intègrent ainsi les différentes technologies allant dans ce sens. C'est notamment le cas des technologies de CAO et de RV. La RV est particulirèment utilisée pour la revue de projet afin d'évaluer le produit et d'anticiper les problèmes au plus tôt. Nous avons passé en revue de façon exhaustive les différentes approches pour l'intégration de données CAO dans les environnements virtuels et fait ressortir les différents problèmes qui peuvent intervenir, comme la conversion des données, ainsi que le manque d'approche permettant la modification de données CAO en environnement immersif.

Les concepts clé de la CAO ont ensuite été décrits afin d'appréhender au mieux les différents mécanismes de notre approche. L'accent a principalement été mis sur la conception volumique et sur les représentations B-Rep (représentation des objets par leurs éléments de frontière) et GHC (graphe d'historique de construction) sur lesquels sont basés la plupart des logiciels de CAO paramétriques à caractéristiques de forme. Le principe de réactivité des objets CAO, permettant d'accéder directement à des nœuds du GHC à partir des éléments de B-Rep — sommet, arête et face — a été explicité ainsi que les différents type de couplage RV-CAO que l'on peut trouver.

En nous basant sur l'approche de l'édition implicite du GHC des objets CAO développée dans le groupe VENISE, nous avons présenté une généralisation de ce concept pour permettre son application sur les systèmes CAO commerciaux. Cette généralisation s'est traduite par la conception d'un modèle de données basé sur plusieurs structures d'encapsulation dans lesquelles sont stockées les données des objets CAO — données des représentations B-Rep et GHC — nécessaires à leur modification. Le mécanisme de *labelling* des éléments de B-Rep sur lequel repose le principe de réactivité des objets CAO a été transformé en un *multi-labelling* permettant la prise en compte des relations de parenté entre des éléments de différents B-Rep. Ce nouveau mécanisme permet ainsi d'accéder à un plus grand nombre de nœuds du GHC à partir d'une seule sélection topologique. Le modèle de données est articulé sur un moteur d'inférence qui traduit alors la sélection d'un élément de B-Rep en l'accès à des paramètres du graphe d'historique. Cette traduction se fait à partir du *multi-labelling* des éléments de B-Rep, et utilise des règles logiques basée sur le langage PROLOG. Deux autres composants viennent compléter cette architecture : un système de visualisation basé sur DRS permettant

une visualisation immersive et stéréoscopique et un moteur d'interaction intégrant VRPN et permettant les manipulations sur la maquette numérique.

Les apports de notre modèle d'interfaçage RV-CAO ont ensuite été décrits. Le principe de réactivé et donc l'édition implicite du GHC d'objet CAO sont ainsi applicables à des objets issus de systèmes CAO paramétriques commerciaux basés sur les caractéristiques de forme (*features*). Cette application requiert l'accès aux fonctionnalités du système CAO cible, et se réalise par un parcours et une analyse des représentations GHC et B-Rep des objets en question. Notre interfaçage permet ainsi la modification d'objets sans aucune conversion de données que ce soit en début ou en fin de session de travail. L'accès et l'intégration des fontionnalités du système CAO visé au sein de notre modèle permet une supervision des modifications et des mises à jour. Cette supervision assure notamment la validité de l'objet CAO au cours de la session de travail ainsi que le caractère temps-réel tant pour l'interaction que pour la visualisation.

Ces différents apports ont été démontrés grâce à CREA-RV, preuve de concept de notre modèle d'interfaçage RV-CAO autour de CATIA V5. Cette implémentation aura constitué un défi majeur de mes travaux de thèse, et particulièrement chronophage. Les résultats positifs obtenus me permettent d'atténuer la frustration de ne pas avoir pu proposer un ensemble plus complet, et notamment un système d'interaction plus évolué et plus intuitif.

Contributions de la thèse

Ces travaux de thèse proposent un modèle d'interfaçage RV-CAO en généralisant les travaux existants et apportent les contributions qui suivent.

Tout d'abord, nous avons apporté une clarification conceptuelle des mécanismes de *labelling* permettant un lien direct entre les éléments topologiques et les nœuds du GHC, et de *persistent naming* permettant d'identifier de façon unique et de chaîner les éléments de B-Rep au sein du GHC.

Le modèle de données proposé permet d'appliquer le principe de réactivité et l'édition implicite du GHC aux objets conçus dans des logiciels CAO paramétriques basés sur des *features*. Notre approche requiert l'accès aux fonctionnalités du système CAO cible et les intègre, permettant ainsi la modification immersive d'objets CAO dans leur format natif, ne nécessitant aucune conversion des données que ce soit en entrée ou en sortie de session de travail.

Un accès simple et direct aux paramètres de la maquette est assuré par le principe de réactivité désormais basé sur un mécanisme de *multi-labelling* agrandissant le nombre de possibilités d'interaction et de modification depuis un même élément de B-Rep en fonction de son historique. Les cibles accessibles depuis un élément topologique sont regroupées au sein d'une seule et unique liste ce qui simplifie leur activation et leur gestion. Cette gestion est dans sa version simple une gestion séquentielle, et dans sa version améliorée est une gestion dynamique où le moteur d'inférence prend en compte les interactions de l'utilisateur.

Des supervisions des modifications et des mises à jours sont disponibles. Différentes stratégies de répercussion des modifications — mise à jour locale, mise à jour globale et mise à jour en cascade — sont proposées afin d'assurer, quelle que soit l'interaction de l'utilisateur, une visualisation directe des modifications apportées aux objets. Un système de détection des erreurs est également mis en place et assure la validité de l'objet CAO tout au long de la séance de travail. Dans le cas des modifications et/ou mises à jour erronnées, l'utilisateur est alerté plus ou moins finement des erreurs survenues, et l'objet retrouve son dernier état valide.

Une preuve de faisabilité en contexte industriel, autour de CATIA V5 a été proposé. Ainsi le démonstrateur CREA-RV permet la visualisation (stéréoscopique) et la modification en environnement immersif, d'objets CAO conçus sous CATIA. Ces objets sont directement importés en environnement immersif sans conversion et peuvent être, une fois sauvegardés, re-chargés directement dans CATIA.

Ces différents apports ont donné lieu à un article de journal actuellement en soumission [111]. De plus, des travaux sur l'interaction haptique, multimodale et collaborative, entamés avant la thèse et complémentaires aux travaux présentés dans ce manuscrit, ont été menés en parallèle et ont donné lieu à trois articles de conférences internationales [113–115].

Perspectives

Ces travaux de thèse offrent de nombreuses perspectives, à la fois sur le plan scientifique que sur le plan recherche.

L'architecture et le modèle de données proposés dans ces travaux, fonctionnels, doivent tout d'abord être étendus afin d'intégrer le plus grand nombre d'opérateurs couramment utilisées pour la conception paramétrique. Ces extensions incluent donc la conception de structures de données pour d'autres opérateurs, ainsi que les répercussions sur le mécanisme de *multi-labelling* et les règles d'inférence.

La gestion des données et les processus de supervision doivent être évalués et optimisés afin de proposer des sessions de modifications immersives les plus robustes et optimales possibles. Une supervision de la complexité des objets ou de parties des objets manipulés permettrait une gestion dynamique, ou du moins plus efficace, des processus de mises à jour des objets CAO.

Les interactions en environnement immersif doivent être étendues dans l'optique de proposer aux utilisateurs une interaction intuitive. Nous visons particulièrement les commandes naturelles — voix, gestes et haptique — ainsi que les approches multimodales (combinaison de modalité) que nous avons abordées au cours de précédents travaux.

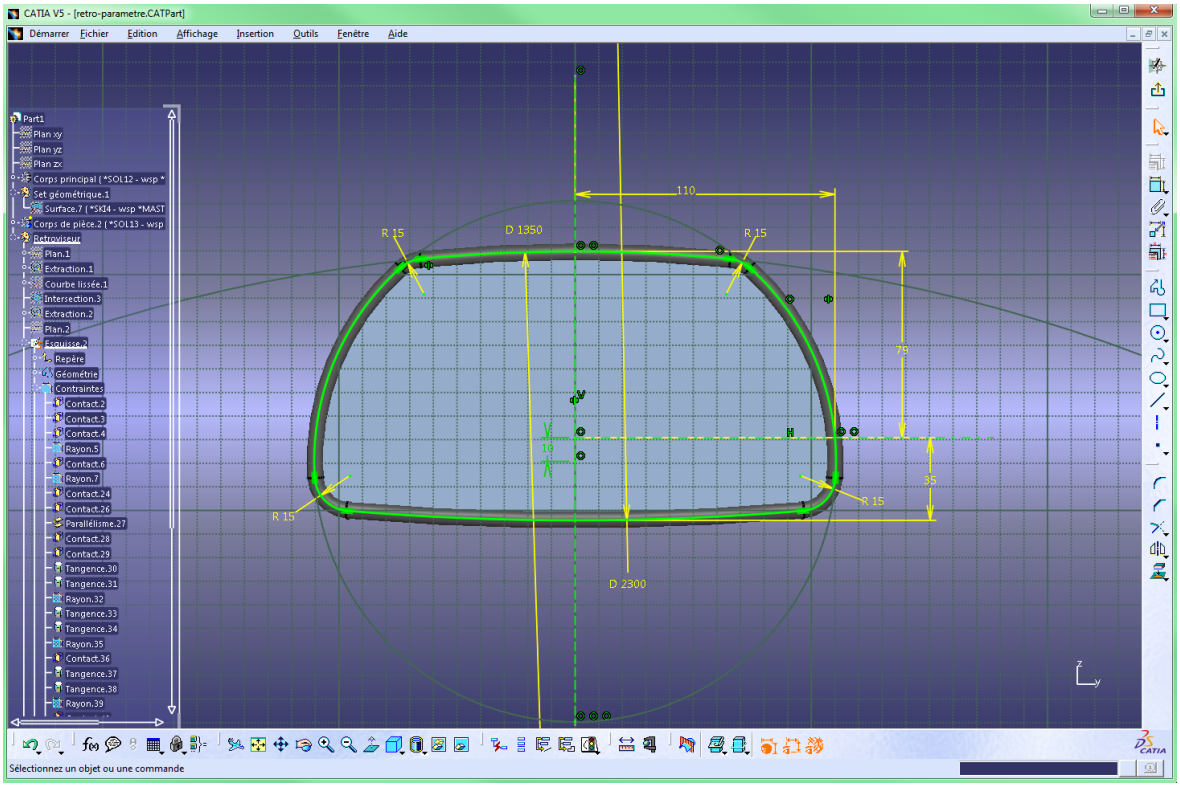
D'un point de vue plus théorique, la problématique des différences d'interprétation du GHC entre l'utilisateur et la machine pourrait être un champ de recherche intéressant. En effet dans le cas où les utilisateurs pourraient complètement spéficier (par une interaction naturelle et intuitive) les modifications qu'ils veulent apporter à la maquette CAO, des écarts pourraient intervenir entre leurs demandes/intentions et ce qu'il est possible de faire. Des techniques d'analyse et de comparaison des graphes, s'appuyant par exemple sur la problématique de la reconnaissance de *features*, pourraient être mises en place pour adapter la commande de l'utilisateur au graphe réel de l'objet.

Enfin d'un point de vue industriel, nos travaux devraient être consolidés, testés, et passés à une plus grande échelle. Ceci permettrait alors d'analyser et d'évaluer l'impact de ces nouvelles approches dans le processus de conception. Il serait également intéressant d'explorer l'adaptation de ces concepts à la conception surfacique afin d'étendre le champ d'activités et de viser un plus grand nombre de métiers et d'utilisateurs, voire de proposer une conception intuitive et immersive sur des postes de travail évolués.

Annexe A

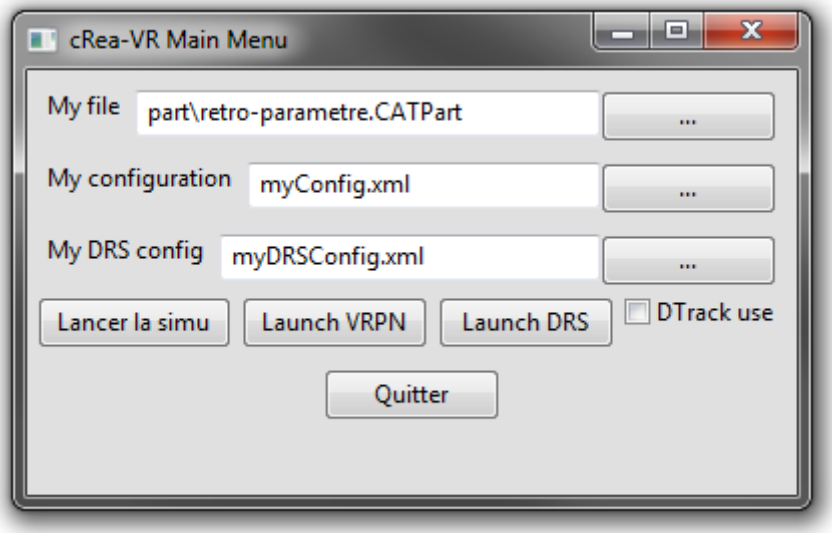
Exemple complet d'une modification de maquette CAO avec CREA-RV

Construction de l'objet sous CATIA V5

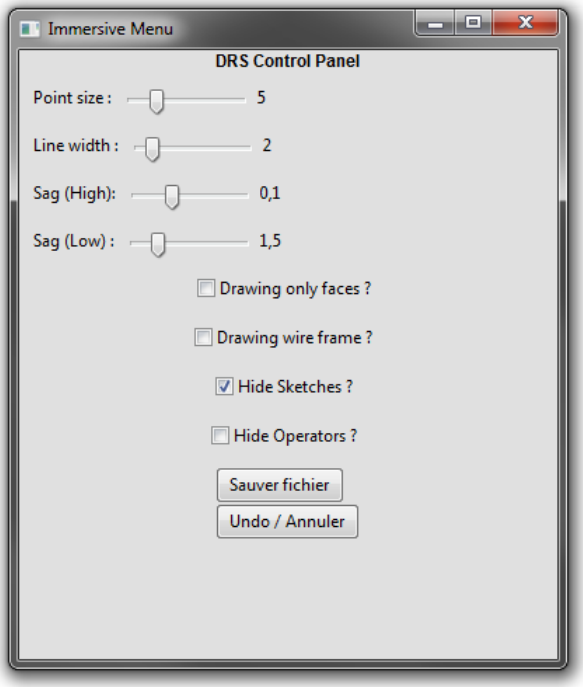


Lancement de CREA-RV - Menu Principal et Menu Immersif

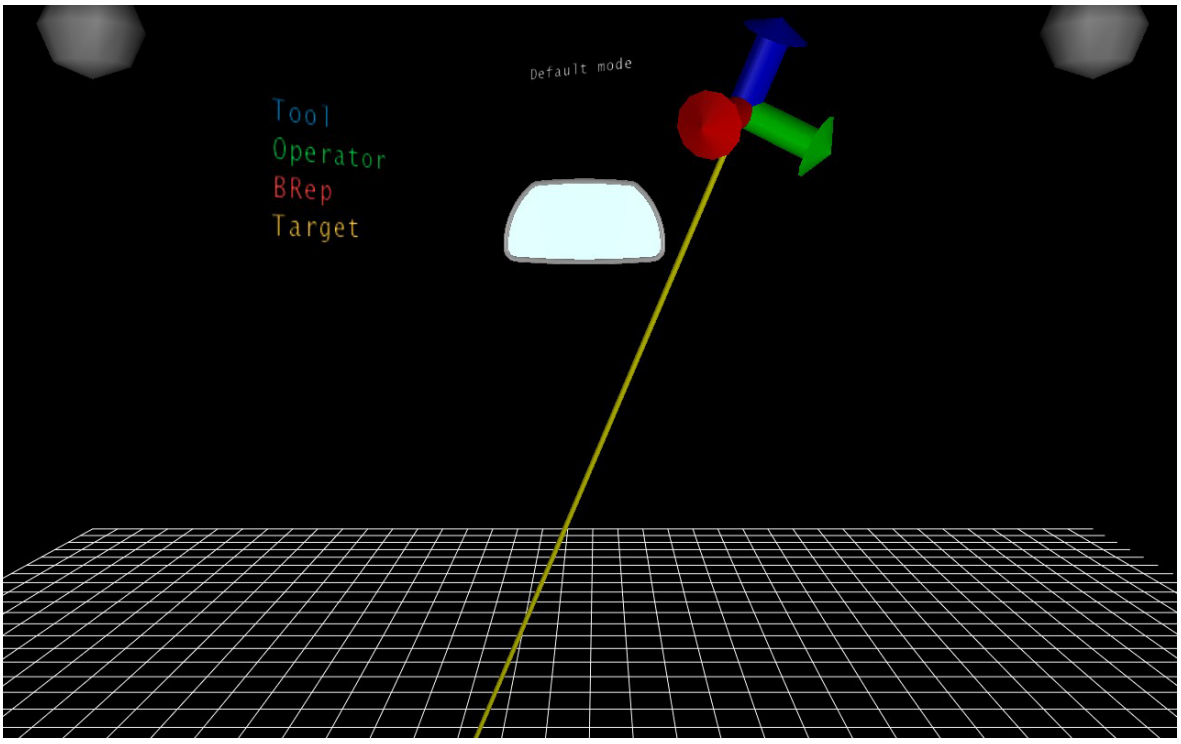
Menu principal pour le lancement de CREA-RV.



Menu immersif pour la session de travail en cours.

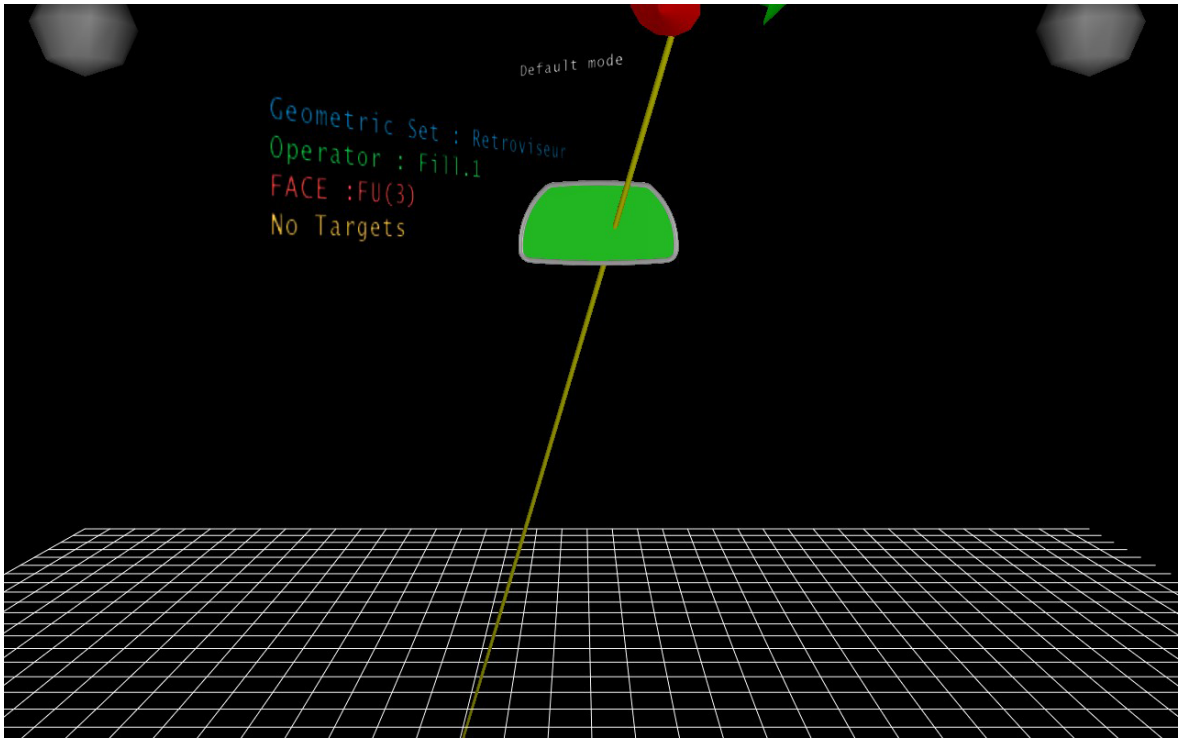


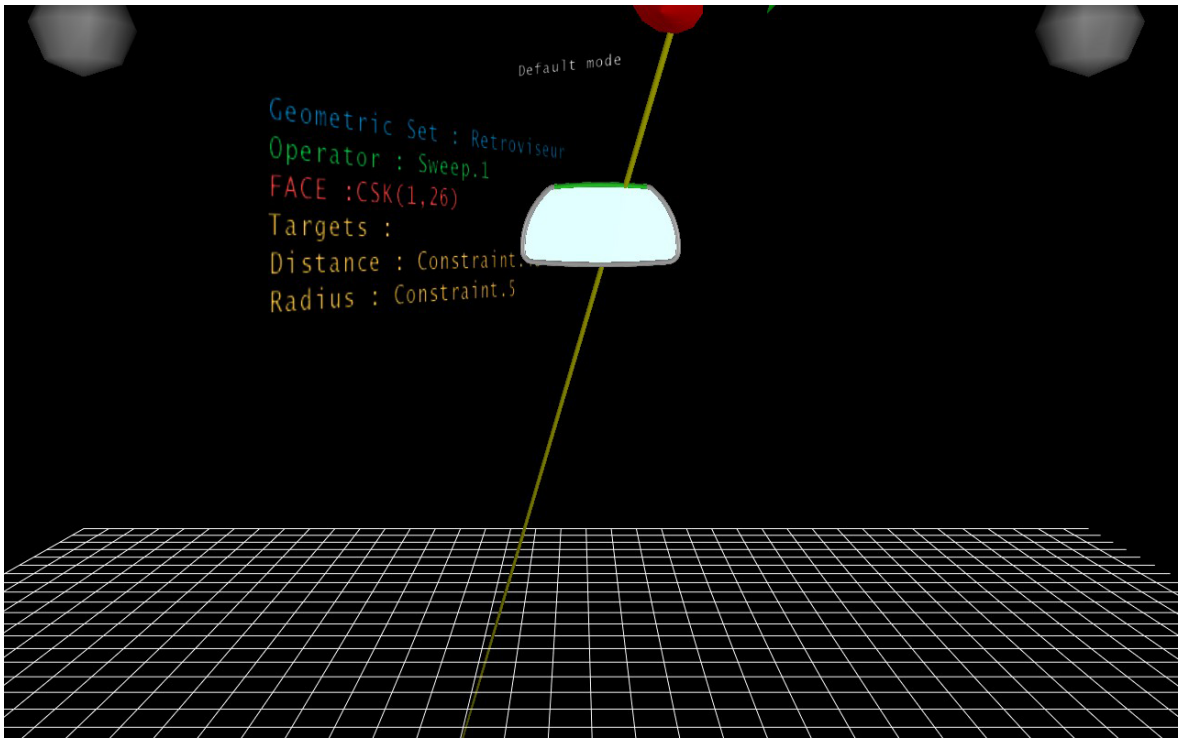
Rétroviseur chargé dans CREA-RV et prêt à être modifié.



Exploration de l'objet

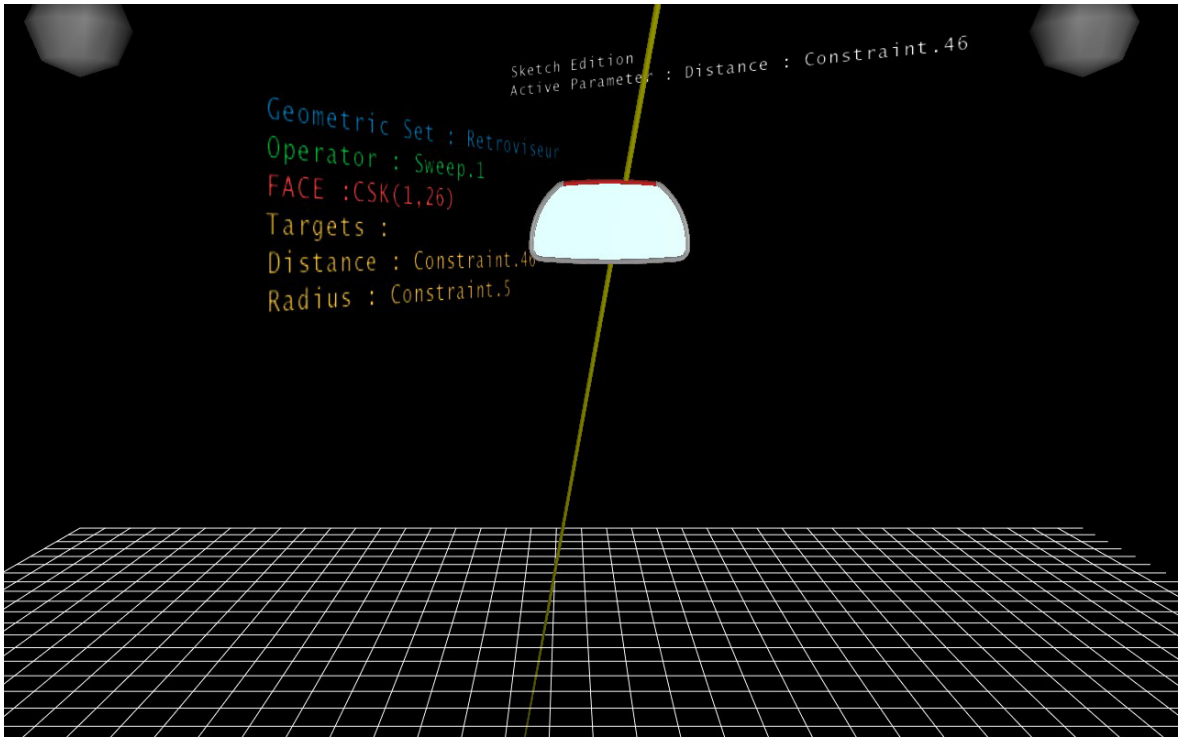
L'utilisateur explore l'objet en pointant les différents éléments. Un élément pointé devient vert.



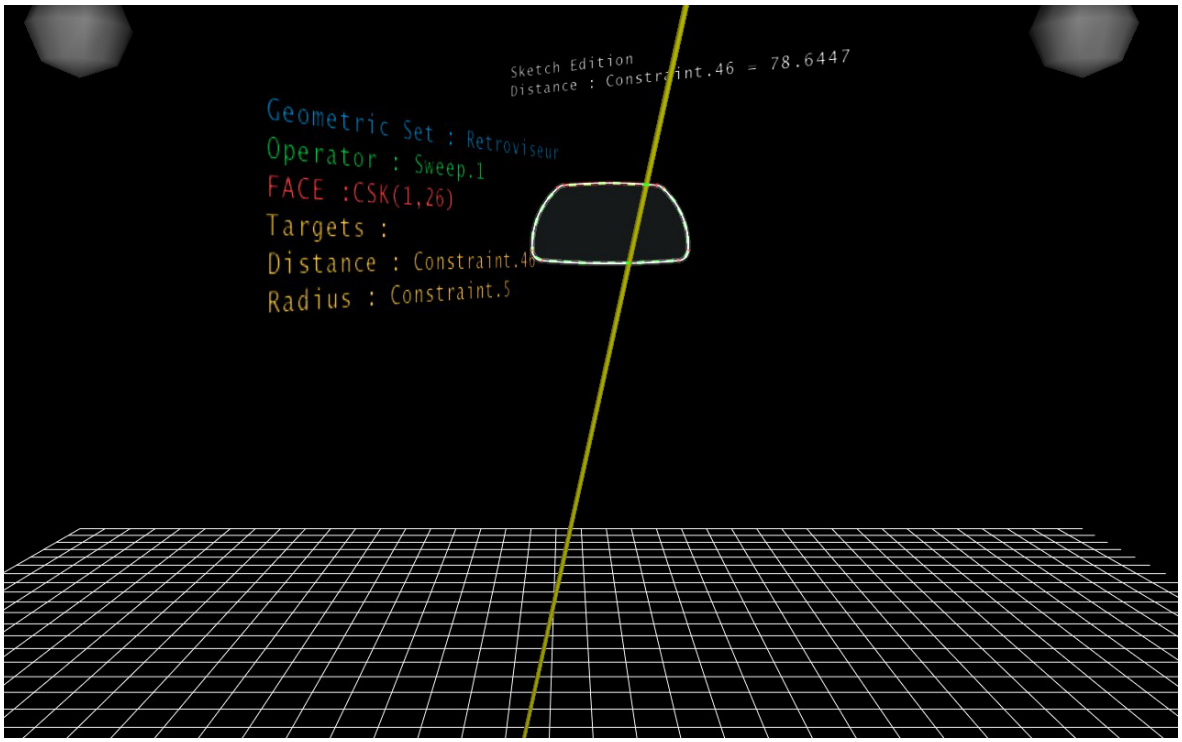


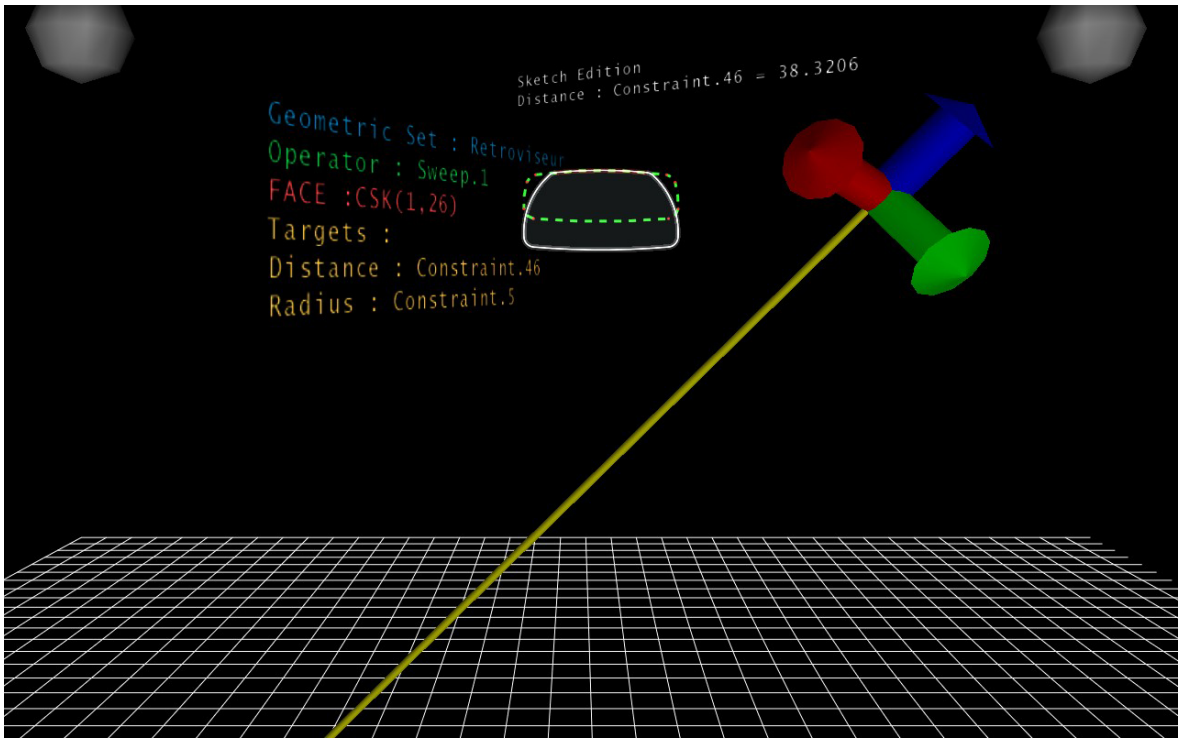
Première sélection et modification

Sélection d'une face topologique (élément rouge). Cette face permet d'accéder à deux cibles qui correspondent à deux contraintes d'esquisse.

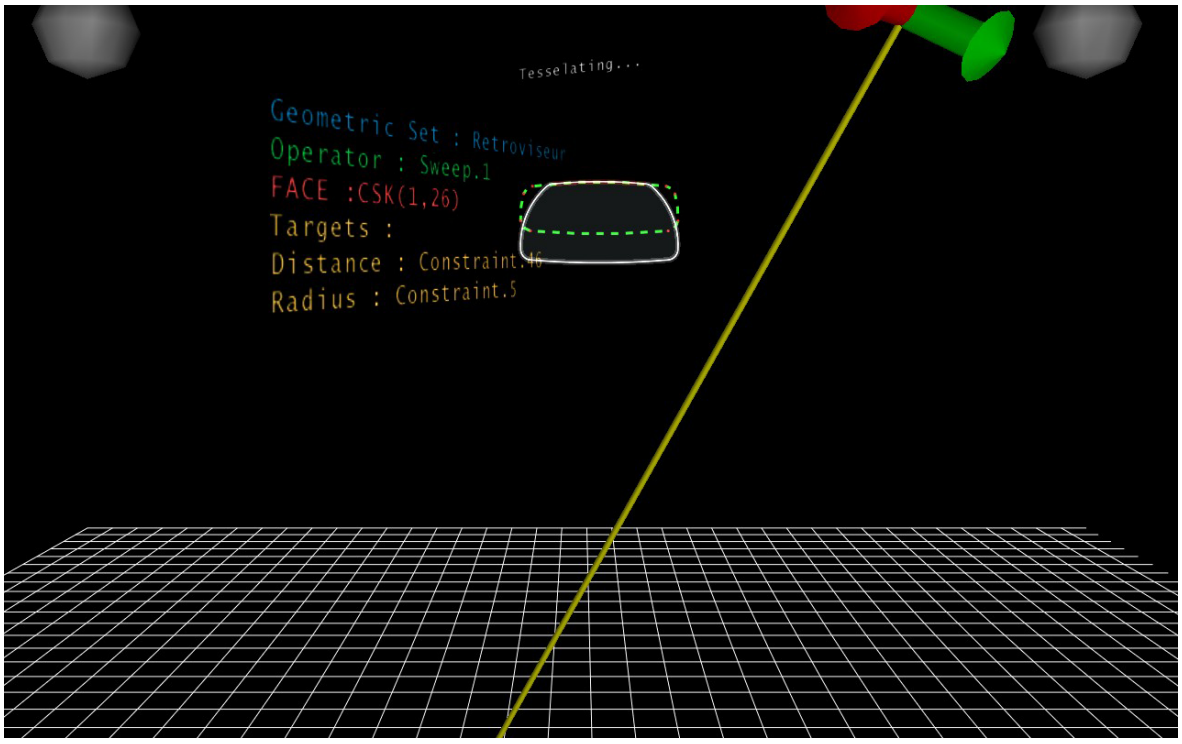


Par défaut, la première contrainte, **Distance - Contrainte.46** est activée. La modification peut alors commencer.



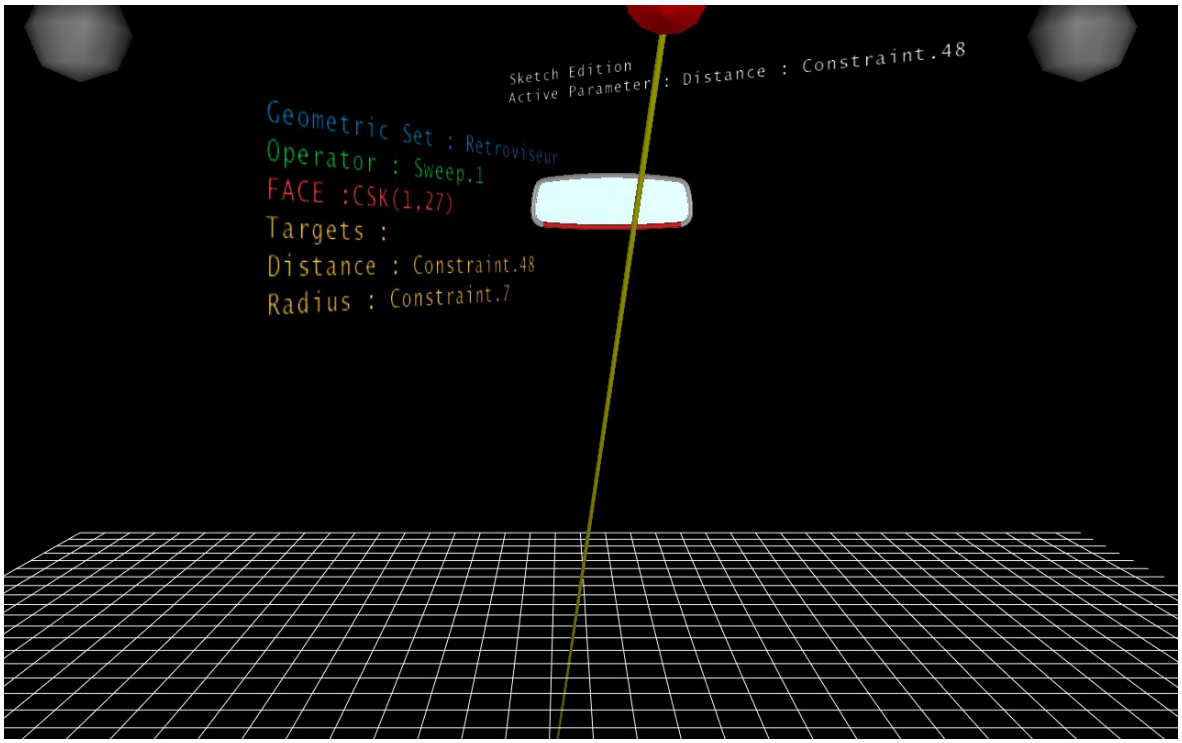


La modification est terminée, l'utilisateur valide sa modification

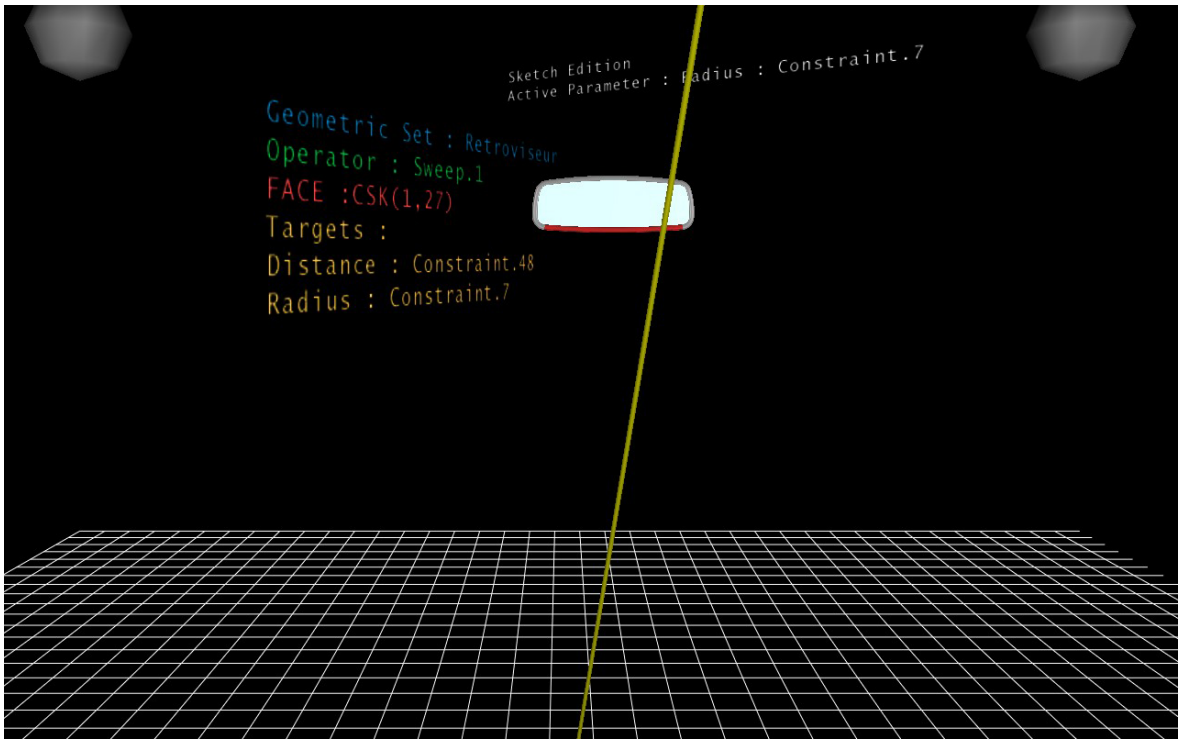


Deuxième sélection et modification

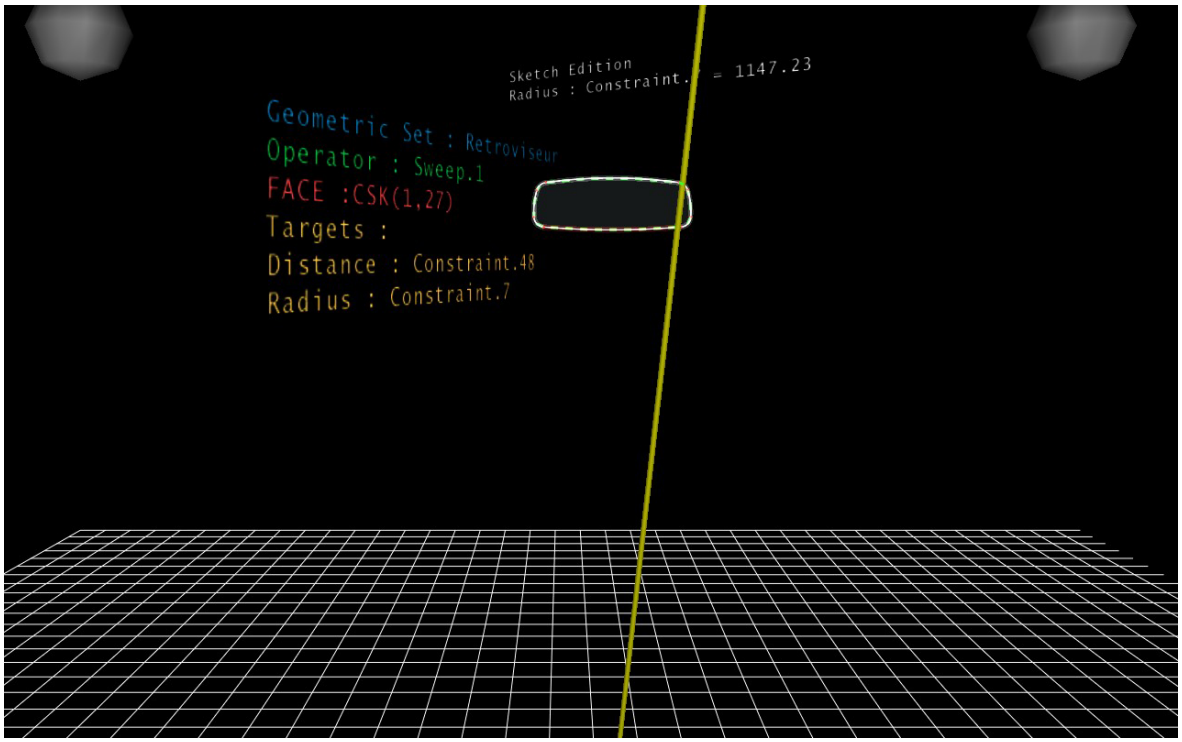
Sélection d'une face topologique (élément rouge). Par défaut, la première contrainte, **Distance - Contrainte.48** est activée.

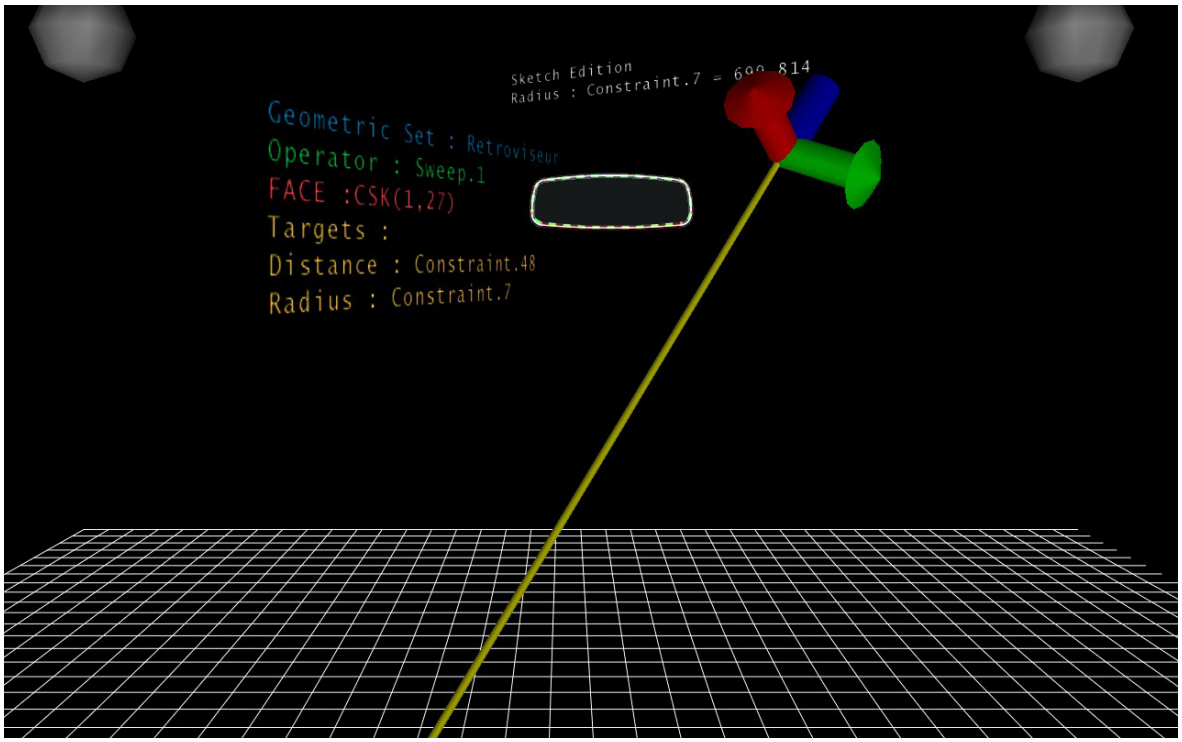


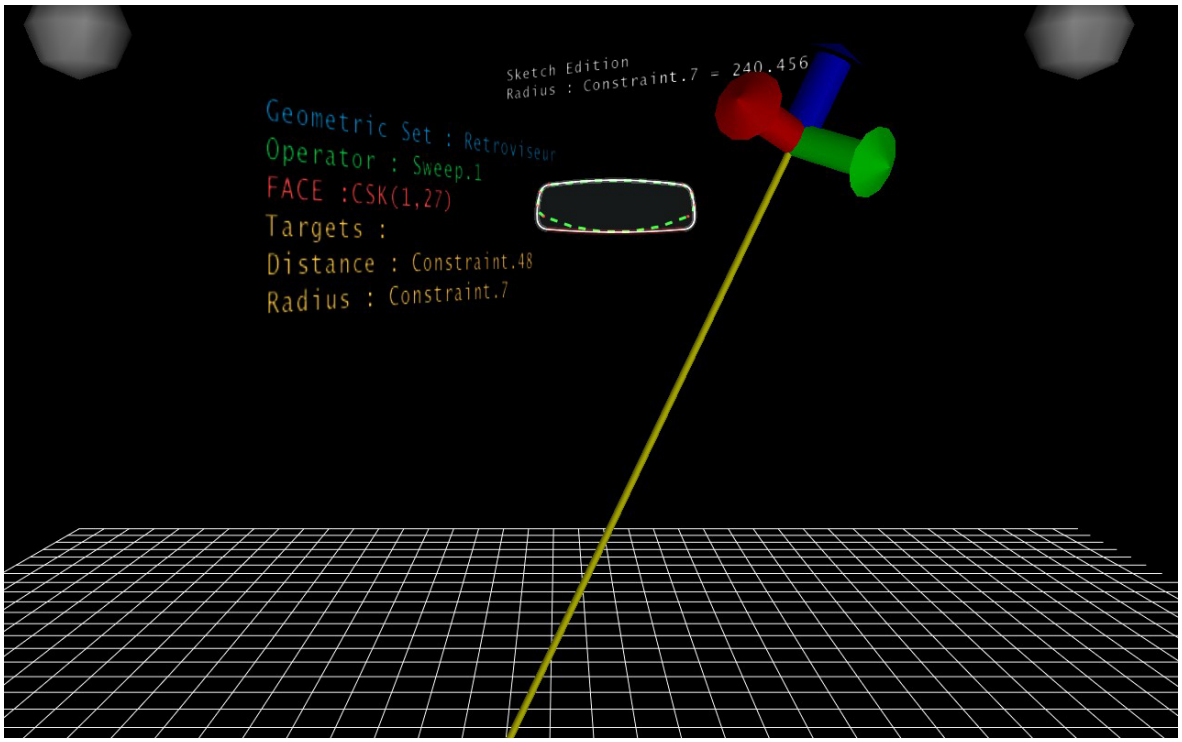
L'utilisateur choisit la cible suivante : le **Rayon - Contrainte.7** est activé.



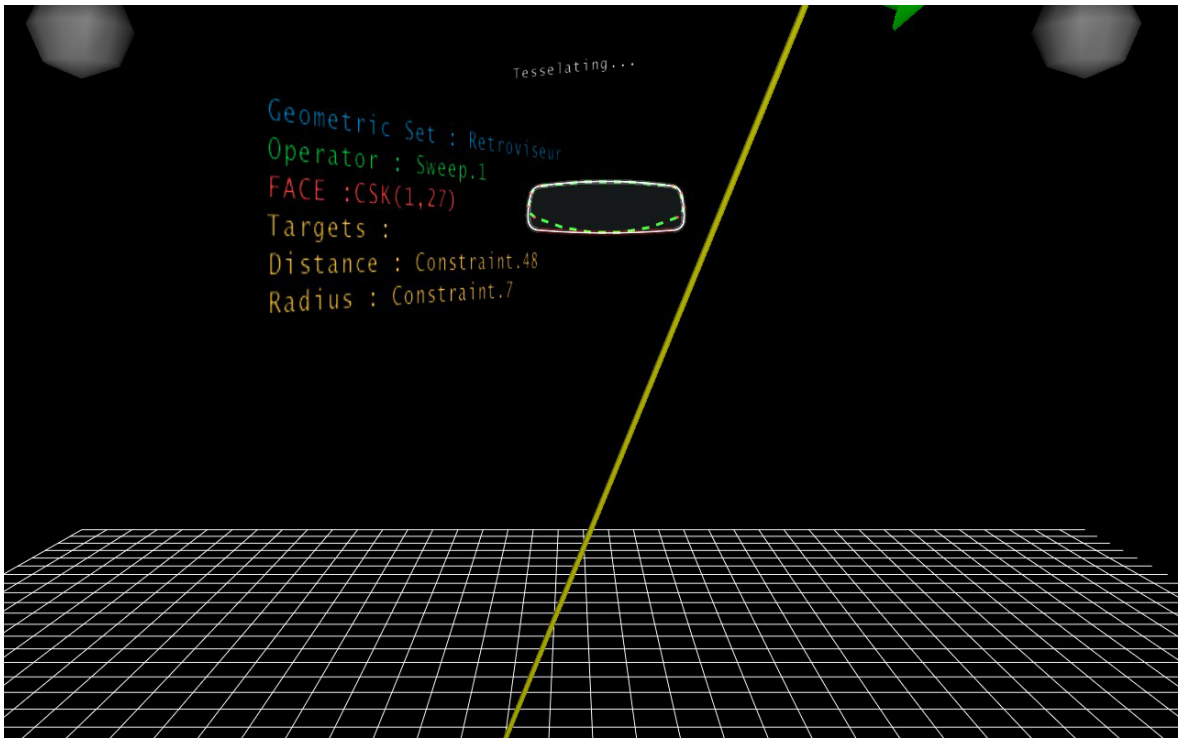
La modification du rayon commence...



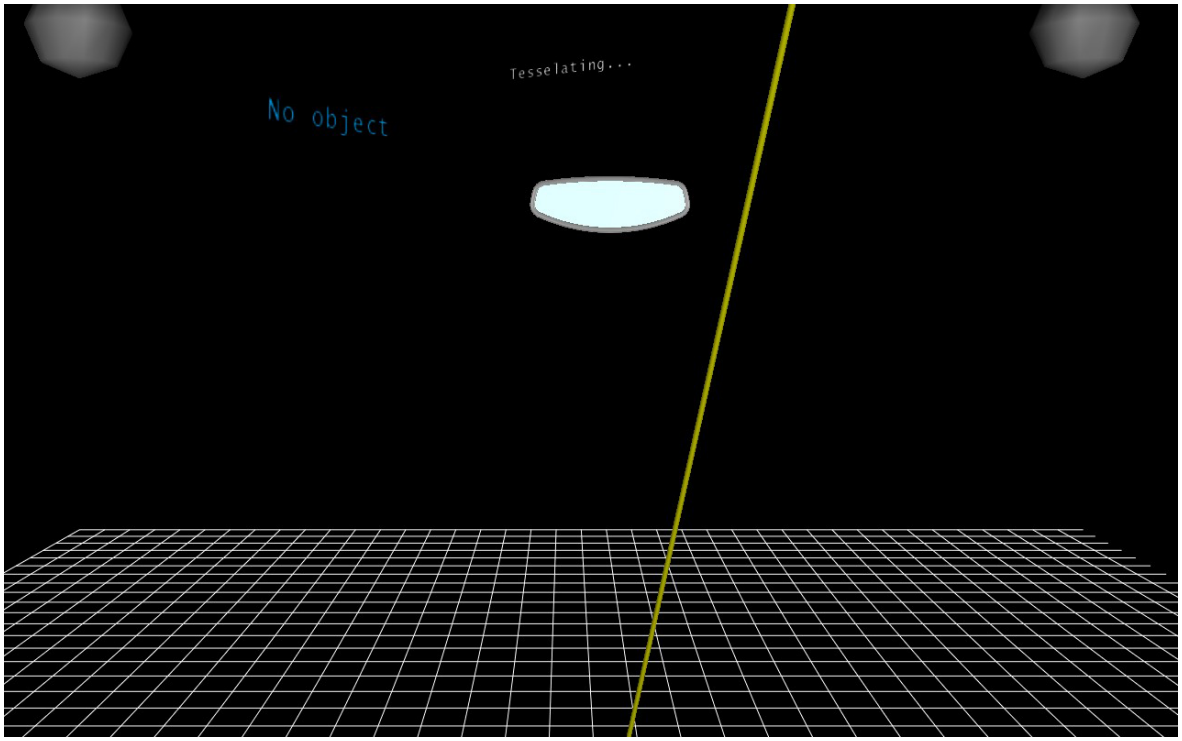




La modification est terminée, l'utilisateur valide sa modification

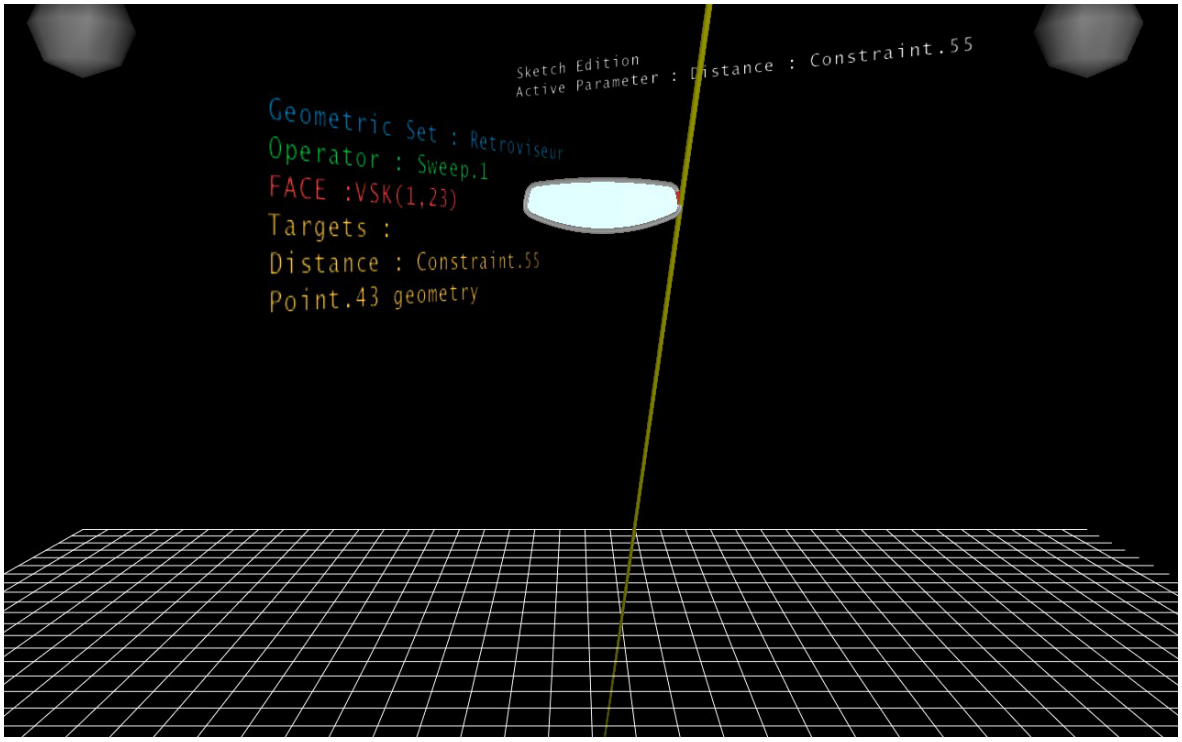


L'objet est complètement à jour.

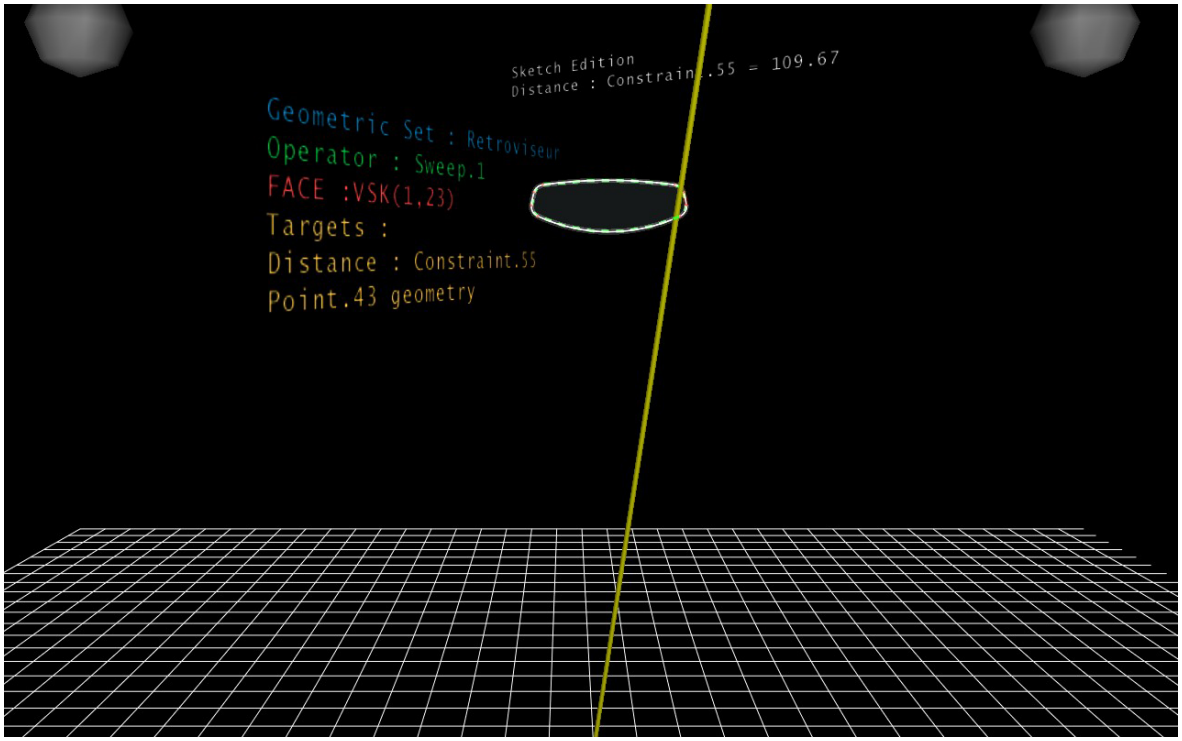


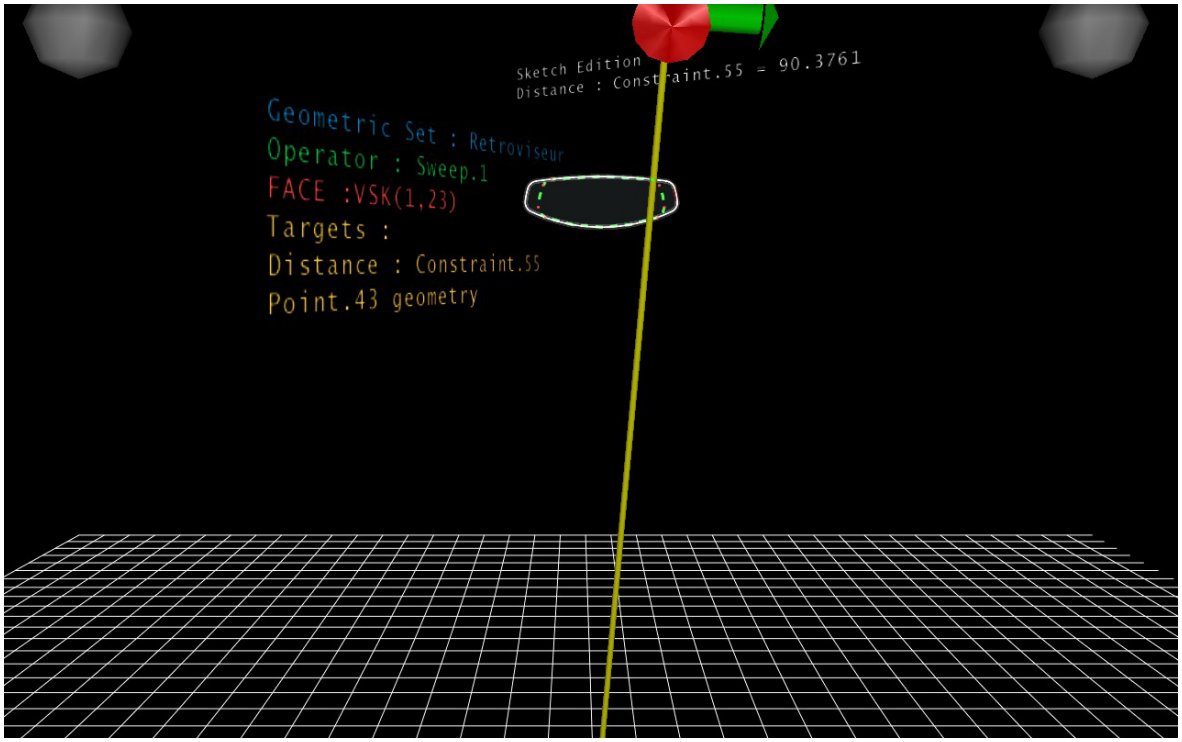
Troisième sélection et modification

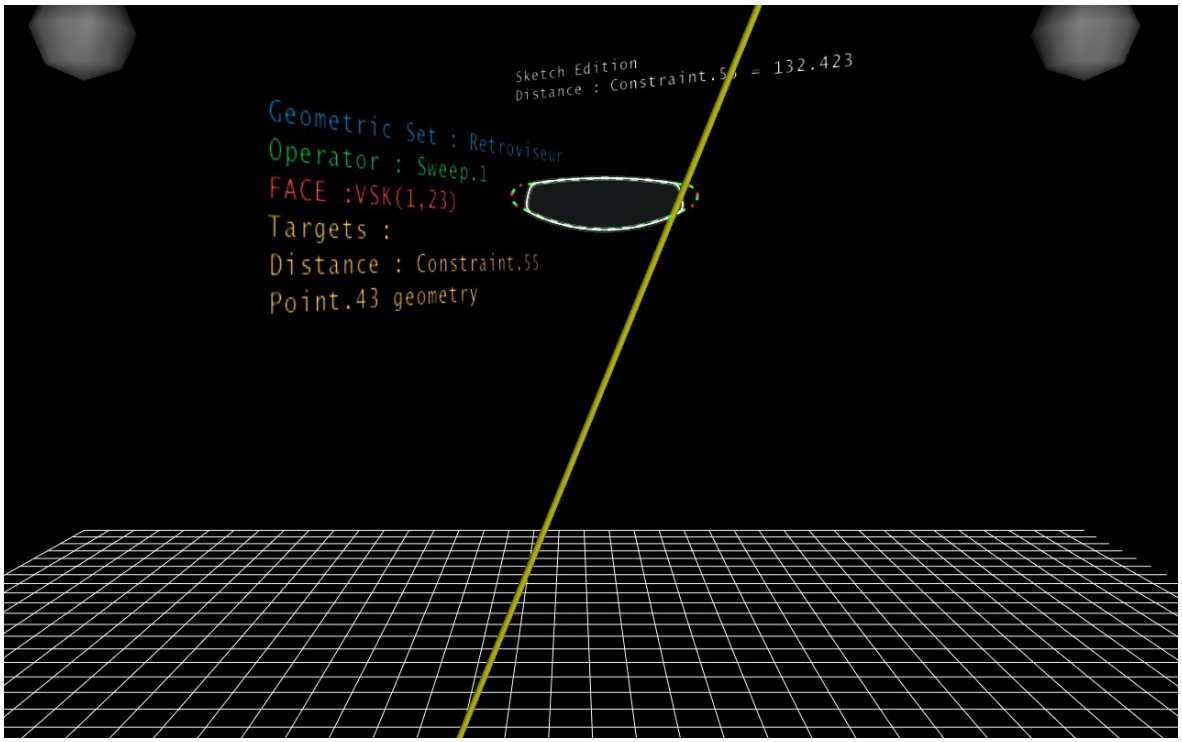
L'utilisateur sélectionne un nouvel élément qui permet de modifier un unique paramètre : la contrainte d'esquisse **Distance - Contrainte.55**.



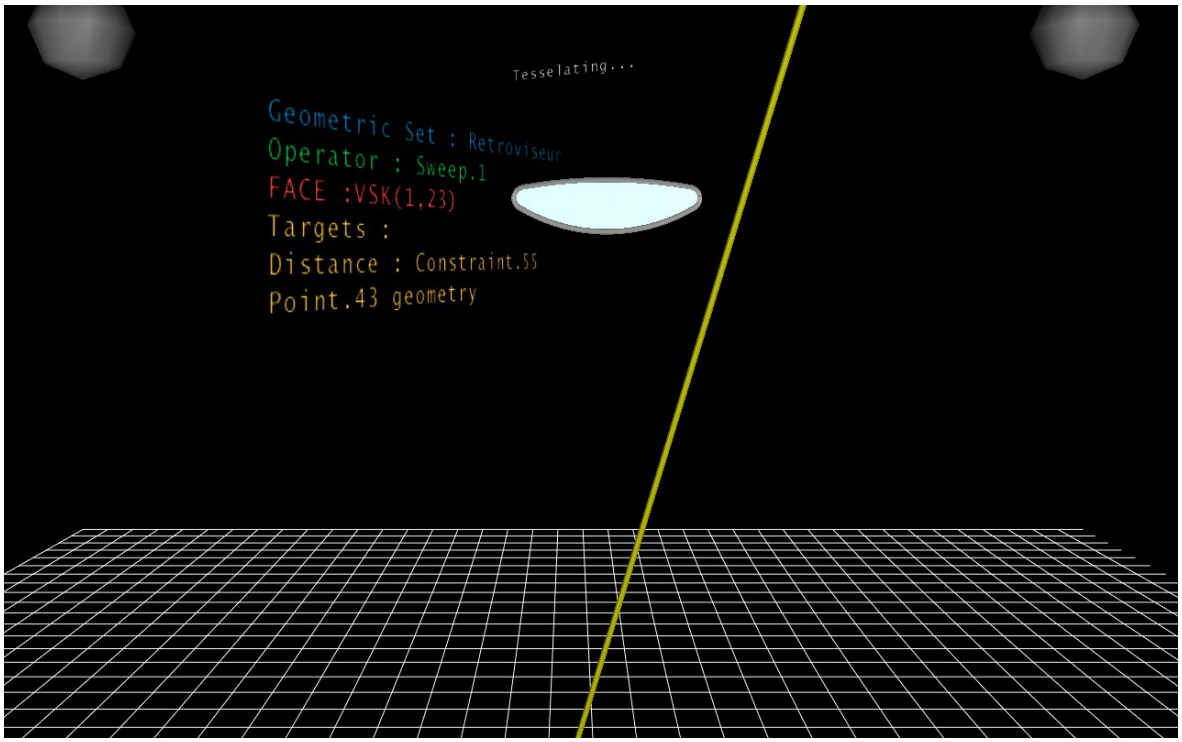
La modification de la distance commence...





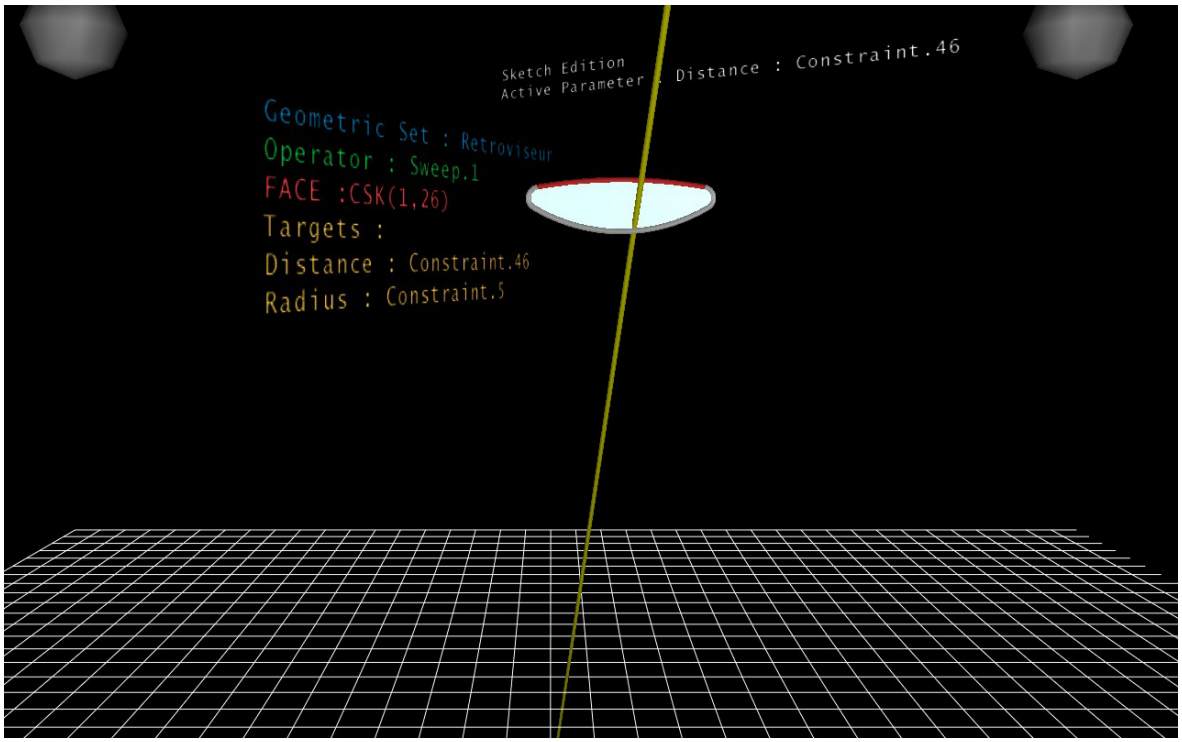


L'utilisateur satisfait valide et peut visualiser l'impact de sa modification.

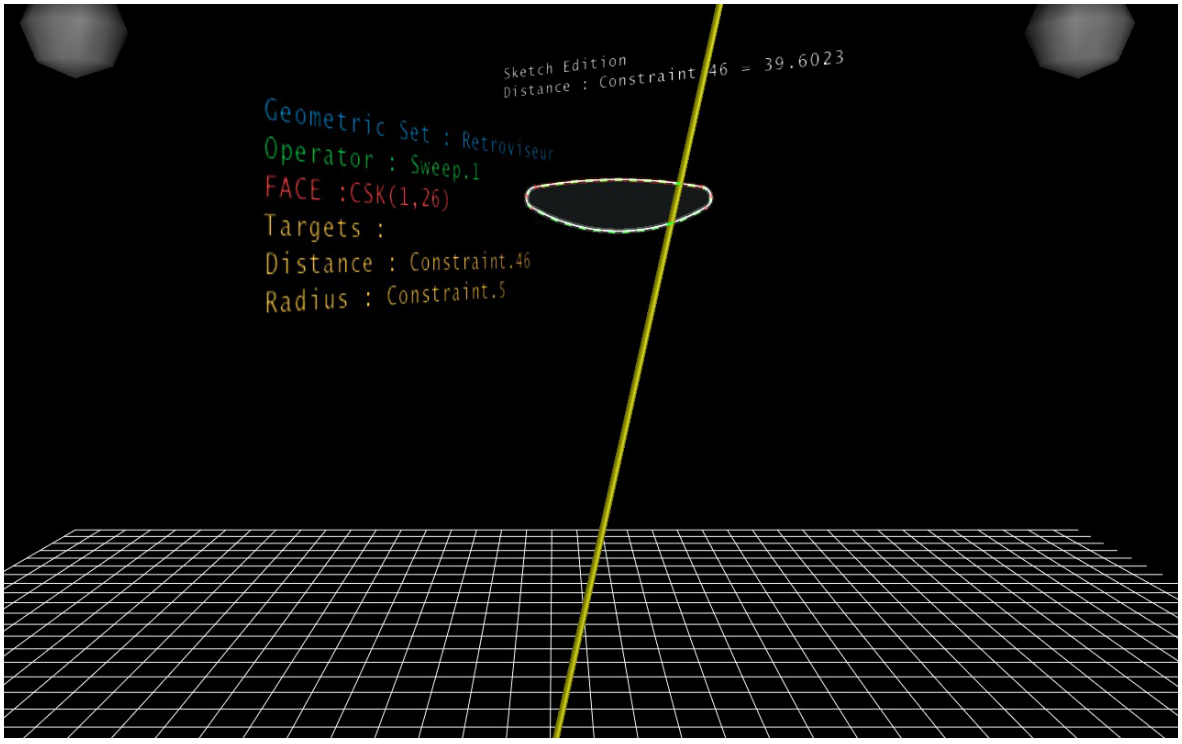


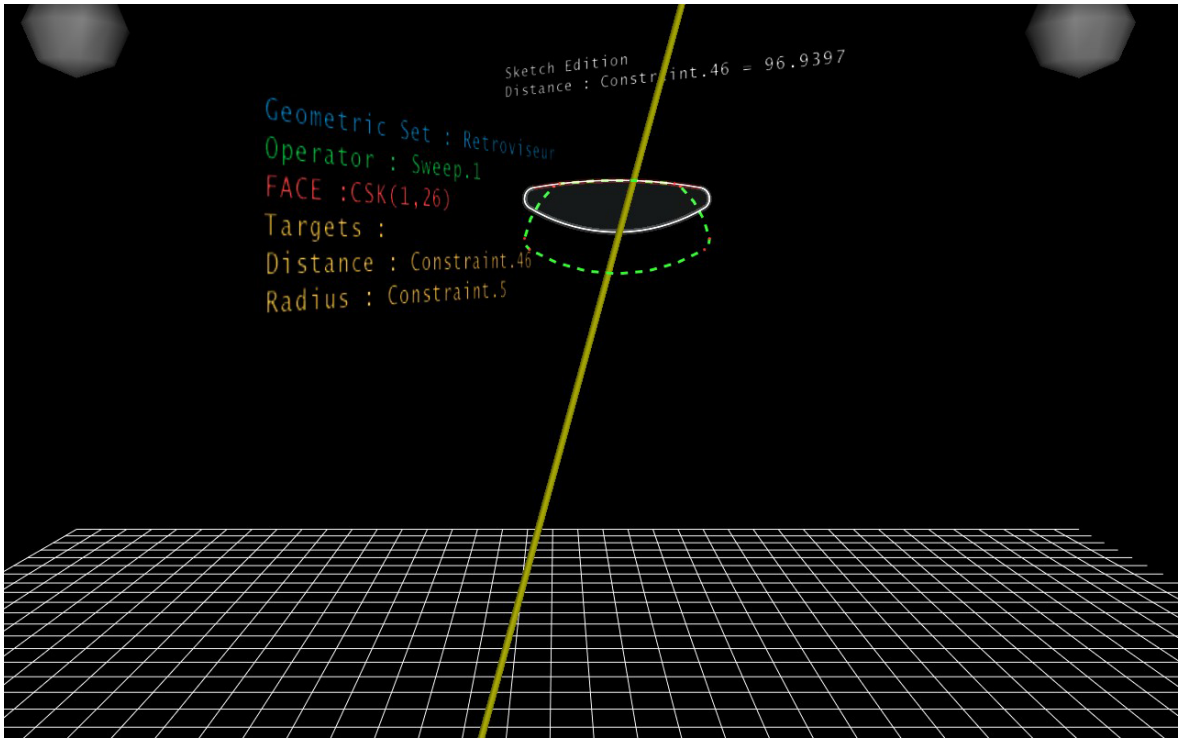
Quatrième sélection et modification

L'utilisateur sélectionne à nouveau la surface supérieure du rétroviseur.

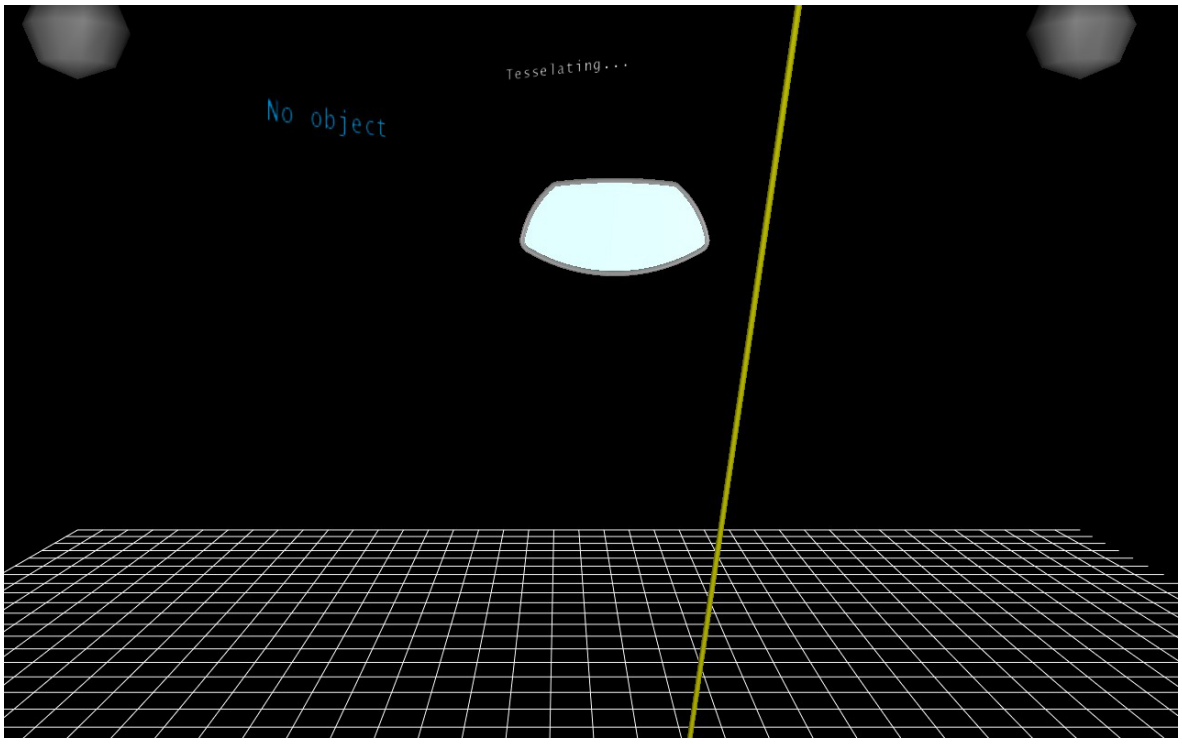


La modification de la contrainte de distance commence...



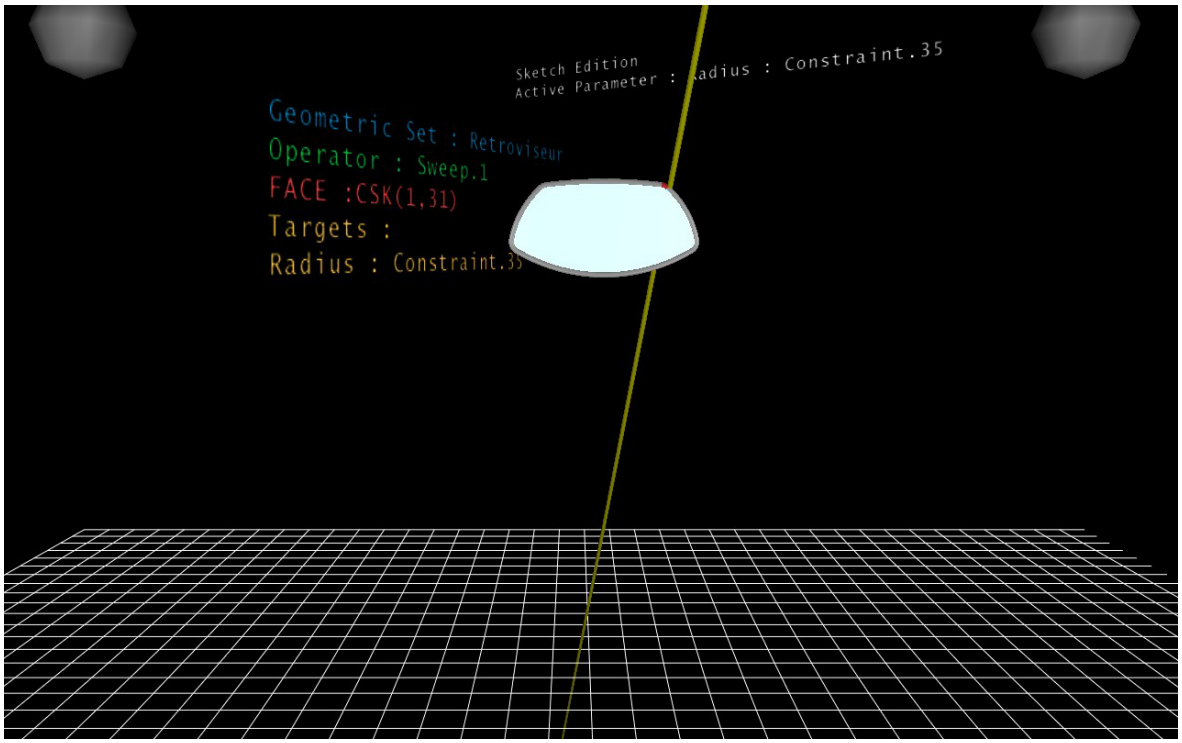


Validation et visualisation de la modification.

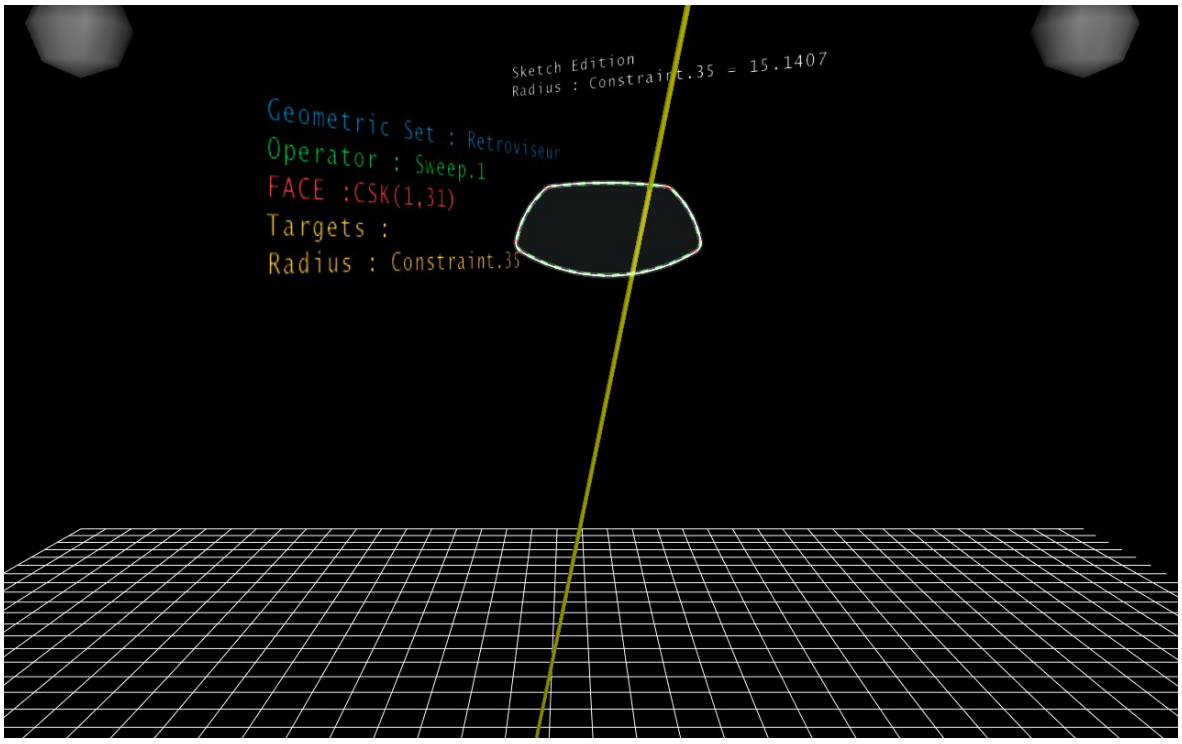


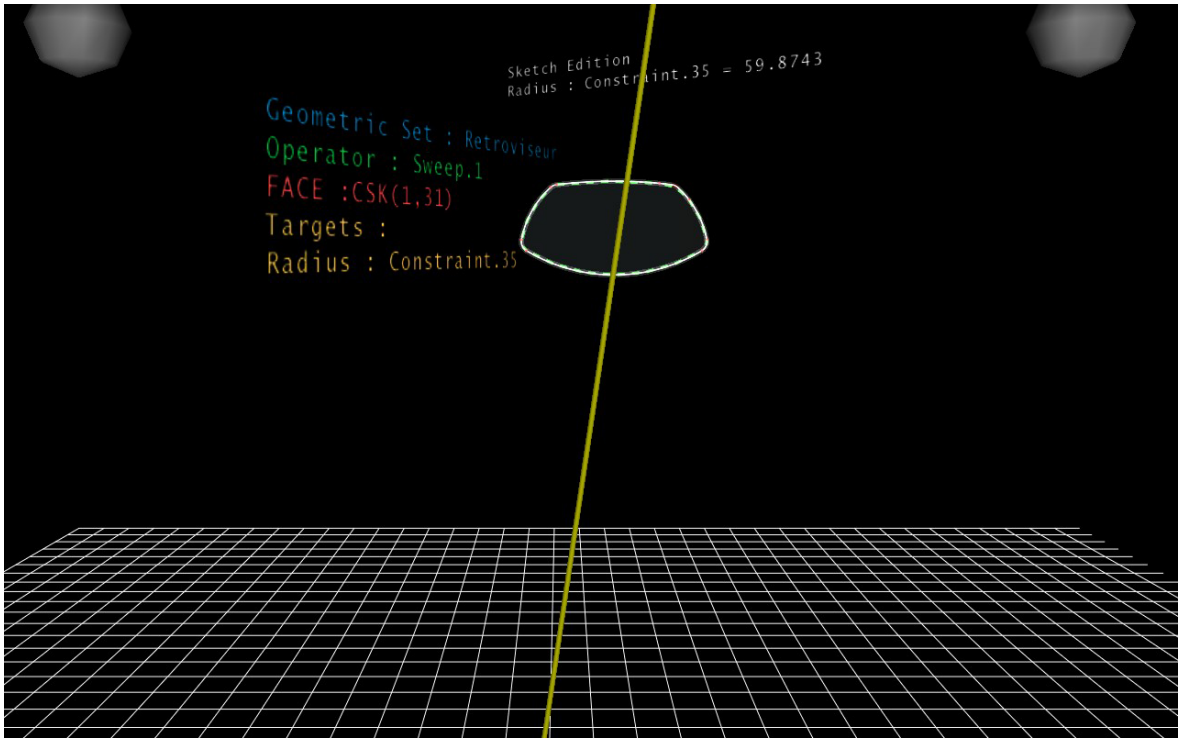
Cinquième et sixième sélections et modifications

L'utilisateur sélectionne cette fois ci un angle du rétroviseur.

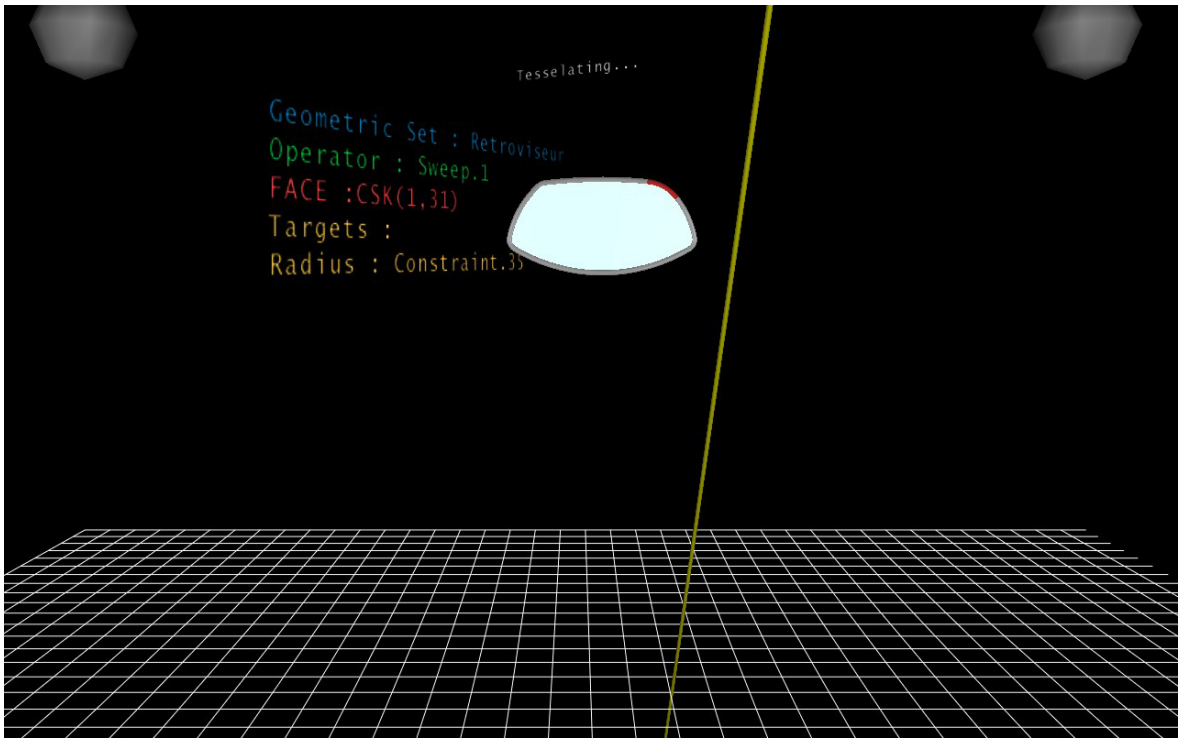


La modification de la contrainte de rayon commence...

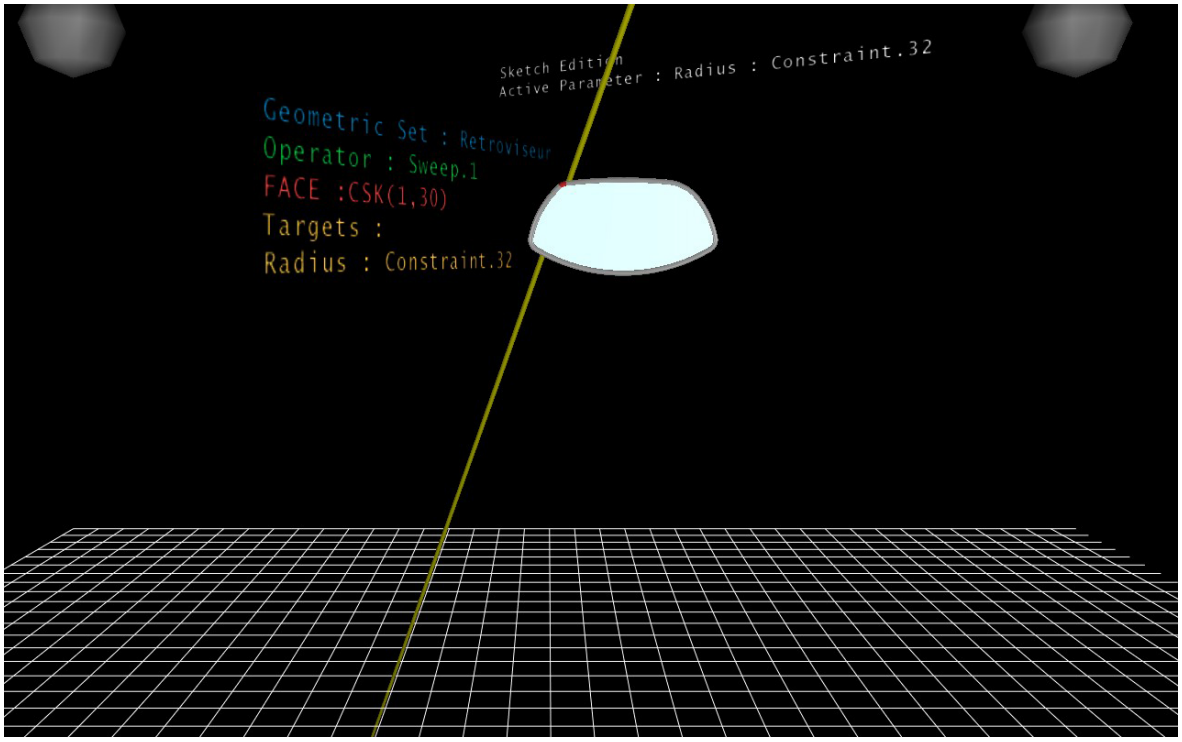




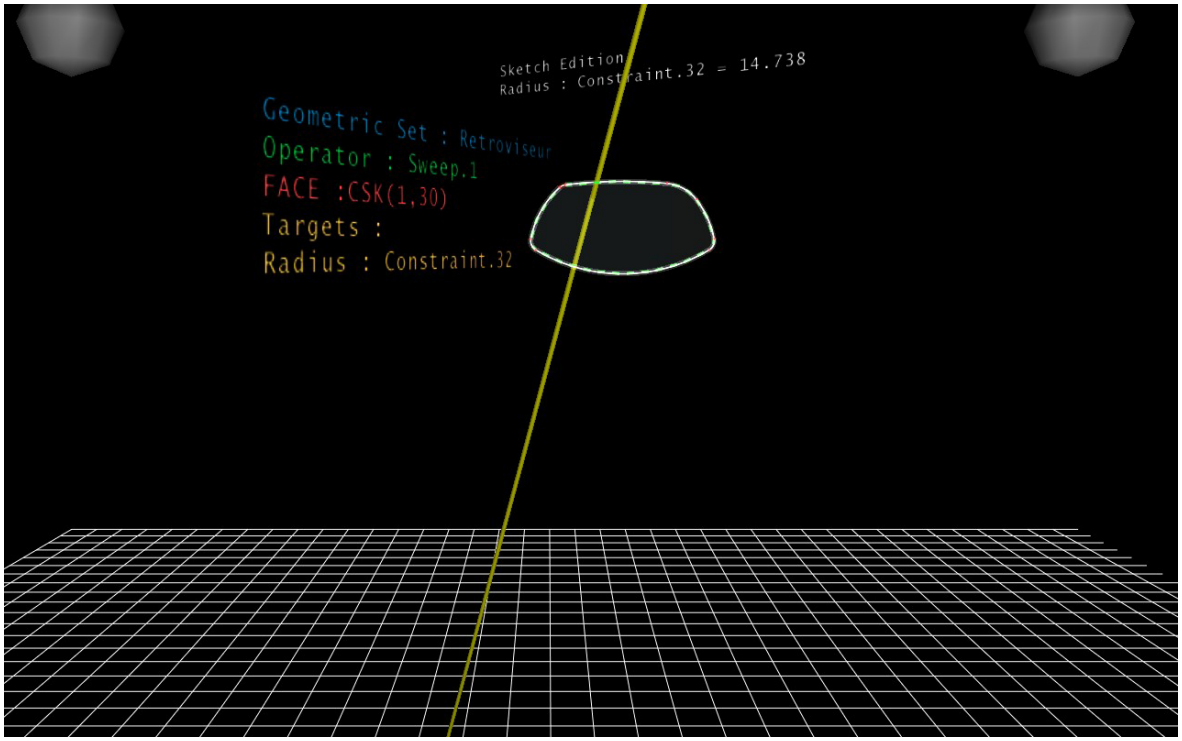
Validation et visualisation de la modification.

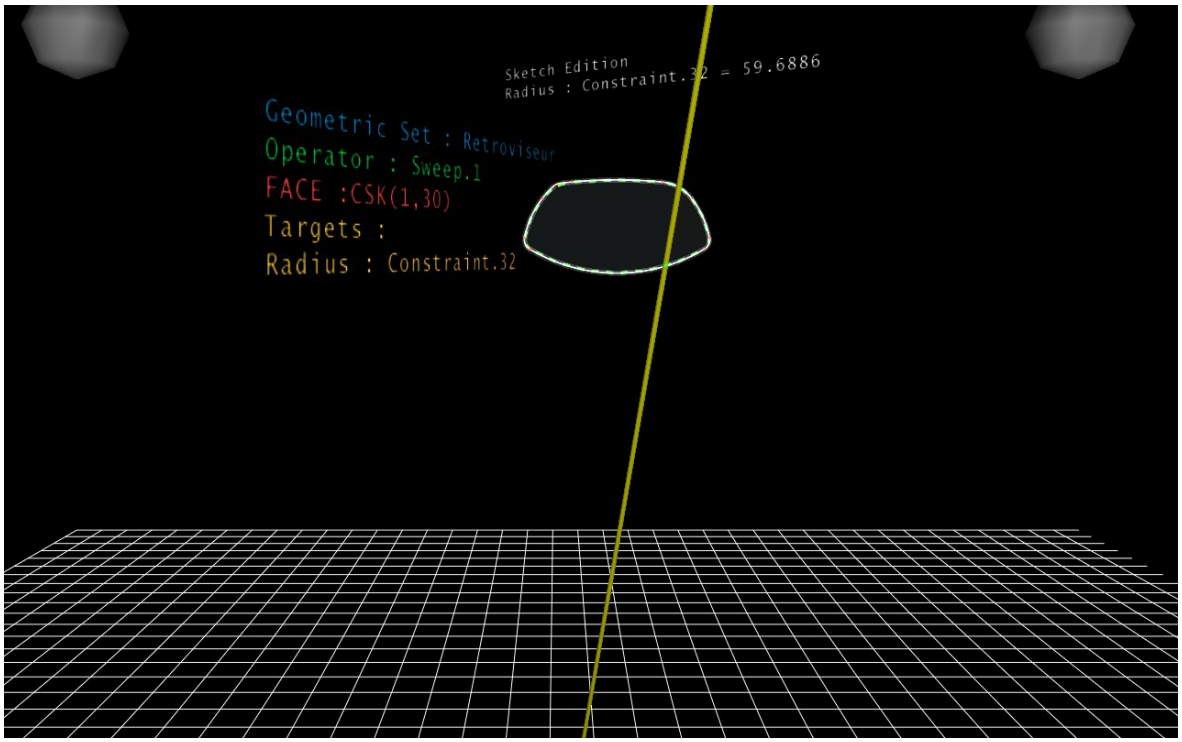


Nouvel angle sélectionné.

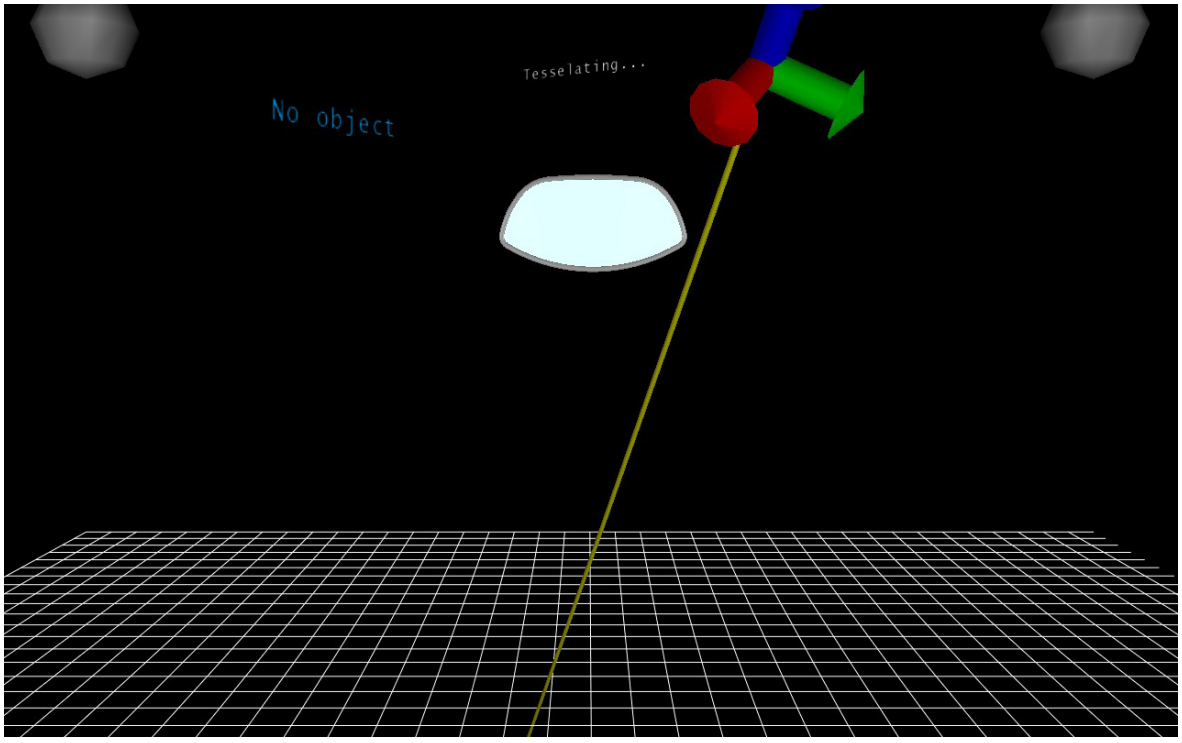


La modification de la contrainte de rayon commence...





L'utilisateur est satisfait de sa modification, il valide.



Sauvegarde finale et retour sur poste de travail

L'utilisateur a fini sa session de travail en sauvegardant le rétroviseur dans son format initial. Il peut alors le recharger sous CATIA afin de vérifier les différentes modifications réalisées en environnement immersif.

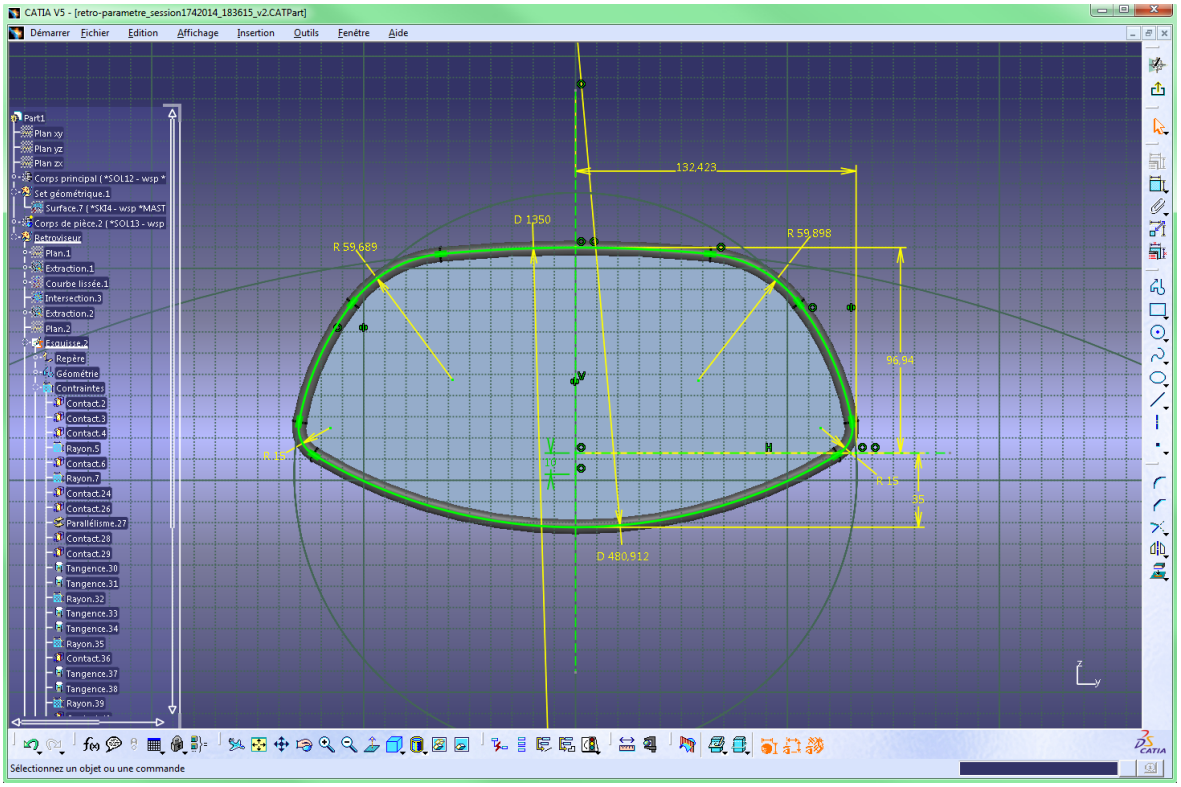


Table des figures

Liste des tableaux

Liste des Algorithmes

Listings

Bibliographie

[1] Andrea F Abate, Mariano Guida, Paolo Leoncini, Michele Nappi et Stefano Ricciardi : A haptic-based approach to virtual training for aerospace industry. *Journal of Visual Languages & Computing*, 20(5):318–325, 2009.

[2] Michael Abramovici : Future trends in product lifecycle management (plm). *In* Frank-Lothar Krause, éditeur : *The Future of Product Development*, pages 665–674. Springer Berlin Heidelberg, 2007.

[3] Dago Agbodan, David Marcheix et Guy Pierra : Persistent naming for parametric models. *In Proceedings of WSCG*, volume 2000, pages 17–38, 2000.

[4] Jorge Alcaide-Marzal, José Antonio Diego-Más, Sabina Asensio-Cuesta et Betina Piqueras-Fiszman : An exploratory study on the use of digital sculpting in conceptual product design. *Design Studies*, 34(2):264–284, 2013.

[5] Csaba Antonya et Doru Talaba : Design evaluation and modification of mechanical systems in virtual environments. *Virtual Reality*, 11(4):275–285, 2007.

[6] Améziane Aoussat : *La pertinence en innovation : nécessité d'une approche plurielle*. Thèse de doctorat, École Nationale Supérieur des Arts et Métiers, Paris, France, 1990.

[7] Bruno Arnaldi, Philippe Fuchs et Jacques Tisseau : Chapitre 1 du Volume 1 du Traité de la Réalité Virtuelle. *Les presses de l'école de Mines de Paris*, pages 1–21, 2003.

[8] Autodesk : 3ds Max. http://www.autodesk.fr/products/autodesk-3ds-max/overview.

[9] Autodesk : Alias Studio Tools. http://www.autodesk.fr/products/autodesk-alias-products/overview.

[10] Autodesk : AutoCAD. http://www.autodesk.fr/products/autodesk-autocad/overview.

[11] Mehdi Baba-Ali : *Système de nomination hiérarchique pour les systèmes paramétriques*. Thèse de doctorat, Université de Poitiers, 2010.

[12] Mehdi Baba-Ali, David Marcheix et Xavier Skapin : A method to improve matching process by shape characteristics in parametric systems. *Computer-Aided Design and Applications*, 6(3):341–350, 2009.

[13] Cornelia Ariadne Bakker : *Environmental information for industrial designers*. Thèse de doctorat, Delft University of Technology, 1995.

[14] Yuki Ban, Takashi Kajinami, Takuji Narumi, Tomohiro Tanikawa et Michitaka Hirose : Modifying an identified angle of edged shapes using pseudo-haptic effects. *In Haptics : Perception, Devices, Mobility, and Communication*, pages 25–36. Springer, 2012.

[15] Jin-Song BAO, Ye JIN, MQ GU, Jun-Qi YAN et Deng-Zhe MA : Immersive virtual product development. *Journal of Materials Processing Technology*, 129(1):592–596, 2002.

[16] Bruce G BAUMGART : Winged edge polyhedron representation. Rapport technique, DTIC Document, 1972.

[17] Bruce G BAUMGART : A polyhedron representation for computer vision. *In Proceedings of the May 19-22, 1975, national computer conference and exposition*, pages 589–596. ACM, 1975.

[18] Lionel BENNES : *Vers une méthodologie de développement d'outils de réalité virtuelle pour faciliter la convergence métiers en conception de produits centrée sur l'homme.* Thèse de doctorat, Université de Technologie de Belfort-Montbeliard, 2013.

[19] Callie BERLINER et James A BRIMSON : *Cost management for today's advanced manufacturing : The CAM-I conceptual design.* Harvard Business School Pr, 1988.

[20] Julien BERTA : Integrating vr and cad. *Computer Graphics and Applications, IEEE*, 19(5):14–19, 1999.

[21] Rafael BIDARRA et Willem F BRONSVOORT : Semantic feature modelling. *Computer-Aided Design*, 32(3):201–225, 2000.

[22] Rafael BIDARRA et Willem F BRONSVOORT : Persistent naming through persistent entities. *In Proceedings of Geometric Modeling and Processing 2002*, pages 233–240. IEEE, 2002.

[23] Rafael BIDARRA, Paulos J NYIRENDA et Willem F BRONSVOORT : A feature-based solution to the persistent naming problem. *Computer-Aided Design and Applications*, 2(1-4):517–526, 2005.

[24] Frank BIOCCA : The cyborg's dilemma : Progressive embodiment in virtual environments [1]. *Journal of Computer-Mediated Communication*, 3(2):0–0, 1997.

[25] Jean-Claude BOCQUET, Marc GABRIEL, Michel GUEURY, Alain JEAN et Jacques NOEL : Maîtriser la conception des produits et des systèmes. *C. Barlier (s/d) : Conception en mécanique industrielle. Partie*, 3, 1996.

[26] Monica BORDEGONI et Umberto CUGINI : Haptic modeling in the conceptual phases of product design. *Virtual Reality*, 9(2-3):192–202, 2006.

[27] Monica BORDEGONI, Francesco FERRISE et Marco AMBROGIO : Haptic interaction and interactive simulation in an ar environment for aesthetic product design. *In Virtual and Mixed Reality*, pages 293–302. Springer, 2009.

[28] B BOSSARD, T CONVARD, A BRAFFORT, D TOURAINE, P BOURDOT et M JARDINO : Un système de reconnaissance de gestes générique pour la Réalité Virtuelle. *In 14eme Congres Francophone AFRIF-AFIA de Reconnaissance des Formes et Intelligence Artificielle*, 2004.

[29] Bruno BOSSARD : *Conception d'un système de reconnaissance automatique de gestes bimanuels : application à la réalité virtuelle et à la langue des signes.* Thèse de doctorat, Université Paris-Sud (XI), 2006.

[30] Patrick BOURDICHON : L'ingénierie simultanée et la gestion d'informations. *Collection systèmes d'information*, 1994.

[31] Patrick BOURDOT : *Reconstruction et interaction 3D : contribution à la réalité virtuelle et augmentée.* Habilitation à dirigier des recherches, Université Paris-Sud (XI), 2002. Notes et Documents LIMSI-CNRS n°2002-11.

[32] Patrick BOURDOT, Thomas CONVARD, Flavien PICON, Mehdi AMMI, Damien TOURAINE et Jean-Marc VÉZIEN : Vr-cad integration : Multimodal immersive interaction and advanced haptic paradigms for implicit edition of cad models. *Computer-Aided Design*, 42(5):445–461, 2010. Advanced and Emerging Virtual and Augmented Reality Technologies in Product Design.

[33] Patrick BOURDOT, Mike KRUS et Rachid GHERBI : Cooperation between reactive 3d objects and a multimodal x window kernel for cad. *In Multimodal Human-Computer Communication*, pages 188–212. Springer, 1998.

[34] Ian C BRAID : The synthesis of solids bounded by many faces. *Communications of the ACM*, 18(4):209–216, 1975.

[35] JC BREZET et C VAN HEMEL : Product development with the environment as innovation strategy–the promise approach. *Delft University of Technology Report*, 1994.

[36] Guillaume BRINCIN et Lionel DOMINJON : VR4D : Virtual Reality for Design. *In Journées de l'Association Française de Réalité Virtuelle (AFRV) 2010*, 2010.

[37] Jean-Marie BURKHARDT : Immersion, réalisme et présence dans la conception et l'évaluation des environnements virtuels. *Psychologie française*, 48(2):35–42, 2003.

[38] W. BUTTERFIELD, M. GREEN, D. SCOTT et W. STOKER : Part features for process planning. Rapport technique, 1986. CAM-I Report R-86-PPP-01.

[39] Jeff BUTTERWORTH, Andrew DAVIDSON, Stephen HENCH et Marc. T. OLANO : 3dm : A three dimensional modeler using a head-mounted display. *In Proceedings of the 1992 Symposium on Interactive 3D Graphics*, I3D '92, pages 135–138, New York, NY, USA, 1992. ACM.

[40] Vasilis CAPOYLEAS, Xiangping CHEN et Christoph M HOFFMANN : Generic naming in generative, constraint-based design. *Computer-Aided Design*, 28(1):17–26, 1996.

[41] Xiangping CHEN et Christoph M HOFFMANN : On editability of feature-based design. *Computer-Aided Design*, 27(12):905–914, 1995.

[42] ACK CHOI, DSK CHAN et AMF YUEN : Application of virtual assembly tools for improving product design. *The International Journal of Advanced Manufacturing Technology*, 19(5):377–383, 2002.

[43] Chi-Cheng P CHU, Tushar H DANI et Rajit GADH : Multi-sensory user interface for a virtual-reality-based computeraided design system. *Computer-Aided Design*, 29(10): 709–725, 1997.

[44] Vincent A CICIRELLO et William C REGLI : Resolving non-uniqueness in design feature histories. *In Proceedings of the fifth ACM symposium on Solid modeling and applications*, pages 76–84. ACM, 1999.

[45] Alain CLAPAUD : Psa : deux fois moins de tests physiques pour concevoir la peugeot 208., 2012. http://pro.01net.com/editorial/570065/psa-deux-fois-moins-de-tests-physiques-pour-concevoir-la-peugeot-208/.

[46] Laurent CLAUDON : Apports et limites des mannequins numériques pour la conception des postes de travail à travers deux études de cas. *In Ergo'IA*, pages 15–17, 2008.

[47] Hugh I CONNACHER, Sankar JAYARAM et Kevin W LYONS : Virtual assembly design environment. *In ASME Comuters in Engineering 1995*, Boston, MA, USA, 1995.

[48] Thomas CONVARD : *Conception assistée par ordinateur en environnement immersif.* Thèse de doctorat, Université Paris-Sud (XI), 2005.

[49] Thomas CONVARD et Patrick BOURDOT : History based reactive objects for immersive cad. *In Proceedings of the Ninth ACM Symposium on Solid Modeling and Applications*, SM '04, pages 291–296. Eurographics Association, 2004.

[50] Thomas CONVARD, Patrick BOURDOT et Jean-Marc VÉZIEN : Managing deformable objects in cluster rendering. *Computational Science-ICCS 2005*, pages 290–297, 2005.

[51] Carolina CRUZ-NEIRA, Daniel J. SANDIN, Thomas A. DEFANTI, Robert V. KENYON et John C. HART : The cave : Audio visual experience automatic virtual environment. *Commun. ACM*, 35(6):64–72, juin 1992.

[52] Umberto CUGINI et Monica BORDEGONI : Touch and design : novel haptic interfaces for the generation of high quality surfaces for industrial design. *The Visual Computer*, 23(4):233–246, 2007.

[53] J.J. CUNNINGHAM et J.R. DIXON : Designing with features : the origin of features. 1988.

[54] Fan DAI et Martin GÖBEL : Virtual prototyping. an approach using vr-techniques. *In ASME Computers in Engineering 1994*, volume 1, pages 311–316, Minneapolis, MN, USA, 1994.

[55] Fan DAI et Peter REINDL : Enabling digital mock up with virtual reality techniques–vision, concept, demonstrator. *In Proceedings of 1996 asme design engineering technical conference and computers in engineering*, 1996.

[56] Tushar H DANI et Rajit GADH : Creation of concept shape designs via a virtual reality interface. *Computer-Aided Design*, 29(8):555–563, 1997.

[57] DASSAULT SYSTÈMES : 3DVIA Virtools. http://www.3ds.com/products-services/3dvia/3dvia-virtools/.

[58] DASSAULT SYSTÈMES : CATIA. http://www.3ds.com/products-services/catia/welcome/.

[59] DASSAULT SYSTÈMES : SOLIDWORKS. http://www.solidworks.fr/.

[60] Raffaele de AMICIS, Fabio BRUNO, André STORK et Maria Laura LUCHI : The eraser pen : a new interaction paradigm for curve sketching in 3d. *In Proceedings of the 7th International Design Conference (DESIGN)*, pages 465–470, 2002.

[61] Raffaele de AMICIS, Michele FIORENTINO et André STORK : Parametric interaction for cad application in virtual reality environment. *In Proceedings of 12th International Conference on Design Tools and Methods in Industrial Engineering*, pages 43–52, 2001.

[62] Bruno R DE ARAÚJO, Géry CASIEZ, Joaquim A JORGE et Martin HACHET : Mockup builder : 3d modeling on and above the surface. *Computers & Graphics*, 37(3):165–178, 2013.

[63] Antonio Gomes de SÀ et Gabriel ZACHMANN : Integrating virtual reality for virtual prototyping. *In Proceedings of the 1997 ASME Design for Engineering Technical Conferences*, 1998.

[64] Antonio Gomes de SÁ et Gabriel ZACHMANN : Virtual reality as a tool for verification of assembly and maintenance processes. *Computers & Graphics*, 23(3):389–403, 1999.

[65] Robert DUCHAMP : *Méthodes de conception de produits nouveaux*. Hermes Science, 1999.

[66] Atila ERTAS et Jesse C JONES : *The engineering design process*, volume 1. John Wiley & Sons, 1993.

[67] Mark EVANS, David WALLACE, David CHESHIRE et Bahar SENER : An evaluation of haptic feedback modelling during industrial design practice. *Design Studies*, 26(5):487–508, 2005.

[68] Mingxian FA, Terrence FERNANDO et Peter M. DEW : Interactive constraint-based solid modeling using allowable motion. *In Proceedings on the Second ACM Symposium on Solid Modeling and Applications*, SMA '93, pages 243–252. ACM, 1993.

[69] Michele FIORENTINO, Raffaele de AMICIS, Giuseppe MONNO et André STORK : Spacedesign : A mixed reality workspace for aesthetic industrial design. *In Proceedings of the 1st International Symposium on Mixed and Augmented Reality*, ISMAR '02, pages 86–, Washington, DC, USA, 2002. IEEE Computer Society.

[70] Michele FIORENTINO, Giuseppe MONNO et Antonio Emmanuele UVA : Smart tools for virtual reality based cad. *In ADM-AIAS International Conference*, 2004.

[71] Andrew FORSBERG, JJ LAVIOLA, Lee MARKOSIAN et Robert C ZELEZNIK : Seamless interaction in virtual reality. *Computer Graphics and Applications, IEEE*, 17(6):6–9, 1997.

[72] Maxim FOURSA, David D'ANGELO, Gerold WESCHE et Manfred BOGEN : A two-user framework for rapid immersive full cycle product customization. *In* Randall SHUMAKER, éditeur : *Virtual and Mixed Reality*, volume 5622 de *Lecture Notes in Computer Science*, pages 566–575. Springer Berlin Heidelberg, 2009.

[73] Association française pour la QUALITÉ : *Guide des outils de la qualité*. AFCIQ/AFW, Paris, 1990.

[74] Philippe FUCHS, Alain BERTHOZ et Jean-Louis VERCHER : Introduction à la réalité virtuelle. *Le Traité de la Réalité Virtuelle*, 1:1–21, 2006.

[75] Philippe FUCHS, Guillaume MOREAU, Sabine COQUILLART et Jean-Marie BURKHARDT : *Le traité de la réalité virtuelle : Interfaçage, immersion et interaction en environnement virtuel*, volume 2. Presses des MINES, 2006.

[76] Shuming GAO, Huagen WAN et Qunsheng PENG : An approach to solid modeling in a semi-immersive virtual environment. *Computers & Graphics*, 24(2):191–202, 2000.

[77] Christoph M HOFFMANN : On the semantics of generative geometry representations. 1993.

[78] Christoph M HOFFMANN et Robert JUAN : Erep an editable high-level representation for geometric design and analysis. 1992.

[79] Vladimir HUBKA, Mogens Myrup ANDREASEN, Wolfgang Ernst EDER et Peter J HILLS : *Practical studies in systematic design*. Butterworths London etc., 1988.

[80] Takeo IGARASHI, Satoshi MATSUOKA et Hidehiko TANAKA : Teddy : A sketching interface for 3d freeform design. *In Proceedings of the 26th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '99, pages 409–416, New York, NY, USA, 1999. ACM Press/Addison-Wesley Publishing Co.

[81] Tommaso INGRASSIA et Francesco CAPPELLO : Virde : a new virtual reality design approach. *International Journal on Interactive Design and Manufacturing (IJIDeM)*, 3(1):1–11, 2009.

[82] Sankar JAYARAM, Uma JAYARAM, Young Jun KIM, Charles DECHENNE, Kevin W LYONS, Craig PALMER et Tatsuki MITSUI : Industry case studies in the use of immersive virtual assembly. *Virtual Reality*, 11(4):217–228, 2007.

[83] Sankar JAYARAM, Uma JAYARAM, Yong WANG, Hrishikesh TIRUMALI, Kevin LYONS et Peter HART : Vade : a virtual assembly design environment. *Computer Graphics and Applications, IEEE*, 19(6):44–50, 1999.

[84] Uma JAYARAM, Sankar JAYARAM, Imtiyaz SHAIKH, YoungJun KIM et Craig PALMER : Introducing quantitative analysis methods into virtual environments for real-time and continuous ergonomic evaluations. *Computers in industry*, 57(3):283–296, 2006.

[85] Anton JEZERNIK et Gorazd HREN : A solution to integrate computer-aided design (cad) and virtual reality (vr) databases in design and manufacturing processes. *The International Journal of Advanced Manufacturing Technology*, 22(11-12):768–774, 2003.

[86] Claude JOUINEAU : Analyse de la valeur. *Techniques de l'ingénieur, Traité Conception des produits industriels AG6, Paris*, 1993.

[87] Bernhard JUNG, Martin HOFFHENKE et Ipke WACHSMUTH : Virtual assembly with construction kits. *In Proceedings of the 1998 ASME Design for Engineering Technical Conferences 1997 Design for Manufacturing Conference (DECT-DFM'98)*, 1997.

[88] Daniel F KEEFE, Daniel ACEVEDO, Jadrian MILES, Fritz DRURY, Sharon M SWARTZ et David H LAIDLAW : Scientific sketching for collaborative vr visualization design. *IEEE Transactions on Visualization and Computer Graphics*, 14(4):835–847, 2008.

[89] Gregory A KEOLEIAN, Dan MENEREY *et al.* : Life cycle design guidance manual : Environmental requirements and the product system. *In Life cycle design guidance manual : environmental requirements and the product system.* EPA, 1993.

[90] Jiri KRIPAC : *Topological ID system.* Thèse de doctorat, Czech Technical Universty in Prague, 1994.

[91] Jiri KRIPAC : A mechanism for persistently naming topological entities in history-based parametric solid models. *In Proceedings of the Third ACM Symposium on Solid Modeling and Applications*, SMA '95, pages 21–30, New York, NY, USA, 1995. ACM.

[92] Jiri KRIPAC : A mechanism for persistently naming topological entities in history-based parametric solid models. *Computer-Aided Design*, 29(2):113–122, 1997.

[93] S KÜHNER, E RANK et M KRAFCZYK : Efficient reduction of 3d simulation results based on spacetree data structures for data analysis in virtual reality environments. *Applied Virtual Reality in Engineering and Construction, Gothenburg, Sweden*, 2001.

[94] Jean-Charles LEBAHAR : *La conception en design industriel et en architecture.* Éditions Lavoisier, 2007.

[95] Anatole LÉCUYER : *Contribution à l'étude des retours haptique et pseudo-haptique et de leur impact sur les simulations d'opérations de montage/démontage en aéronautique.* Thèse de doctorat, Université Paris-Sud (XI), 2001.

[96] Anatole LECUYER, Sabine COQUILLART, Abderrahmane KHEDDAR, Paul RICHARD et Philippe COIFFET : Pseudo-haptic feedback : Can isometric input devices simulate force feedback ? *In Virtual Reality, 2000. Proceedings. IEEE*, pages 83–90. IEEE, 2000.

[97] Sylvain LENFLE et Christophe MIDLER : Expansion des produits, des usages, des marchés et dynamique du système de conception : l'exemple de la voiture communicante. *Les nouveaux régimes de la conception. Langages, théories, métiers*, pages 125–147, 2008.

[98] Jiandong LIANG et Mark GREEN : Jdcad : A highly interactive 3d modeling system. *Computers & Graphics*, 18(4):499–506, 1994.

[99] LIGHT & SHADOWS : http://light-and-shadows.com/.

[100] Matthew LOMBARD et Theresa DITTON : At the heart of it all : The concept of presence. *Journal of Computer-Mediated Communication*, 3(2):0–0, 1997.

[101] Joseph S LOMBARDO, Edward MIHALAK et Scott R OSBORNE : Collaborative virtual prototyping. *Johns Hopkins APL Technical Digest*, 17(3):295, 1996.

[102] Weiyin MA, Yongmin ZHONG, Shiu-Kit TSO et Tianxiang ZHOU : A hierarchically structured and constraint-based data model for intuitive and precise solid modeling in a virtual reality environment. *Computer-Aided Design*, 36(10):903–928, 2004.

[103] Morad MAHDJOUB : *La réalité virtuelle pour une conception de systèmes mécaniques centrée sur l'utilisateur.* Thèse de doctorat, Université de Technologie de Belfort-Montbeliard, 2007.

[104] Paul MAITRE, Jacques-Didier MIQUEL et Pascale BRENET : *De l'idée au produit : guide de la valorisation industrielle de la recherche.* Eyrolles, 1992.

[105] Fabrizia MANTOVANI, Gianluca CASTELNUOVO, Andrea GAGGIOLI et Giuseppe RIVA : Virtual reality training for health-care professionals. *CyberPsychology & Behavior*, 6(4):389–395, 2003.

[106] Martti MANTYLA : A note on the modeling space of euler operators. *Computer vision, graphics, and image processing*, 26(1):45–60, 1984.

[107] Martti MÄNTYLÄ : *An introduction to solid modeling.* Computer science press, 1988.

[108] David MARCHEIX et Guy PIERRA : A survey of the persistent naming problem. *In Proceedings of the seventh ACM symposium on Solid modeling and applications*, pages 13–22. ACM, 2002.

[109] Jacques MARSOT : Conception et ergonomie. méthodes et outils pour intégrer l'ergonomie dans le cycle de conception des outils à mains. *Note scientifique et technique n ° 219 - INRS*, 2002.

[110] Gale L MARTIN : The utility of speech input in user-computer interfaces. *Int. J. Man-Mach. Stud.*, 30(4):355–375, avril 1989.

[111] Pierre MARTIN, Patrick BOURDOT et Stéphane MASFRAND : A VR-CAD data model for immersive and intuitive conception. En cours de soumission à International Journal on Interactive Manufacturing and Design (IJIDeM).

[112] Pierre MARTIN, Patrick BOURDOT et Damien TOURAINE : A reconfigurable architecture for multimodal and collaborative interactions in virtual environments. *In 3D User Interfaces (3DUI), 2011 IEEE Symposium on*, pages 11–14. IEEE, 2011.

[113] Pierre MARTIN, Nicolas FÉREY, Céline CLAVEL et Patrick BOURDOT : A haptic paradigm to learn how to drive a non-motorised vehicle manipulated through an articulated mechanism. *In Proceedings of Joint Virtual Reality Conference of* ICAT-EGVE-E*uro*VR *2012 (JVRC 2012)*, pages 81–84, 2012.

[114] Pierre MARTIN, Nicolas FÉREY, Céline CLAVEL, Françoise DARSES et Patrick BOURDOT : Sensorimotor feedback for interactive realism : evaluation of a haptic driving paradigm for a forklift simulator. *In Haptics : Perception, Devices, Mobility, and Communication*, pages 314–325. Springer, 2012.

[115] Pierre MARTIN, Anthony TSEU, Nicolas FÉREY, Damien TOURAINE et Patrick BOURDOT : A hardware and software architecture to deal with multimodal and collaborative interactions in multiuser virtual reality environments. *In IS&T/SPIE Electronic Imaging.* International Society for Optics and Photonics, 2014.

[116] Herman MEINDERS : Point of no return. *Philips EcoDesign Guidelines*, 1997.

[117] Daniel MESTRE et Philippe FUCHS : Immersion et présence. *Le Traité de la Réalité Virtuelle*, pages 309–338, 2006.

[118] Vincent MEYRUEIS : *Modification interactive de formes en Réalité Virtuelle. Application à la conception d'un produit.* Thèse de doctorat, École nationale supérieure des mines de Paris, 2011.

[119] Vincent MEYRUEIS, Alexis PALJIC et Philippe FUCHS : D 3 : an immersive aided design deformation method. *In Proceedings of the 16th ACM Symposium on Virtual Reality Software and Technology*, pages 179–182. ACM, 2009.

[120] G MICHALOS, S MAKRIS, N PAPAKOSTAS, D MOURTZIS et G CHRYSSOLOURIS : Automotive assembly technologies review : challenges and outlook for a flexible and adaptive approach. *CIRP Journal of Manufacturing Science and Technology*, 2(2):81–91, 2010.

[121] Sushmita MITRA et Tinku ACHARYA : Gesture recognition : A survey. *Systems, Man, and Cybernetics, Part C : Applications and Reviews, IEEE Transactions on*, 37(3):311–324, 2007.

[122] TS MUJBER, T SZECSI et MSJ HASHMI : Virtual reality applications in manufacturing process simulation. *Journal of materials processing technology*, 155:1834–1838, 2004.

[123] Duhwan MUN et Soonhung HAN : Identification of topological entities and naming mapping for parametric cad model exchanges. *International Journal of CAD/CAM*, 5(1), 2005.

[124] Gerhard PAHL et Wolfgang BEITZ : Engineering design : a systematic approach. *NASA STI/Recon Technical Report A*, 89:47350, 1988.

[125] PARAMETRIC TECHNOLOGY CORPORATION (PTC) : PRO/ENGINEER. http://www.proengineer.com/.

[126] Flore PERRIN-BRUNEAU : *Proposition d'une démarche d'intégration de nouvelles méthodes en conception : éléments pour la définition du rôle de l'intégrateur'méthodes'.* Thèse de doctorat, École Nationale Supérieure des Arts et Métiers ParisTech, 2005.

[127] Claude PETITDEMANGE : La maîtrise de la valeur. *AFNOR*, 1985.

[128] Flavien PICON : *Interaction haptique pour la conception de formes en CAO immersive.* Thèse de doctorat, Université Paris-Sud (XI), 2010.

[129] Flavien PICON, Mehdi AMMI et Patrick BOURDOT : Case study of haptic methods for selection on cad models. *In Virtual Reality Conference, 2008. VR'08. IEEE*, pages 209–212. IEEE, 2008.

[130] Flavien PICON, Mehdi AMMI et Patrick BOURDOT : Force model for cad selection. *In Proceedings of the 2008 ACM symposium on Virtual reality software and technology*, pages 283–284. ACM, 2008.

[131] Flavien PICON, Mehdi AMMI et Patrick BOURDOT : Haptically-aided extrusion for object edition in cad. *In Haptics : Perception, Devices and Scenarios*, pages 736–741. Springer, 2008.

[132] Jean-Louis POMIAN, Thierry PRADÈRE et Irène GAILLARD : Ingénierie et ergonomie. *Cépaduès, Toulouse*, 1997.

[133] PTC : http://www.ptc.com/product/division.

[134] Andreas PUSCH, Olivier MARTIN et Sabine COQUILLART : Hemp-hand-displacement-based pseudo-haptics : A study of a force field application and a behavioural analysis. *International journal of human-computer studies*, 67(3):256–268, 2009.

[135] Danielle QUARANTE : *Eléments de design industriel*. Éditions Maloine, Paris, France, 1994.

[136] Srinivas RAGHOTHAMA et Vadim SHAPIRO : Consistent updates in dual representation systems. *Computer-Aided Design*, 32(8):463–477, 2000.

[137] Alberto RAPOSO, Eduardo TL CORSEUIL, Gustavo N WAGNER, Ismael HF dos SANTOS et Marcelo GATTASS : Towards the use of cad models in vr applications. *Proceedings of the 2006 ACM international conference on Virtual reality continuum and its applications*, pages 67–74, 2006.

[138] Chris RAYMAEKERS, Gert VANSICHEM et Frank VAN REETH : Improving sketching by utilizing haptic feedback. *In AAAI Spring Symposium on Sketch Understanding*, pages 113–117, 2002.

[139] Aristides AG REQUICHA et Herbert B VOELCKER : Constructive solid geometry. *Technical Memo 25, Production Automation Project*, 1977.

[140] Aristides G REQUICHA : Representations for rigid solids : Theory, methods, and systems. *ACM Computing Surveys (CSUR)*, 12(4):437–464, 1980.

[141] Emanuel SACHS, Andrew ROBERTS et David STOOPS : 3draww : A tool for designing 3d shapes. *IEEE Comput. Graph. Appl.*, 11(6):18–26, novembre 1991.

[142] Dominique SADOUL : La maquette numérique : Un exemple de développement industriel. *Mécanique & industries*, 1(4):373–382, 2000.

[143] Barb SCHMITZ : Analyse des tendances de la cao à l'échelle mondiale, 2011. http://fr.creo.ptc.com/2011/12/06/breaking-down-global-cad-trends/.

[144] Bahar SENER, Owain PEDGLEY, Paul WORMALD et Ian CAMPBELL : Incorporating the freeform haptic modelling system into new product development. *In Proceedings of EuroHaptics*, 2003.

[145] Abhishek SETH, Judy M VANCE et James H OLIVER : Virtual reality for assembly methods prototyping : a review. *Virtual reality*, 15(1):5–20, 2011.

[146] J. J. SHAH : Assessment of features technology. *Computer-Aided Design*, 23(5):331–343, 1991.

[147] Jami J SHAH : Feature transformations between application-specific feature spaces. *Computer-Aided Engineering Journal*, 5(6):247–255, 1988.

[148] Jami J. SHAH et Martti MANTYLA : *Parametric and Feature Based CAD/Cam : Concepts, Techniques, and Applications*. John Wiley & Sons, Inc., New York, NY, USA, 1st édition, 1995.

[149] Mel SLATER : Measuring presence : A response to the witmer and singer presence questionnaire. *Presence : Teleoperators and Virtual Environments*, 8(5):560–565, 1999.

[150] Mel SLATER et Sylvia WILBUR : A framework for immersive virtual environments (five) : Speculations on the role of presence in virtual environments. *Presence : Teleoperators and virtual environments*, 6:603–616, 1997.

[151] Gunnar SOHLENIUS : Concurrent engineering. *CIRP Annals-Manufacturing Technology*, 41(2):645–655, 1992.

[152] SPATIAL CORPORATION : 3D ACIS modeler. http://www.spatial.com/products/3d-acis-modeling.

[153] Rainer STARK, Johann Habakuk ISRAEL et Thomas WÖHLER : Towards hybrid modelling environments - merging desktop-cad and virtual reality-technologies. *CIRP Annals-Manufacturing Technology*, 59(1):179–182, 2010.

[154] André STORK et Martin MAIDHOF : Efficient and precise solid modelling using a 3d input device. *In Proceedings of the fourth ACM symposium on Solid modeling and applications*, pages 181–194. ACM, 1997.

[155] Ivan E SUTHERLAND : A man-machine graphical communication system. *In Proceedings of AFIPS Spring Joint Computer Conference*, pages 329–346, 1963.

[156] Loïc TCHING : *Traitement générique des interactions haptiques pour l'assemblage d'objets issus de CAO*. Thèse de doctorat, INSA de Rennes, 2010.

[157] OpenCascade TECHNOLOGY : OpenCascade. http://www.opencascade.org/occt/.

[158] TECHVIZ : TechViz XL. http://www.techviz.net/products/techviz-xl-driver/.

[159] Mădălina Ioana TOMA, Florin GÎRBACIA et Csaba ANTONYA : A comparative evaluation of human interaction for design and assembly of 3d cad models in desktop and immersive environments. *International Journal on Interactive Design and Manufacturing (IJIDeM)*, 6(3):179–193, 2012.

[160] Sanjeev N TRIKA, Prashant BANERJEE et Rangasami L KASHYAP : Virtual reality interfaces for feature-based computer-aided design systems. *Computer-Aided Design*, 29(8):565–574, 1997.

[161] David G ULLMAN : *The mechanical design process*, volume 2. McGraw-Hill New York, 1992.

[162] Etienne van WYK et Ruth de VILLIERS : Virtual reality training applications for the mining industry. *In Proceedings of the 6th international conference on computer graphics, virtual reality, visualisation and interaction in Africa*, pages 53–63. ACM, 2009.

[163] G Gary WANG : Definition and review of virtual prototyping. *Journal of Computing and Information Science in engineering*, 2(3):232–236, 2002.

[164] Qing-Hui WANG, Jing-Rong LI, Bao-Li WU et Xiao-Ming ZHANG : Live parametric design modifications in cad-linked virtual environment. *The International Journal of Advanced Manufacturing Technology*, 50(9-12):859–869, 2010.

[165] Yan WANG et Bartholomew O. NNAJI : Geometry-based semantic id for persistent and interoperable reference in feature-based parametric modeling. *Computer-Aided Design*, 37(10):1081–1093, septembre 2005.

[166] WEB 3D CONSORTIUM : Virtual Reality Markup Langage. http://www.web3d.org/x3d/vrml/.

[167] Dieter WEIDLICH, László CSER, Thomas POLZIN, D CRISTIANO et Holger ZICKNER : Virtual reality approaches for immersive design. *International Journal on Interactive Design and Manufacturing (IJIDeM)*, 3(2):103–108, 2009.

[168] Junjun WU, Tianbing ZHANG, Xinfang ZHANG et Ji ZHOU : A face based mechanism for naming, recording and retrieving topological entities. *Computer-Aided Design*, 33(10): 687–698, 2001.

[169] Bernard YANNOU et Jean-François PETIOT : Needs, perceptions, functions and products : highlight on promising design methods linking them. *In IDMME2002 : 4th International Conference on Integrated Design and Manufacturing in Mechanical Engineering, Clermont-Ferrand*, 2002.

[170] Jilin YE, R Ian CAMPBELL, Tom PAGE et Kevin S BADNI : An investigation into the implementation of virtual reality technologies in support of conceptual design. *Design Studies*, 27(1):77–97, 2006.

[171] Gabriel ZACHMANN et Alexander RETTIG : Natural and robust interaction in virtual assembly simulation. *In Eighth ISPE International Conference on Concurrent Engineering : Research and Applications (ISPE/CE2001)*, volume 1, pages 425–434, 2001.

[172] Yongmin ZHONG, Weiyin MA et Bijan SHIRINZADEH : A methodology for solid modelling in a virtual reality environment. *Robotics and Computer-Integrated Manufacturing*, 21(6):528–549, 2005.

[173] Farbod ZORRIASSATINE, Catherine WYKES, Robert PARKIN et Nabil GINDY : A survey of virtual prototyping techniques for mechanical product development. *Proceedings of the institution of mechanical engineers, Part B : Journal of engineering manufacture*, 217(4):513–530, 2003.