

Filière Systèmes industriels

Orientation Infotronics

Travail de bachelor

Diplôme 2021

Gaëtan Fumeaux

*Génération automatique de support de
frittage pour impression SG-3DP*

- Professeur
Medard Rieder
- Expert
Kevin Cardoso
- Date de la remise du rapport
20.08.2021

Ce rapport est l'original remis par l'étudiant.
Il n'a pas été corrigé et peut donc contenir des inexactitudes ou des erreurs.

| | | |
|--|---|---|
| Filière / Studiengang SYND | Année académique / <i>Studienjahr</i> 2020/21 | No TD / Nr. DA IT/2021/52 |
| Mandant / <i>Auftraggeber</i> <input checked="" type="checkbox"/> HES—SO Valais <input type="checkbox"/> Industrie <input type="checkbox"/> Etablissement partenaire <i>Partnerinstitution</i> | Etudiant / <i>Student</i> Gaëtan Fumeaux | Lieu d'exécution / <i>Ausführungsart</i> <input checked="" type="checkbox"/> HES—SO Valais <input type="checkbox"/> Industrie <input type="checkbox"/> Etablissement partenaire <i>Partnerinstitution</i> |
| Travail confidentiel / <i>vertrauliche Arbeit</i> <input type="checkbox"/> oui / ja ¹ <input checked="" type="checkbox"/> non / nein | Expert / <i>Experte (données complètes)</i> Kevin Cardoso Chemin des Palettes 9 1967 Bramois +41 79 847 84 98, kevin-cardoso@bluewin.ch | |

Titre / Titel**Génération automatique de support de frittage pour impression SG-3DP****Description / Beschreibung**

Le groupe technologie des poudres de la HES-SO Valais est spécialisé dans la technique d'impression 3D "Solvent on Granules"

(https://www.researchgate.net/publication/349553916_A_Comparative_Study_of_Cemented_Carbide_Parts_Produced_by_Solvent_on_Granules_3D-Printing_SG-3DP_Versus_Press_and_Sinter).

Une imprimante de ce type a été développée entièrement dans le cadre de l'école.

Après l'impression proprement dite, les pièces sont consolidées par frittage. Lors de cette opération, les pièces peuvent s'affaisser si elles ne sont pas soutenues par un support adéquat. Le projet proposé a pour but de réaliser un logiciel (ou partie de logiciel) effectuant une génération automatique d'un support adapté à la forme de la pièce imprimée.

Objectifs / Ziele

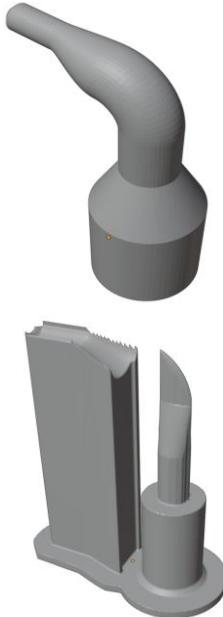
- Étudier et comprendre l'imprimante 3D, son fonctionnement et surtout son contrôle par un fichier "3D". Comprendre le format et la structure de ce fichier.
- Proposer / choisir un algorithme capable de déterminer la forme du support en partant de la forme de l'objet à imprimer en 3D.
- Concevoir et implémenter un logiciel qui est capable de créer un fichier "3D" qui représente le support et qui peut être utilisé pour piloter l'imprimante 3D.
- Tester et optimiser ce logiciel avec des formes simples.
- Établir une documentation technique et un rapport final du travail réalisé

| | |
|---|--|
| Signature ou visa / <i>Unterschrift oder Visum</i> Responsable de l'orientation / filière <i>Leiter der Vertiefungsrichtung / Studiengang:</i>  | Délais / <i>Termine</i> Attribution du thème / <i>Ausgabe des Auftrags:</i> 10.05.2021 Présentation intermédiaire / <i>Zwischenpräsentation</i> 07 – 08.06.2021 Remise du rapport / <i>Abgabe des Schlussberichts:</i> 20.08.2021, 12:00 Exposition / <i>Ausstellung der Diplomarbeiten:</i> 25 – 27.08.2021 (si autorisé / falls genehmigt) Défense orale / <i>Mündliche Verfechtung:</i> 30.08 – 09.09.2021 |
| ¹ Etudiant / <i>Student</i> :  | |

¹ Par sa signature, l'étudiant-e s'engage à respecter strictement la directive DI.1.2.02.07 liée au travail de diplôme.
Durch seine Unterschrift verpflichtet sich der/die Student/in, sich an die Richtlinie DI.1.2.02.07 der Diplomarbeit zu halten.

Rapport reçu le / *Schlussbericht erhalten am* Visa du secrétariat / *Visum des Sekretariats*

Rapport reçu le / *Schlussbericht erhalten am* Visa du secrétariat / *Visum des Sekretariats*



Travail de diplôme | édition 2021 |

Filière
Systèmes industriels

Domaine d'application
Infotronics

Professeur responsable
Medard Rieder
Medard.rieder@hevs.ch

Génération automatique de support de frittage

 Diplômant/e Gaëtan Fumeaux

Objectif du projet

L'objectif de ce projet est de réaliser un logiciel effectuant une génération automatique d'un support adapté à la forme de la pièce imprimée qui peut être utilisé pour la soutenir pendant le frittage afin qu'elle ne s'affaisse pas.

Méthodes | Expériences | Résultats

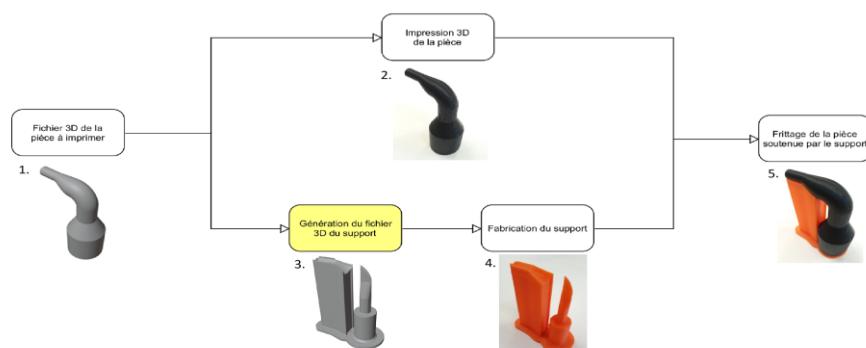
Pour ce travail, il a été décidé que le logiciel développé puisse importer un fichier STL, générer les supports de l'objet importé grâce à des options et exporter le fichier STL de ces supports.

Les programmes 3D testés pour générer les supports sont Meshmixer et Blender. Après des résultats peu concluants avec Meshmixer, le choix du programme 3D s'est porté sur Blender avec lequel il a été possible de créer un script pour générer les supports.

Grâce à l'API de Blender, il a été possible de développer un panel de contrôle avec des boutons et des sliders pour la génération des supports qui contient plusieurs fonctionnalités tels que l'import/export, la rotation, la génération, le redimensionnement, etc.

Après avoir vérifié le bon fonctionnement des différentes options implémentées et du panel de contrôle, des tests ont été réalisés avec des prototypes imprimés en plastique pour voir si les supports générés sont réalisables.

Les résultats de ces tests ont montré que le logiciel développé est capable de générer des supports qui soutiennent les pièces et qu'il est donc maintenant possible de tester ces supports avec des pièces métalliques.



Etapes de la fabrication d'une pièce avec la méthode d'impression « Solvent on Granules » et utilisant des supports pour la soutenir lors du frittage.

Table des matières

| | | |
|--------|---|----|
| 1. | Introduction | 1 |
| 1.1. | Contexte..... | 1 |
| 1.2. | Objectifs..... | 1 |
| 1.3. | Structure..... | 2 |
| 2. | Cahier des charges..... | 2 |
| 3. | Analyse | 3 |
| 3.1. | Solvent on Granule 3D Printing (SG-3DP) | 3 |
| 3.2. | Fichier 3D | 4 |
| 3.3. | Materialise | 6 |
| 3.4. | Génération des supports..... | 8 |
| 4. | Conception | 9 |
| 5. | Meshmixer..... | 10 |
| 5.1. | Tests..... | 10 |
| 5.2. | Résultats..... | 11 |
| 6. | Blender | 12 |
| 6.1. | Tests..... | 12 |
| 6.2. | Résultats..... | 13 |
| 7. | Implémentation..... | 14 |
| 7.1. | API de Blender | 14 |
| 7.2. | Logiciel | 14 |
| 7.2.1. | Fichiers | 14 |
| 7.2.2. | Menus | 15 |
| 7.2.3. | Diagramme de classe..... | 15 |
| 7.3. | Import/Export | 17 |
| 7.3.1. | Unité et volume | 17 |
| 7.3.2. | Import/Export | 17 |
| 7.4. | Rotations et offset..... | 18 |
| 7.5. | Génération | 20 |
| 7.5.1. | Sélection des faces | 20 |
| 7.5.2. | Génération des supports | 27 |
| 7.5.3. | Génération d'un moule | 29 |
| 7.6. | Options de génération | 32 |
| 7.6.1. | Aire minimale..... | 32 |
| 7.6.2. | Génération du fond..... | 34 |
| 7.6.3. | Génération d'un socle..... | 36 |

| | | |
|---------|--------------------------------------|----|
| 7.7. | Sélection et redimensionnement | 38 |
| 7.7.1. | Sélection | 38 |
| 7.7.2. | Redimensionnement | 41 |
| 7.7.3. | Autres options | 43 |
| 7.8. | Modification du maillage..... | 44 |
| 7.8.1. | Modificateur voxel | 44 |
| 7.8.2. | Réduction de faces | 45 |
| 7.8.3. | Modificateur blocs..... | 45 |
| 7.9. | Mesure | 45 |
| 7.10. | Fonctions non-implémentées | 46 |
| 7.10.1. | Sélection d'arêtes..... | 46 |
| 7.10.2. | Génération de pointes | 47 |
| 8. | Validation | 48 |
| 8.1. | Transfert de connaissances | 48 |
| 8.2. | Procédure de tests | 50 |
| 8.3. | Résultats..... | 55 |
| 9. | Conclusion..... | 57 |
| 10. | Date et signature | 57 |
| 11. | Bibliographie | 58 |
| 12. | Annexes | 58 |
| 13. | Liste des figures et tableaux | 58 |

1. Introduction

1.1. Contexte

Depuis quelques années, l'industrie des poudres est en pleine expansion. Avec la possibilité de faire des pièces bien plus complexes qu'avec l'usinage traditionnel, de créer des pièces avec des propriétés mécaniques spécifiques et de faire du prototypage rapide, la technologie des poudres séduit de plus en plus de clients. L'augmentation de la popularité de cette technologie a permis de faire des avancées technologiques dans plusieurs domaines et notamment dans celui de l'impression 3D métallique.

Actuellement, l'impression 3D métallique permet déjà d'imprimer des pièces de bonne qualité avec des matières différentes. Cependant, étant donné que l'impression 3D métallique est une technologie plutôt récente, il reste de nombreux outils et poudres à développer pour perfectionner cette technique et ainsi améliorer les propriétés et la précision des pièces imprimées.

1.2. Objectifs

Le groupe technologie des poudres de la HES-SO Valais est spécialisé dans la technique d'impression 3D « Solvent on Granules ». Après l'impression proprement dite, les pièces sont consolidées par frittage. Le frittage consiste à chauffer les grains de la pièce pour qu'ils se soudent entre eux. Lors de cette opération, les pièces peuvent s'affaisser si elles ne sont pas soutenues par un support adéquat.

En effet, la méthode d'impression 3D « Solvent on Granules » ne nécessite pas de supports. Cependant, ceux-ci deviennent nécessaires lors du frittage. Actuellement, les pièces sont soit frittées sans support, soit frittées en étant soutenues par de simples plaques de céramique frittées, puis usinées.

Le projet proposé a pour but de réaliser un logiciel (ou partie de logiciel) effectuant une génération automatique d'un support adapté à la forme de la pièce imprimée, qui peut être utilisé pour soutenir cette dernière pendant l'opération de frittage, afin qu'elle ne s'affaisse pas.

Les étapes pour fabriquer une pièce en 3D avec le logiciel créé sont les suivantes (*Figure 1*) :

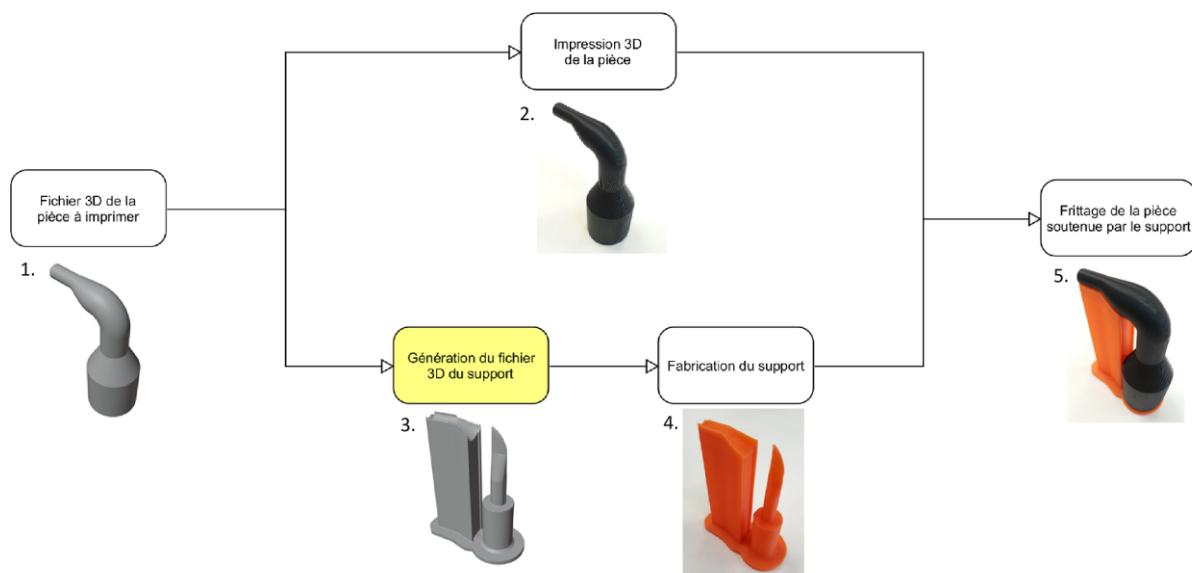


Figure 1 : Etapes de fabrication d'une pièce 3D

1. La première étape permettant d'imprimer une pièce consiste à créer un fichier 3D de cette dernière.
2. Ce fichier 3D est ensuite découpé en tranches 2D pour être converti en instructions pour l'imprimante. La pièce peut ensuite être imprimée en 3D avec la méthode « Solvent on Granules ».
3. Parallèlement, le fichier 3D est également envoyé à un logiciel qui permet de générer un fichier 3D des supports pour l'objet du fichier 3D. Cette étape est réalisée grâce au logiciel créé pour cette thèse.
4. Les supports générés doivent ensuite être fabriqués.
5. Finalement, la pièce est mise dans le four de frittage avec son support créé pour éviter qu'elle ne s'affaisse.

L'utilité d'avoir un logiciel qui peut automatiquement créer des supports est que cela permet d'avoir un outil simple et rapide, afin d'éviter un éventuel affaissement de la pièce. Le but principal de cette thèse est de développer un outil, ainsi que la méthode pour y parvenir. Cela donne la possibilité de créer des supports de frittage, tout en comprenant la structure des fichiers 3D.

1.3. Structure

Ce rapport présente dans un premier temps le cahier des charges de ce projet. Puis, il explique les différents éléments à disposition nécessaires à la compréhension de cette thèse tels que la méthode d'impressions « Solvent on Granules », le format des fichiers 3D, *Materialise* et la génération des supports. Ce rapport aborde ensuite la conception du logiciel, ainsi que les programmes testés pour y parvenir. Puis, une partie sur l'implémentation explique les différents choix des fonctionnalités implémentées. Finalement, la partie validation indique les tests effectués, afin de vérifier le logiciel créé et, pour finir, une conclusion termine ce rapport.

2. Cahier des charges

Les objectifs du cahier des charges, à réaliser pour ce travail de bachelor, sont les suivants :

- Etudier et comprendre l'imprimante 3D, son fonctionnement et surtout son contrôle par un fichier « 3D ».
Comprendre le format et la structure de ce fichier.
- Proposer / choisir un algorithme capable de déterminer la forme du support en partant de la forme de l'objet à imprimer en 3D.
- Concevoir et implémenter un logiciel qui est capable de créer un fichier « 3D » qui représente le support et qui peut être utilisé pour piloter l'imprimante 3D.
- Tester et optimiser ce logiciel avec des formes simples.
- Etablir une documentation technique et un rapport final du travail réalisé.

3. Analyse

3.1. Solvent on Granule 3D Printing (SG-3DP)

Le groupe technologie des poudres de la HES-SO a développé un procédé génératif appelé « Solvent on Granules » [1][2]. Ce procédé consiste à déposer sélectivement un agent liant pour joindre des particules de poudre ou des granulés (agglomérats poudre-liant) et ainsi construire un objet couche par couche. Le développement de cette méthode s'inspire de la technologie « three-dimensional printing ».

Dans la technologie classique, une colle ou un liant est imprimé sur des couches de poudre. Ce liant est giclé par une tête d'impression dans une aire 2D pour consolider une couche. La plateforme descend ensuite d'une couche d'épaisseur. Cette opération est répétée jusqu'à l'obtention de la forme voulue. Finalement, la poudre en excès est retirée et le « corps vert » obtenu est soumis à un déliantage, suivi d'un frittage.

L'innovation du procédé SG-3DP consiste à déposer un solvant sur une poudre composite métal-polymère. Le solvant ramollit le liant et les granulés restent collés ensemble après l'évaporation. Le « corps vert » généré couche par couche est ensuite consolidé par déliantage et frittage.

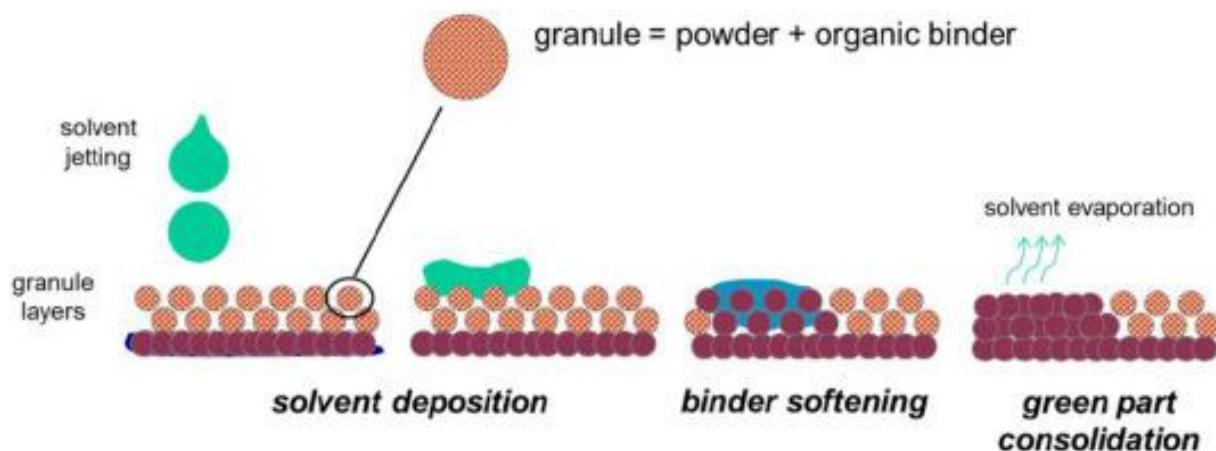


Figure 2 : Principe de l'impression 3D « Solvent on Granules » (SG-3DP)

Source : https://www.researchgate.net/publication/349553916_A_Comparative_Study_of_Cemented_Carbide_Parts_Produced_by_Solvent_on_Granules_3D-Printing_SG-3DP_Versus_Press_and_Sinter

Cette méthode donne plus de flexibilité pour le choix des poudres et des liants et permet d'éviter le problème de blocage de buses d'impression par des fluides visqueux.

Les pièces réalisées par cette méthode ont des propriétés mécaniques et une densité proche de celles des pièces pressées et frittées obtenues par les procédés de fabrication conventionnels. Le procédé génératif « Solvent on Granules » ouvre de nombreuses possibilités pour le développement de pièces avec des géométries complexes qui ne sont pas réalisables avec des méthodes de fabrication conventionnelles.

3.2. Fichier 3D

Pour imprimer une pièce en 3D, il faut avoir un modèle 3D de cette pièce contenu dans un fichier 3D. Le fichier 3D est ensuite transmis au slicer pour être imprimé. Un slicer est un logiciel qui permet de convertir le modèle du fichier 3D en une multitude de plans 2D. Ces informations sont ensuite transmises et traduites, si besoin, en instructions pour l'imprimante, afin de pouvoir imprimer l'objet.

Pour ce projet, il a été demandé de travailler avec des fichiers 3D de type STL. STL est un format de fichier 3D créé par *3D Systems* et possédant plusieurs acronymes dont le principal est « *Stereolithography* ». Le but de ce format de fichier est d'encoder la géométrie de surface externe du modèle 3D à l'aide de la tessellation. La tessellation consiste à pavier une surface à l'aide d'une forme géométrique, qui est le triangle dans ce cas. STL n'encode que les informations des triangles et il n'y a donc pas d'informations sur l'échelle ou sur la couleur de l'objet.

Chaque triangle est défini par son vecteur unitaire normal (n) qui pointe vers l'extérieur de l'objet et par les 3 sommets du triangle dans un système de coordonnées cartésiennes (x, y, z).

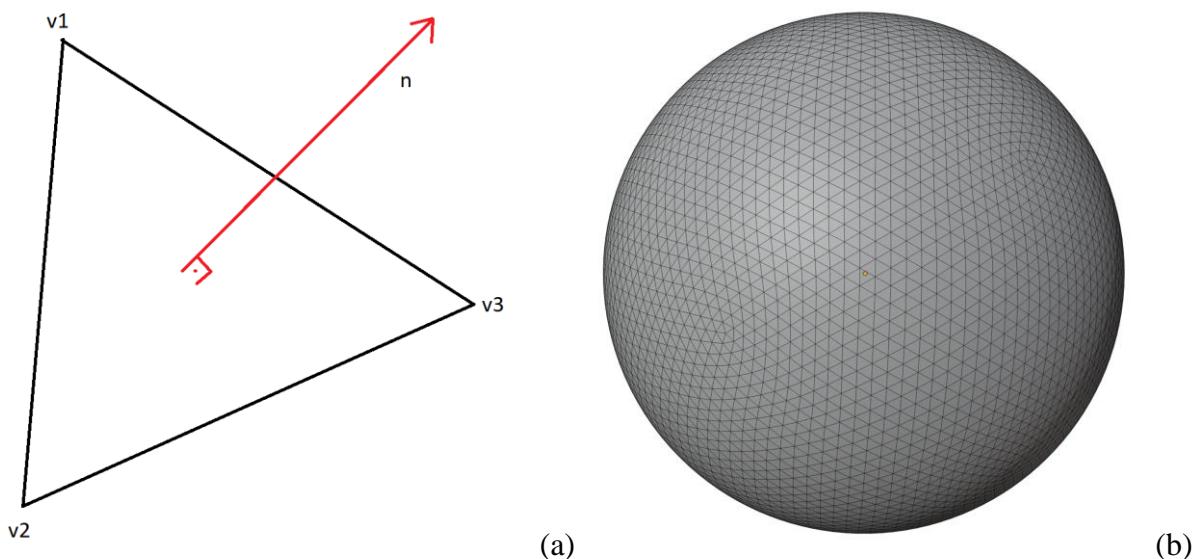


Figure 3 : Paramètres d'un triangle STL (a) et tessellation d'une sphère (b)

Il existe deux méthodes pour stocker les informations en STL : le codage ASCII et le codage binaire.

Le codage ASCII se présente sous cette forme :

```

1. solid name
2.
3. facet normal ni nj nk
4.   outer loop
5.     vertex v1x v1y v1z
6.     vertex v2x v2y v2z
7.     vertex v3x v3y v3z
8.   endloop
9. endfacet
10.
11. // Other triangles
12.
13. endsolid name

```

Où chaque n et v sont des nombres à virgules flottantes.

Les fichiers ASCII sont très faciles à créer, mais peuvent rapidement devenir volumineux pour des objets complexes. C'est pourquoi les fichiers binaires, qui sont plus compacts, sont davantage utilisés. De plus, le format d'encodage binaire est plus facile à lire. Le codage binaire se présente sous la forme suivante :

```
1.  UINT8[80]      // Header           80 bytes
2.  UINT32         // Number of triangles   4 bytes
3.
4.  foreach triangle          50 bytes:
5.    REAL32[3]    // Normal vector     12 bytes
6.    REAL32[3]    // Vertex 1           12 bytes
7.    REAL32[3]    // Vertex 2           12 bytes
8.    REAL32[3]    // Vertex 3           12 bytes
9.  UINT16         // Attribute byte count  2 bytes
10. end
```

La compréhension de la structure des fichiers STL est intéressante et nécessaire au projet. En effet, ces informations sont utiles, car le principe pour développer un algorithme qui génère des supports se base sur les faces de l'objet.

3.3. Materialise

Pour avoir une base de comparaison avec le logiciel voulu, il est intéressant de connaître les fonctionnalités d'un logiciel de création de supports existant. Le groupe technologie des poudres de la HES-SO Valais utilise le logiciel *Magics* de *Materialise*, afin de créer les fichiers pour les impressions 3D de technologie « Selective Laser Melting » (SLM). *Magics* est un programme qui permet de préparer les fichiers nécessaires à l'impression 3D. Il possède aussi plusieurs modules payants comme celui qui permet de générer des supports.

Contrairement à la méthode « Solvent on Granules », la méthode « Selective Laser Melting » utilise des lasers pour venir fusionner la poudre. Chaque fois que le ou les lasers finissent une couche, une nouvelle couche de poudre est appliquée. Ces étapes se répètent jusqu'à la fin de la fabrication de l'objet. Puis, la poudre qui n'a pas été fusionnée est retirée. Les supports doivent être retirés par usinage si la pièce en possède (*Figure 4*).



Figure 4 : Etapes du procédé d'impression 3D SLM - crédit Materialise

Source : <https://www.a3dm-magazine.fr/news/fabrication-additive-metallique/procede-de-fabrication-additive-slm>

Une des différences entre les méthodes « Solvent on Granules » et « Selective Laser Melting » est que « Solvent on Granules » n'a pas besoin de supports pendant l'impression. Elle en a seulement besoin pour l'opération de frittage. Donc, les supports sont séparés de la pièce.

A contrario, la méthode « Selective Laser Melting » nécessite des supports pour l'impression. Elle en a besoin pour soutenir la pièce lorsque les lasers fusionnent la poudre et ces supports peuvent être nécessaires pour dissiper la chaleur de la fusion. Les supports font donc partie de la pièce. Cela peut être un désavantage, car il faut ensuite les retirer de la pièce par usinage et ces opérations coûtent parfois plus chères que la fabrication en elle-même.

Cette différence dans l'utilisation des supports pour ces deux méthodes explique pourquoi *Magics* n'est pas utilisé pour générer les supports de la méthode « Solvent on Granules », car ceux-ci ne seraient pas forcément adaptés. De plus, *Magics* est un programme très complet qui contient beaucoup d'options. Ce travail a pour but de créer un logiciel qui permet de générer rapidement et simplement des supports.

La figure 5 montre les différentes étapes permettant d'imprimer le fichier 3D à l'aide de *Magics*.

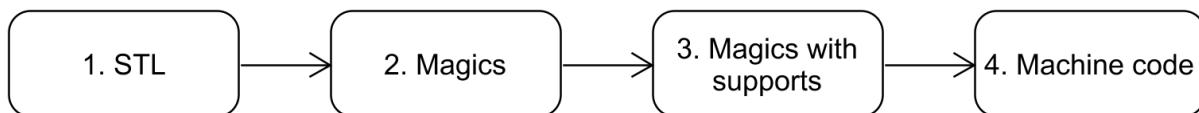


Figure 5 : Etapes de préparation du fichier 3D avec Magics

1. Tout d'abord, le fichier STL contenant la pièce en 3D doit être créé.
2. Puis, Magics convertit le fichier STL en fichier magics. Pendant cette étape, les erreurs du fichier STL sont fixées.
3. La ou les pièces sont ensuite positionnées sur le plateau et des supports sont créés pour les soutenir.
4. Finalement, l'ensemble est coupé en couches (« slicing ») et converti en langage machine pour l'imprimante voulue. Dans ce cas, il s'agit d'une imprimante de type SLM.

Comme annoncé dans les objectifs, le logiciel voulu n'a pas pour vocation de créer les fichiers pour imprimer une pièce, mais il se concentre seulement sur la génération du fichier 3D des supports. Par conséquent, la partie 3 des étapes de préparation du fichier 3D avec *Magics* est intéressante, car c'est la partie qui génère des supports grâce au module des supports de *Magics*.

Ce module possède beaucoup d'options dont les plus intéressantes sont décrites ci-après.

Magics peut générer plusieurs types de supports tels que ceux de projection, pointes, lignes, cônes, blocs, manuels. Cette diversité dans les supports offre la possibilité de tester plusieurs supports pour déterminer lequel est le meilleur.

De plus, il est possible d'avoir une prévisualisation des supports et de connaître leur volume. Cela est utile, car l'orientation de la pièce peut être modifiée, ce qui met à jour la prévisualisation et le volume. Il est donc possible de tester différentes orientations tout en cherchant facilement les meilleurs supports grâce à la prévisualisation et au volume.

Le module des supports sépare aussi les supports générés en plusieurs groupes. Il est ensuite possible de sélectionner chacun de ces groupes et de les modifier manuellement pour améliorer les supports.

Magics de *Materialise* n'est pas étudié dans le but de copier les méthodes de ce programme dans le logiciel créé, mais d'avoir une idée sur les différentes fonctionnalités que possède un programme spécialisé dans l'industrie. C'est pour cela qu'il a été décidé de développer d'abord l'algorithme avant de découvrir les fonctionnalités de *Magics*.

3.4. Génération des supports

Les supports que le logiciel doit générer sont différents des supports habituellement utilisés pour les impressions 3D. En effet, les supports classiques pour les impressions 3D sont générés dans le but d'être imprimés en même temps que la pièce à soutenir. Tandis que pour ce travail, et comme énoncé dans les objectifs, le logiciel doit créer des supports qui servent à soutenir la pièce lors de l'opération de frittage et qui sont fabriqués indépendamment de cette dernière.

Les spécificités des supports voulus et du frittage vont imposer plusieurs contraintes pour la génération de ceux-ci. Une contrainte à prendre en compte est le facteur de rétrécissement de la pièce lors du frittage (*Figure 6*). En effet, les supports générés ne doivent pas imposer des contraintes à la pièce lors du frittage pour éviter des déformations ou la destruction de cette dernière. Pour ne pas avoir de contraintes, les supports doivent avoir le même facteur de rétrécissement que la pièce et doivent donc être imprimés avec la même matière ou une matière similaire à cette pièce. Etant donné que les supports seront sûrement frittés en même temps que l'objet qu'ils doivent soutenir, il ne faut pas qu'ils se soudent avec l'objet pendant le frittage. Cela peut être évité en utilisant une laque qui va isoler les supports de l'objet. Les supports générés doivent également être assez simples pour diminuer la quantité de matière et donc le coût des poudres utilisées pour leur fabrication.

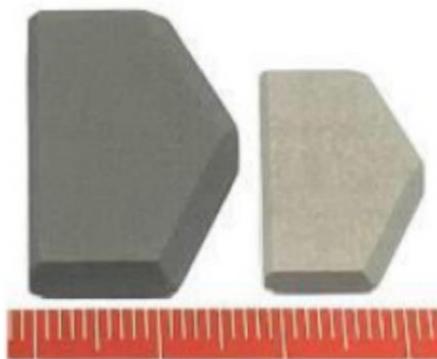


Figure 6 : Comparaison entre le « corps vert » et la pièce frittée

Source : https://www.researchgate.net/publication/349553916_A_Comparative_Study_of_Cemented_Carbide_Parts_Produced_by_Solvent_on_Granules_3D-Printing_SG-3DP_Versus_Press_and_Sinter

Le type et la forme des supports sont une autre contrainte. En effet, le fichier STL de la pièce aura une certaine résolution (quantité de détails).

Pour que les supports ne modifient pas la pièce lors du frittage, ils doivent :

- Soit avoir une quantité de détails semblables ou plus élevés que la pièce si les supports épousent parfaitement les faces de la pièce qui ont besoin d'être soutenues.
- Soit que les supports aient peu de points de contact avec la pièce, mais que ces points soutiennent la pièce aux endroits importants (pointes, lignes, cônes).

La forme des supports que le logiciel doit générer varie aussi fortement selon l'orientation et la forme de la pièce. De plus, les supports doivent être stables et robustes pour ne pas avoir eux-mêmes besoin de supports.

Toutes ces contraintes vont influencer la méthode utilisée pour générer des supports. Toutefois, c'est seulement avec des tests sur l'imprimante 3D métallique que l'importance de ces contraintes sur la génération des supports pourra être établie et que des mesures pourront être prises, afin d'améliorer la méthode de génération.

4. Conception

Après quelques discussions, il a été décidé que l'utilisateur devait être capable, sur le logiciel, d'importer un fichier STL de l'objet ayant besoin de supports, de générer des supports pour cet objet et de les exporter dans un fichier STL. L'utilisateur a également accès à plusieurs options, afin de l'aider à mieux générer les supports (*Figure 7*).

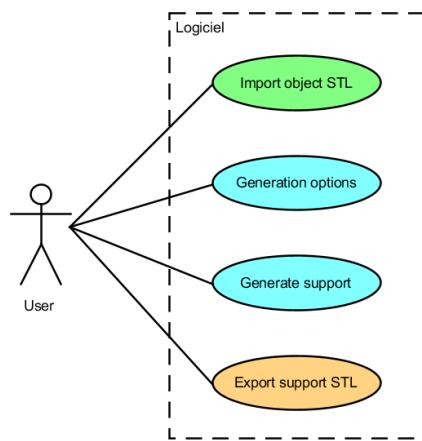


Figure 7 : « Use case » du logiciel

L'avantage d'avoir un logiciel qui importe un fichier STL et en exporte un autre est que le programme n'est pas dépendant de l'imprimante SG-3DP et peut ainsi fonctionner avec différentes imprimantes 3D qui ont besoin de faire un frittage aux pièces imprimées.

Pour créer le logiciel, il faut chercher un programme qui possède des outils de modélisation 3D, afin de générer les supports et une interface de programmation d'applications (API). L'API permet de pouvoir écrire du code, afin d'automatiser des scripts qui vont interagir avec le programme.

Le but est de créer un add-on avec l'API du programme 3D choisi pour que l'utilisateur puisse envoyer des événements (boutons, touches du clavier, souris) au programme, afin de déclencher des scripts qui vont contrôler les outils du programme et permettre de faire les actions demandées par le « use case » (*Figure 8*).

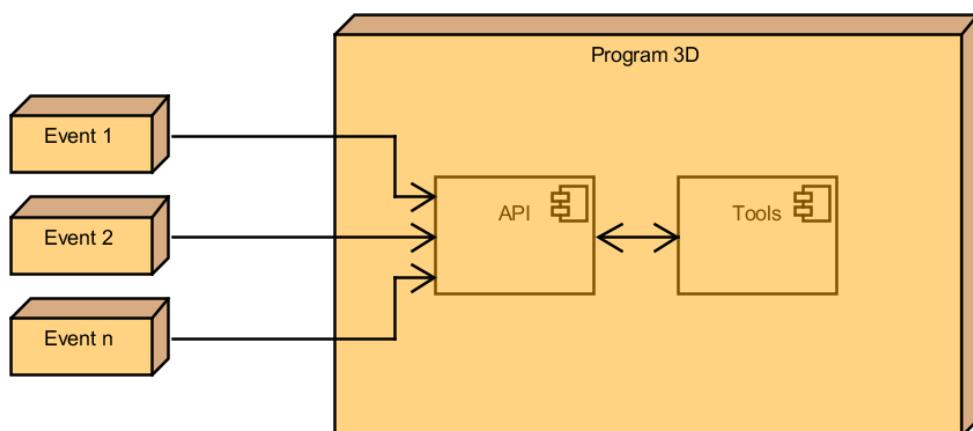


Figure 8 : Diagramme physique du logiciel

5. Meshmixer

Meshmixer est un logiciel gratuit et open source, proposé par Autodesk, qui offre plusieurs fonctionnalités utiles dans la modélisation 3D. *Meshmixer* est principalement utilisé pour la réparation et le nettoyage de maillages, la préparation à l'impression 3D et la modification d'objets 3D. *Meshmixer* est, comme le surnomme Autodesk, une sorte de « couteau suisse » pour les maillages 3D.

Le choix d'étudier *Meshmixer* pour générer des supports est dû au fait qu'il a été proposé pour débuter la génération de supports et qu'il s'agit d'un logiciel gratuit et open source. De plus, il possède une interface de programmation d'applications (API) qui permet d'automatiser des scripts avec des lignes de code en Python et donc de faire une génération de supports automatiques.

5.1. Tests

Afin de générer des supports, il a d'abord fallu vérifier que l'API était capable de réaliser les actions demandées par le « use case » (Figure 7). Puis, il a ensuite fallu rechercher une méthode pouvant générer les supports.

Un premier script a été réalisé en utilisant l'outil « Overhangs » de *Meshmixer* (Figure 9).

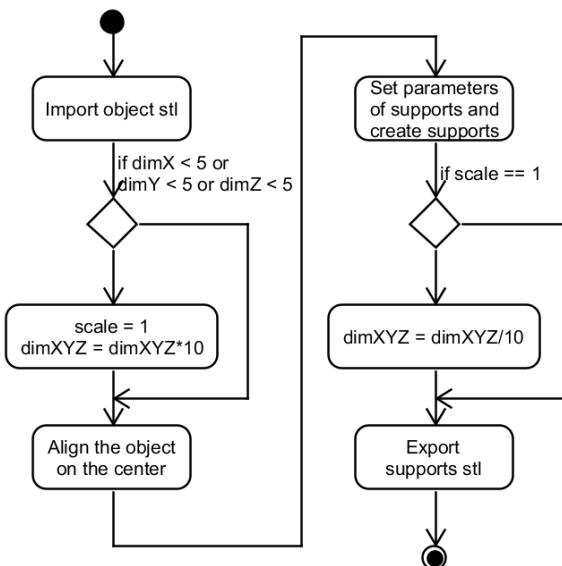


Figure 9 : Diagramme d'état de la génération des supports avec l'outil « Overhangs » de Meshmixer

Ce script importe le fichier STL de l'objet. Puis, il agrandit l'objet si ses dimensions sont trop petites, sinon l'outil « Overhangs » ne fonctionne pas. Ensuite, l'objet est placé au centre, l'outil « Overhangs » est paramétré et les supports sont générés. Finalement, les dimensions des supports sont remises à l'échelle si nécessaire et les supports sont exportés en fichier STL.

L'outil « Overhangs » de *Meshmixer* permet de générer automatiquement des supports. Cependant, cet outil est utilisé pour les impressions 3D plastiques et fonctionne sur le principe de « tree support ». Comme son nom l'indique, le « tree support » imite la structure d'un arbre avec son tronc et ses branches. Les avantages de ce type de supports sont la quantité de matière et la facilité de séparation des supports de la pièce. Par ailleurs, ce qui est un avantage pour l'impression 3D plastique ne l'est pas pour les supports de frittage. En effet, étant donné que les supports de frittage ne sont pas reliés à la pièce, ils doivent être stables et simples, afin de pouvoir être imprimés sans support. Malgré des supports qui pourraient fonctionner pour des formes simples (Figure 10a), il existe plusieurs cas où l'utilisation de « tree support » ne convient pas, peu importe les réglages des supports (Figure 10b).

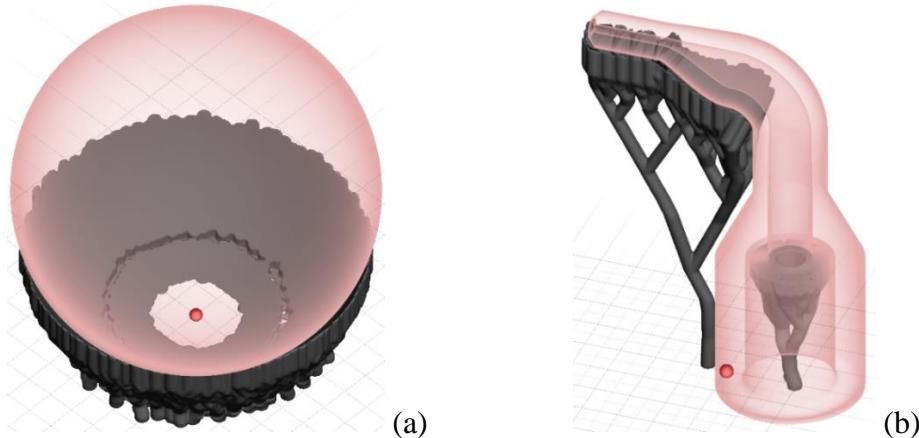


Figure 10 : Résultats de l'outil support de *Meshmixer* avec une forme simple (a) et une forme complexe (b)

Un autre script a été réalisé sur *Meshmixer*, afin d'essayer une méthode différente pour obtenir des supports (*Figure 11a*). Ce script importe un objet, en agrandissant son échelle si ses dimensions sont trop petites et en permettant aussi à l'utilisateur de le faire pivoter. Puis, l'utilisateur choisit s'il veut un support avec une forme de cube ou de cylindre. Il modifie ensuite la position et les dimensions du support pour le placer à l'endroit qu'il veut. Finalement, le script soustrait l'objet du support qui est ensuite remis à l'échelle si nécessaire et est exporté en fichier STL.

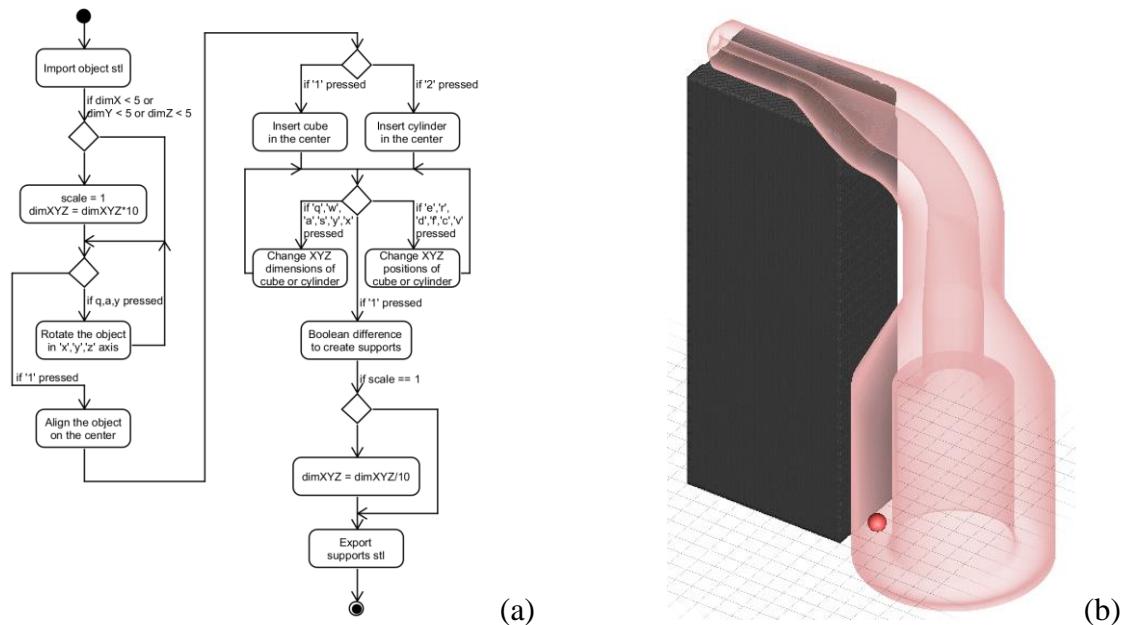


Figure 11 : Diagramme d'état de la génération des supports à l'aide d'une forme simple (a) et son résultat (b)

Ce script permet de montrer que l'on peut faire des supports très simples pour des pièces compliquées (*Figure 11b*). Néanmoins, puisque ce script n'est pas automatique et est très peu ergonomique et flexible à cause des options assez limitées de *Meshmixer*, il n'a pas été développé plus loin.

5.2. Résultats

Les scripts réalisés sur *Meshmixer* ont montré que les fonctionnalités de l'API de ce logiciel n'étaient pas les plus adaptées à l'utilisation voulue. De plus, *Meshmixer* a presque été abandonné par Autodesk. La dernière mise à jour remonte au 17 avril 2018 et son API utilise Python 2.7. Pour ces raisons, il a été décidé de rechercher un autre programme permettant de générer les supports de frittage.

6. Blender

Blender est un logiciel gratuit et open source développé par la *Fondation Blender*. Il prend en charge l'intégralité du pipeline 3D : modélisation, rigging, animation, simulation, rendu, composition et suivi de mouvement, voire même montage vidéo et création de jeux (<https://www.blender.org/about/>, The Software, consulté le 08 août 2021).

Le choix de *Blender* s'explique par le fait que ce logiciel est gratuit, open source et qu'il possède de nombreuses fonctionnalités 3D. De plus, *Blender* dispose d'une interface de programmation d'applications (API) pour automatiser des scripts avec des lignes de code en Python, ce qui permet de générer automatiquement des supports.

6.1. Tests

Blender possède de nombreuses fonctionnalités favorisant la création de supports. Le script doit suivre les actions du « use case » de la figure 7. Sous *Blender*, ces actions peuvent être réalisées en suivant le diagramme d'état suivant (*Figure 12*) :

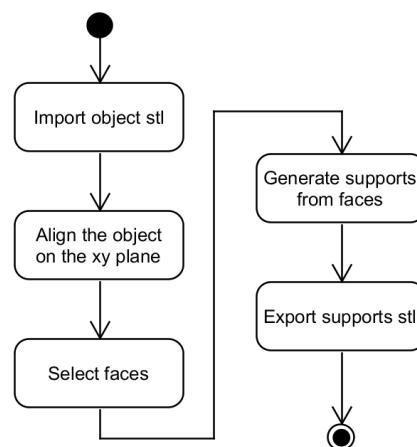


Figure 12 : Diagramme d'état de la génération automatique des supports pour Blender

Le script importe le fichier STL de l'objet. Puis, l'objet est ensuite placé sur le plan xy. Le programme sélectionne ensuite les faces à supporter. Par la suite, le script génère des supports à partir des faces sélectionnées. Cela permet finalement d'obtenir les supports qui peuvent être exportés en fichier STL.

Ce script est capable de générer des supports assez simples qui partent verticalement du plateau et qui peuvent s'adapter à différentes formes (*Figure 13*).

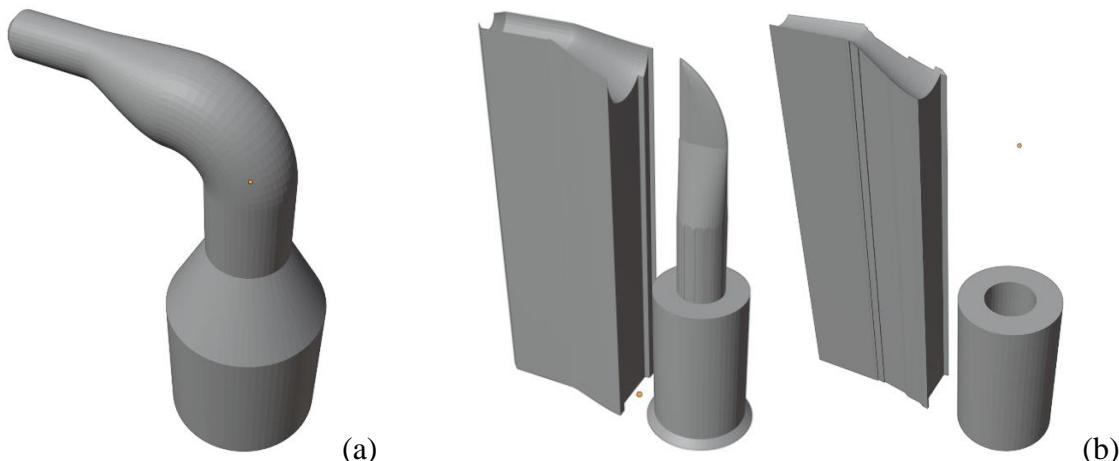


Figure 13 : Objet de base (a) et exemples de supports générés avec Blender (b)

6.2. Résultats

Afin de tester la fonctionnalité des supports, il a été décidé d'imprimer, à l'aide d'une imprimante plastique, un prototype d'une pièce de base avec ses supports. La pièce de base a d'abord été imprimée (*Figure 15a*), puis les supports ont été imprimés avec une *Zortrax M200* (*Figure 14*).

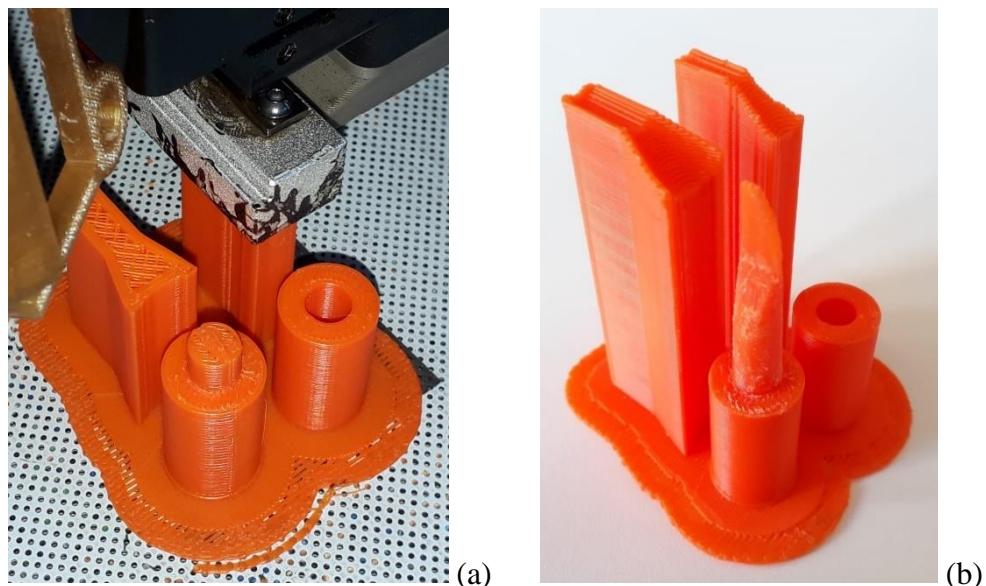


Figure 14 : Impression des supports (a) et résultat de l'impression (b)

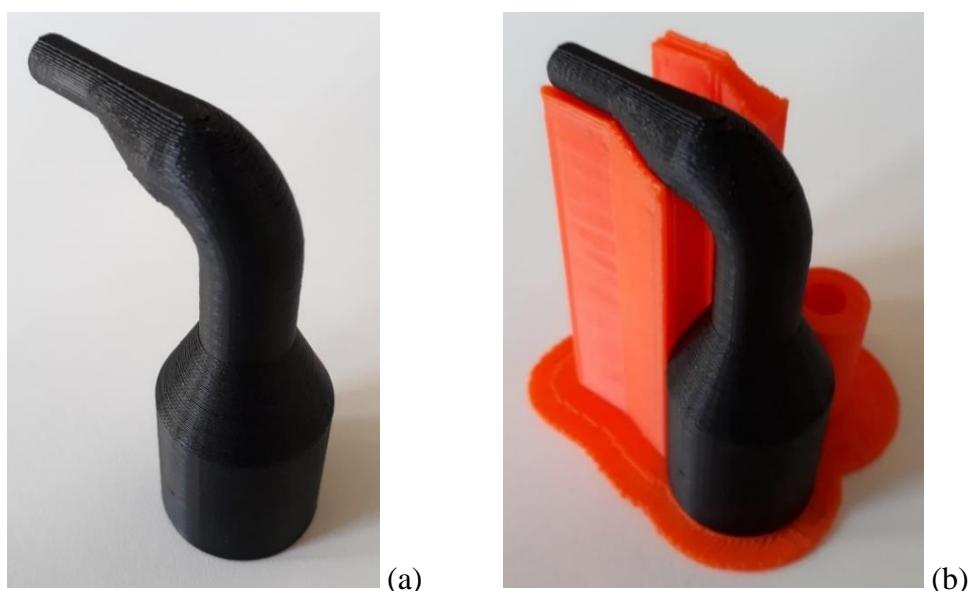


Figure 15 : Pièce de base imprimée (a) et pièce de base soutenue par les supports (b)

Une fois les supports et la pièce de base imprimés, un test a été effectué afin de les mettre ensemble. Cependant, il n'était pas possible d'insérer les supports dans la pièce, car il n'y avait pas de jeu entre les deux. Après avoir limé les pièces pour résoudre ce problème, il a été possible d'assembler la pièce de base et les supports (*Figure 15b*).

Suite à ce test, il a été décidé que la génération des supports avec le programme *Blender* était une bonne base. Cependant, comme l'a démontré cet essai, il faut continuer à développer et à améliorer des outils pour gérer les problèmes rencontrés et ainsi créer un panel de contrôle proposant plusieurs options dans le but de générer de meilleurs supports.

7. Implémentation

7.1. API de Blender

L'API de *Blender* offre de nombreuses méthodes permettant de contrôler les outils de *Blender* et dont la description des fonctionnalités se trouve dans la documentation de l'API [3]. Elle permet de développer des tableaux de contrôle avec des menus, des sliders et des boutons qui peuvent exécuter des scripts quand on appuie dessus, mais aussi de créer des paramètres (*FloatProperty* et *IntProperty*) pouvant être considérés comme des variables globales de *Blender*. Le code des fonctionnalités doit être écrit en Python et doit ensuite être ajouté comme add-on sur *Blender* pour y avoir accès. Des modules Python donnent les méthodes qui permettent d'utiliser tous les outils de *Blender* à l'aide de code. Le code ci-dessous montre quelques exemples de méthodes donnant accès aux outils de *Blender*.

```

1. # Switch in edit mode
2. bpy.ops.object.mode_set(mode = 'EDIT')
3.
4. # Select all the faces
5. bpy.ops.mesh.select_all(action='SELECT')
6.
7. # Extrude the support
8. bpy.ops.mesh.extrude_region_move(''arguments'')
9.
10. # Select all
11. bpy.ops.mesh.select_all(action='SELECT')
12.
13. # Bissect and delete the element under the xy plane
14. bpy.ops.mesh.bisect(plane_co=(0, 0, 0.001), plane_no=(0, 0, 1), use_fill=False,
15.                     clear_inner=True, xstart=942, xend=1489, ystart=872, yend=874,
16.                     flip=False)
17.
18. # Switch in object mode
19. bpy.ops.object.mode_set(mode = 'OBJECT')
```

7.2. Logiciel

7.2.1. Fichiers

Le logiciel a été séparé en 4 fichiers Python.

Le fichier `__init__.py` :

Il est l'élément central de l'add-on. Il contient toutes les classes des menus et des boutons, ainsi que la déclaration de tous les paramètres. De plus, ce fichier contient les deux méthodes `register()` et `unregister()`. La méthode `register()` est appelée lors de l'installation de l'add-on et permet d'enregistrer toutes les classes des fichiers de l'add-on et de créer tous les paramètres. La méthode `unregister()` est appelée lors de la désinstallation de l'add-on et fait exactement le contraire de `register()`.

Le fichier `getter_and_setter.py` :

Il contient les méthodes `get()` et `set()` des paramètres. Certains paramètres possèdent ces méthodes, afin de pouvoir exécuter des scripts dynamiquement quand le slider lié aux paramètres change de valeurs.

Le fichier `import_export.py` :

Il possède les méthodes nécessaires pour pouvoir ouvrir un explorateur de fichier qui ne peut sélectionner que des fichiers STL pour l'import et l'export.

Le fichier `operations.py` :

Il contient toutes les méthodes qui vont s'exécuter lorsqu'un bouton est appuyé.

7.2.2. Menus

Dans *Blender*, chaque menu est une classe héritée de la classe *Panel* fournie par *Blender*. Elle possède plusieurs attributs dont les plus importants sont *bl_idname*, *bl_label*, *bl_space_type*, *bl_region_type* et *bl_category* qui indiquent respectivement le nom de la classe, le nom qui est affiché, la région où le menu est utilisé et la catégorie du menu. Elle a aussi une méthode *draw()* qui permet de dessiner les boutons, les paramètres sous forme de sliders et les labels désirés. Chaque bouton est une classe héritée de la classe *Operator* fournie par *Blender*. Les attributs utilisés sont *bl_idname* qui est le nom de la classe, *bl_label* qui est le nom qui est affiché, *bl_description* qui donne une description de l'utilisation du bouton et *bl_options* qui permet d'ajouter des options à l'opérateur et notamment celle pour revenir en arrière « *ctrl + z* ». Les boutons possèdent aussi une méthode *execute()* qui est exécutée quand le bouton est appuyé.

Les fonctionnalités du logiciel ont été séparées en 8 menus. L'explication détaillée de chaque bouton et paramètre de ces menus est disponible dans la documentation technique (Annexe 1).

Menu Import/Export :

Il possède des boutons pour gérer l'import et l'export de fichiers STL. Il a aussi des fonctions pour définir les unités en mm, pour calculer le volume de l'objet et pour rendre toutes ses faces triangulaires. Des labels affichent le nombre de faces ainsi que la taille du fichier STL.

Menu Rotation and offset :

Il s'occupe de tourner la pièce dans tous les sens et de pouvoir translater verticalement l'objet à l'aide des paramètres qui sont dans des sliders.

Menu Generation :

Il permet de sélectionner les faces qui ont besoin de supports à l'aide d'un paramètre d'angle max. Il peut aussi générer des supports de cette sélection ou faire un moule s'il y a un offset z négatif. Ce menu possède aussi des options pour générer le fond du support et un socle.

Menu Area :

Il permet de séparer les faces sélectionnées et de supprimer les groupes de faces dont l'aire est inférieure à un paramètre d'aire minimale.

Menu Resize :

Il s'occupe de sélectionner automatiquement toutes les faces qui sont connectées à la sélection. Il peut aussi sélectionner automatiquement les faces qui sont connectées à la sélection et dont l'angle se trouve entre les paramètres d'angle min et max. Il permet de redimensionner ces faces grâce à un paramètre. Il peut aussi boucher les trous d'un plan, inverser la sélection des faces et supprimer les faces sélectionnées.

Menu Lattice :

Il peut créer un objet « Treillis » aux dimensions de la pièce, modifier la position et la taille de ce « Treillis » et sélectionner les faces à l'intérieur de ce « Treillis ». Il peut aussi supprimer ce « Treillis ».

Menu Remesh :

Il possède des options pour recalculer le maillage de l'objet et ainsi nettoyer ce maillage. Il peut recalculer le maillage à l'aide de voxel, diminuer le nombre de faces de l'objet ou simplifier le maillage à l'aide de blocs.

Menu Measure :

Il permet de mesurer la distance x, y, z et la distance totale entre deux sommets.

7.2.3. Diagramme de classe

Avec les informations ci-dessus, il est maintenant possible de faire le diagramme de classe visible sur la figure 16 et de développer les fonctionnalités citées ci-dessus.

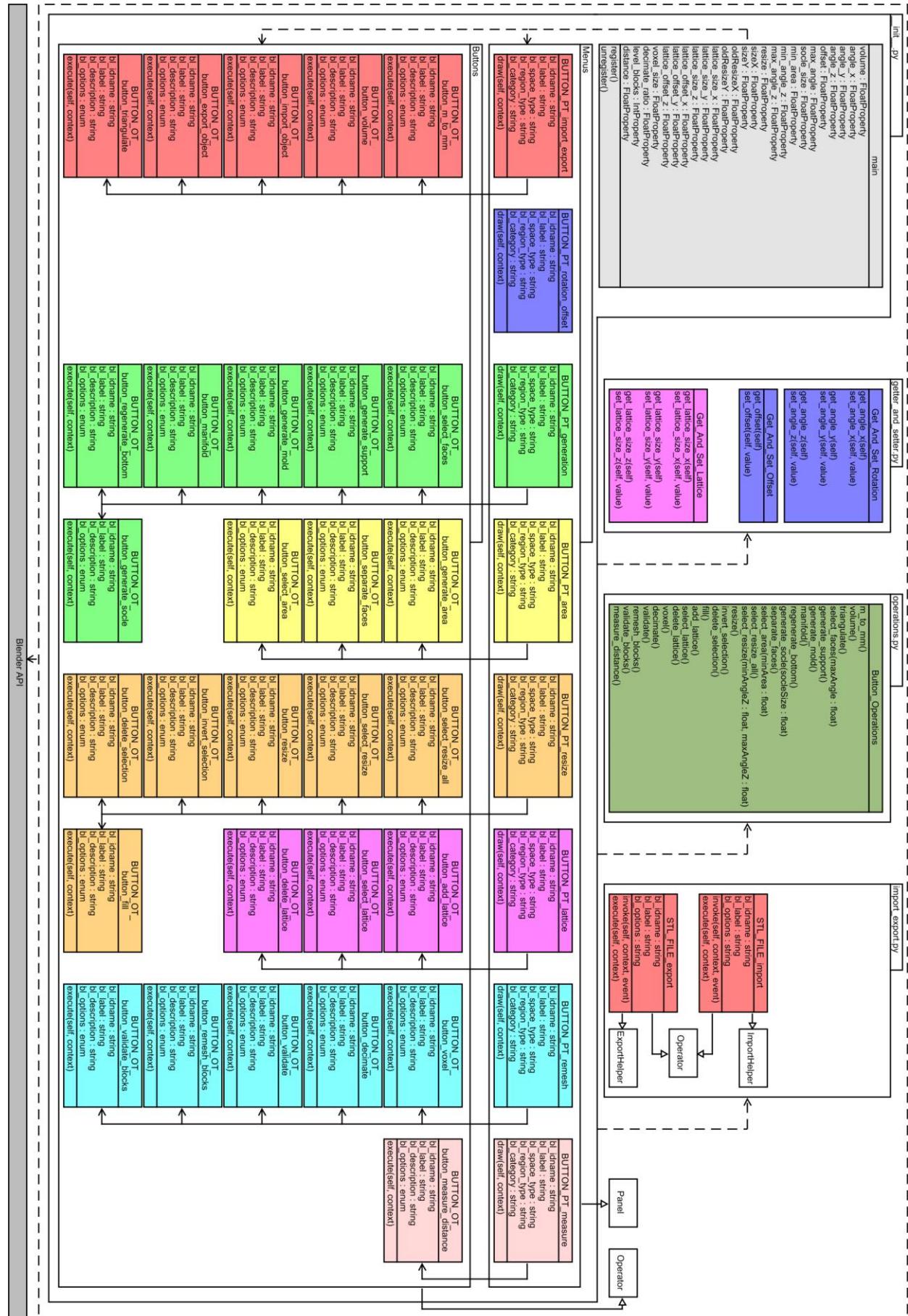


Figure 16 : Diagramme de classe du logiciel

7.3. Import/Export

7.3.1. Unité et volume

Comme expliqué dans la partie sur les fichiers STL, un fichier STL ne stocke aucune information sur les unités. L'interprétation de la longueur des vecteurs dépend donc de *Blender*. De base, *Blender* affiche les distances en mètre. Etant donné que la majorité des logiciels de création d'objets vont utiliser le millimètre comme unité pour le STL, il faut passer l'unité de *Blender* du mètre au millimètre.

L'information du volume est utile pour savoir la quantité de matière nécessaire à la fabrication de l'objet. Pour connaître cette information, le module « *bmesh* » de *Blender* possède la méthode *calc_volume()* qui utilise un algorithme pour calculer le volume de l'objet en fonction des normales des faces.

7.3.2. Import/Export

L'import et l'export des fichiers STL sont une partie importante du logiciel. La première méthode pour gérer cet aspect était d'importer le premier fichier STL d'un répertoire défini et d'exporter le résultat dans ce répertoire après la génération des supports. Cependant, cette méthode n'était pas très pratique. Après quelques recherches, il a été trouvé que l'API de *Blender* possède des modules permettant d'ouvrir un explorateur de fichiers pour importer ou exporter des fichiers. Il est aussi possible, grâce à cette méthode, de spécifier l'importation ou l'exportation uniquement de fichiers STL.

Pour exporter un fichier STL, il est parfois important de connaître sa taille, car les logiciels qui convertissent le fichier STL pour l'imprimante 3D prennent plus de temps plus le fichier est volumineux. Afin de pouvoir estimer la taille d'un fichier STL, un test a été effectué pour déterminer la présence d'un lien entre le nombre de faces du fichier et la taille du fichier.

Tableau 1 : Données du rapport entre le nombre de faces et la taille du fichier

| | | | | | | | | |
|------------------------------|--------|--------|--------|--------|--------|--------|--------|--------|
| Taille du fichier STL [Ko] | 65 | 223 | 443 | 695 | 1919 | 3246 | 4883 | 7656 |
| Nombre de faces du fichier | 1310 | 4563 | 9068 | 14220 | 39295 | 66472 | 100000 | 156672 |
| Rapport nbre de faces/taille | 20,154 | 20,462 | 20,470 | 20,460 | 20,477 | 20,478 | 20,479 | 20,464 |

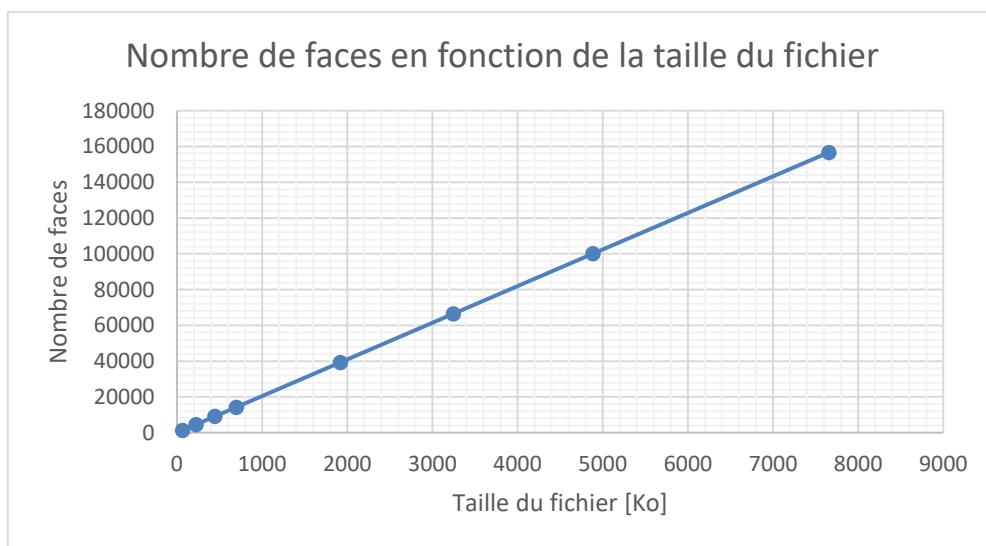


Figure 17 : Graphique de la relation entre la taille d'un fichier 3D et de son nombre de faces

Comme le montre la figure 17, il y a bien un lien entre le nombre de faces et la taille du fichier. Le graphique montre que le rapport entre le nombre de faces et la taille du fichier est une fonction linéaire où un fichier STL de 1Ko sera composé de 20,47 faces. Ce lien permet de savoir, dans *Blender*, quelle sera la taille du fichier STL grâce au nombre de ses faces. Ce rapport fonctionne seulement avec les faces triangulaires. Si, au moment d'exporter un fichier, il possède des faces non triangulaires, *Blender* va automatiquement les transformer en faces triangulaires avant d'exporter le fichier. Cette action a pour conséquence d'augmenter le nombre de faces et donc de changer la taille du fichier. C'est pourquoi il existe une option pour transformer toutes les faces non triangulaires de l'objet en faces triangulaires.

7.4. Rotations et offset

Afin de pouvoir tester facilement différents types de supports, il doit être possible de tourner l'objet à 360° sur tous les axes et de pouvoir régler la position en z de l'objet.

Dans *Blender*, il existe deux systèmes d'axes pour obtenir la position et la rotation d'un objet. Il y a le système d'axes locaux qui définit la position de l'objet par rapport à son centre. L'autre système d'axes est le système d'axes globaux qui permet d'obtenir la position du centre de l'objet par rapport à l'origine des axes globaux (0, 0, 0).

Il existe plusieurs modes pour faire une rotation de l'objet. Celui qui a été utilisé est le mode Euler XYZ. Le mode Euler est un système d'axes qui a la particularité d'avoir une relation hiérarchique entre les axes. Dans le cas du XYZ, l'axe x a comme enfant immédiat l'axe y qui, lui-même, a comme enfant immédiat l'axe z. L'avantage de ce mode est que si une valeur de rotation x, y et z est appliquée à l'objet, le résultat sera le même, peu importe l'ordre dans lequel les rotations sont réalisées. Le principal problème de ce mode est qu'il perd un axe de rotation quand la valeur de l'axe du milieu (y) est proche de 90° ou d'angles équivalents (blocage de cardan). Cependant, ce n'est pas un problème dans le cas du projet, car l'objet peut tourner à 360° dans les trois axes de rotations. Il est donc possible d'obtenir la rotation voulue par un autre moyen quand il y a un blocage de cardan. Les rotations de l'objet se font par rapport aux axes globaux.

Toutes les informations sur les sommets, les arêtes ou les faces (coordonnées, vecteurs normaux, ...) sont des vecteurs indiqués par rapport aux axes locaux. Quand il y a des opérations qui nécessitent d'utiliser les axes globaux, il faut donc modifier les informations des axes locaux en informations pour les axes globaux et vice-versa grâce aux matrices.

Chaque objet dans *Blender* possède un paramètre `matrix_world` qui est la matrice de transformation de l'espace global. Il est possible d'avoir `matrix_world` en tant que matrice 3x3. Cette matrice contient la direction normalisée des trois axes x, y et z locaux.

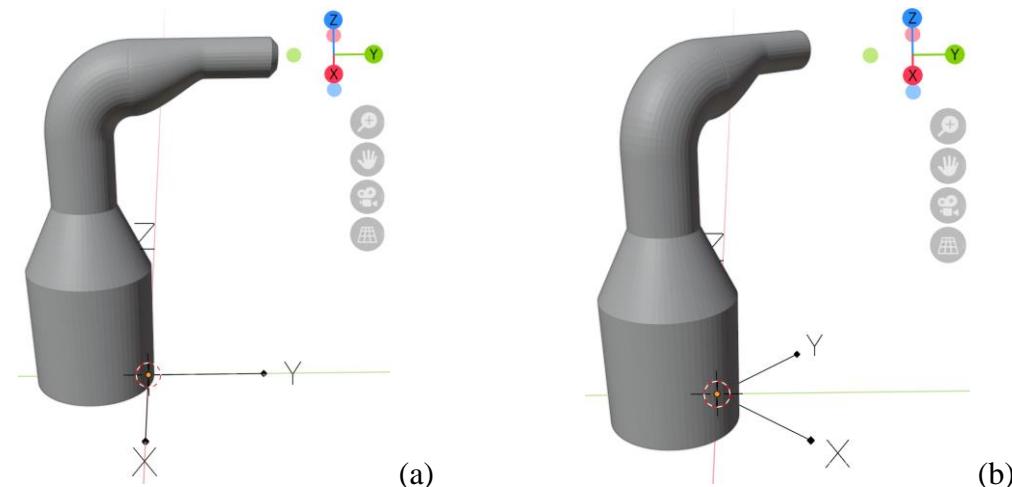


Figure 18 : Axes globaux et locaux sans rotation (a) et avec une rotation de 45° en z (b)

La figure 18 montre un objet respectivement sans aucune rotation (a) et avec rotation (b). Sur les figures, le système d'axes globaux est en haut à droite et le système d'axes locaux est au centre de l'objet. Si l'objet n'a aucune rotation, alors les axes globaux pointent dans les mêmes directions que les axes locaux comme le montre la figure 18a. La matrice de transformation aura donc la forme suivante :

$$\text{matrix}_{\text{world}} = \begin{pmatrix} x & y & z \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

S'il y a maintenant une rotation de 45° sur l'axe z, il y a une différence entre les axes globaux et les axes locaux (*Figure 18b*). L'axe local x pointe comme la bissectrice des axes x et y globaux, l'axe local y pointe comme la bissectrice des axes -x et y globaux et l'axe z local pointe toujours comme l'axe z global. Cela donne la matrice de transformation suivante :

$$\text{matrix}_{\text{world}} = \begin{pmatrix} x & y & z \\ 0.71 & -0.71 & 0 \\ 0.71 & 0.71 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

Pour obtenir le vecteur global d'un vecteur local, il faut donc faire :

$$v_{\text{global}} = \text{matrix}_{\text{world}} * v_{\text{local}}$$

Cependant, pour obtenir le vecteur local à partir du vecteur global, il faut connaître quelques règles dans les matrices. Une matrice carrée est une matrice qui a le même nombre de lignes que de colonnes (n x n).

Une matrice identité I est une matrice carrée de dimensions n qui contient des 1 dans la diagonale de haut-gauche à bas-droite et des 0 partout ailleurs. La matrice identité de dimensions n = 3 vaut :

$$I = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

De plus, quand une matrice identité est multipliée à une matrice A, on obtient comme résultat la matrice A : $A I = I A = A$.

Une matrice inverse existe si, pour deux matrices carrées de même taille A et B, elle possède la propriété $A B = B A = I$. On dit alors que A est l'inverse de B et que B est l'inverse de A. Cela s'écrit : $A = B^{-1}$ et $B = A^{-1}$.

Ces règles permettent d'obtenir le vecteur local à partir du vecteur global. Pour cela, il faut d'abord multiplier à gauche par l'inverse de matrix_world :

$$\text{matrix}_{\text{world}}^{-1} * v_{\text{global}} = \text{matrix}_{\text{world}}^{-1} * \text{matrix}_{\text{world}} * v_{\text{local}}$$

Si une matrice est multipliée par son inverse, le résultat est la matrice identité I :

$$\text{matrix}_{\text{world}}^{-1} * v_{\text{global}} = I * v_{\text{local}}$$

Si une matrice identité I est multiplié à une matrice A, le résultat est la matrice A et donc pour trouver le vecteur, il faut faire :

$$v_{\text{local}} = \text{matrix}_{\text{world}}^{-1} * v_{\text{global}}$$

7.5. Génération

7.5.1. Sélection des faces

La sélection de faces est l'élément central de la génération des supports. Cette fonctionnalité permet de déterminer les faces de l'objet qui ont besoin de supports. Etant donné qu'il a été décidé que les supports doivent partir du plateau et qu'ils sont seulement verticaux, seules les faces qui ont un angle inférieur à une variable angle max (entre 0° et 90°) par rapport au vecteur descendant (*Figure 19*) et qui ne possèdent pas de faces en dessous d'elles peuvent avoir besoin de supports.

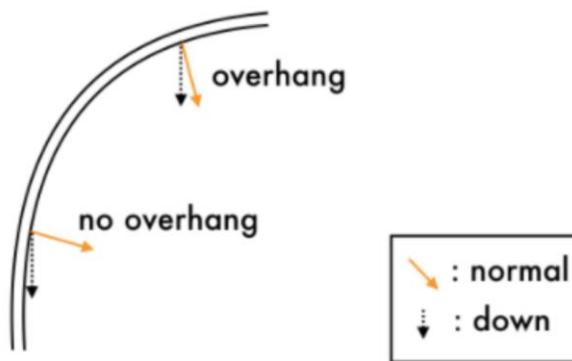


Figure 19 : Principe des faces qui ont besoin de supports en fonction de l'angle

Source : <https://help.autodesk.com/view/MSHMXR/2019/ENU/?guid=GUID-E507C608-8856-47C4-B0B8-BC570DAFD18A>

Méthodes

Afin de trouver quelles sont les faces qui ont besoin de supports, trois méthodes ont été développées :

La méthode de la sélection utilisateur

C'est la méthode la plus simple, mais elle nécessite une action de l'utilisateur et n'est donc pas automatique. Le principe de cette méthode est que l'utilisateur choisisse manuellement les faces qui ont besoin de supports. Cette méthode s'utilise en complément des deux autres méthodes qui sont automatiques, car elle permet d'affiner manuellement la sélection des faces.

La méthode de la caméra

Cette méthode utilise la caméra de *Blender*. Le principe consiste à placer la caméra en dessous de la pièce. Puis, la caméra zoomé pour que la taille de l'objet dans l'objectif soit maximale (*Figure 20a*). La caméra prend ensuite une photo et toutes les faces visibles sur cette photo sont sélectionnées (*Figure 20b*). Les faces dont l'angle entre le vecteur normal et le vecteur descendant est plus grand que l'angle max sont finalement désélectionnées.

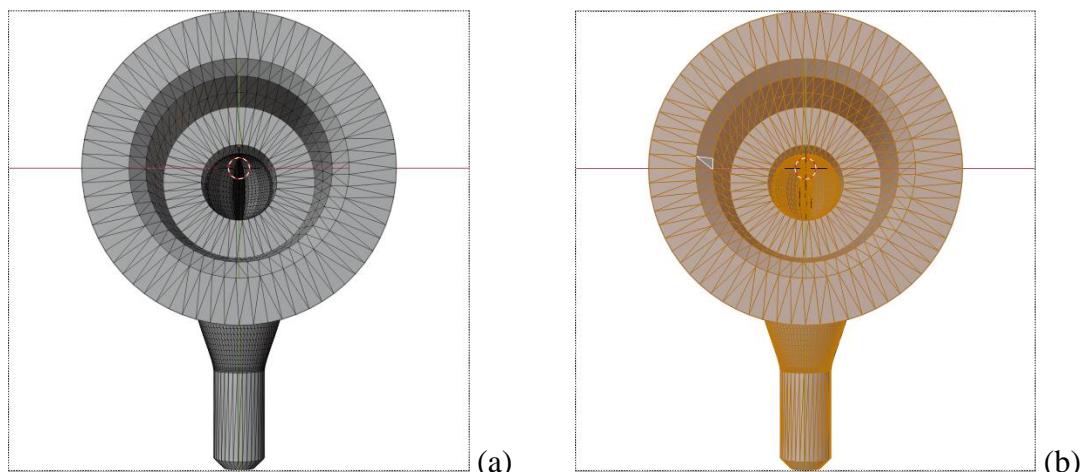


Figure 20 : Vue de la caméra après le zoom (a) et Sélection des faces visibles (b)

Le problème de cette méthode est qu'elle n'est pas assez précise. En effet, toutes les faces visibles ne sont pas forcément sélectionnées à cause de la focale de la caméra, de la taille et de l'inclinaison de certaines faces (*Figure 21a*). Cette incapacité à sélectionner toutes les faces peut amener à créer des supports relativement morcelés (*Figure 21b*).

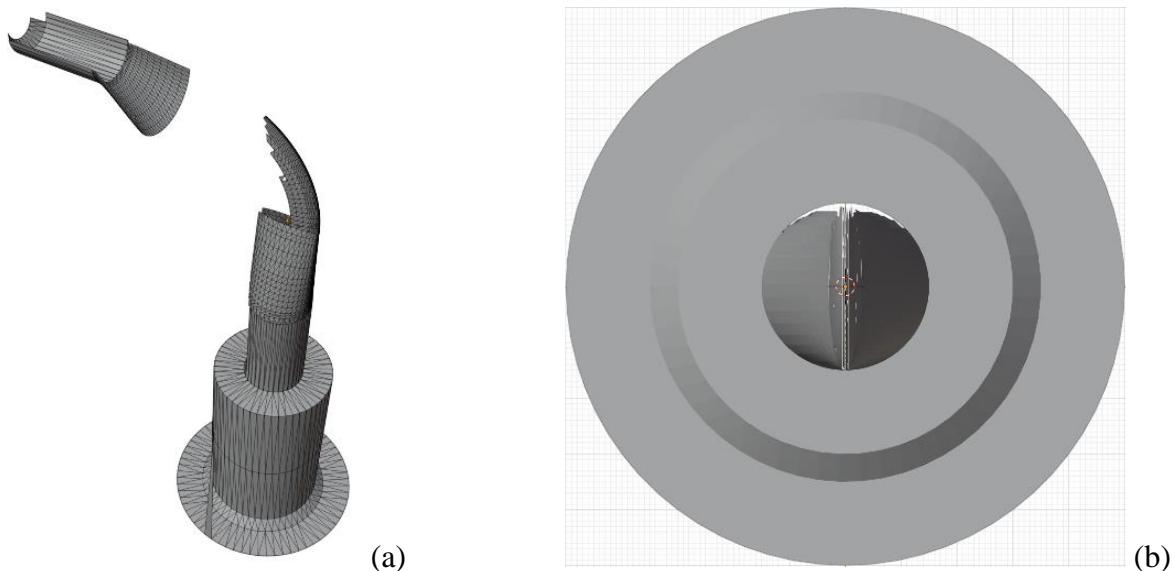


Figure 21 : Sélection avec la méthode de la caméra vu de côté (a) et vu de dessus (b)

La méthode du calcul des faces

La méthode du calcul des faces est une méthode purement mathématique dont le fonctionnement est le suivant (*Figure 22*) :

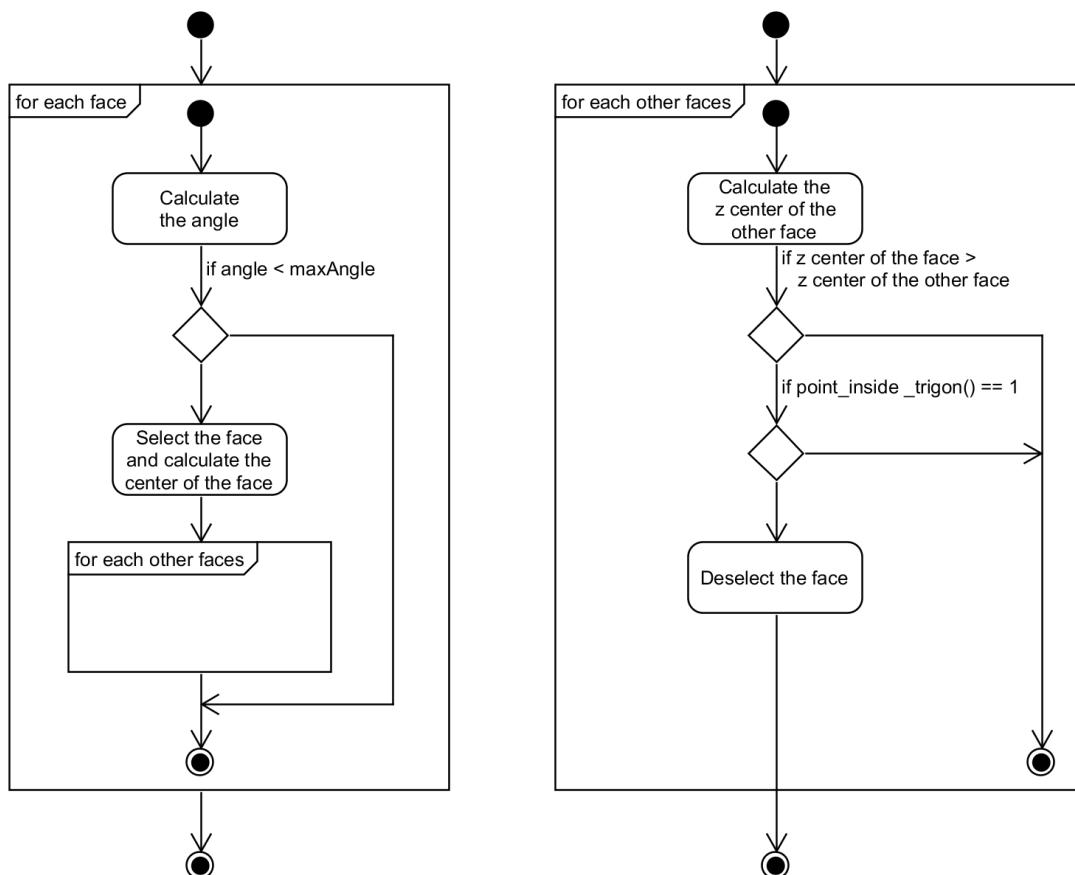


Figure 22 : Diagramme d'état de la méthode du calcul des faces

Cette méthode sélectionne toutes les faces où l'angle entre le vecteur normal de la face et le vecteur de direction descendante est plus petit que l'angle max. Pour les faces où c'est le cas, la méthode vérifie ensuite qu'il n'y a pas de faces en dessous d'elles en parcourant toutes les faces de l'objet et en contrôlant que le centre de la face ne se projette pas dans l'aire des faces qui sont plus basses qu'elles (méthode `point_inside_trigon()`). Seules les faces qui respectent ces conditions sont finalement sélectionnées.

L'angle entre deux vecteurs est calculé à l'aide de la norme et du produit scalaire. La norme d'un vecteur \vec{a} est sa longueur et s'écrit $\|\vec{a}\|$.

Pour un vecteur \vec{a} qui vaut $\begin{pmatrix} a_x \\ a_y \\ a_z \end{pmatrix}$, $\|\vec{a}\| = \sqrt{a_x^2 + a_y^2 + a_z^2}$.

Le produit scalaire de deux vecteurs \vec{a} et \vec{b} est le nombre $\vec{a} * \vec{b} = \|\vec{a}\| * \|\vec{b}\| * \cos(\varphi)$. Le résultat du produit scalaire vaut :

$$\vec{a} * \vec{b} = \begin{pmatrix} a_x \\ a_y \\ a_z \end{pmatrix} * \begin{pmatrix} b_x \\ b_y \\ b_z \end{pmatrix} = a_x b_x + a_y b_y + a_z b_z$$

Ces informations permettent de calculer l'angle voulu. L'angle φ qui est l'angle entre les deux vecteurs s'obtient avec le calcul suivant :

$$\varphi = \arccos\left(\frac{\vec{a} * \vec{b}}{\|\vec{a}\| * \|\vec{b}\|}\right)$$

La méthode `point_inside_trigon()`, qui sert à déterminer si un point est dans un triangle, utilise une projection pour passer à un problème 2D. Etant donné qu'il y a une projection, elle ne prend en compte que les coordonnées x et y des faces. Afin de déterminer si une face est dans un triangle, la méthode utilise le produit vectoriel. Le produit vectoriel de deux vecteurs \vec{a} et \vec{b} ($\vec{a} \times \vec{b}$) est le vecteur où :

- $\vec{a} \times \vec{b}$ est perpendiculaire à \vec{a} et \vec{b} .
- Le résultat de $\vec{a} \times \vec{b}$ vaut : $\vec{a} \times \vec{b} = \begin{pmatrix} a_1 \\ a_2 \\ a_3 \end{pmatrix} \times \begin{pmatrix} b_1 \\ b_2 \\ b_3 \end{pmatrix} = \begin{pmatrix} a_2 b_3 - a_3 b_2 \\ a_3 b_1 - a_1 b_3 \\ a_1 b_2 - a_2 b_1 \end{pmatrix}$.

Dans le cas d'un problème 2D, a_3 et b_3 sont nuls : $\vec{a} \times \vec{b} = \begin{pmatrix} 0 \\ 0 \\ a_1 b_2 - a_2 b_1 \end{pmatrix}$.

- Si on place le centre de la main droite au point de départ commun de \vec{a} et \vec{b} avec le pouce sur \vec{a} , l'index sur \vec{b} et le majeur perpendiculaire à la paume, le majeur pointe dans la direction de $\vec{a} \times \vec{b}$ (règle du tire-bouchon), donc $\vec{a} \times \vec{b} = -(\vec{b} \times \vec{a})$.

Grâce à la règle du tire-bouchon, il est possible de déterminer si un point S se trouve dans un triangle ABC avec la méthode expliquée ci-après.

Tout d'abord, il faut regarder de quel côté du vecteur \overrightarrow{AB} se trouve le point S avec le produit vectoriel ($\overrightarrow{AB} \times \overrightarrow{AS}$). Puis, il faut faire pareillement avec le vecteur \overrightarrow{AC} ($\overrightarrow{AC} \times \overrightarrow{AS}$). Si le point S est du même côté les deux fois, alors le vecteur résultant du produit vectoriel pointe dans la même direction et le point S ne peut pas être dans le triangle (*Figure 23a*). Sinon, le point S se trouve entre les deux demi-plans AB et AC (*Figure 23b*).

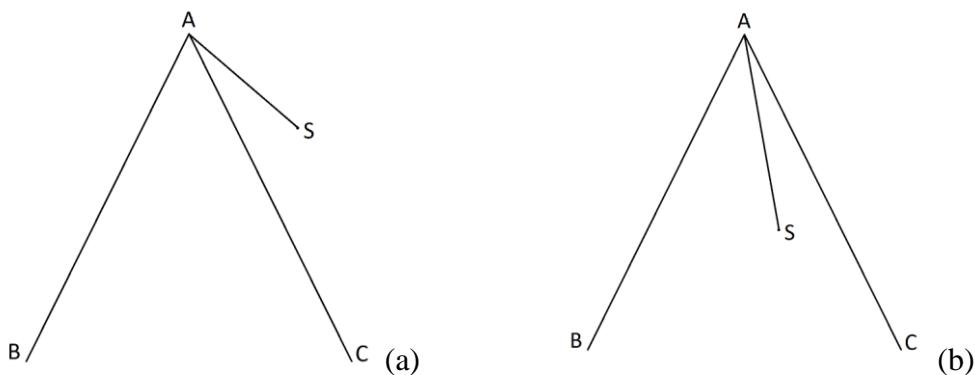


Figure 23 : Schéma pour déterminer si un point est dans les demi-plans AB et AC

Il reste finalement à contrôler de quel côté du vecteur \vec{BC} se trouve le point S avec le même principe qu'avant ($\vec{BC} \times \vec{BS}$) pour déterminer si le point est dans le triangle (Figure 24).

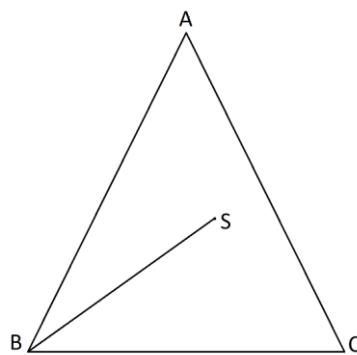


Figure 24 : Schéma pour déterminer si un point est dans le triangle ABC

Cet algorithme donne le code suivant :

```

1. int point_inside_trigon(float sx, float sy, float ax, float ay, float bx, float by, float
   cx, float cy)
2. {
3.     float as_x = sx-ax;
4.     float as_y = sy-ay;
5.
6.     int s_ab = 0;
7.     if((bx-ax)*as_y-(by-ay)*as_x > 0)
8.     {
9.         s_ab = 1;
10.    }
11.    if((cx-ax)*as_y-(cy-ay)*as_x > 0 == s_ab) return 0;
12.    if((cx-bx)*(sy-by)-(cy-by)*(sx-bx) > 0 != s_ab) return 0;
13.    return 1;
14. }
```

Choix

Contrairement à la méthode de la caméra, la méthode du calcul des faces sélectionne toutes les faces nécessaires. Le désavantage de cette méthode est que son temps d'exécution peut devenir très important pour des objets avec beaucoup de faces. En effet, l'exécution peut prendre des dizaines de minutes. La solution pour diminuer ce temps d'exécution qui est dû à la lenteur de Python est d'utiliser le module « ctypes » de Python qui permet d'appeler des fonctions C en Python. Le code C s'exécute beaucoup plus rapidement que le code Python et cela a permis de réduire le temps d'exécution du script à quelques secondes. Pour pouvoir appeler des fonctions C, le module « ctypes » demande que le fichier C soit compilé en un fichier dll. Etant donné

que le logiciel *Blender* et sa version de Python sont en 64 bits, il faut que le fichier dll soit également en 64 bits. Le fichier C a été compilé avec une version de MinGW 64 bits qui se trouve sur le site <https://nuwen.net/mingw.html#install>. Cette version de MinGW contient la version 9.2.0 de GCC et la ligne de commande pour pouvoir compiler le fichier dans MinGW est :

```
1. gcc -shared -o function.dll function.c
```

GCC possède des options pour optimiser la compilation et donc permettre une exécution plus rapide du code. La ligne de commande pour optimiser l'exécution du code (niveau O3) est :

```
1. gcc -O3 -shared -o function.dll function.c
```

Au vu des avantages de la méthode du calcul des faces par rapport à celle de la caméra, il a donc été décidé d'utiliser la méthode du calcul des faces pour la sélection des faces.

Améliorations

Cependant, après plusieurs tests, il a été montré que la méthode du calcul des faces avait quelques bugs et qu'elle devait être améliorée. Un des problèmes de cette version est dû au fait que, pour savoir si une face possède une face en-dessous, on regarde si le centre de la face est projeté dans l'aire des autres faces. Cela pose un problème lorsque seulement une partie de la face est couverte par une autre sans que son centre soit couvert (*Figure 25a*). L'autre problème est causé par la méthode pour savoir si une face est dessous ou dessus une autre. Cette méthode consiste à comparer la hauteur en z du centre des deux faces pour savoir quelle face est dessus l'autre. Le problème est que la taille des faces peut varier et donc si, comme le montre la figure 25b, il y a des grandes et des petites faces, selon l'angle, des faces qui devraient être sélectionnées ne le sont pas.



Figure 25 : Problèmes de la méthode du calcul des faces

Pour résoudre le premier problème, à la place de vérifier si le centre de la face est projeté sur une autre face, il faut vérifier si les trois sommets ne sont pas projetés sur une autre face pour sélectionner la face. A noter que, comme les faces sont reliées entre elles et donc possèdent des sommets communs, la méthode *point_inside_trigon()* ne doit pas considérer que les points qui se trouvent sur le périmètre et les sommets de la face testée soient dans cette face. Un point est considéré comme étant dans une face que s'il se trouve à l'intérieur de cette dernière. Par conséquent, il a fallu légèrement modifier la méthode *point_inside_trigon()*.

```

1. int point_inside_trigon(float sx, float sy, float ax, float ay, float bx, float by, float
   cx, float cy)
2. {
3.     float as_x = sx-ax;
4.     float as_y = sy-ay;
5.
6.     int s_ab = 0;
7.     if((bx-ax)*as_y-(by-ay)*as_x >= 0)
8.     {
9.         s_ab = 1;
10.    }
11.    if(((cx-ax)*as_y-(cy-ay)*as_x >= 0) == s_ab) return 0;
12.    if(((cx-bx)*(sy-by)-(cy-by)*(sx-bx) >= 0) != s_ab) return 0;
13.    return 1;
14. }
```

La solution au deuxième problème est qu'il faut calculer la distance perpendiculaire d'un point à un plan formé par la face testée à la place de contrôler la hauteur en z des centres pour déterminer si le point se trouve dessous ou dessus la face. Cette distance est aussi calculée à l'aide de la norme et du produit scalaire. Comme déjà présenté pour le calcul de l'angle entre deux vecteurs, la norme d'un vecteur \vec{a} est sa longueur et s'écrit $\|\vec{a}\|$ et le produit scalaire de deux vecteurs \vec{a} et \vec{b} est le nombre $\vec{a} \cdot \vec{b} = \|\vec{a}\| * \|\vec{b}\| * \cos(\varphi)$.

Le produit scalaire possède une caractéristique qui permet de faire la projection orthogonale d'un vecteur sur l'autre (*Figure 26*). La projection de \vec{a} sur \vec{b} est donc le vecteur $\frac{\vec{a} \cdot \vec{b}}{\|\vec{b}\|^2} \vec{b}$ de longueur $\frac{|\vec{a} \cdot \vec{b}|}{\|\vec{b}\|}$.

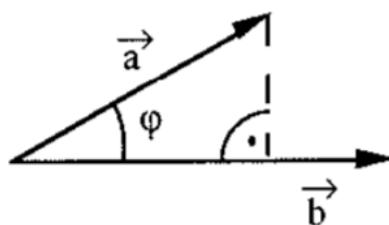


Figure 26 : Projection orthogonale de \vec{a} sur \vec{b}

Donc, pour calculer la distance perpendiculaire d'un point P au plan formé par le triangle ABC, il faut connaître le vecteur normal \vec{n} et la position du point P et d'un des sommets du triangle (A).

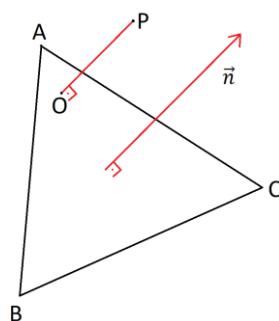


Figure 27 : Distance perpendiculaire du point P au plan formé par le triangle ABC

Puis, il faut faire la projection du vecteur \vec{AP} sur le vecteur \vec{n} , ce qui donne comme longueur :

$$dist(P, O) = \frac{|\vec{AP} \cdot \vec{n}|}{\|\vec{n}\|}$$

Etant donné que le vecteur normal des faces est un vecteur unitaire, sa norme est de 1. De plus, selon le fait que le point soit d'un côté ou de l'autre de ce triangle, cela inverse le signe de la distance. C'est ce signe qui va permettre de déterminer si le point est dessus ou dessous la face. Il faut donc enlever la valeur absolue pour conserver ce signe, ce qui donne finalement :

$$dist(P, O) = \overrightarrow{AP} * \vec{n} = AP_x n_x + AP_y n_y + AP_z n_z$$

Cette distance permet de savoir de quel côté de la face se trouve le point. Pour déterminer si ce point est au-dessus de la face avec la distance, il faut regarder la direction du vecteur normal. Dans le premier cas, le vecteur normal pointe vers le haut (positif en z), le point P est dessus la face quand sa distance à cette face est positive (*Figure 28a*). Dans le deuxième cas, le vecteur normal pointe vers le bas (négatif en z), le point P se trouve au-dessus quand la distance à cette face est négative (*Figure 28b*). Sinon, le point P n'est pas dessus la face.

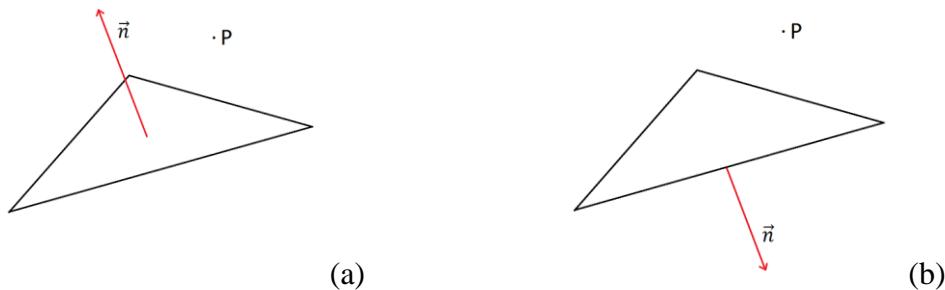


Figure 28 : Cas pour déterminer quand le point P est dessus la face

Ces différents changements font que la sélection des faces est la suivante (*Figure 29*) :

Le calcul de l'angle pour savoir si une face a besoin de support est toujours le même qu'avant. Les faces qui ont besoin de supports sont ensuite sélectionnées, puis comparées avec toutes les autres faces. Pour ce contrôle, la distance perpendiculaire entre chacun des trois sommets de la face est calculée, ce qui permet de savoir si la face se trouve dessus une autre. Dans ce cas, on vérifie si aucun des trois sommets de la face n'est projeté dans l'aire de l'autre face pour garantir qu'il n'y ait aucun obstacle entre les faces sélectionnées et le sol (plan xy).

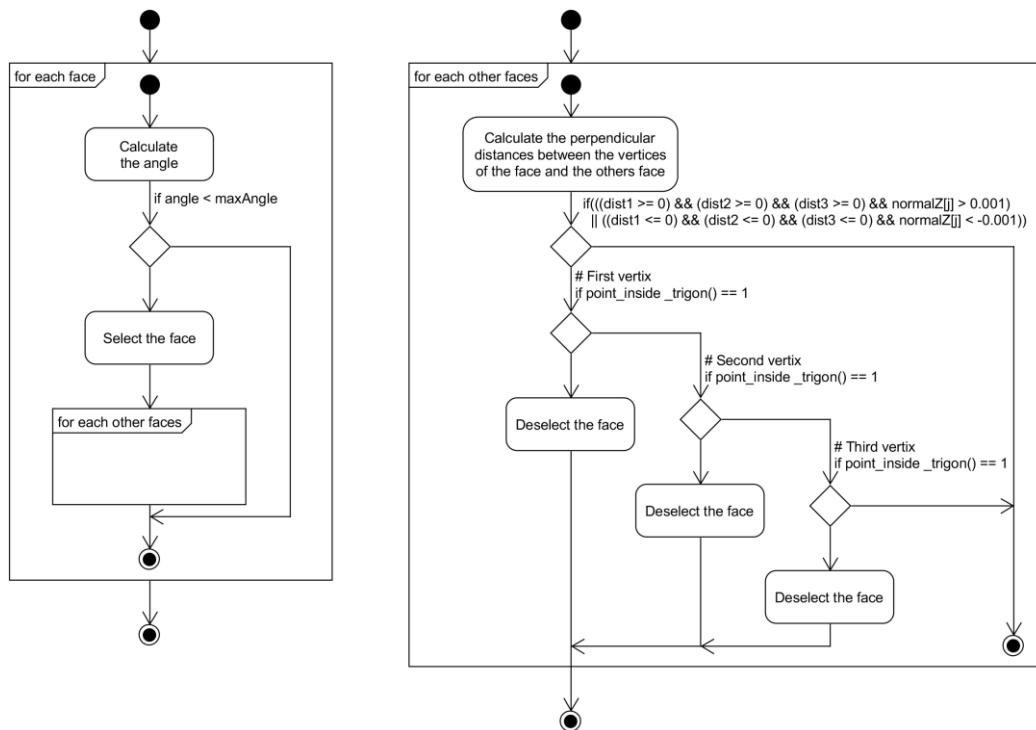


Figure 29 : Diagramme d'état de la version modifiée du calcul des faces

7.5.2. Génération des supports

Cette étape est l'étape qui va créer les supports grâce aux informations du fichier STL importé. Afin de créer les supports, plusieurs méthodes ont été testées, comme par exemple l'outil « Bridge Edge Loop » qui permet de relier plusieurs boucles d'arêtes fermées par des faces (*Figure 30*).

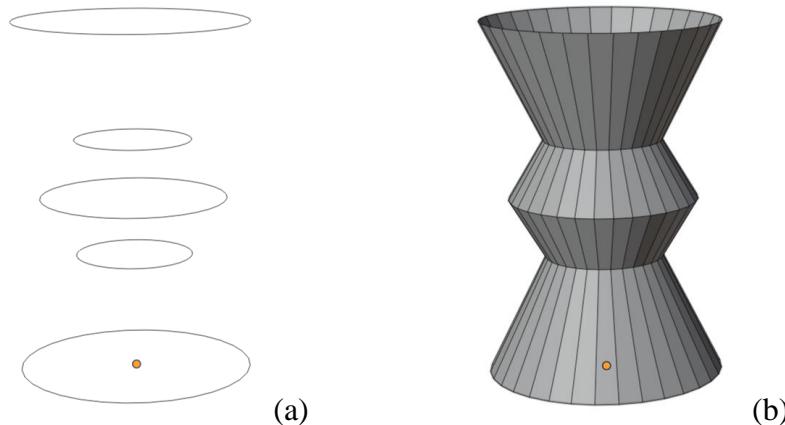


Figure 30 : Boucles d'arêtes (a) et création des faces avec l'outil « Bridge Edge Loop » (b)

Cependant, étant donné la difficulté à sélectionner les arêtes nécessaires et du fait que l'outil ne générera pas toujours les faces comme attendu, notamment avec des formes complexes, l'outil « Bridge Edge Loop » a très vite été abandonné.

La réflexion a également porté sur un moyen d'extruder les groupes de faces sélectionnées jusqu'au prochain groupe de faces, afin d'éviter d'avoir des faces internes, mais cette méthode est compliquée à automatiser. Par conséquent, le problème des faces internes sera résolu après la génération des supports (post-traitement).

Finalement, sur les différentes méthodes testées, la seule donnant de bons résultats, peu importe la forme et la complexité de la pièce ayant besoin de supports, est la suivante (*Figure 31*) :

Après avoir sélectionné les faces qui ont besoin de supports, le script va séparer les faces sélectionnées de l'objet. Puis, il va les extruder vers le bas et couper ce qui se trouve sous le plan xy. Le fond du support est ensuite généré et, enfin, l'objet de base est supprimé pour qu'il ne reste que le support.

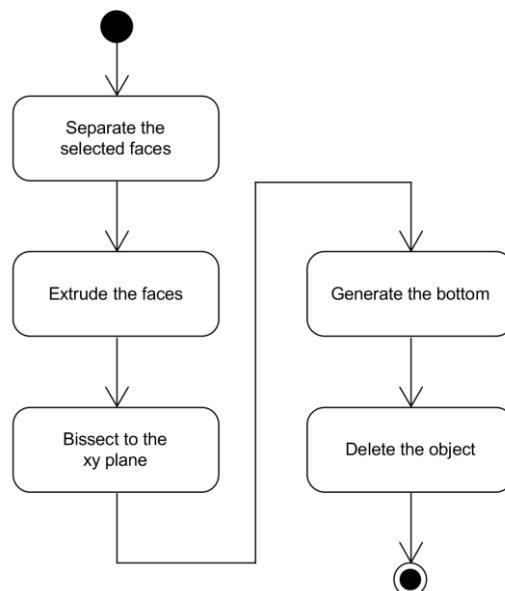


Figure 31 : Diagramme d'état de l'automatisation de la génération des supports

Cette méthode permet de générer facilement des supports verticaux à partir des faces sélectionnées, en fonction desquelles le résultatat de la génération change. Plus l'angle maximal choisi comme seuil pour sélectionner les faces est grand, plus le nombre de faces sélectionnées est grand et donc plus il y a de supports générés (*Figure 32*). Avec un angle maximal à 90° , toutes les faces visibles depuis dessous la pièce ont besoin de supports.

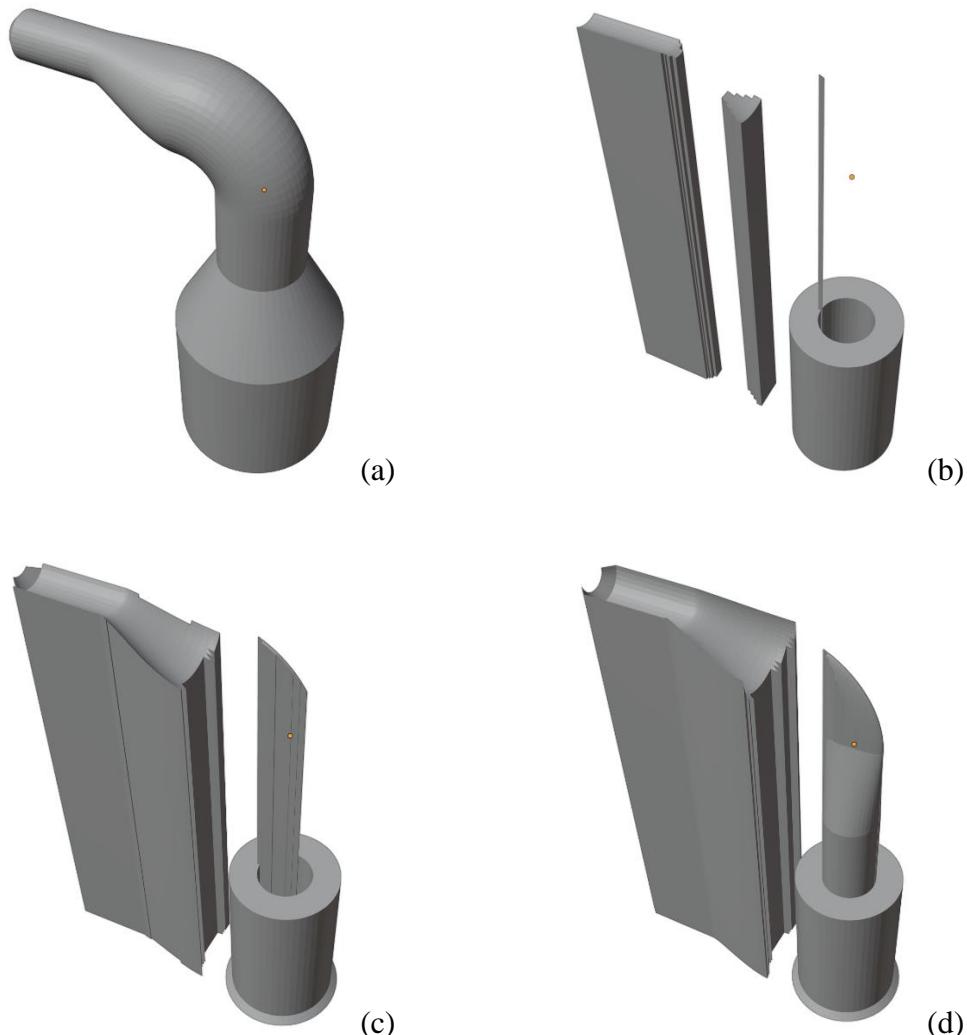


Figure 32 : Objet de base (a) et supports créés avec le calcul des faces avec un angle max de 30° (b), 60° (c), 90° (d)

Cette méthode pour créer les supports fonctionne, car seules les faces qui ne possèdent pas de faces en dessous d'elles peuvent être sélectionnées. En effet, vu que les faces sélectionnées sont extrudées vers le bas, les supports créés ne seraient pas réalisables si des faces ayant d'autres faces en dessous étaient sélectionnées.

L'avantage de cette méthode est sa faculté à créer des supports simples et robustes étant donné qu'ils partent verticalement du plateau.

7.5.3. Génération d'un moule

La réflexion s'est également focalisée sur une autre façon de générer des supports pour que la pièce soit moulée. Cette méthode peut s'avérer plus pratique que la méthode de génération basique des supports pour certaines pièces. Le but de cette fonctionnalité est de mouler toutes les parties de la pièce qui se trouvent dessous le plan xy. La pièce peut être déplacée grâce à l'offset en z. La principale contrainte, sans prendre en compte le redimensionnement des parties internes pour laisser du jeu, est que la pièce puisse être insérée et retirée du moule.

La première méthode utilisée pour créer le moule est la suivante (*Figure 33*) :

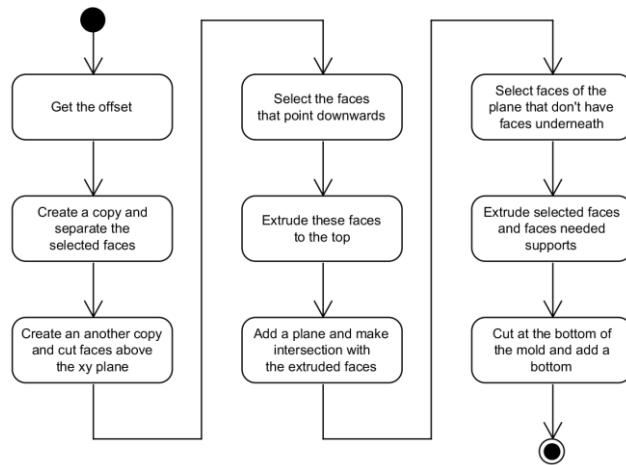


Figure 33 : Diagramme d'état de la première méthode pour générer des moules

Tout d'abord, il faut récupérer l'offset de la pièce. Puis, il faut créer une copie des faces sélectionnées qui ont besoin de supports. Une seconde copie est aussi créée ; elle va servir à récupérer les informations pour permettre de rendre le moule possible. Pour cela, il faut supprimer toutes les faces qui ne sont pas en dessous du plan xy à la copie, car elles ne sont pas utiles pour le moule. Ensuite, il faut sélectionner toutes les faces dont l'angle entre leur vecteur normal et le vecteur descendant est inférieur à 90°. Ce sont toutes les faces qui pointent vers le bas. Puis, ces faces sont extrudées vers le haut et un objet « Plan », qui a les dimensions de la pièce est ajouté. L'intersection entre le plan et l'extrusion des faces permet de projeter la partie inférieure de la pièce sur le plan, puis la partie extrudée est coupée pour ne garder que le plan. Finalement, les faces de la première copie, les faces qui ont besoin de supports, ainsi que les faces du plan qui ne possèdent pas de faces en-dessous sont extrudées, avant d'être coupées et un fond est ajouté.

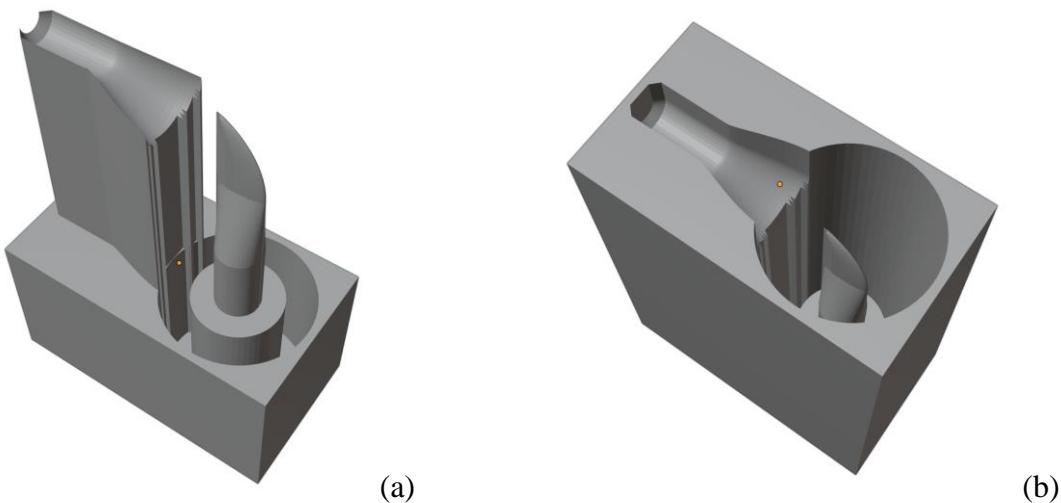


Figure 34 : Moules générés avec la première méthode

Cette méthode permet de générer des moules où il est possible d'insérer et retirer la pièce, comme le montre les figures 34a et b. Cependant, il y a parfois des erreurs lors de la projection des faces en dessous du plan xy sur le « Plan ». En effet, celles qui sont au niveau du « Plan » ne se projettent pas bien sur ce dernier. De plus, cette méthode a cessé de fonctionner lors de l'amélioration de la sélection des faces (*Figure 35*).

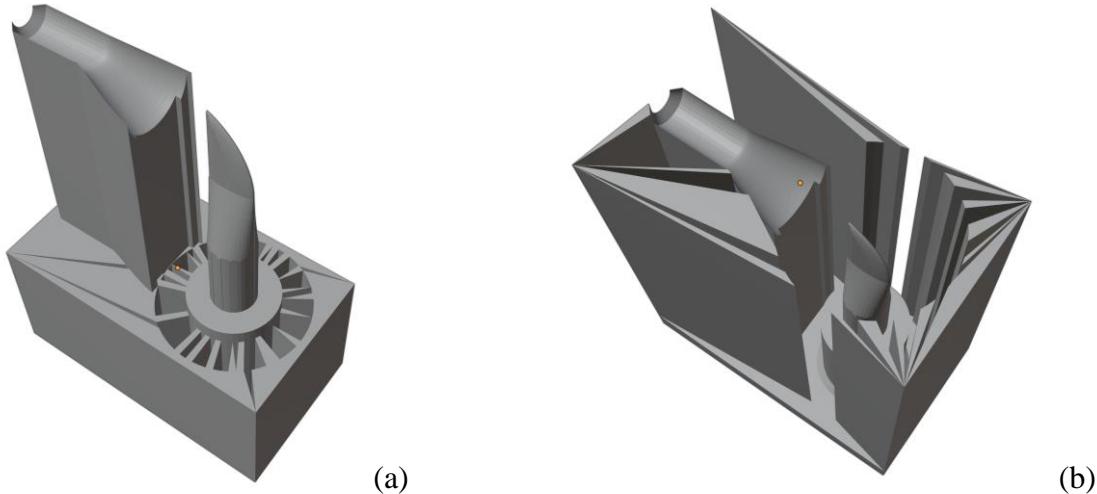


Figure 35 : Erreurs lors de la génération du moule à cause du changement de la sélection des faces

Cela est dû au fait que l'on vérifie les trois sommets de la face à la place de son centre lors de la sélection pour savoir si une face n'a pas de face en dessous d'elle. Avec la projection des faces inférieures sur le plan et la triangulation de ce plan pour pouvoir utiliser la méthode de la sélection, la précision des faces du plan ne peut pas être garantie. Par conséquent, il se peut que des faces se superposent légèrement entre le plan et les faces qui ont besoin de supports, ce qui crée des erreurs.

Afin de résoudre ce problème, le principe de cette méthode a été modifié (*Figure 36*).

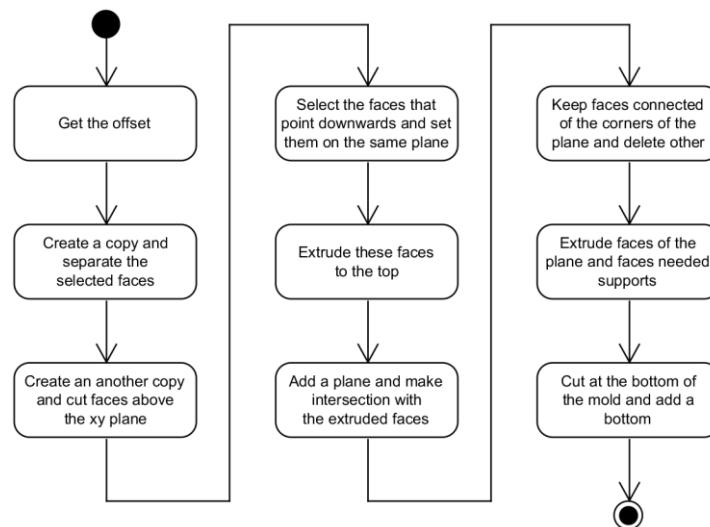


Figure 36 : Diagramme d'état de la deuxième méthode pour générer des moules

Après avoir récupéré la valeur de l'offset et créé une copie pour récupérer les faces qui ont besoin de supports, une deuxième copie est créée pour obtenir les informations, afin d'avoir un moule possible. Pour cela, la méthode est la même que pour la première méthode créée, mais après avoir sélectionné toutes les faces qui pointent vers le bas, les faces sélectionnées sont toutes positionnées sur le même plan, puis extrudées vers le haut. Cela permet de supprimer des

erreurs quand l'intersection avec le plan utilisé pour le moule est faite. Ensuite, toutes les faces qui ne sont pas connectées aux quatre coins du plan sont supprimées, afin de créer les trous pour laisser passer la pièce. Finalement, les faces du plan et celles qui ont besoin de supports sont extrudées, puis coupées et un fond est ajouté.

Ces modifications permettent de générer à nouveau des moules (*Figure 37*).

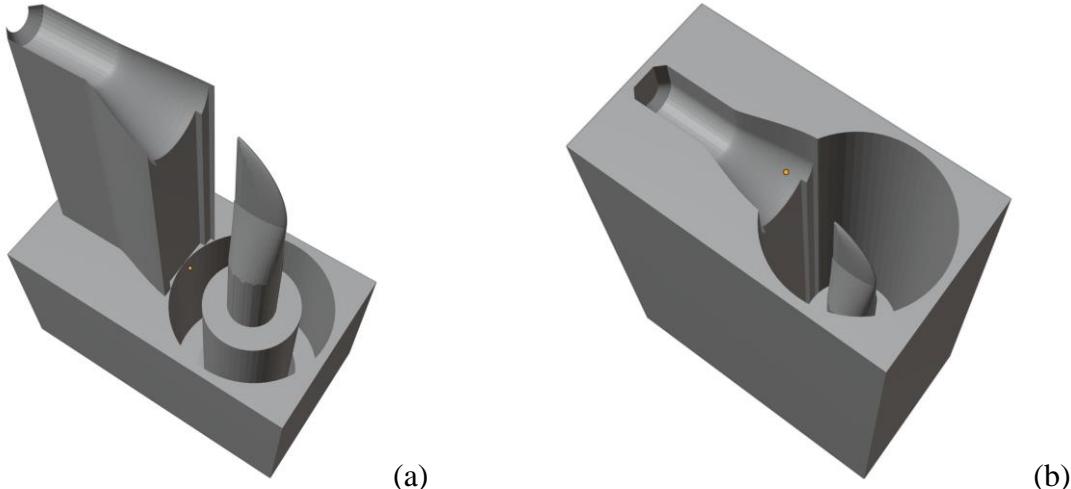


Figure 37 : Moules générés avec la deuxième méthode

Avec cette méthode, il est possible de générer des moules pour les pièces. Cependant, vu qu'il s'agit d'un moule, il y a beaucoup de supports qui doivent être redimensionnés pour laisser du jeu avec la pièce de base et, comme il est expliqué dans la partie redimensionnement (point 7.7.2), il est difficile de bien redimensionner des ensembles de faces qui ont des formes complexes. Par conséquent, le redimensionnement des parties internes du moule demande plus de travail pour les pièces complexes, afin de garantir que celles-ci puissent être insérées et retirées du moule.

De plus, un moule consomme plus de matière que les supports classiques, donc son utilisation est moins économique, mais elle peut être plus intéressante pour certaines pièces. Néanmoins, comme le montre la figure 37a, il n'est pas obligatoire de faire un moule intégral de la pièce. En outre, le moule peut être utilisé pour faire des rainures, afin d'avoir un guide pour placer facilement la pièce sur les supports (*Figure 38*).

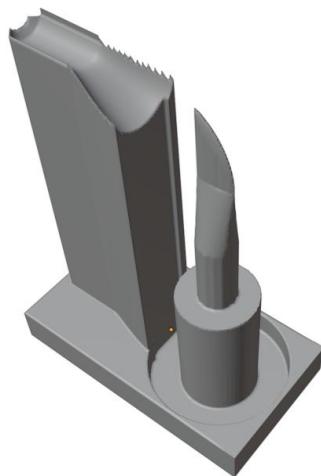


Figure 38 : Moule généré pour créer une rainure afin de guider la pièce

7.6. Options de génération

Différentes options ont été développées dans le but de permettre à l'utilisateur de mieux gérer la génération des supports.

7.6.1. Aire minimale

Lors de la sélection des faces, il arrive souvent que les faces sélectionnées soient réparties en plusieurs groupes de faces. Le problème est que, si ces groupes ont peu de faces ou des petites faces, les supports générés de ces faces ne sont pas viables ou utiles mécaniquement.

La solution pour résoudre ce problème est de calculer l'aire des faces de chaque groupe et de ne pas générer de supports pour les groupes qui ont une aire inférieure à une valeur minimale choisie (*Figure 39*).

Il faut d'abord séparer les faces sélectionnées qui ont besoin de supports et supprimer les autres faces qui ne sont plus utiles. Ensuite, toutes les faces restantes sont stockées dans un tableau et la première est sélectionnée. *Blender* possède la méthode `select_all()` qui permet de sélectionner tous les sommets (et donc les faces) connectés à cette face. Le premier groupe de faces est maintenant sélectionné, puis il est retiré du tableau des faces et stocké dans un autre tableau. Les autres groupes de faces sont séparés avec la même méthode jusqu'à ce que le tableau des faces soit vide. Après avoir isolé les faces de chaque groupe, il est possible de calculer l'aire et de ne sélectionner que les groupes avec l'aire minimale requise.

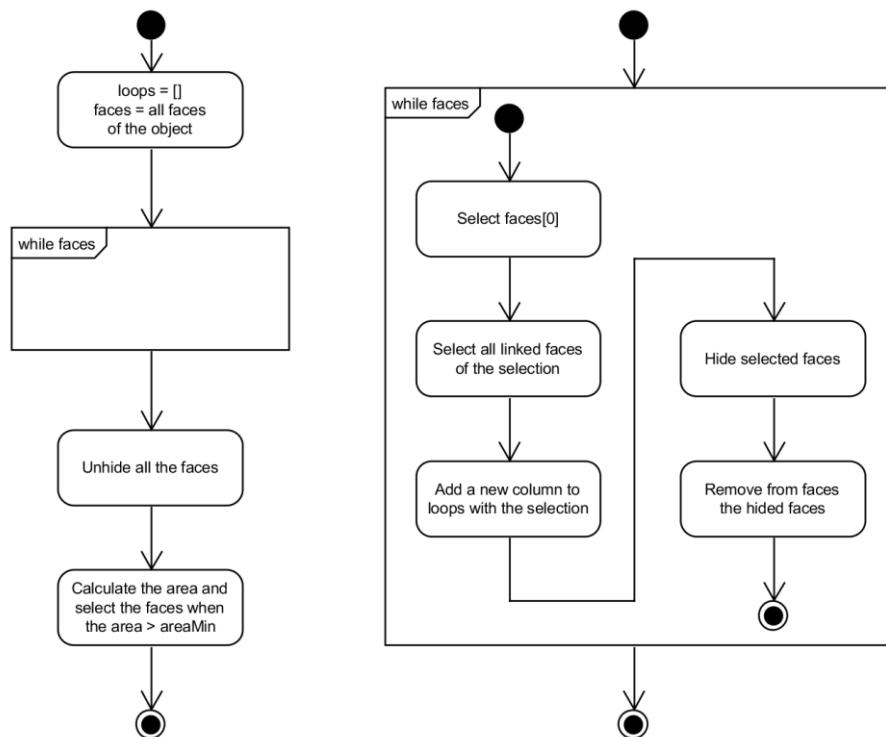


Figure 39 : Diagramme d'état de la méthode de l'aire minimale

Le paramètre area min indique l'aire minimale que doivent avoir les faces interconnectées pour pouvoir générer les supports. Cela évite que des supports trop petits soient générés. La figure 40 montre que la sélection des faces a créé quatre groupes. L'aire de ces faces est calculée et comparée à la valeur d'area min. La comparaison montre que pour ce cas de figure, le groupe 3 a une aire trop petite et qu'il n'y aura pas de supports générés à partir de ces faces (*Figure 41b*).

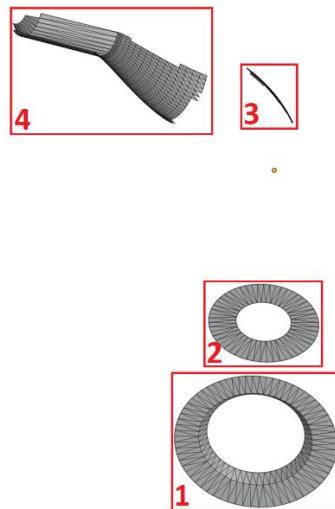


Figure 40 : Groupes de faces qui ont besoin de supports avant le calcul de l'aire minimale

La figure 41 montre bien la différence entre les supports générés sans l'aire minimale (a) et ceux générés avec (b).

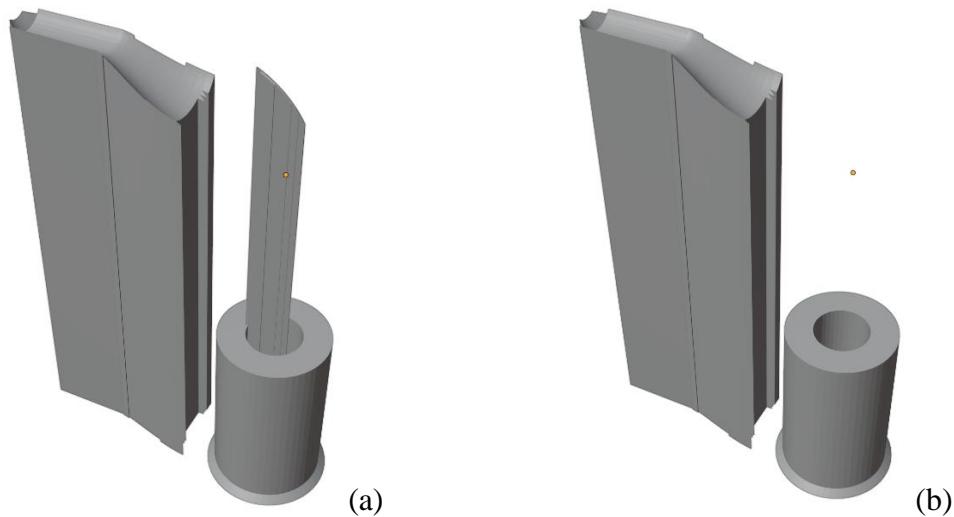


Figure 41 : Supports générés sans l'aire minimale (a) et avec l'aire minimale (b)

Comme déjà expliqué, l'aire minimale permet de créer des supports en supprimant les parties jugées inutiles, car les groupes de faces qui ont une aire inférieure à la valeur voulue ne génèrent pas de supports. Elle permet de ne garder que les groupes de faces principaux qui sont les surfaces soutenant réellement la pièce.

7.6.2. Génération du fond

Comme expliqué dans la génération des supports, il faut combler le fond pour fermer le fichier STL après avoir coupé les parties extrudées sous le plan xy.

Pour cela, *Blender* possède la méthode `select_non_manifold()` qui permet de sélectionner tous les sommets où l'objet n'est pas fermé. Avec les sommets concernés, *Blender* peut ajouter des faces et des arêtes pour fermer le fichier STL avec la méthode `edge_face_add()`.

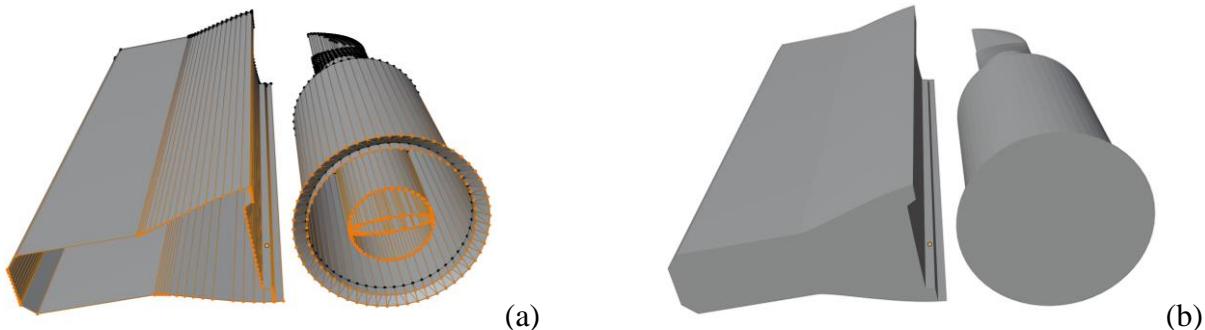


Figure 42 : Exemple de supports sans le fond (a) et avec le fond généré avec le « `edge_face_add()` » (b)

La figure 42b montre que le fond est correctement généré grâce à cette méthode. Cependant, comme le montre la figure 43b, il arrive parfois que la méthode pour générer le fond ne ferme pas complètement le fichier STL, ce qui pose problème pour fabriquer la pièce.

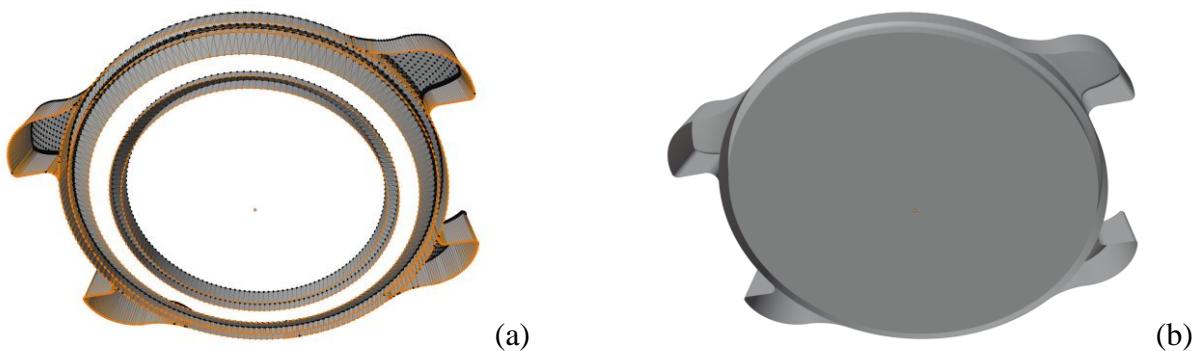


Figure 43 : Autre exemple de supports sans le fond (a) et avec le fond généré avec le « `edge_face_add()` » (b)

Une autre méthode pour générer le fond a donc été développée :

Le principe de cette méthode est d'ajouter un objet de type « Plan » dans *Blender*. Ce « Plan » fait la taille en x et y du support qui a besoin d'un fond et il est positionné légèrement en dessus du plan xy pour qu'il croise les faces des supports générés. Ensuite, l'intersection entre le « Plan » et le support est faite pour pouvoir marquer les arêtes et les sommets du support sur le plan (Figure 44).

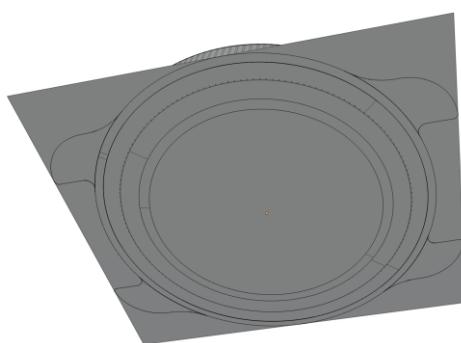


Figure 44 : Supports qui sont intersectés avec l'objet « Plan »

Il reste alors à supprimer les quatre sommets d'origine du plan. Cela va permettre de supprimer les faces à l'extérieur du support pour ne garder que le fond du support.

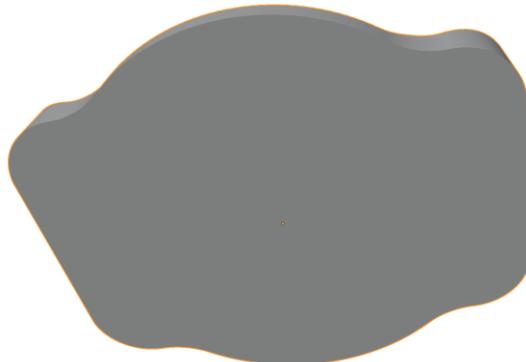


Figure 45 : Fond généré avec l'objet de type « Plan »

Comme le montre la figure 45, le fond de l'objet qui n'avait pas été généré correctement avec la première méthode est correctement généré avec la deuxième. Il possède même des faces qui ne sont pas nécessaires, car elles remplissent un endroit où il n'y a pas besoin de faces. Cela n'est pas un problème, car ces faces seront supprimées si une modification du maillage, qui est expliqué plus loin, est appliquée ou si les faces non nécessaires sont sélectionnées à la main et supprimées. Il est facile de supprimer les faces qui ne sont pas utiles, car le « Plan » est constitué d'une seule face et que l'intersection avec le support crée des faces qui ne sont pas triangulaires. Pour l'exemple de la figure 45, il suffit donc de sélectionner trois faces pour obtenir la figure 46.



Figure 46: Fond généré avec l'objet de type « Plan » sans les faces non-nécessaires

Finalement, les deux méthodes ont été gardées. La première méthode a l'avantage d'être très simple, tandis que la deuxième permet de mieux s'adapter aux pièces.

7.6.3. Génération d'un socle

Certains supports générés possèdent plusieurs parties séparées. La génération d'un socle en-dessous du support généré permet de fusionner les différentes parties ensemble. L'utilisation d'un socle est utile pour éviter d'avoir plusieurs supports et pour permettre de placer facilement la pièce sur ceux-ci lors du frittage.

La méthode la plus simple pour générer un socle à la pièce serait d'ajouter un parallélépipède rectangle dessous la pièce et de les fusionner ensemble. Cependant, cette méthode générerait beaucoup de matière inutile qui coûte de l'argent (*Figure 47*). D'autres méthodes, plus économies en matière, ont donc été développées.

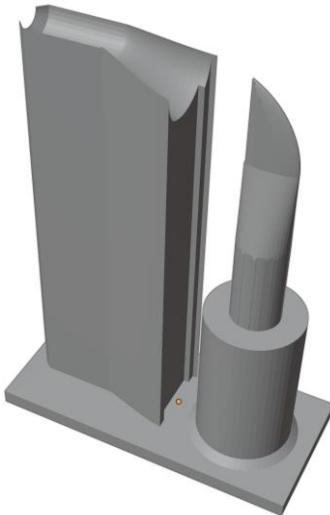


Figure 47 : Crédit de socle en ajoutant un parallélépipède rectangle

La première méthode imaginée pour générer le socle est d'utiliser la fonction *resize()* de *Blender*. Cette méthode permet de modifier l'échelle des faces sélectionnées. Le principe de cette méthode est de sélectionner le fond du support qui doit être correctement généré. Pour sélectionner le fond, il faut créer une copie du support et ne garder que les sommets, arêtes et faces qui se trouvent sur le plan xy. Ensuite, il faut séparer tous les groupes de faces avec la même méthode utilisée pour l'aire minimale. Chaque groupe de faces est redimensionné pour créer le fond et finalement toutes les faces sont extrudées verticalement, puis réassemblées avec le support.

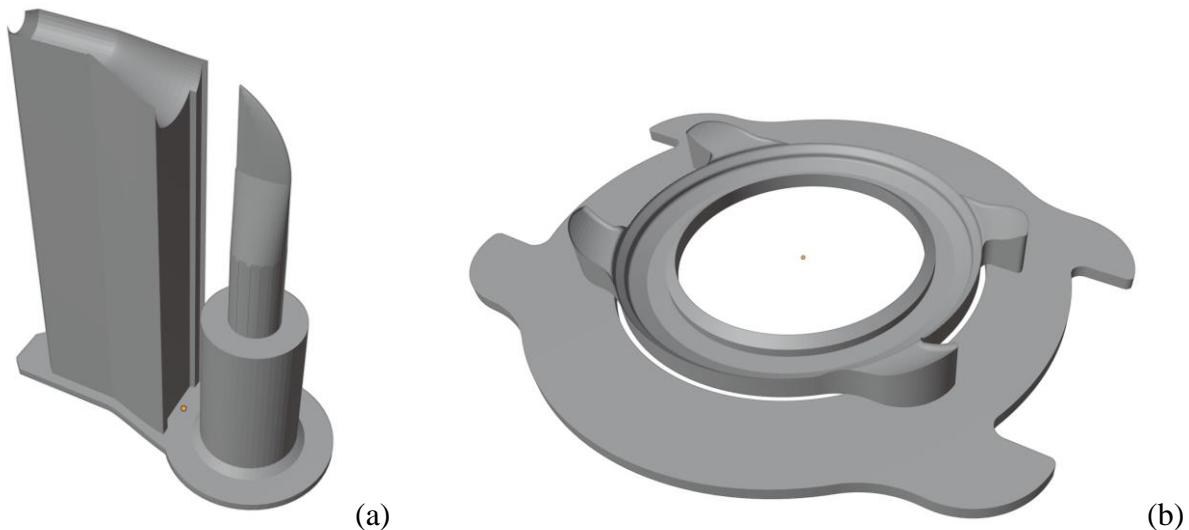


Figure 48 : Crédit de socles avec la méthode resize()

Les figures 48a et b montre des socles générés par cette méthode. Celle-ci rencontre toutefois quelques problèmes ; vu que l'on modifie l'échelle de chaque groupe de faces, il est difficile de créer un socle qui possède la même distance horizontale du support, défaut bien visible sur la figure 48a. Un autre problème plus important est que cette méthode ne peut pas gérer les supports qui possèdent des trous. En effet, la méthode `resize()` va agrandir l'échelle des faces sélectionnées. Si ces faces ont un trou, ce trou va s'agrandir et poser le problème visible sur la figure 48b. Pour le résoudre, il suffirait de boucher le trou, mais cela créerait de la matière inutile.

C'est pourquoi une deuxième méthode a été créée. Celle-ci sélectionne aussi le fond du support, comme pour la première. Puis, ce fond est extrudé verticalement. Ensuite, toutes les faces verticales sont sélectionnées. Les faces verticales sont celles qui ont été créées par l'extrusion verticale (*Figure 49*). Les faces sélectionnées sont ensuite extrudées le long de leur normale. Etant donné que les faces sont verticales, leur normale est horizontale, ce qui va donc générer le socle. Finalement, le socle est réassemblé avec le support.

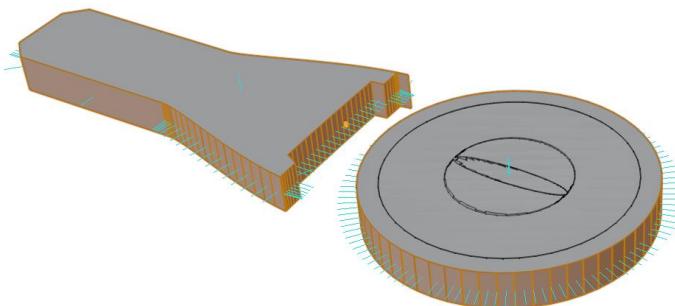


Figure 49 : Socle avant d'extruder les faces verticales le long des vecteurs normaux

Comme le montre les figures 50a et b, cette méthode corrige les problèmes de la première méthode. L'extrusion le long des vecteurs normaux permet de contrôler la longueur de l'extrusion et de s'adapter à toutes les formes, étant donné que l'extrusion suit la normale. C'est la raison pour laquelle cette méthode a été conservée.

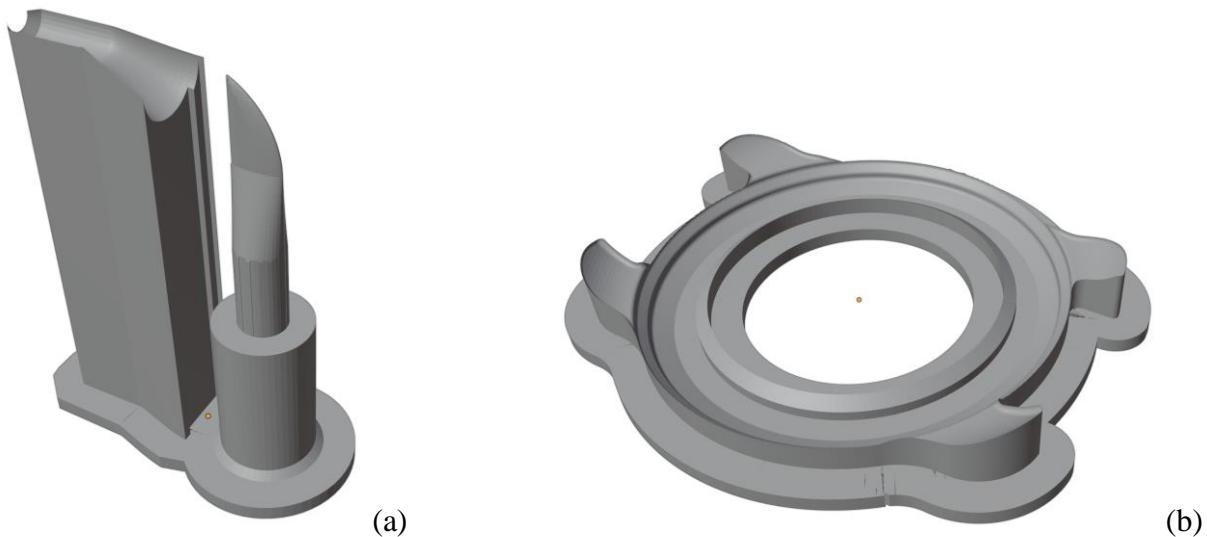


Figure 50 : Crédit de socles avec l'extrusion des faces le long des vecteurs normaux

7.7. Sélection et redimensionnement

Comme cela a été constaté lors de la première impression plastique en 3D (*Figure 14*), les supports internes n'ont pas pu être insérés dans la pièce, car il n'y avait pas de jeu entre les deux. Afin de résoudre ce problème, il a été décidé d'ajouter une option pour redimensionner les parties sélectionnées.

7.7.1. Sélection

Pour ne pas avoir besoin de sélectionner manuellement toutes les faces qui doivent être redimensionnées, plusieurs méthodes de sélection ont été développées. Pour chaque méthode, les valeurs min et max des faces en x et en y sont sauvegardées à chaque sélection. Ces valeurs sont nécessaires lors du redimensionnement pour pouvoir choisir une valeur d'agrandissement ou de rétrécissement en mm.

La première méthode sélectionne toutes les faces qui sont connectées aux faces sélectionnées. Le module « *bmesh* », fourni avec *Blender*, permet de savoir quelles faces sont connectées à un sommet grâce à l'attribut *link_faces* qui est disponible pour chaque sommet.

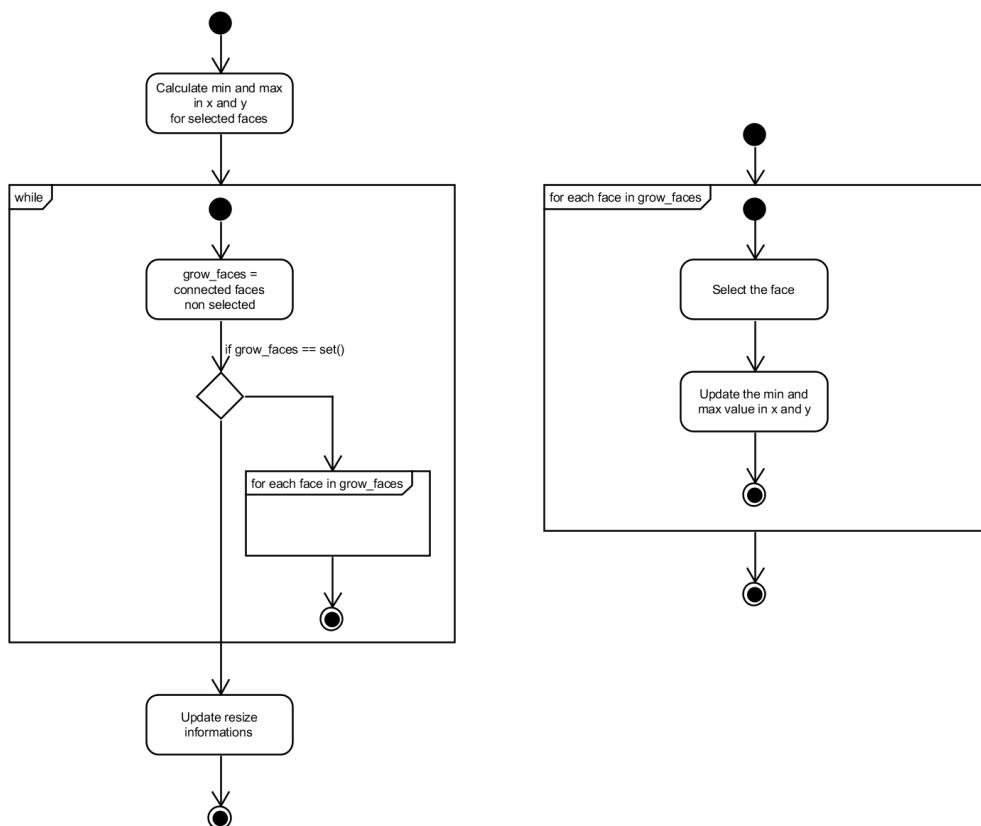


Figure 51 : Diagramme d'état de la sélection de toutes les faces connectées

Le principe de cette méthode est de savoir quelles faces sont sélectionnées, ainsi que leurs positions min et max en x et y. Ensuite, toutes les faces connectées aux faces sélectionnées sont à leur tour sélectionnées et les positions min et max sont mises à jour. Cette étape est répétée jusqu'à ce qu'il n'y ait plus aucune face répondant aux critères. Pour terminer, les informations pour le redimensionnement sont mises à jour.

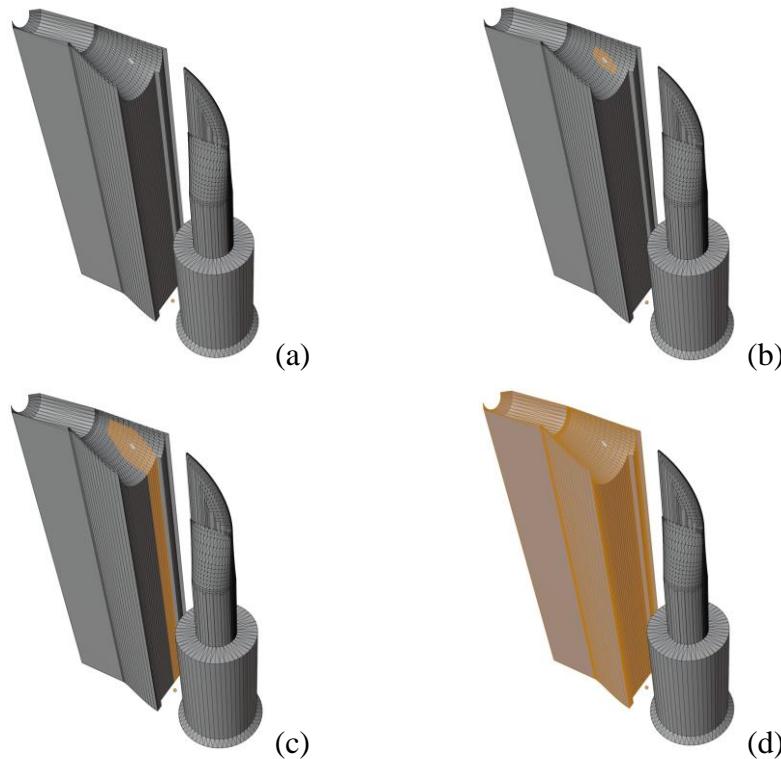


Figure 52 : Sélection de base (a), après 2 passages dans la boucle (b), après 5 passages (c) et à la fin (d)

La deuxième méthode sélectionne toutes les faces qui sont connectées aux faces sélectionnées et dont l'angle entre leur normale et le vecteur descendant est compris entre une valeur minimum et maximum choisie. Cette méthode se base sur le même principe que la première, mais avec une condition en plus.

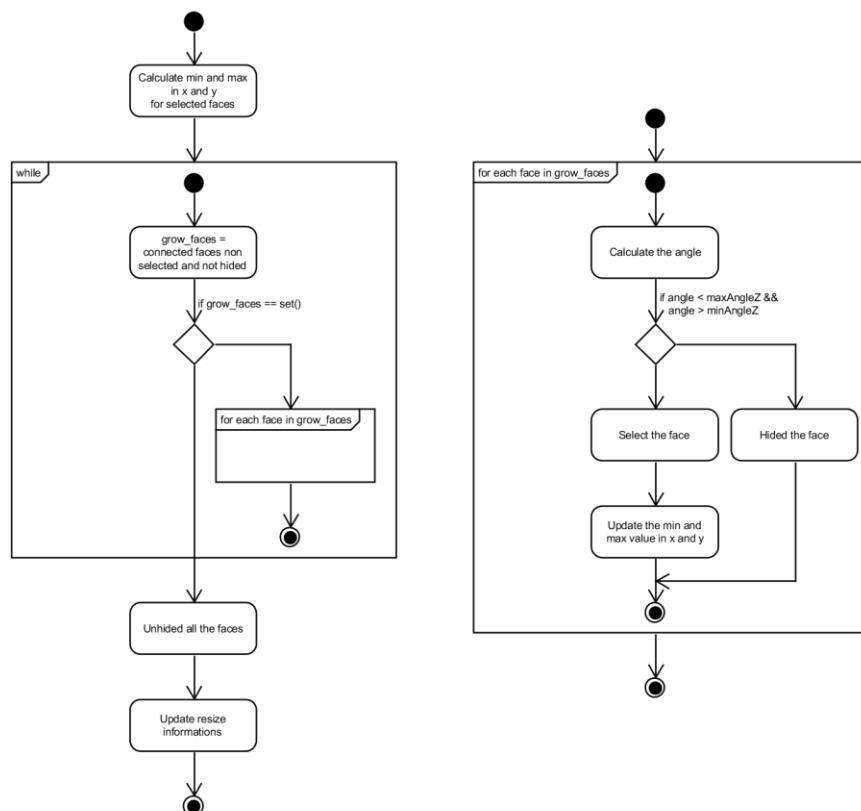


Figure 53 : Diagramme d'état de la sélection des faces connectées dont l'angle est compris entre des valeurs min et max

Le début est le même que pour la première méthode. Les positions min et max en x et en y sont enregistrées pour les faces sélectionnées. Puis, toutes les faces non sélectionnées et non cachées, qui sont connectées aux faces sélectionnées, sont recherchées. Pour ces faces, l'angle entre le vecteur normal et le vecteur descendant est calculé. Si cet angle se situe entre les valeurs min et max choisies, la face est sélectionnée et les positions min et max sont mises à jour, sinon la face est cachée. Cacher une face permet d'éviter de vérifier une deuxième fois une face dont l'angle calculé ne correspond pas aux valeurs min et max choisies. Cette étape est répétée jusqu'à ce qu'il n'y ait plus aucune face répondant aux critères. Finalement, toutes les faces cachées sont réaffichées et les informations pour le redimensionnement sont mises à jour.

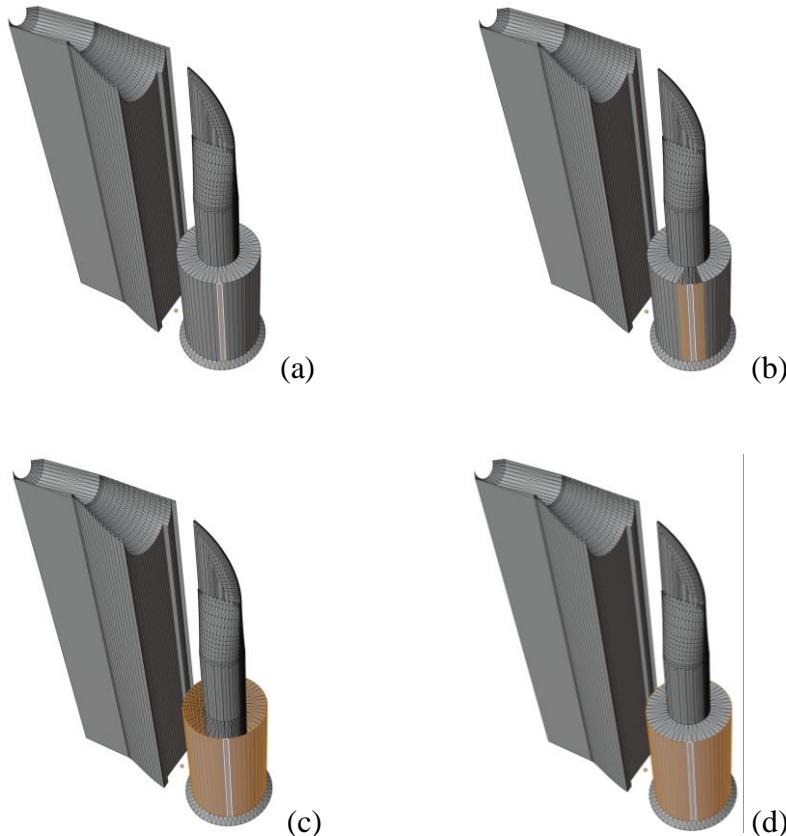


Figure 54 : Sélection de base (a), après 2 passages dans la boucle (b), à la fin (c) et après l'affichage des faces cachées (d)

La dernière méthode utilise un objet de type « Treillis » de *Blender*. Cet objet est normalement utilisé pour faire des déformations à la pièce, mais dans ce cas, il est utilisé pour sélectionner toutes les faces qui se trouvent à l'intérieur du « Treillis ».

Il y a plusieurs options avec le « Treillis ». La première option permet d'ajouter un « Treillis », s'il n'y en a aucun, et de faire en sorte que le « Treillis » ait les dimensions de l'objet (*Figure 55a*). Ensuite, la taille et la position du « Treillis » peuvent être réglées (*Figure 55b*). La deuxième option permet de sélectionner les faces qui se trouvent dans le « Treillis » et de mettre à jour les différentes informations pour le redimensionnement (*Figure 55c*). Actuellement, les faces qui sont considérées comme étant à l'intérieur du « Treillis » sont celles dont le centre est à l'intérieur. Toutefois, d'autres méthodes seraient également envisageables comme, par exemple, vérifier si un des sommets de la face est dans le « Treillis ». La dernière option permet de supprimer le « Treillis » s'il y en a un (*Figure 55d*).

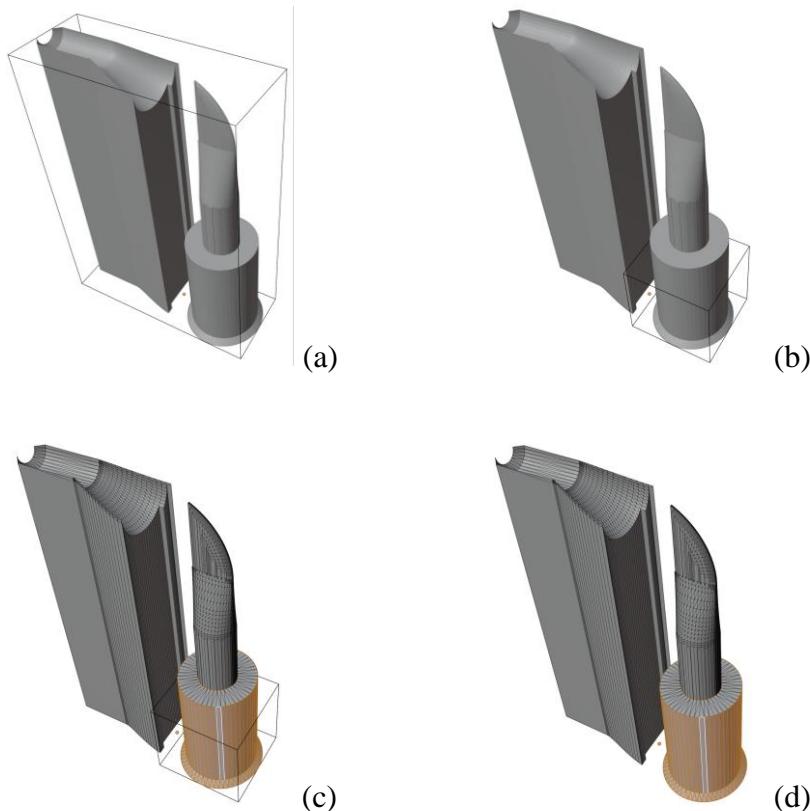


Figure 55 : Ajout d'un « Treillis » (a), modification de la taille et la position du « Treillis » (b), sélection des faces dans le « Treillis » et suppression du « Treillis » (d)

Ces différentes options permettent de facilement sélectionner les faces voulues qui sont utiles pour le redimensionnement ou pour d'autres options qui utilisent les faces.

7.7.2. Redimensionnement

Avec les faces sélectionnées, il est maintenant possible de les redimensionner. La méthode de redimensionnement est la suivante :

La valeur « $|d|$ » est la longueur du redimensionnement.

- Si $d < 0$, il s'agit d'un rétrécissement (Figure 56a).
- Si $d > 0$, il s'agit d'un agrandissement (Figure 56b).

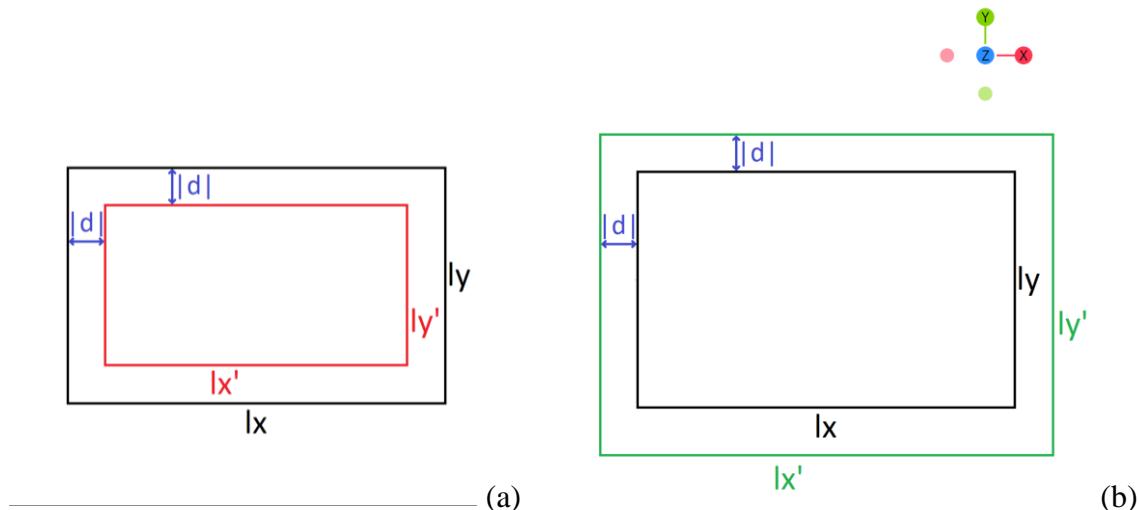


Figure 56 : Principe du redimensionnement vu de dessus

Comme le montre les figures 56a et b, le retrait ou l'ajout est le même en x et en y.

Donc

$$lx' = lx + 2 * d$$

$$ly' = ly + 2 * d$$

Pour faire le redimensionnement, la méthode `resize()` de *Blender* est utilisée. Cependant, elle ne permet que de modifier l'échelle des faces sélectionnées. Par conséquent, pour avoir la longueur « $|d|$ » du redimensionnement, il faut connaître les distances « lx » et « ly » qui ont été calculées pendant la sélection des faces.

Le calcul pour trouver l'échelle x et y à appliquer est :

$$scaleX = 1 + \frac{2 * d}{lx} \quad scaleY = 1 + \frac{2 * d}{ly}$$

Afin de permettre à l'utilisateur de tester plusieurs redimensionnements pour la même sélection, les paramètres `oldResizeX` et `oldResizeY` sauvegardent les dernières valeurs de l'échelle appliquée. Avec ces paramètres, il est possible d'annuler le redimensionnement pour retrouver la forme d'origine en appliquant l'inverse de l'ancienne échelle. Quand il y a une nouvelle sélection, `oldResizeX` et `oldResizeY` passent à 1 pour signifier que c'est une nouvelle sélection.

Le principe du redimensionnement est de calculer l'inverse de `oldResizeX` et `oldResizeY` et de redimensionner les faces sélectionnées afin de retrouver la forme d'origine. Ensuite, les échelles à appliquer sont calculées. Si `scaleX` ou `scaleY` est négatif, alors le redimensionnement est impossible et donc annulé (`scaleX` et `scaleY` valent 1). Finalement, le redimensionnement est appliqué et les anciennes échelles sont enregistrées dans `oldResizeX` et `oldResizeY` (Figure 57).

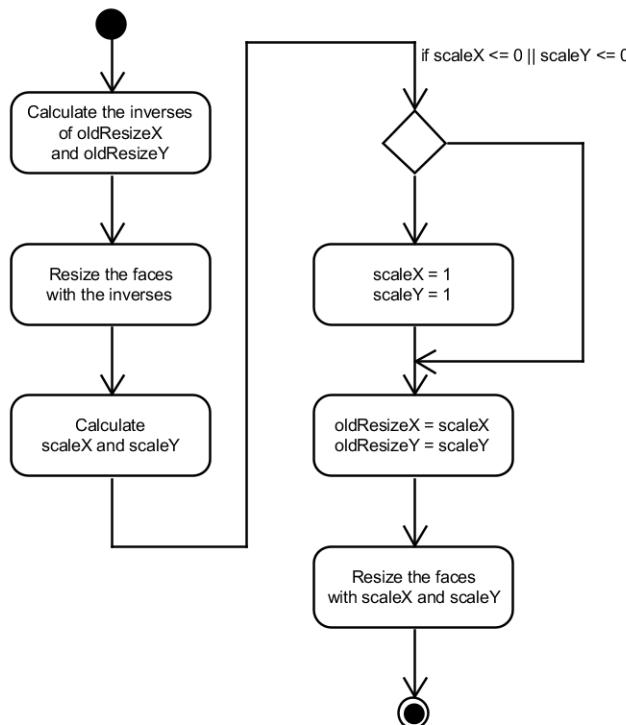


Figure 57 : Diagramme d'état du principe du redimensionnement

Avec le redimensionnement, il est désormais possible de créer des supports internes. Toutefois, un élément à prendre en compte avec cette méthode est qu'elle diminue la distance entre les points min et max en x et en y d'un facteur $2 * d$ en réduisant l'échelle des faces sélectionnées. Par conséquent, les sélections qui ont des formes géométriques simples gardent leur forme lors du redimensionnement. Cependant, il est possible que le redimensionnement déforme la forme des sélections avec des formes géométriques complexes.

7.7.3. Autres options

Il existe plusieurs options demandant que des faces de l'objet soient sélectionnées et permettant d'affiner les supports voulus.

La première option permet de combler un trou pour la sélection. Il faut cependant que les faces se trouvent sur le même plan pour que cette option fonctionne (*Figure 58*).

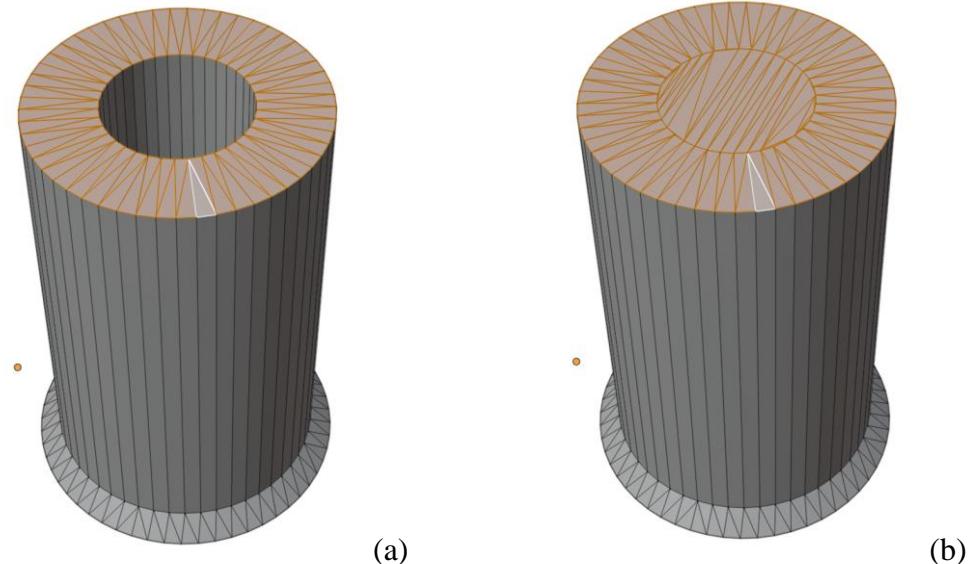


Figure 58 : Fonction pour remplir les trous des faces sélectionnées sur un même plan

La deuxième option permet d'inverser l'état de sélection de toutes les faces (*Figure 59b*). Cette option peut notamment être utile pour la troisième option, qui supprime toutes les faces sélectionnées (*Figure 59c*).

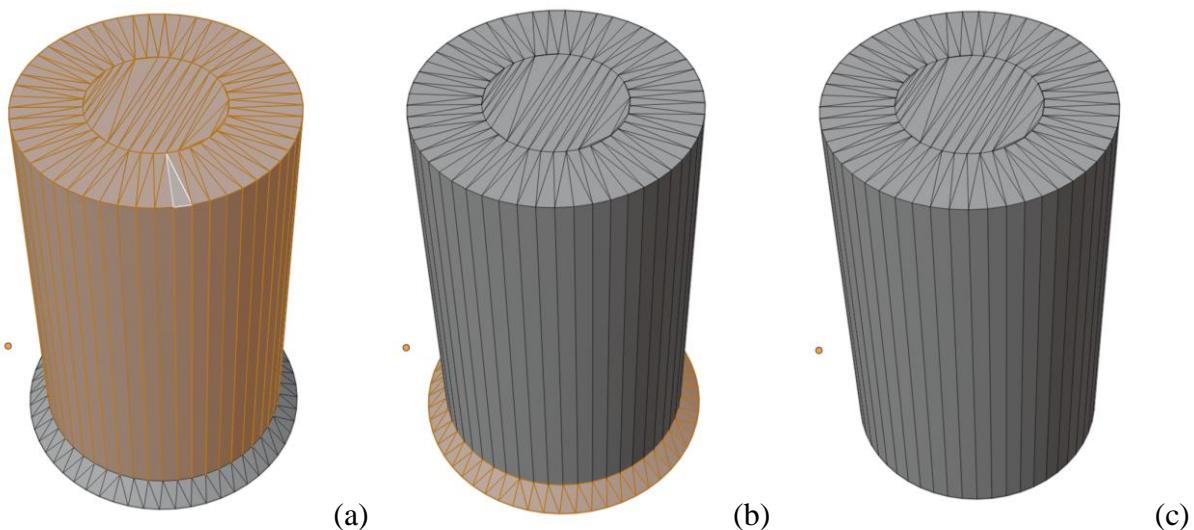


Figure 59 : Sélection (a), inversion de la sélection (b) et suppression de la sélection (c)

7.8. Modification du maillage

Plusieurs outils ont été développés pour modifier le maillage. Ces outils sont nécessaires, afin de pouvoir terminer proprement le fichier STL des supports. En effet, à ce moment-là, certains supports ont des problèmes de faces internes, causés par la présence de groupes de faces lors de la génération des supports (*Figure 60*). Cela peut poser problème plus tard, car un fichier STL décrit seulement la surface externe d'un objet fermé et donc, il devient compliqué pour un programme de déterminer le volume qui se trouve dans l'objet.

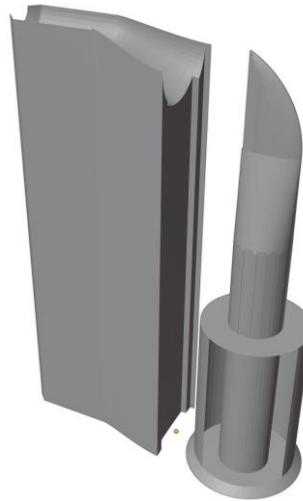


Figure 60 : Exemple du problème des faces internes

7.8.1. Modificateur voxel

Afin d'éliminer ces faces internes, il faut recalculer le maillage de l'objet. La solution pour recalculer ce maillage est d'utiliser le modificateur voxel de *Blender*. « Un voxel est à la 3D ce qu'un pixel est à la 2D » (<https://fr.wikipedia.org/wiki/Voxel>, consulté le 13 août 2021). Bien évidemment, la couleur ou l'intensité des voxels n'a pas d'importance pour le modificateur voxel, car son but est de recalculer le maillage en utilisant des voxels. La taille du voxel va déterminer la résolution des détails.

Cet outil reconstruit automatiquement la topologie de l'objet en remplaçant toutes les faces actuelles par des voxels. Cette reconstruction permet de rendre le maillage plus propre et de le nettoyer. Les faces internes qui sont inutiles vont donc être automatiquement supprimées (*Figure 61*).

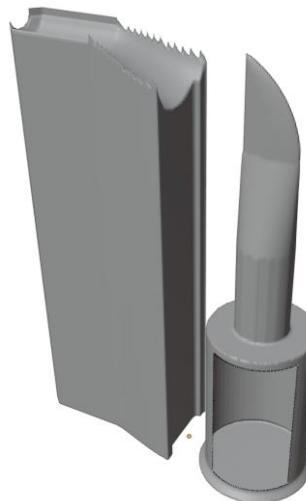


Figure 61 : Suppression des faces internes à l'aide des voxels

7.8.2. Réduction de faces

Comme énoncé dans le point précédent, le modificateur voxel permet de nettoyer le maillage et de supprimer les faces internes. En outre, plus la taille du voxel est petite, plus les détails sont précis. Cependant, cela va aussi créer plus de faces et, comme expliqué dans la partie Import/Export, il faut faire attention à ce que les fichiers STL ne soient pas trop grands et donc qu'ils n'aient pas un trop grand nombre de faces.

La réduction de faces utilise le modificateur decimate de *Blender* qui permet de diminuer le nombre de faces de l'objet. Qui dit réduction de faces, dit diminution de la résolution. Il faut donc faire attention à ce que les supports ne perdent pas trop de précision pour qu'ils soient toujours viables. Cela peut être vérifié avec l'outil « Mesure ».

7.8.3. Modificateur blocs

Cette option utilise le modificateur blocs de *Blender*, avant de faire une intersection avec le support de base. L'intersection permet de garantir que le volume du bloc ne dépasse pas de celui d'origine. Le modificateur blocs va reconstruire le maillage de l'objet avec un ensemble de blocs (*Figure 62b*). Le but de cette option est de tester les options de reconstruction du maillage pour essayer de simplifier un objet. Ce n'est donc pas une option forcément utile pour créer les supports.

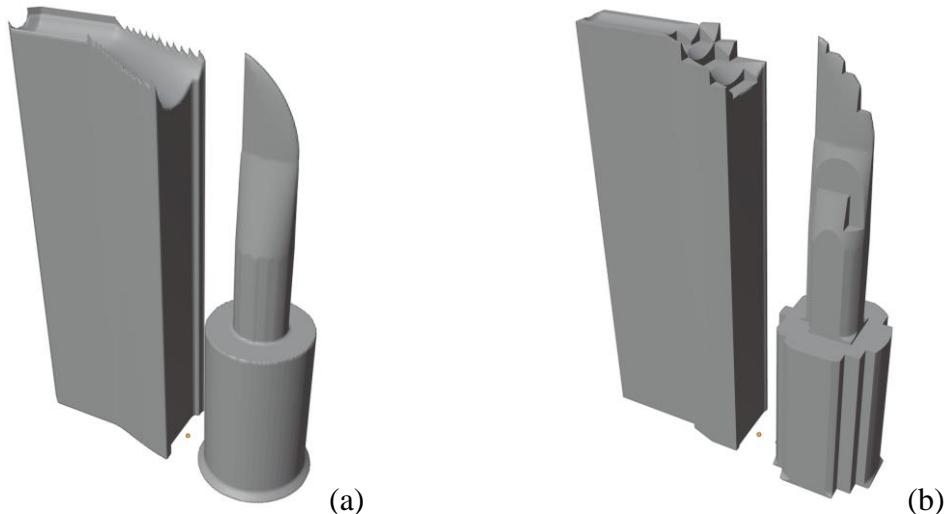


Figure 62 : Supports avant le modificateur blocs (a), après le modificateur blocs (b)

7.9. Mesure

Comme il est possible que le maillage d'un objet soit modifié, il est important qu'il existe un outil permettant de mesurer des distances. En effet, quand le fichier STL de l'objet de base est créé, il est possible de spécifier la résolution des détails de l'objet. Il est recommandé que les supports de l'objet n'aient pas une résolution plus basse que l'objet de base. Afin de pouvoir vérifier la résolution, il faut mesurer la distance entre les points.

Le principe de cette méthode est de mesurer les distances en x, y, z, ainsi que la distance totale entre 2 sommets. Pour 2 sommets $P1 = (x_1, y_1, z_1)$ et $P2 = (x_2, y_2, z_2)$, ces distances valent :

$$distanceX = |x_2 - x_1|$$

$$distanceY = |y_2 - y_1|$$

$$distanceZ = |z_2 - z_1|$$

$$distance_{totale} = \sqrt{(distanceX)^2 + (distanceY)^2 + (distanceZ)^2}$$

7.10. Fonctions non-implémentées

Blender étant un programme qui possède beaucoup de fonctionnalités, plusieurs fonctions ont été développées, mais elles n'ont finalement pas été implémentées pour différentes raisons.

7.10.1. Sélection d'arêtes

Une des fonctions étudiées utilise la méthode `region_to_loop()` de *Blender*. Cette méthode permet de sélectionner les arêtes de la bordure des faces sélectionnées. Le principe est d'utiliser cette méthode pour connaître les différentes arêtes de bordure et de séparer ensuite toutes les arêtes sélectionnées en groupe en fonction des arêtes connectées, avec le même principe que la méthode pour séparer les faces connectées de l'aire minimale. Il est par ailleurs possible de sélectionner individuellement ces groupes d'arêtes connectées.

Avoir la possibilité de sélectionner un groupe d'arêtes permet d'avoir plus d'options pour le redimensionnement étant donné que seules les arêtes sont redimensionnées. Par exemple, la sélection de faces de la figure 63a possède deux groupes d'arêtes (*Figure 63b et c*). Il est maintenant possible d'appliquer un redimensionnement à la première bordure (*Figure 64a*) et à la seconde (*Figure 64b*) pour obtenir un redimensionnement différent que celui obtenu avec les faces.

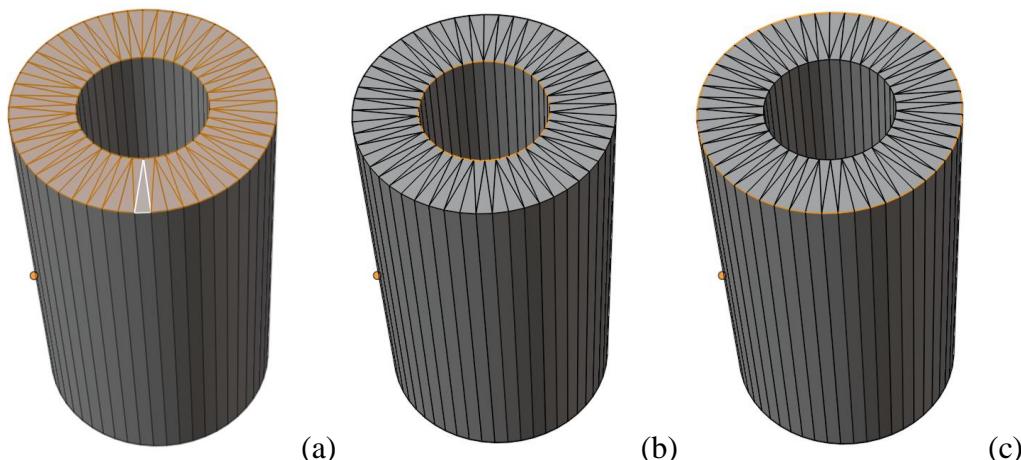


Figure 63 : Sélection de faces (a), première bordure de la sélection (b) et deuxième bordure de la sélection (c)

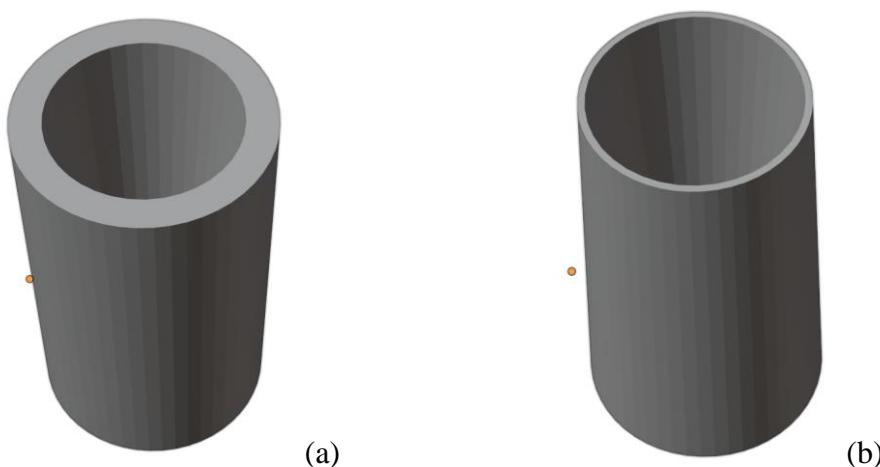


Figure 64 : Redimensionnement de la première bordure (a) et de la deuxième (b)

Cette méthode n'a finalement pas été intégrée, car elle n'apporte pas beaucoup d'avantages par rapport au redimensionnement des faces et il n'est pas très pratique de devoir enregistrer tous les groupes d'arêtes et de switcher d'un groupe à l'autre. Cependant, cette méthode pourrait être utile pour d'autres utilisations.

7.10.2. Génération de pointes

Une autre fonctionnalité étudiée est le fait de générer seulement des pointes pour soutenir les parties importantes de l'objet à la place de la méthode actuellement utilisée. Pour obtenir ces pointes, chaque face qui a besoin de supports est séparée des faces qui lui sont connectées grâce à la méthode `edge_split()` de *Blender*. La suppression des liaisons entre les faces permet de les redimensionner individuellement pour obtenir la surface des pointes. Il faut ensuite générer des supports à partir des surfaces obtenues pour avoir les pointes et faire un socle pour lier les pointes entre elles (*Figure 65*).

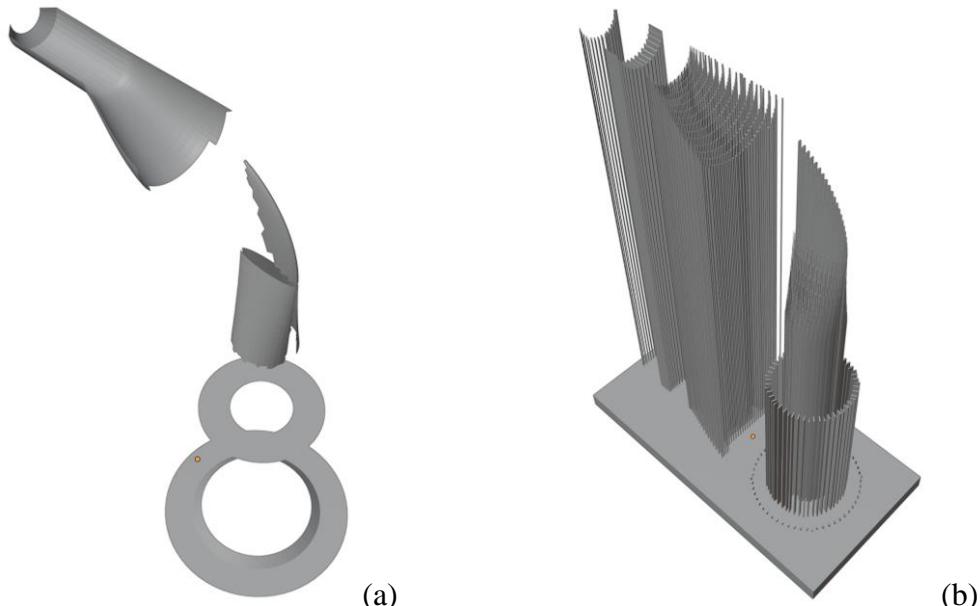


Figure 65 : Faces qui ont besoin de supports (a) et supports créés avec la méthode des pointes à partir de la sélection (b)

Cette méthode n'a pas été ajoutée, car elle n'a pas été développée plus loin. En effet, actuellement, les pointes sont créées en redimensionnant les faces qui ont besoin de supports avec un certain facteur, qui est le même pour toutes les faces. Cela implique que la taille des pointes varie beaucoup si la face est grande ou petite. Le fait qu'une pointe soit générée pour chaque face est un autre défaut de cette méthode, car le nombre de pointes dépend donc du nombre de faces.

Finalement, cette méthode peut être une bonne base pour essayer de développer une façon de générer des supports à l'aide de pointes ou d'autres méthodes similaires.

8. Validation

8.1. Transfert de connaissances

Un transfert des connaissances a eu lieu, afin de pouvoir facilement transférer le logiciel avec la personne qui va récupérer le projet. Pour que cette personne puisse comprendre comment fonctionne le programme, elle a été invitée à installer *Blender* pour pouvoir tester l'add-on créé.

Ces tests ont aussi permis de régler des problèmes de l'add-on et d'augmenter son ergonomie.

Premier test

Lors du premier test, deux problèmes ont été rapidement détectés, entraînant l'interruption du test afin de les corriger.

1. L'utilisation d'une variable pour spécifier un chemin de base dans l'explorateur de fichiers pour l'import et l'export n'est pas pratique.
2. Un problème lors de la sélection de faces a été découvert.

Le premier problème a été résolu en supprimant la variable qui permettait de spécifier le répertoire par défaut où l'explorateur de fichiers s'ouvrirait. Il a été décidé de la supprimer, car il aurait fallu modifier cette variable pour chaque ordinateur, étant donné qu'elle contenait un chemin local.

Le deuxième problème a été corrigé en améliorant la méthode de la sélection des faces. Avec cette modification, il a aussi fallu changer la méthode de génération du moule. Ce problème a d'abord dû être résolu avant de poursuivre la phase de tests.

Deuxième test

Lors du deuxième test, aucune erreur n'a été détectée au niveau des fonctionnalités mises à disposition pour la génération de supports. Néanmoins, quelques remarques ont été faites de la part du testeur concernant la convivialité (« user friendliness ») de l'interface utilisateur.

1. Les fonctionnalités d'importation/exportation et de transformation géométrique de l'objet ne sont pas indispensables, car elles sont mises à disposition par *Blender*.
2. A part cela, du point de vue de la personne qui teste, il faudrait mettre en évidence 3 groupes de fonctionnalités parmi les menus existants :
 - o La génération de supports proprement dite.
 - o Des fonctions de prétraitement, qui s'appliquent à l'objet, pour améliorer la génération de supports.
 - o Des fonctions de post-traitement, qui s'appliquent aux supports, ou qui sont à effectuer en complément de la génération de supports (« generate mold » et « generate socle »).

Ainsi, le programme gagnerait beaucoup en « user friendliness » si ces catégories d'opérations étaient bien visibles.

3. Il faudrait distinguer visuellement les boutons qui servent à choisir des paramètres de ceux qui servent à lancer une opération.
4. Au lieu d'informer dans la documentation dans quelles conditions un bouton peut être utilisé, c'est au programmeur d'activer le bouton lorsqu'il peut être utilisé et le désactiver lorsque ce n'est pas le cas.
5. Le manque d'une fonction pour pouvoir revenir en arrière « ctrl + z » a été constaté.

Concernant le point 1, il a finalement été jugé que les fonctionnalités d'importation/exportation et de transformation géométrique de l'objet étaient quand même utiles. En effet, ces fonctions exécutent plusieurs étapes. Par exemple, la fonction d'importation importe un fichier STL et place ensuite l'objet importé sur le plan xy, contrairement à l'option mise à disposition par *Blender*, qui ne va qu'importer le fichier STL. Le fait d'avoir ces fonctions permet aussi à l'utilisateur d'avoir toutes les options pour générer les supports dans le panel de contrôle et, ainsi, d'éviter de rechercher ces options dans *Blender*.

Pour le point 2, le problème soulevé est que le rôle de certains menus n'est pas clairement indiqué. En effet, pour créer les supports, il y a principalement 3 étapes en dehors de l'import/export et de la mesure. Ce sont les étapes de prétraitement de l'objet, de génération des supports et de post-traitement de ceux-ci. Pour éviter d'ajouter des bugs avec la création de nouveaux sous-menus, il a été décidé de ne pas modifier la structure des menus. Cependant, afin d'aider l'utilisateur à mieux différencier ces étapes, des informations visuelles ont été ajoutées et la description des boutons a été améliorée.

Afin de résoudre le point 3, à savoir pouvoir facilement différencier les boutons des paramètres contenus dans les sliders, il a suffi d'ajouter une barre de chargement à ces sliders. Grâce à cela, la différence entre le slider et le bouton est facilement visible et beaucoup plus claire qu'avant, comme le montre la figure 66.

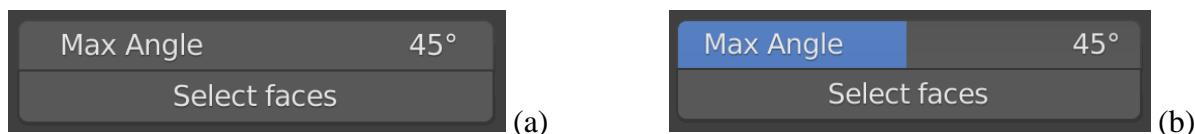


Figure 66 : Différence entre un slider (haut) et un bouton (bas) avant le changement (a) et après (b)

La solution pour résoudre le point 4 est de griser le bouton quand les conditions d'utilisation de ce dernier ne sont pas remplies (Figure 67). Cela permet en effet à l'utilisateur d'éviter d'appuyer sur un bouton qui ne va pas s'exécuter correctement dans le cas où l'utilisateur a oublié de respecter les conditions d'utilisation.



Figure 67 : Différence entre un bouton qui peut être appuyé (haut) et un bouton grisé (bas)

Pour le point 5, l'absence d'un retour en arrière n'est pas pratique. En effet, l'utilisateur ne peut pas revenir en arrière s'il a commis une erreur lors de la génération des supports et est obligé de recommencer depuis le début. De plus, s'il essaie de faire un « ctrl + z », *Blender* peut planter. En effectuant quelques recherches, il a été trouvé qu'il fallait choisir une option dans *bl_options* pour chaque bouton afin d'autoriser le retour en arrière.

Présentation du code et de l'API

Finalement, après avoir effectué les différents changements signalés, le testeur a été invité pour une présentation de la structure du logiciel et des fonctionnalités de l'API de *Blender*. Cette présentation a eu lieu, afin de permettre au testeur de prendre en main le logiciel créé pour ne pas perdre trop de temps à la compréhension du code effectué.

8.2. Procédure de tests

Après avoir vérifié que les fonctions du logiciel créées fonctionnaient correctement, des tests ont été réalisés pour savoir si les supports créés étaient applicables. Le but des tests ci-dessous est de vérifier la réalisation et la viabilité des supports et donc, pour des raisons de coûts et de prototypage rapide, les pièces ont été imprimées en plastique. Malheureusement, aucun test n'a pu être réalisé avec l'imprimante 3D métallique.

Pour réaliser ces tests, une pièce de test a été imprimé avec une imprimante 3D plastique ainsi que différents supports réalisés exclusivement à l'aide du logiciel créé. Les différents tests ont été réalisés avec la pièce suivante comme pièce de référence (*Figure 68*) :



Figure 68 : Rendu 3D du fichier STL de la pièce de base (a) et impression 3D plastique (b)

Pour le premier test, le support a été créé de la façon jugée la plus optimale. Cela signifie que l'orientation de la pièce a été réfléchie afin de diminuer le volume du support, car le support coûte aussi de l'argent. L'orientation trouvée est obtenue en tournant la pièce de 90° (*Figure 69a*) ce qui permet d'obtenir le support de la figure 69b pour un angle maximal des faces à soutenir de 80° .

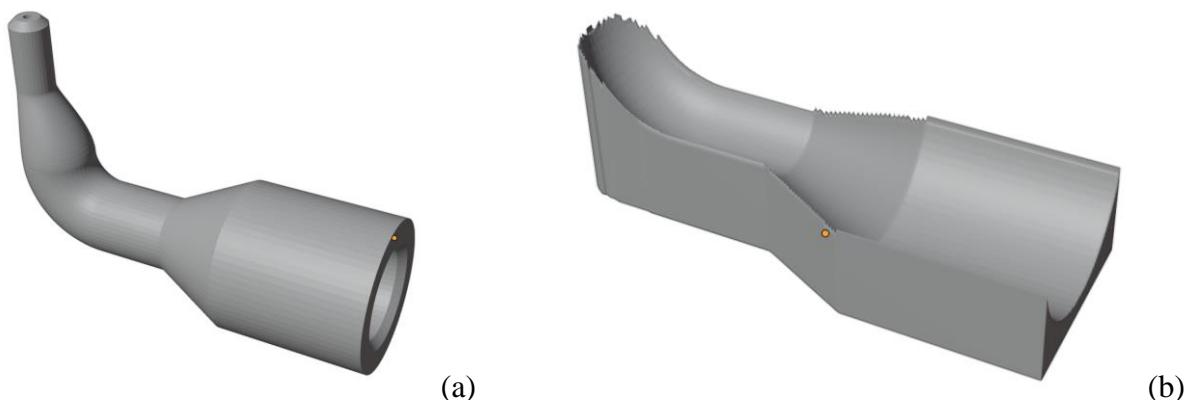


Figure 69 : Pièce de référence tournée de 90° (a) et support obtenu (b)

Comme le montre la figure 69a, dans cette position, la pièce possède des surfaces internes qu'il pourrait être intéressant de soutenir, pour éviter tout risque d'affaissement. Cependant, la méthode pour créer les supports ne permet pas de générer des supports pour ces faces dans cette orientation, car les supports créés par le logiciel doivent partir verticalement du plateau. Afin de soutenir les faces mentionnées, il faut créer un autre support en remettant la pièce en position initiale (*Figure 70a*). Après avoir généré les supports (*Figure 70b*), la seule partie intéressante est celle qui va à l'intérieur de la pièce et donc les parties inutiles sont supprimées (*Figure 70c*).

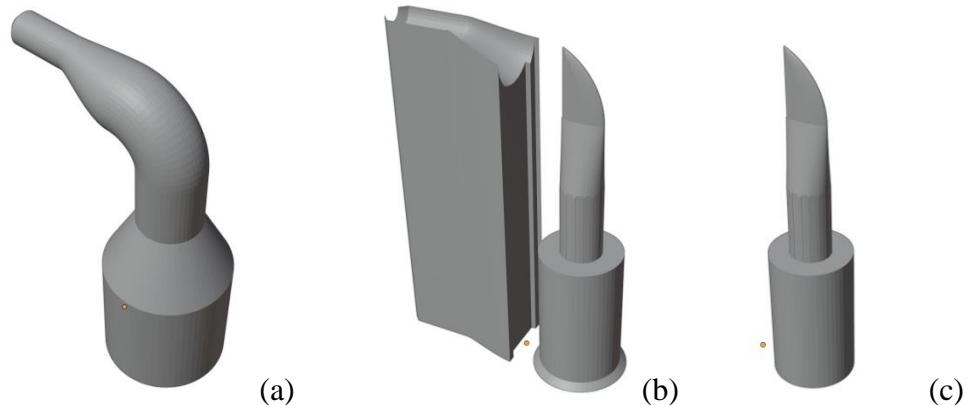


Figure 70 : Pièce sans rotation (a), supports générés (b) et partie interne utile (c)

Il faut ensuite redimensionner le deuxième support créé, étant donné qu'il faut laisser du jeu pour qu'il puisse rentrer dans la pièce. Pour ce test, deux supports ont été créés pour la partie interne ; le premier soutient toute la partie interne (*Figure 71a*), tandis que le deuxième ne soutient que la partie cylindrique (*Figure 71b*),

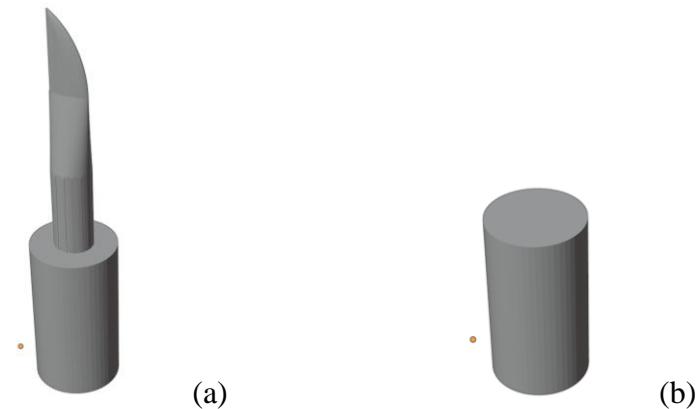


Figure 71 : Supports internes qui soutiennent toutes les faces (a) et la partie cylindrique (b)

Maintenant que tous les supports ont été générés, ils peuvent être imprimés. Il est ensuite possible de tester ces supports. Comme le montre la figure 72, la pièce est correctement soutenue par les supports. De plus, il est possible de rentrer les supports internes dans la pièce, ce qui veut dire que le redimensionnement a correctement fonctionné.

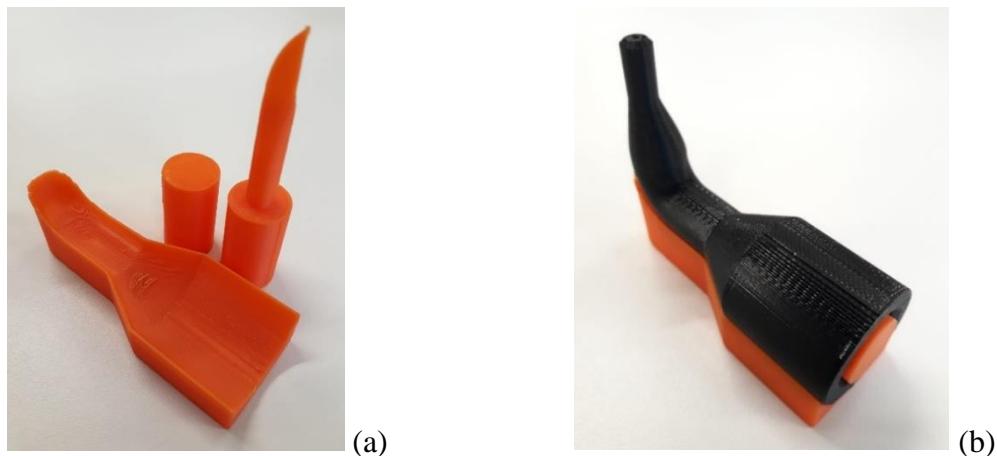


Figure 72 : Supports imprimés (a) et montage avec la pièce de référence (b)

Ce premier test montre que les supports créés peuvent correctement soutenir la pièce.

Après avoir testé la faisabilité des supports, il faut ensuite tester les autres fonctions disponibles pour la génération des supports.

Ce deuxième test va principalement expérimenter la fonctionnalité socle du logiciel. Lorsqu'aucune rotation n'est appliquée à la pièce de référence, deux blocs de supports sont créés pour un angle maximal de la sélection des faces de 85° (*Figure 73a*). Afin de relier ces deux blocs pour pouvoir facilement placer la pièce de référence, la fonction socle est utilisée. Comme le montre la figure 73b, les supports ne forment qu'un bloc après avoir généré le socle. Avant d'exporter le fichier pour l'impression, il faut encore redimensionner les parties internes pour laisser du jeu.

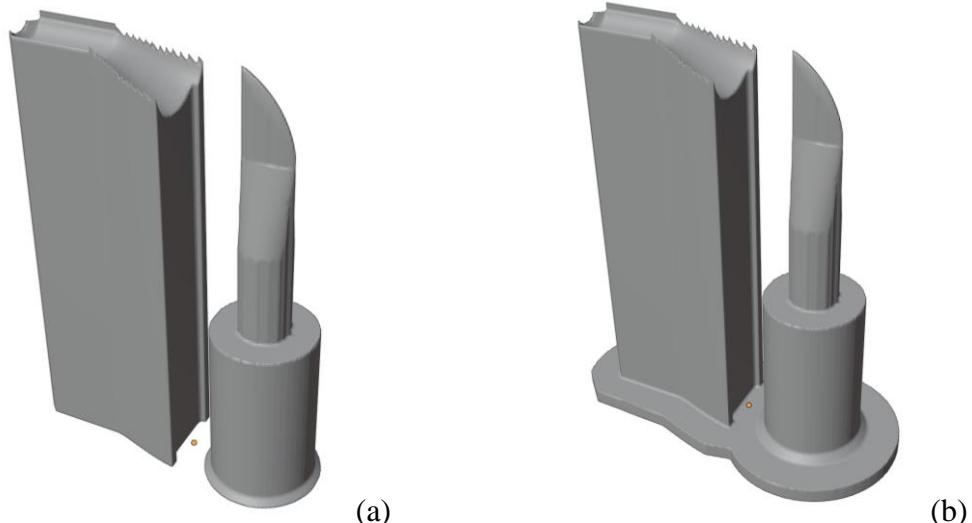


Figure 73 : Supports sans le socle (a) et avec le socle (b)

Les supports peuvent ensuite être imprimés pour être testés (*Figure 74a*). Comme le montre la figure 74b, la pièce de référence peut facilement être placée sur les supports tout en étant correctement soutenue par ceux-ci. Le socle remplit bien sa fonction et il n'y a pas de problème, même si certaines parties des supports ont été redimensionnées.

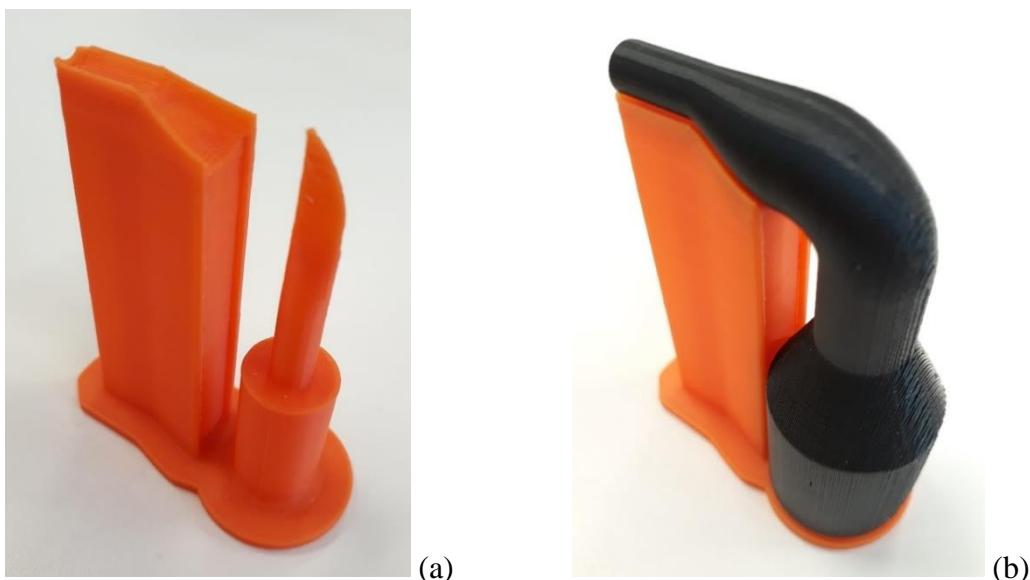


Figure 74 : Supports imprimés (a) et montage avec la pièce de référence (b)

Une autre fonctionnalité du logiciel qui a aussi été testée est la génération d'un moule. Un offset est appliqué à la pièce de référence sans rotation pour qu'elle soit entièrement moulée. Il faut aussi appliquer un redimensionnement pour que la pièce de référence puisse rentrer dans le moule. Ces différentes opérations donnent le support suivant (*Figure 75*) :



Figure 75 : Support généré (moule)

Après avoir imprimé ce support (*Figure 76a*), il faut tester s'il est possible de placer correctement la pièce de référence dans le moule. Comme le montre la figure 76b, il est en effet possible d'insérer la pièce ainsi que de la ressortir. Les figures 77a et b montrent une vue de dessus qui permet de bien observer comment la pièce de référence peut être insérée dans le moule. Pour le cas de cette pièce, le fait de créer un moule intégré n'est pas utile car, comme vu précédemment, il existe de meilleurs supports qui demandent moins de matière. Cependant, ce test montre qu'il est possible de réaliser un moule pour une pièce.

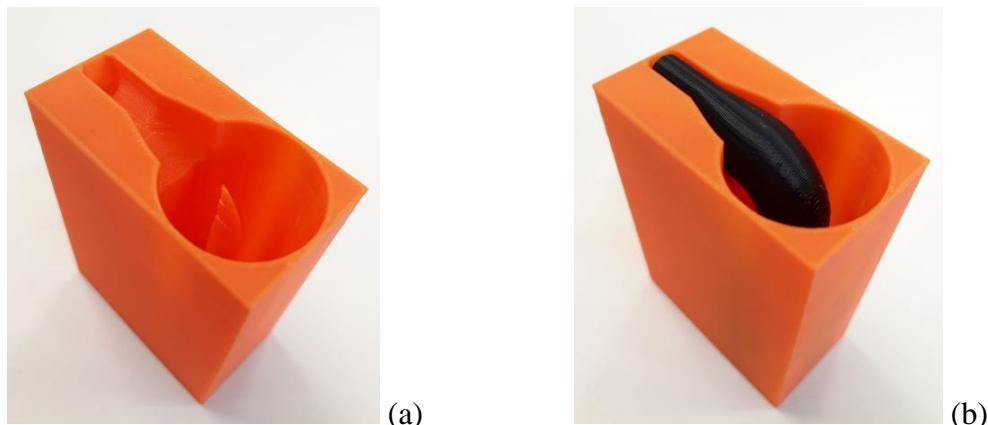


Figure 76 : Support imprimé (a) et montage avec la pièce de référence (b)

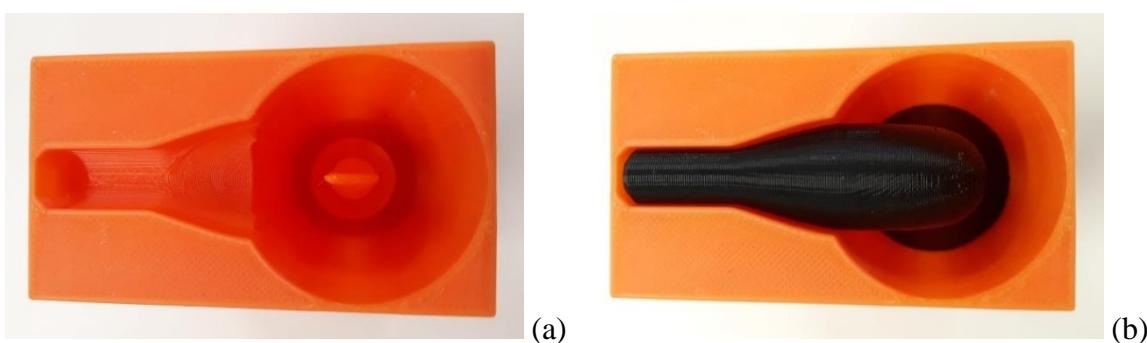


Figure 77 : Vue de dessus du support (a) et du montage (b)

Lors de l'utilisation de l'outil moule, il n'est pas obligatoire de mouler toute la pièce. En effet il est possible de mouler seulement une partie de la pièce. Il faut toujours redimensionner certaines parties du support pour pouvoir placer correctement la pièce de référence. Cela permet de mouler seulement ce qui est nécessaire ou d'avoir des rainures pour placer la pièce. Le moule obtenu est le suivant (*Figure 78*) :

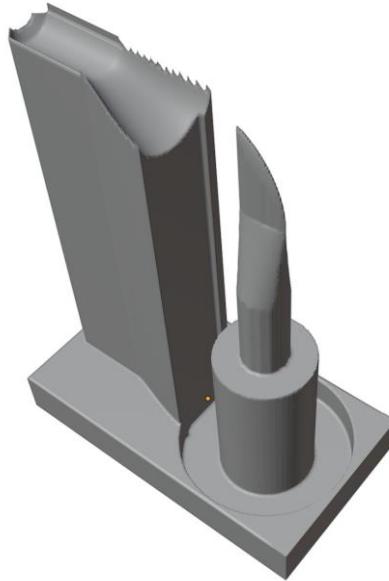


Figure 78 : Support généré (Rainure)

Après l'impression de ce support (*Figure 79a*), il faut essayer de placer la pièce de référence sur celui-ci. Comme le montre la figure *79b*, la pièce peut facilement être placée dans la rainure et elle est bien soutenue par le support.

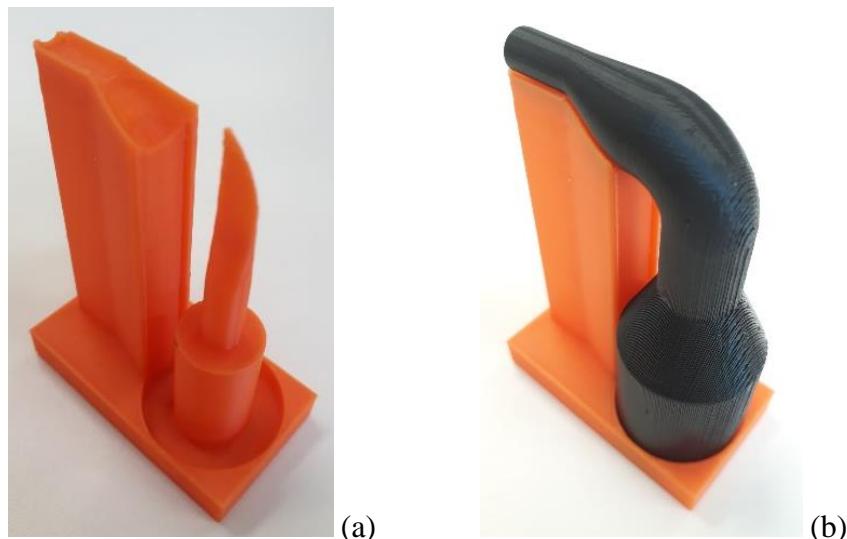


Figure 79 : Support imprimé (a) et montage avec la pièce de référence (b)

8.3. Résultats

Les tests précédents ont permis de valider le fonctionnement des différentes options développées, ainsi que la possible réalisation de supports avec ce logiciel. Cette partie montre les options créées que le logiciel est capable de réaliser pour générer les supports.

- Import/Export (Menu *Import/Export*)
 - Importer un fichier STL.
 - Exporter un fichier STL.
 - Communiquer des informations sur le volume et le nombre de faces de l'objet ainsi que la taille de son fichier.
- Prétraitement de l'objet (Menu *Rotation and offset*)
 - Modifier l'orientation de l'objet.
 - Régler l'offset vertical de l'objet.
- Génération de supports et options de génération (Menu *Génération et Area*)
 - Sélectionner les faces qui ont besoin de supports.
 - L'aire minimale permet de ne pas sélectionner les groupes de faces dont l'aire est trop petite.
 - Générer les supports à partir des faces sélectionnées.
 - Générer des supports classiques.
 - Générer des supports et un moule pour la partie de l'objet sous le plan xy.
 - Régénérer le fond des supports en cas de problème.
 - Générer un socle pour les supports.
- Post-traitement des supports (Menu *Resize, Lattice et Remesh*)
 - Sélectionner des faces.
 - Toutes les faces connectées à la sélection.
 - Toutes les faces connectées à la sélection et dont l'angle entre la normale et le vecteur descendant est compris entre les valeurs min et max.
 - Toutes les faces dont le centre est dans l'objet « Treillis » qui peut être ajouté et dont la position et la taille peuvent être modifiées.
 - Redimensionner des faces.
 - Combler un trou pour des faces coplanaires, supprimer les faces sélectionnées et inverser l'état des faces.
 - Remailler les supports avec des voxels.
 - Diminuer le nombre de faces des supports.
 - Remailler les supports avec des blocs.
- Outils de mesure (Menu *Measure*)
 - Mesurer la distance entre deux sommets des faces des supports.

Toutes les options du logiciel ont permis de générer les supports qui sont visibles sur les figures suivantes :

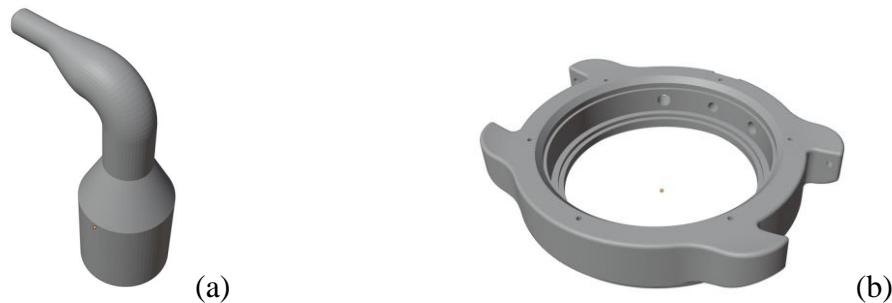


Figure 80 : Pièces d'exemple pour générer les supports

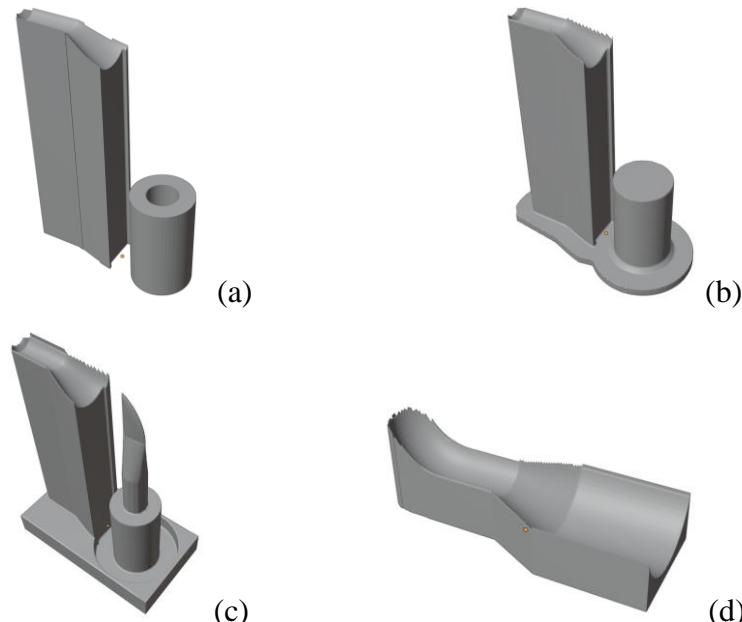


Figure 81 : Exemple de supports pour la pièce de la figure 80a

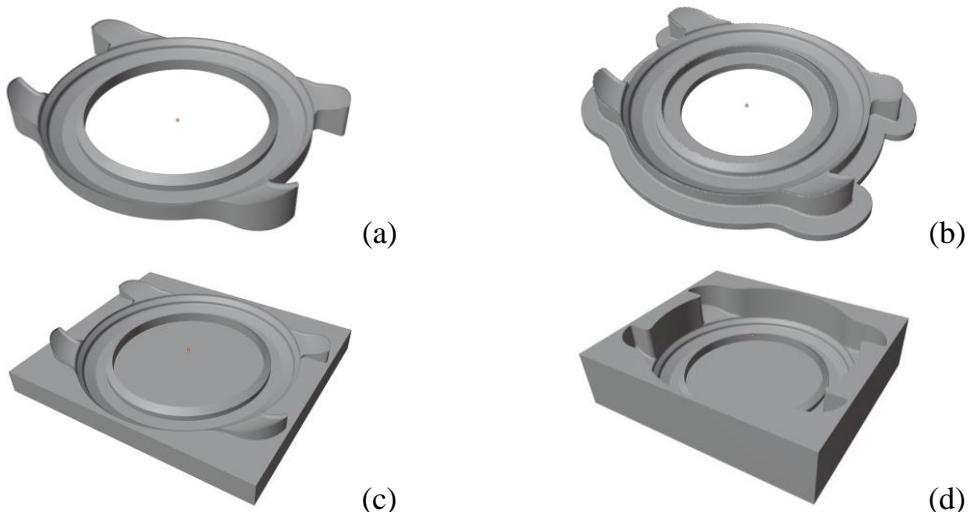


Figure 82 : Exemple de supports pour la pièce de la figure 80b

Les exemples ci-dessus montrent les possibilités de génération de différents supports pour une pièce donnée. Ces deux exemples montrent aussi que le logiciel est capable de générer des supports peu importe la forme de la pièce de base.

9. Conclusion

Le groupe technologie des poudres de la HES-SO Valais est spécialisé dans la technique d'impression 3D « Solvent on Granules ». Pour éviter que les pièces ne s'affaissent lors de l'opération de frittage, le groupe souhaite développer un logiciel capable de créer des supports adaptés aux pièces pour les soutenir.

Les points importants du logiciel développé sont de pouvoir importer un fichier STL d'une pièce, de lui créer des supports grâce à plusieurs options et d'exporter ces supports créés. Pour créer ce logiciel, il a été décidé de trouver un programme avec des fonctionnalités 3D qui possède une API pour pouvoir automatiser des scripts et développer un panel de contrôle. Le premier programme testé a été *Meshmixer*. Cependant, les résultats obtenus avec ce programme n'étaient pas satisfaisants et donc il a été décidé de trouver un autre programme 3D. Le choix s'est porté sur *Blender* avec lequel il a été possible de coder un script de génération de supports fonctionnels.

Grâce à l'API de *Blender*, il a été possible de développer un panel de contrôle pour la génération des supports qui contient plusieurs fonctionnalités tels que l'import/export, la rotation, la génération, le redimensionnement, etc.

Des tests ont été réalisés pour vérifier les fonctionnalités du logiciel et pour démontrer que les supports générés étaient réalisables et fonctionnels. Ces tests ont montré que le logiciel était capable de générer des supports pour la pièce demandée.

Evidemment, le logiciel peut encore être amélioré. Certes, les tests ont prouvé que les supports créés étaient réalisables, mais ils ont été réalisés avec une imprimante 3D plastique. Pour valider les supports, des tests devront être réalisés avec l'imprimante 3D métallique pour confirmer qu'il n'y ait aucun problème.

Par ailleurs, d'autres fonctions peuvent être développées pour améliorer la génération des supports et augmenter le nombre d'options pour les créer. Par exemple, il pourrait être intéressant d'avoir un outil de placement automatique de la pièce qui chercherait son orientation pour obtenir les supports optimaux. Les recherches déjà effectuées sur le sujet peuvent être un bon point de départ [4].

Il serait aussi intéressant de développer d'autres méthodes pour générer les supports comme celle des pointes qui a déjà été un peu expérimentée (point 7.10.2) ou des méthodes qui créent des lignes ou des cônes. Le logiciel *Magics* de *Materialise* possède ces méthodes pour la génération des supports. Le fait d'avoir plusieurs méthodes permettrait d'avoir plus de possibilités pour créer des supports qui s'adaptent le mieux à la forme et cela permettrait aussi de faire des tests de ces différentes méthodes pour savoir laquelle est la plus efficace, économique ou pratique.

Finalement, ce travail a permis de prouver qu'il est possible de créer des supports simples et efficaces en développant du code sur un programme gratuit et open source. L'utilisation de ce logiciel permet à l'utilisateur d'avoir plusieurs options, afin de créer facilement et rapidement un fichier STL de supports à partir d'un fichier STL d'une pièce de base.

10. Date et signature

Sion, le 20 août 2021

Gaëtan Fumeaux

11. Bibliographie

- [1] E. Carreño-Morelli, P. Alveen, S. Moseley, M. Rodriguez-Arbaizar, K. Cardoso, A Comparative Study of Cemented Carbide Parts Produced by Solvent on Granules 3D-Printing (SG-3DP) Versus Press and Sinter
- [2] HES-SO Valais-Wallis, Institut Systèmes Industriels, <https://www.hevs.ch/fr/rad-instituts/institut-systemes-industriels/projets/impression-3d-2656> (Consulté le 4 juin 2021)
- [3] Python API documentation for Blender, the free and open source 3D creation suite, <https://docs.blender.org/api/current/index.html> (Consulté le 08 août 2021)
- [4] Better 3D Print Positioning: Free Open Source Software, <https://www.salzburgresearch.at/blog/3d-print-positioning/> (Consulté le 08 août 2021)

12. Annexes

Annexe 1 : Documentation technique

13. Liste des figures et tableaux

| | |
|--|----|
| Figure 1 : Etapes de fabrication d'une pièce 3D | 1 |
| Figure 2 : Principe de l'impression 3D « Solvent on Granules » (SG-3DP) | 3 |
| Figure 3 : Paramètres d'un triangle STL (a) et tessellation d'une sphère (b)..... | 4 |
| Figure 4 : Etapes du procédé d'impression 3D SLM - crédit Materialise | 6 |
| Figure 5 : Etapes de préparation du fichier 3D avec Magics | 7 |
| Figure 6 : Comparaison entre le « corps vert » et la pièce frittée..... | 8 |
| Figure 7 : « Use case » du logiciel | 9 |
| Figure 8 : Diagramme physique du logiciel | 9 |
| Figure 9 : Diagramme d'état de la génération des supports avec l'outil « Overhangs » de Meshmixer..... | 10 |
| Figure 10 : Résultats de l'outil support de Meshmixer avec une forme simple (a) et une forme complexe (b) | 11 |
| Figure 11 : Diagramme d'état de la génération des supports à l'aide d'une forme simple (a) et son résultat (b) | 11 |
| Figure 12 : Diagramme d'état de la génération automatique des supports pour Blender | 12 |
| Figure 13 : Objet de base (a) et exemples de supports générés avec Blender (b)..... | 12 |
| Figure 14 : Impression des supports (a) et résultat de l'impression (b) | 13 |
| Figure 15 : Pièce de base imprimée (a) et pièce de base soutenue par les supports (b)..... | 13 |
| Figure 16 : Diagramme de classe du logiciel | 16 |
| Figure 17 : Graphique de la relation entre la taille d'un fichier 3D et de son nombre de faces | 17 |
| Figure 18 : Axes globaux et locaux sans rotation (a) et avec une rotation de 45° en z (b)..... | 18 |
| Figure 19 : Principe des faces qui ont besoin de supports en fonction de l'angle | 20 |
| Figure 20 : Vue de la caméra après le zoom (a) et Sélection des faces visibles (b)..... | 20 |
| Figure 21 : Sélection avec la méthode de la caméra vu de côté (a) et vu de dessus (b)..... | 21 |
| Figure 22 : Diagramme d'état de la méthode du calcul des faces | 21 |
| Figure 23 : Schéma pour déterminer si un point est dans les demi-plans AB et AC | 23 |
| Figure 24 : Schéma pour déterminer si un point est dans le triangle ABC | 23 |
| Figure 25 : Problèmes de la méthode du calcul des faces | 24 |
| Figure 26 : Projection orthogonale de a sur b | 25 |

| | |
|---|----|
| Figure 27 : Distance perpendiculaire du point P au plan formé par le triangle ABC | 25 |
| Figure 28 : Cas pour déterminer quand le point P est dessus la face | 26 |
| Figure 29 : Diagramme d'état de la version modifiée du calcul des faces | 26 |
| Figure 30 : Boucles d'arêtes (a) et création des faces avec l'outil « Bridge Edge Loop » (b) .. | 27 |
| Figure 31 : Diagramme d'état de l'automatisation de la génération des supports | 27 |
| Figure 32 : Objet de base (a) et supports créés avec le calcul des faces avec un angle max de 30° (b), 60° (c), 90° (d) | 28 |
| Figure 33 : Diagramme d'état de la première méthode pour générer des moules | 29 |
| Figure 34 : Moules générés avec la première méthode | 29 |
| Figure 35 : Erreurs lors de la génération du moule à cause du changement de la sélection des faces | 30 |
| Figure 36 : Diagramme d'état de la deuxième méthode pour générer des moules | 30 |
| Figure 37 : Moules générés avec la deuxième méthode | 31 |
| Figure 38 : Moule généré pour créer une rainure afin de guider la pièce | 31 |
| Figure 39 : Diagramme d'état de la méthode de l'aire minimale | 32 |
| Figure 40 : Groupes de faces qui ont besoin de supports avant le calcul de l'aire minimale .. | 33 |
| Figure 41 : Supports générés sans l'aire minimale (a) et avec l'aire minimale (b) | 33 |
| Figure 42 : Exemple de supports sans le fond (a) et avec le fond généré avec le « edge_face_add() » (b) | 34 |
| Figure 43 : Autre exemple de supports sans le fond (a) et avec le fond généré avec le « edge_face_add() » (b) | 34 |
| Figure 44 : Supports qui sont intersectés avec l'objet « Plan » | 34 |
| Figure 45 : Fond généré avec l'objet de type « Plan » | 35 |
| Figure 46 : Fond généré avec l'objet de type « Plan » sans les faces non-nécessaires | 35 |
| Figure 47 : Création du socle en ajoutant un parallélépipède rectangle | 36 |
| Figure 48 : Création de socles avec la méthode resize() | 36 |
| Figure 49 : Socle avant d'extruder les faces verticales le long des vecteurs normaux | 37 |
| Figure 50 : Création de socles avec l'extrusion des faces le long des vecteurs normaux | 37 |
| Figure 51 : Diagramme d'état de la sélection de toutes les faces connectées | 38 |
| Figure 52 : Sélection de base (a), après 2 passages dans la boucle (b), après 5 passages (c) et à la fin (d) | 39 |
| Figure 53 : Diagramme d'état de la sélection des faces connectées dont l'angle est compris entre des valeurs min et max | 39 |
| Figure 54 : Sélection de base (a), après 2 passages dans la boucle (b), à la fin (c) et après l'affichage des faces cachées (d) | 40 |
| Figure 55 : Ajout d'un « Treillis » (a), modification de la taille et la position du « Treillis » (b), sélection des faces dans le « Treillis » et suppression du « Treillis » (d) | 41 |
| Figure 56 : Principe du redimensionnement vu de dessus | 41 |
| Figure 57 : Diagramme d'état du principe du redimensionnement | 42 |
| Figure 58 : Fonction pour remplir les trous des faces sélectionnées sur un même plan | 43 |
| Figure 59 : Sélection (a), inversion de la sélection (b) et suppression de la sélection (c) | 43 |
| Figure 60 : Exemple du problème des faces internes | 44 |
| Figure 61 : Suppression des faces internes à l'aide des voxels | 44 |
| Figure 62 : Supports avant le modificateur blocs (a), après le modificateur blocs (b) | 45 |
| Figure 63 : Sélection de faces (a), première bordure de la sélection (b) et deuxième bordure de la sélection (c) | 46 |
| Figure 64 : Redimensionnement de la première bordure (a) et de la deuxième (b) | 46 |
| Figure 65 : Faces qui ont besoin de supports (a) et supports créés avec la méthode des pointes à partir de la sélection (b) | 47 |

| | |
|---|----|
| Figure 66 : Différence entre un slider (haut) et un bouton (bas) avant le changement (a) et après (b) | 49 |
| Figure 67 : Différence entre un bouton qui peut être appuyé (haut) et un bouton grisé (bas) . | 49 |
| Figure 68 : Rendu 3D du fichier STL de la pièce de base (a) et impression 3D plastique (b). . | 50 |
| Figure 69 : Pièce de référence tournée de 90° (a) et support obtenu (b)..... | 50 |
| Figure 70 : Pièce sans rotation (a), supports générés (b) et partie interne utile (c)..... | 51 |
| Figure 71 : Supports internes qui soutiennent toutes les faces (a) et la partie cylindrique (b). . | 51 |
| Figure 72 : Supports imprimés (a) et montage avec la pièce de référence (b) | 51 |
| Figure 73 : Supports sans le socle (a) et avec le socle (b)..... | 52 |
| Figure 74 : Supports imprimés (a) et montage avec la pièce de référence (b) | 52 |
| Figure 75 : Support généré (moule) | 53 |
| Figure 76 : Support imprimé (a) et montage avec la pièce de référence (b) | 53 |
| Figure 77 : Vue de dessus du support (a) et du montage (b)..... | 53 |
| Figure 78 : Support généré (Rainure)..... | 54 |
| Figure 79 : Support imprimé (a) et montage avec la pièce de référence (b) | 54 |
| Figure 80 : Pièces d'exemple pour générer les supports..... | 56 |
| Figure 81 : Exemple de supports pour la pièce de la figure 80a | 56 |
| Figure 82 : Exemple de supports pour la pièce de la figure 80b | 56 |
| Tableau 1 : Données du rapport entre le nombre de faces et la taille du fichier | 17 |