

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* Mikel Rodriguez Arbaizar |
|  | *Date de la remise du rapport* 20.08.2021 |

Rawyl1_RVBRawyl1_RVBRawyl1_RVB



Table des matières

[Introduction 1](#_Toc77863517)

[Contexte 1](#_Toc77863518)

[Objectifs 1](#_Toc77863519)

[Structure 2](#_Toc77863520)

[Cahier des charges 2](#_Toc77863521)

[Analyse (Changer le nom du titre peut-être) 3](#_Toc77863522)

[Connaissances (Changer le nom du titre peut-être) 3](#_Toc77863523)

[Solvent on Granule 3D Printing (SG-3DP) (Passer en titre 2) 4](#_Toc77863524)

[Principe (Supprimer) 4](#_Toc77863525)

[File 3D 5](#_Toc77863526)

[Génération de support 6](#_Toc77863527)

[Conception (Lien logiciel utilisé et l’api) 7](#_Toc77863528)

[Meshmixer 8](#_Toc77863529)

[Tests 8](#_Toc77863530)

[Résultats 9](#_Toc77863531)

[Blender 10](#_Toc77863532)

[Tests 10](#_Toc77863533)

[Résultat 11](#_Toc77863534)

[Logiciel 12](#_Toc77863535)

[Import/Export 14](#_Toc77863536)

[Unités et volume 14](#_Toc77863537)

[Import/Export 14](#_Toc77863538)

[Rotations et offset 15](#_Toc77863539)

[Génération 17](#_Toc77863540)

[Sélection des faces 17](#_Toc77863541)

[Génération des supports 24](#_Toc77863542)

[Options de génération 26](#_Toc77863543)

[Aire minimum 26](#_Toc77863544)

[Génération du fond 27](#_Toc77863545)

[Génération d’un socle 27](#_Toc77863546)

[Sélection et redimensionnement 28](#_Toc77863547)

[Validation 29](#_Toc77863548)

[Conclusion 29](#_Toc77863549)

[Bibliographie 29](#_Toc77863550)

# Introduction

## 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 traditionnelle, 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, vu 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.

## 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 a 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.

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 imprimé.

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

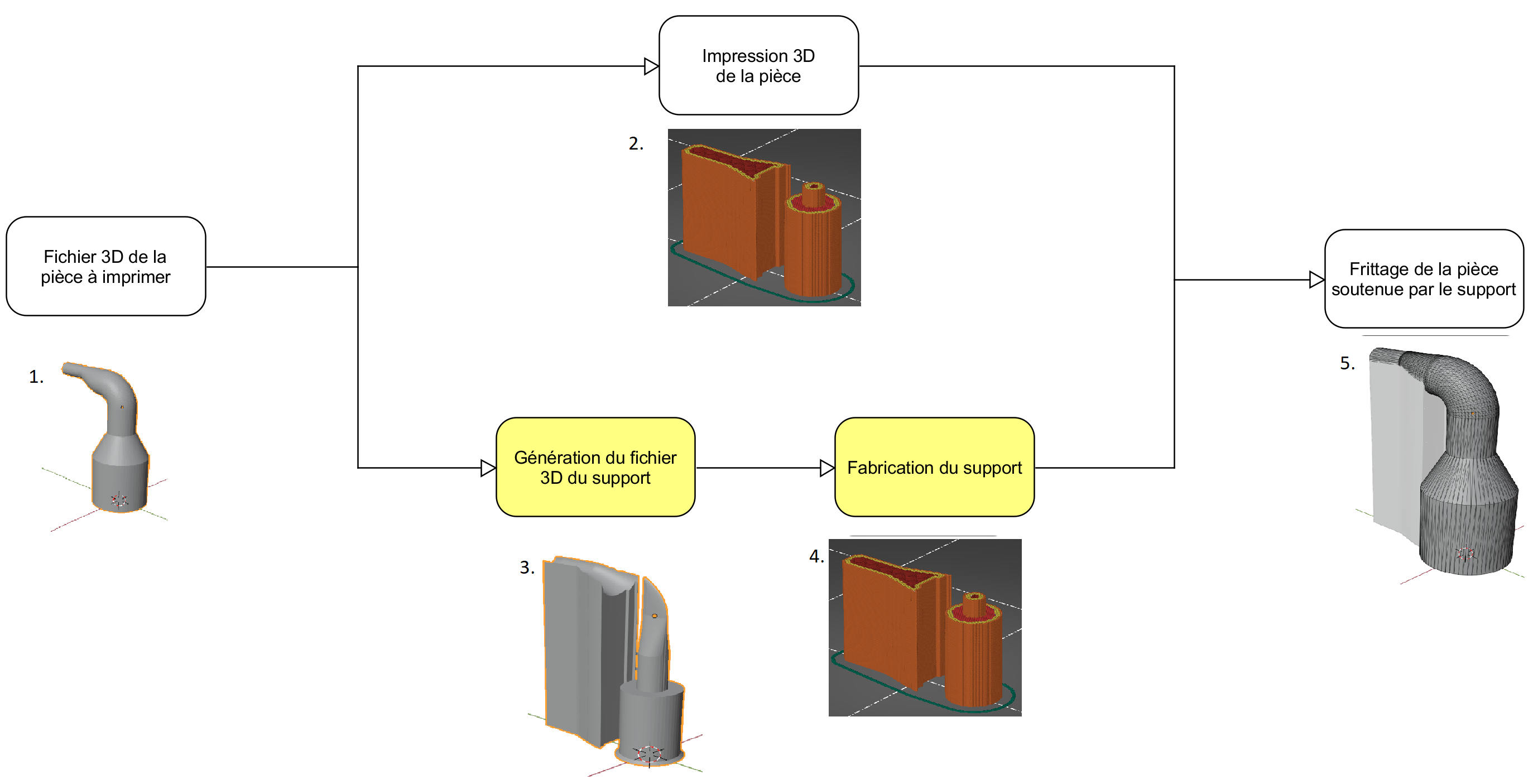


Figure 1 : Etapes de fabrication d'une pièce 3D

1. Pour imprimer une pièce, il faut d’abord avoir un fichier 3D de la pièce à imprimer.
2. Ce fichier 3D est ensuite converti en instructions pour l’imprimante, puis imprimé en 3D avec la méthode « Solvent on Granules ».
3. Parallèlement, le fichier 3D est aussi envoyé au logiciel créé pour ce Bachelor qui va générer un fichier 3D des supports pour l’objet.
4. Le support est ensuite fabriqué.
5. La pièce est finalement mise dans le four de frittage avec le support pour éviter que la pièce ne s’affaisse.

(Plus développé chaque étape, montré un Exemple final en .6, voir image pièce affaissé et demandé plus d’info dessus)

## Structure

Ce rapport va d’abord expliquer le fonctionnement de la « Solvent on Granules » avec son principe, le format des fichiers 3D et l’utilité des supports. Puis, ce rapport abordera les logiciels utilisés pour générer les supports.

# Cahier des charges

Les objectifs du cahier des charges 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

# Analyse (Changer le nom du titre peut-être)

## Connaissances (Changer le nom du titre peut-être)

(Demander l’état actuel des connaissances à la partie mécanique)

(Parler de Materialise, comparer ce que le logiciel doit remplacer, avant/après)

# Solvent on Granule 3D Printing (SG-3DP) (Passer en titre 2)

## Principe (Supprimer)

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 consolidée par déliantage et frittage.

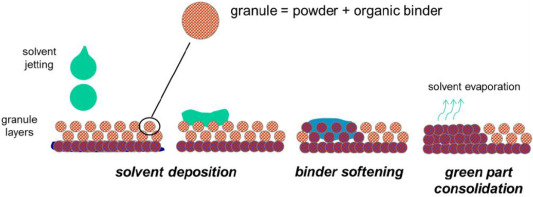


Figure 2 : Principe de l'impression 3D par solvant sur granules (SG-3DP)  
Source : https://www.researchgate.net/publication/349553916\_A\_Comparative\_Study\_of\_Cemented\_Carbide\_Parts  
\_Produced\_by\_Solvent\_on\_Granules\_3D-Printing\_SG-3DP\_Versus\_Press\_and\_Sinter

Cette méthode donne plus de flexibilité pour le choix des poudres et liant 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.

## File 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 multitudes de plan 2D. Ces informations sont ensuite transmises et traduites, si besoin, en instructions pour l’imprimante.

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 paver 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).

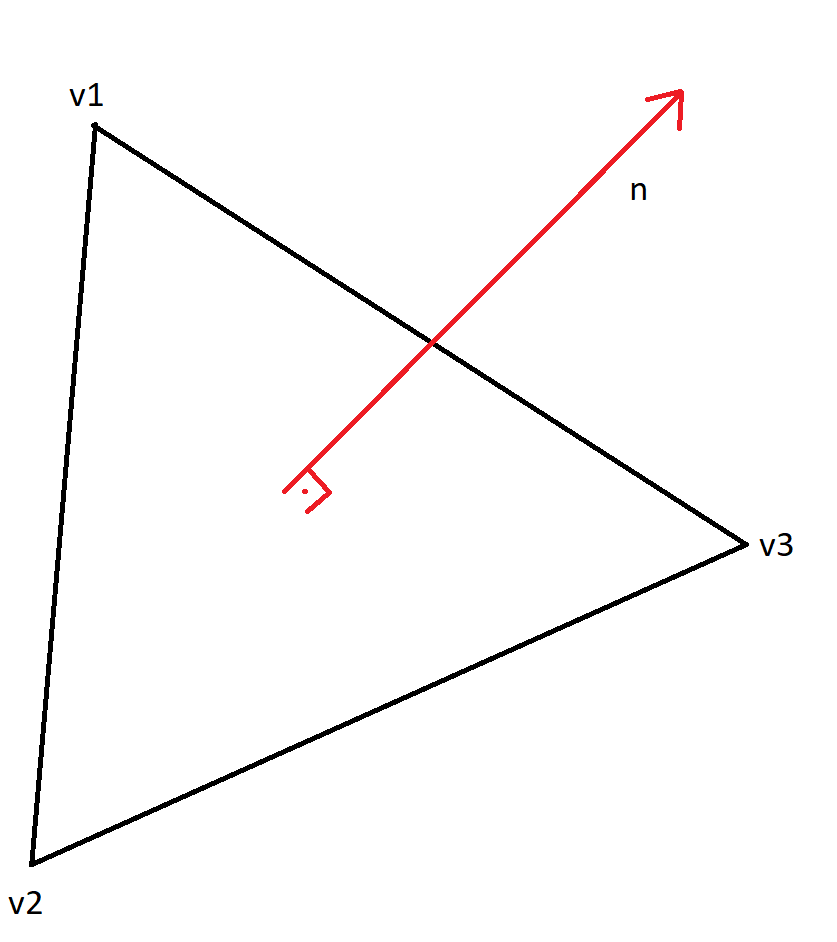
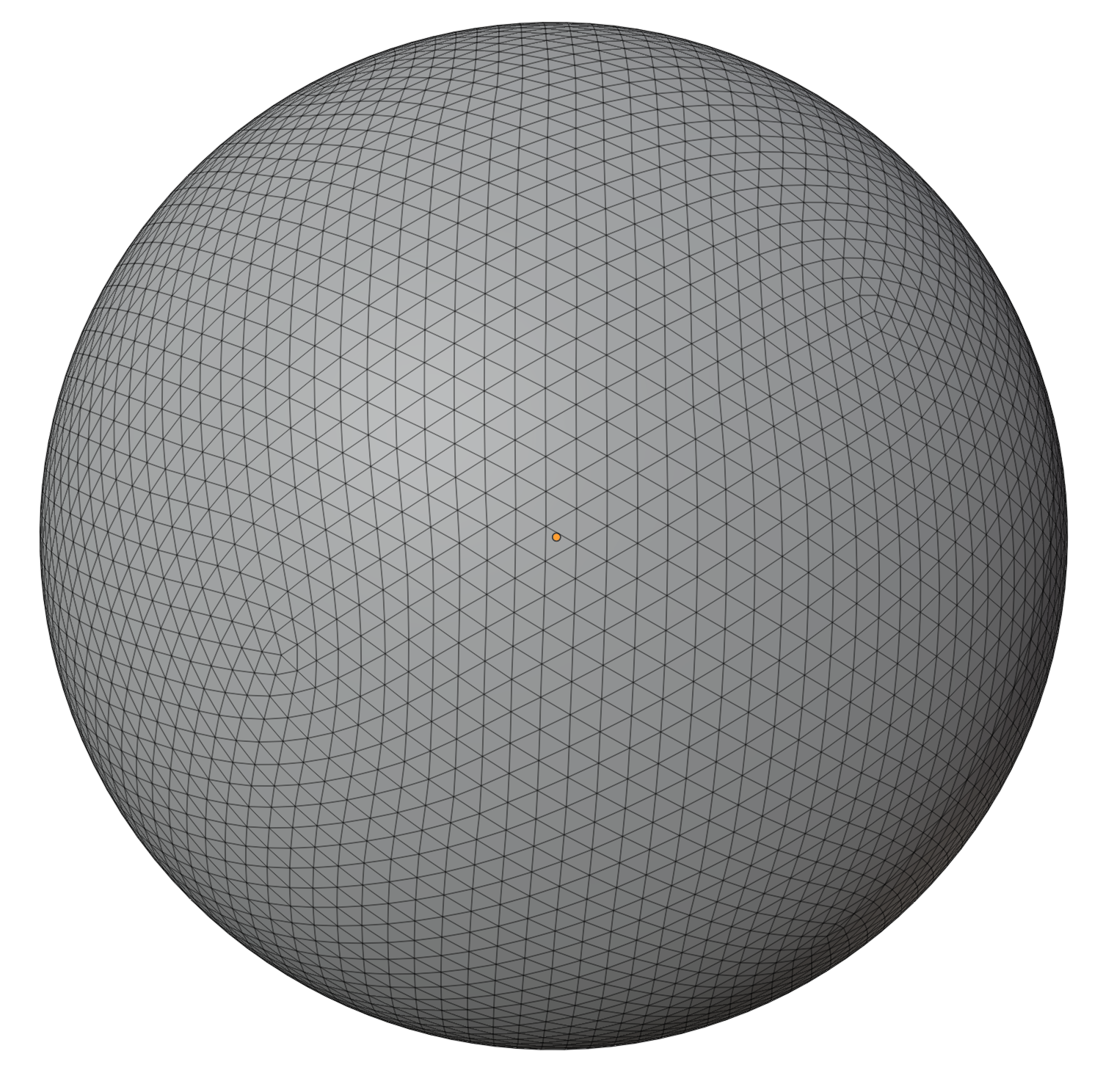
 a  b

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

Il existe de 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
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
11. // Others triangles
13. endsolid name

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

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 plus 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
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

## Génération de support

(Parler des probleme qui peut y avoir pour la génération smooth faisabilité complexité, type de support)

La forme des supports va varier fortement selon l’orientation et la forme de la pièce. Une autre contrainte à prendre en compte est le facteur de rétrécissement de la pièce lors du frittage (Figure 3). En effet, le support généré doit éviter d’imposer des contraintes à la pièce lors du frittage pour éviter des déformations ou la destruction de cette dernière.

Une image contenant texte, matériau de construction

Description générée automatiquement

Figure 4 : 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

L’algorithme de génération des supports doit être capable de créer des supports plutôt simples et robustes en fonction des objets. Il prend en entrée un fichier STL de l’objet qui a besoin de support et doit ressortir un fichier STL des supports générés (Figure 4).

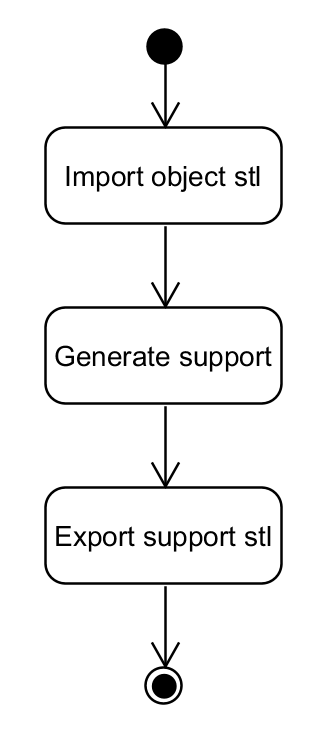


Figure 5 : Diagramme d'état de l'algorithme

# Conception (Lien logiciel utilisé et l’api)

# 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’objet 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 support et que ce soit 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.

## Tests

Afin de générer des supports, il a d’abord fallu vérifier que l’API était capable de réaliser les actions nécessaires à l’algorithme (Figure 4). Puis, il a fallu regarder comment générer les supports.

Un premier script a été réalisé en utilisant l’outil « Overhangs » de Meshmixer (Figure 5).

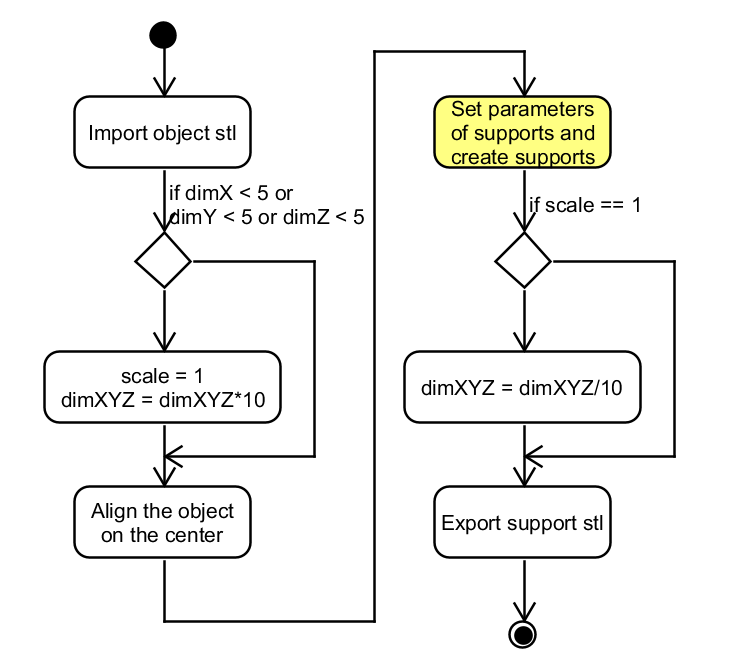


Figure 6 : Diagramme d'état de la génération des supports avec l'outil « Overhangs » de Meshmixer

Ce script importe le fichier STL de l’objet. Puis, il agrandit l’objet si ses dimensions sont trop petites car sinon l’outil « Overhangs » ne fonctionne pas. Ensuite, l’objet est placé au centre et les supports sont générés. Finalement, les dimensions du support est remis à l’échelle si besoin et le support est exporté 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 utilise 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. Ce qui est un avantage pour l’impression 3D plastique n’est pas le cas pour les supports de frittage. En effet, vu que les supports de frittage ne sont pas reliés à la pièce, ils doivent être stables et simples afin de pouvoir être imprimer sans support. Malgré des supports qui pourraient fonctionner pour des formes simples (Figure 6a), il y a plusieurs cas où l’utilisation de tree supports ne convient pas, peu importe les réglages des supports (Figure 6b).

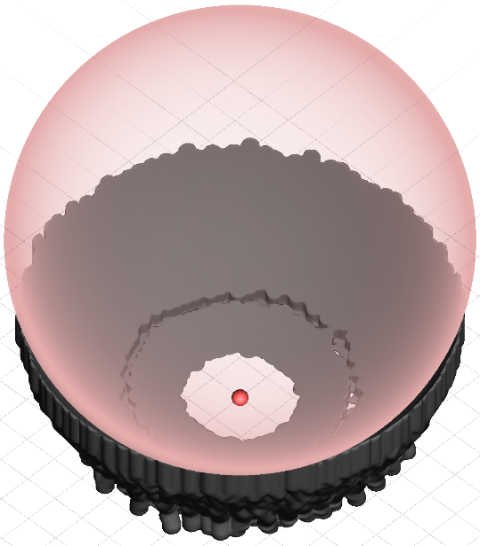
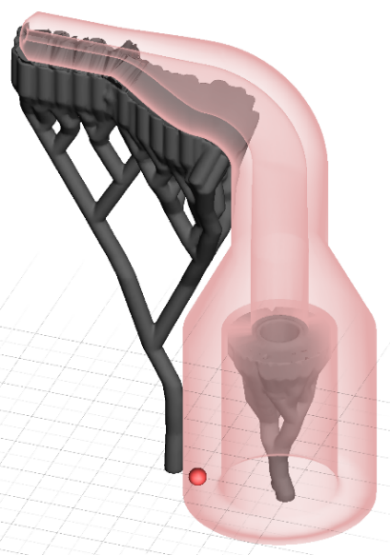
a  b

Figure 7 : 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 (Figure 7a). Ce script importe un objet, en agrandissant son échelle si ses dimensions sont trop petites et permet à l’utilisateur de tourner l’objet. 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 soustraie l’objet du support qui est ensuite remis à l’échelle si besoins et exporté en fichier STL.

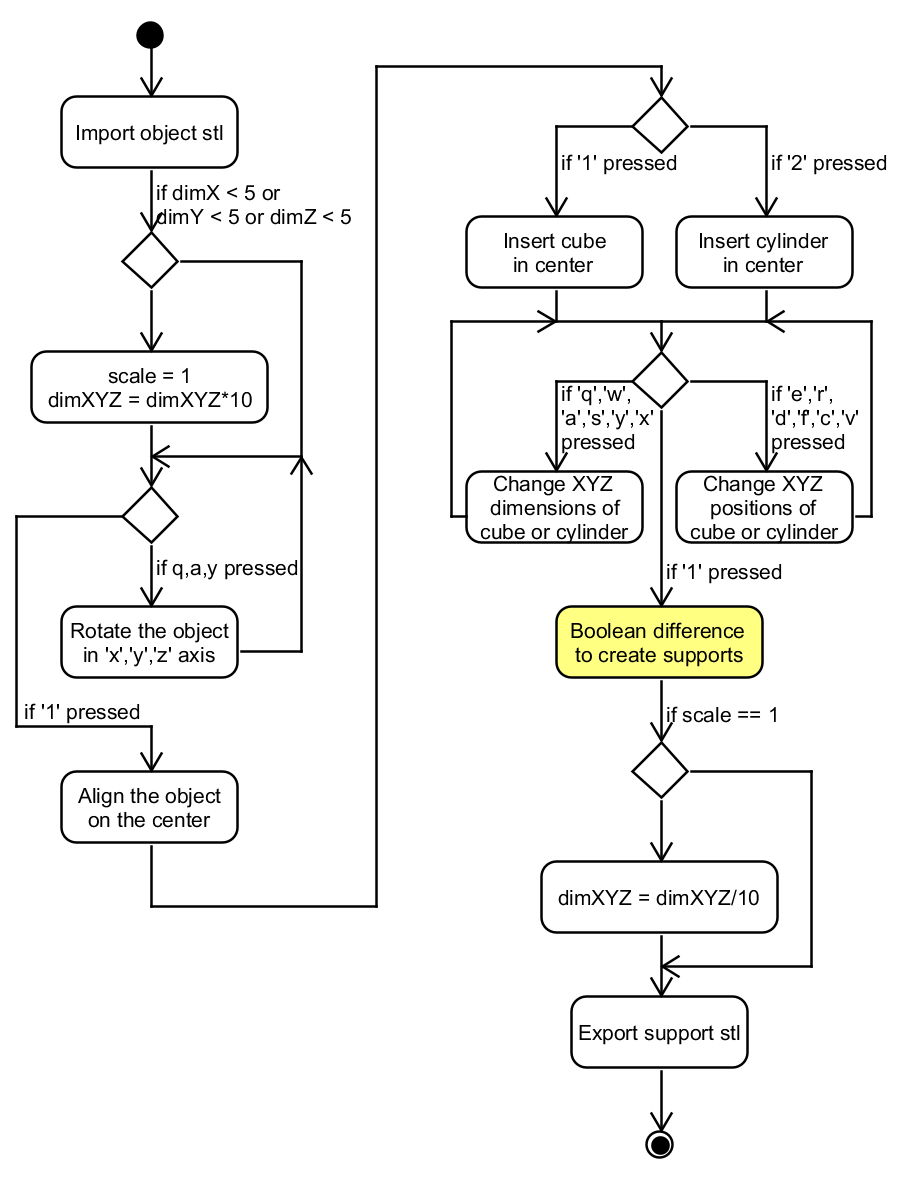
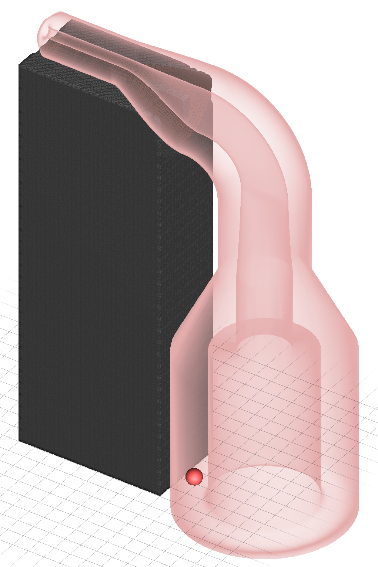
 a 

Figure 8 : 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 7b). Néanmoins, vu que 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.

## 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 pour générer les supports de frittage.

# 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, voir même montage vidéo et création de jeux (<https://www.blender.org/about/>, The Software).

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

## Tests

Blender possède de nombreuses fonctionnalités qui permettent la création de supports. Le script doit suivre les actions de la figure 4 ce qui, sous Blender, donne le diagramme d’état suivant (Figure 8) :

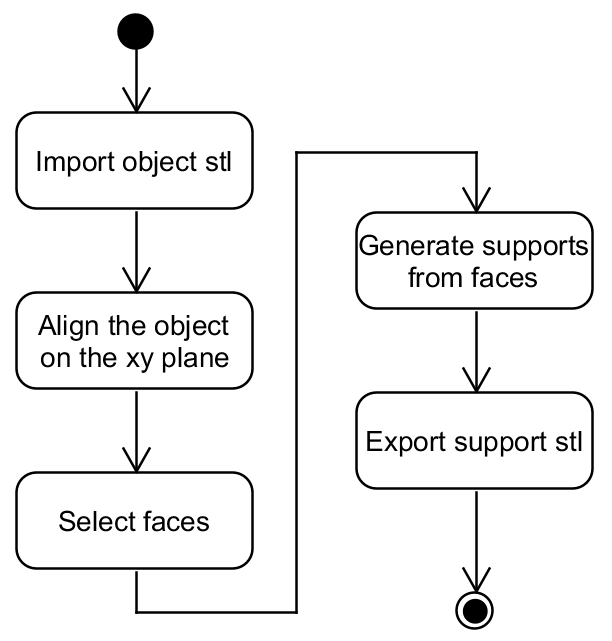


Figure 9 : Diagramme d'état de l'automatisation de la génération des supports pour Blender

Le script importe le fichier STL de l’objet. L’objet va ensuite être placer sur le plan xy. Le programme va ensuite sélectionner les faces à supporter. Ensuite, le script va générer des supports à partir des faces sélectionner. Cela permet finalement d’obtenir le support qui peut être exporté en fichier STL.

Ce script permet de générer des supports assez simples et qui peuvent s’adapter à différentes formes.

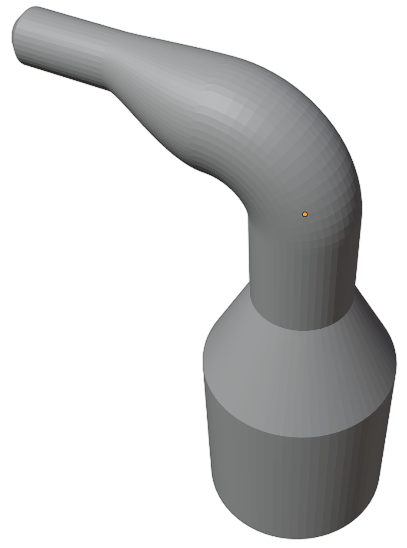
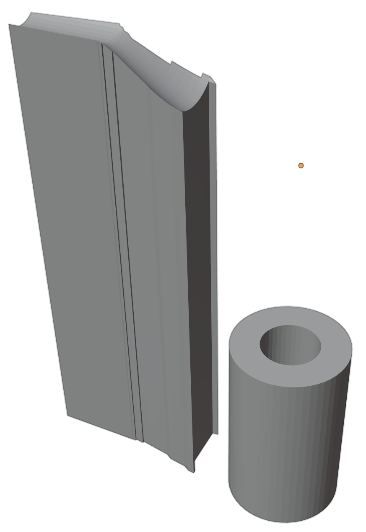
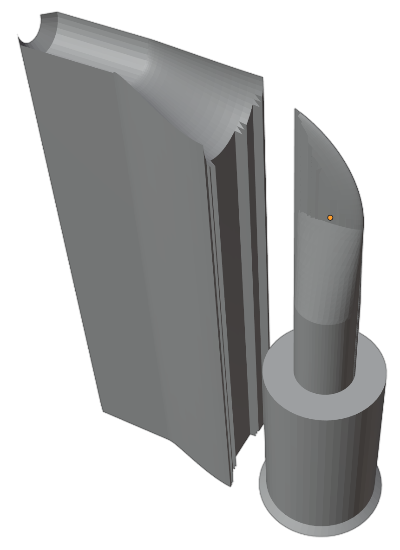
 a  b

Figure 10 : Objet de base (a) et des exemples de support générer avec Blender

## Résultat

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 été imprimé avec (NOM DE L’IMPRIMANTE) (Figure 12a), tandis que les supports ont été imprimés avec une Zortrax M200 (Figure 11).

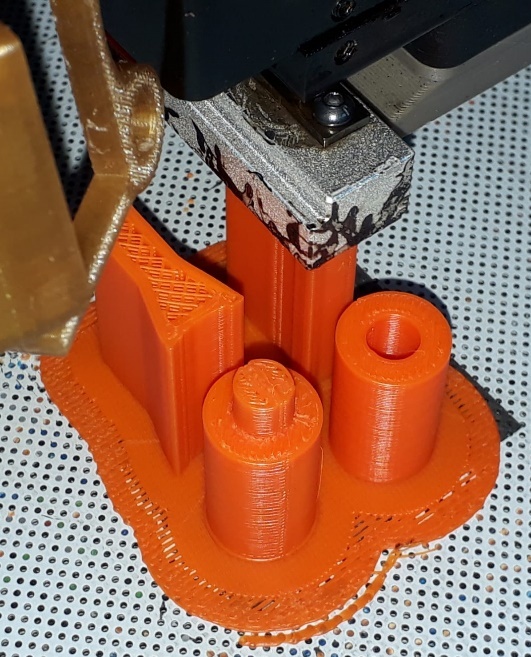
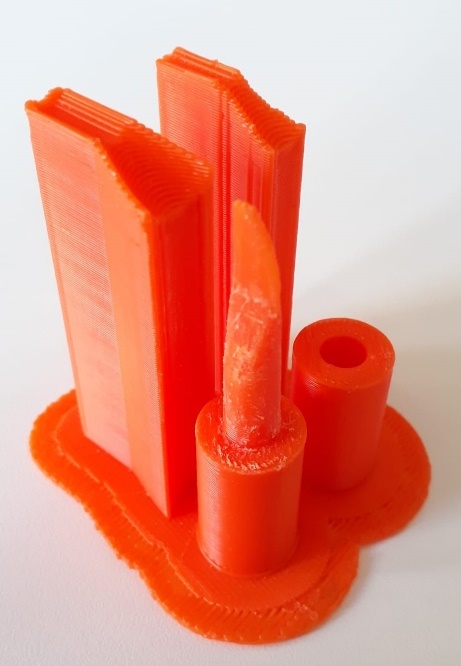
 a  b

Figure 11 : Impression des supports (a) et le résultat final (b)

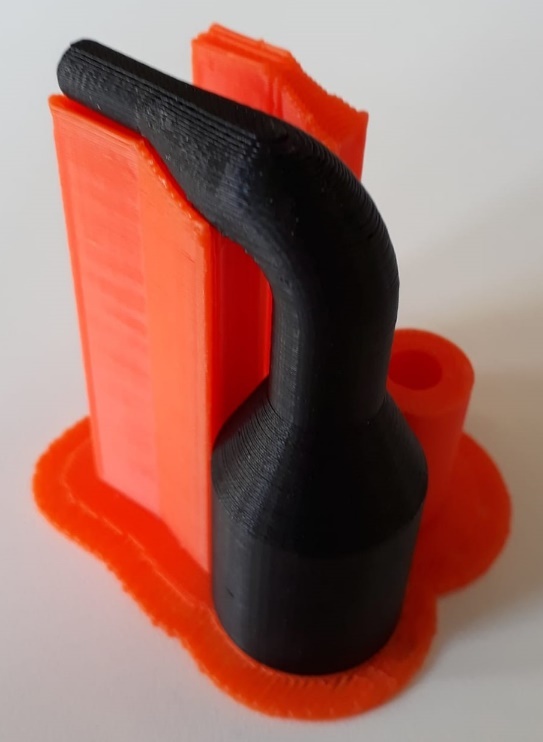
 a  b

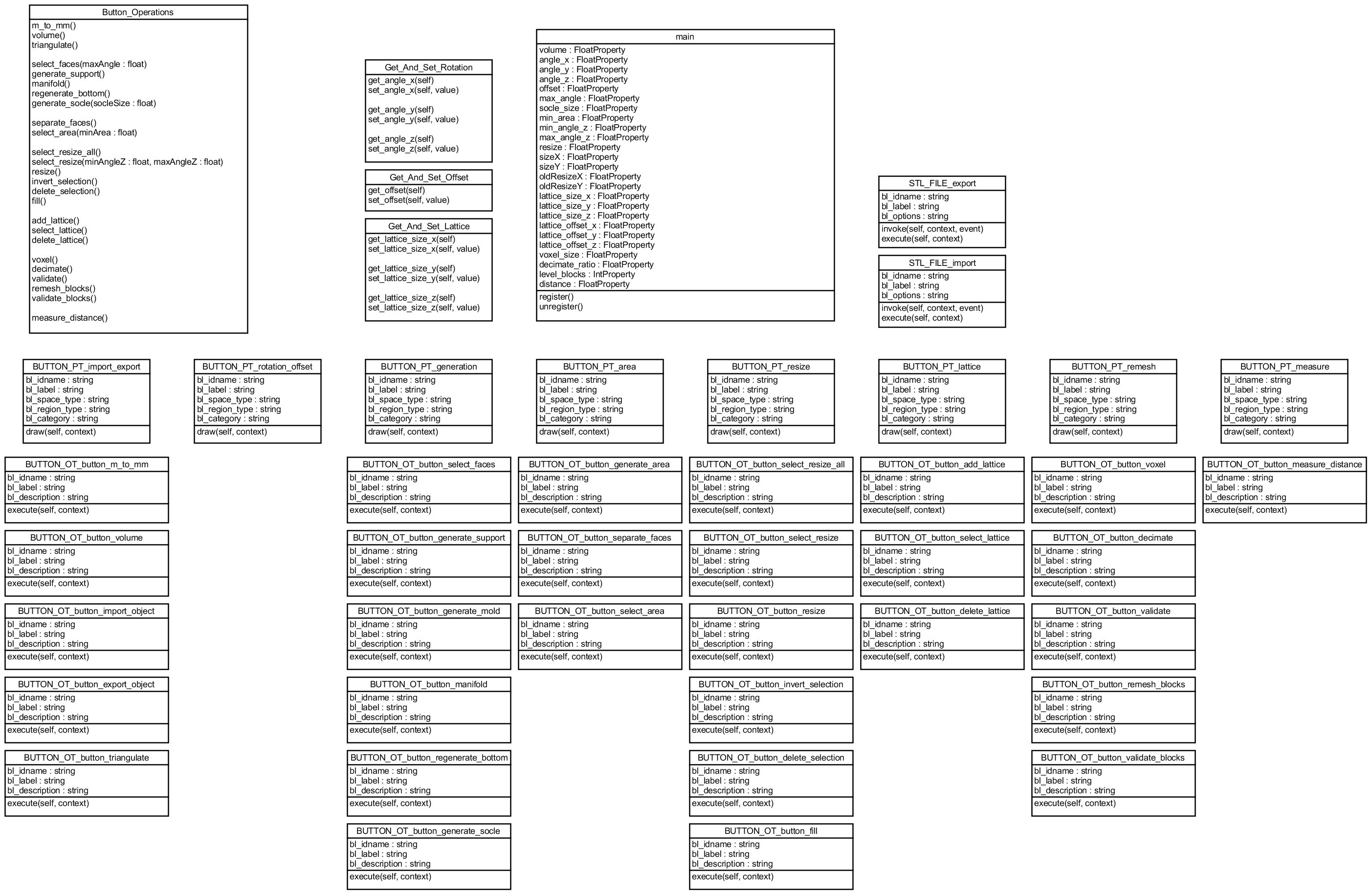
Figure 12 : L'objet de base imprimé (a) et l'objet de base soutenu par les supports (b)

Une fois les supports et la pièce de base imprimés, il a été essayé de mettre ensemble les deux. Cependant, il n’était pas possible de rentrer le support dans la pièce à cause du fait qu’il n’y avait pas de jeu entre les deux. Après avoir constaté ce problème et avoir limé les pièces, il a été possible d’assembler la pièce de base et les supports.

Avec ce test, il a été décidé que la génération des supports avec Blender été une bonne base et qu’il fallait continuer avec Blender tout en améliorant et en ajoutant des outils pour gérer les problèmes rencontrés et créer un panel de contrôle pour générer les supports.

# Logiciel

L’API de Blender permet de développer des tableaux de contrôle avec des menus, des sliders et des boutons qui peuvent exécuter des scripts quand on appuie dessus. Le code des fonctionnalités doit être écrit en python et doit ensuite être ajouté comme add-on sur Blender pour y avoir accès. Les fonctionnalités du logiciel peuvent être divisées en X catégories…



## Import/Export

### Unités 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, 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.

### Import/Export

L’import et l’export des fichiers STL est 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 possédait des modules afin d’ouvrir un explorateur de fichiers pour importer ou exporter des fichiers. Il est aussi possible, grâce à cette méthode, de spécifier de ne pouvoir importer ou exporter que des fichiers STL.

Pour exporter un fichier STL, il est parfois important de connaître sa taille car les logiciels qui vont convertir le fichier STL pour l’imprimante 3D vont prendre plus de temps plus le fichier est volumineux. Afin de pouvoir estimer la taille d’un fichier STL, il a été testé s’il y avait un lien entre le nombre de faces du fichier 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 13 : Graphique de la relation entre la taille d'un fichier 3D et de son nombre de faces

Comme le montre la figure 14, 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 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 triangulaire, Blender va automatiquement transformer ces faces en faces triangulaires avant d’exporter le fichier ce qui va augmenter le nombre de faces et donc changer la taille du fichier. C’est pourquoi il y a une option pour transformer toutes les faces non triangulaires en faces triangulaires.

## Rotations et offset

Afin de pouvoir tester facilement différents types de support, il doit être possible de tourner l’objet à 360° sur tous 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’axe local qui définit la position de l’objet par rapport à son centre. L’autre système d’axes est le système d’axes global qui permet d’obtenir la position du centre de l’objet par rapport à l’origine de l’axe global (0, 0, 0).

Il existe plusieurs modes pour faire une rotation de l’objet et celui qui a été utilisé et 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 appliqué à l’objet, le résultat sera le même peu importe l’ordre dans lequel les rotations sont faites. 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 et donc il est possible d’obtenir la rotation voulue par un autre moyen quand il y a un blocage de cardan. La rotation se fait par rapport à l’axe global.

Toutes les informations sur les sommets, les arêtes ou les faces (coordonnées, vecteurs normaux, …) sont des vecteurs indiqués par rapport à l’axe local. Quand il y a des opérations qui nécessite d’utiliser l’axe global, il faut donc modifier les informations de l’axe local en informations pour l’axe global 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.

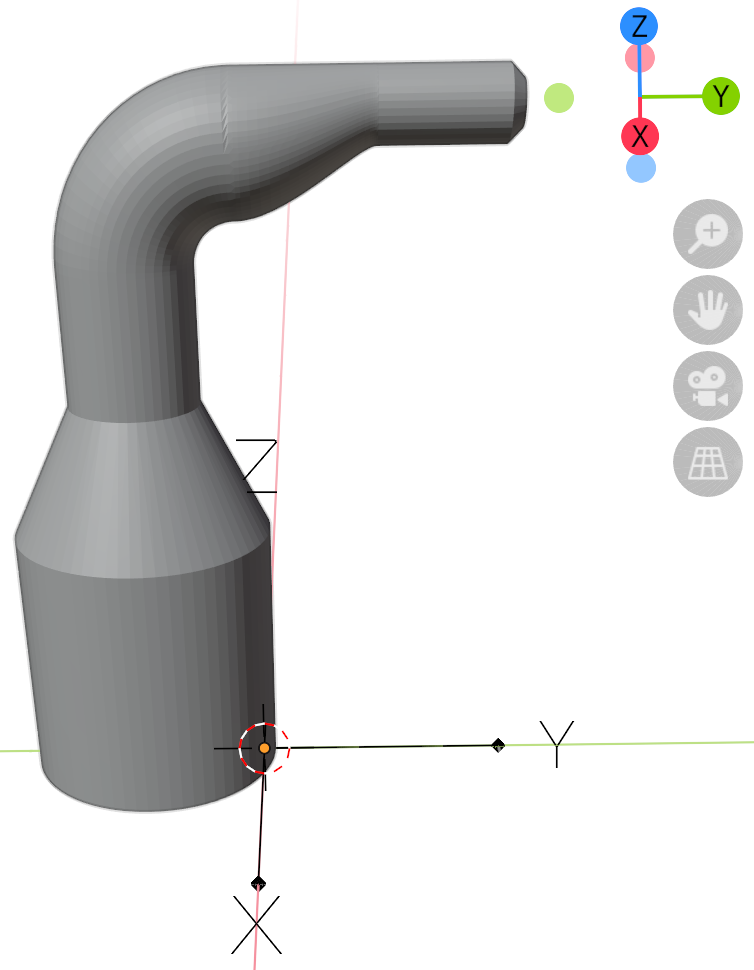
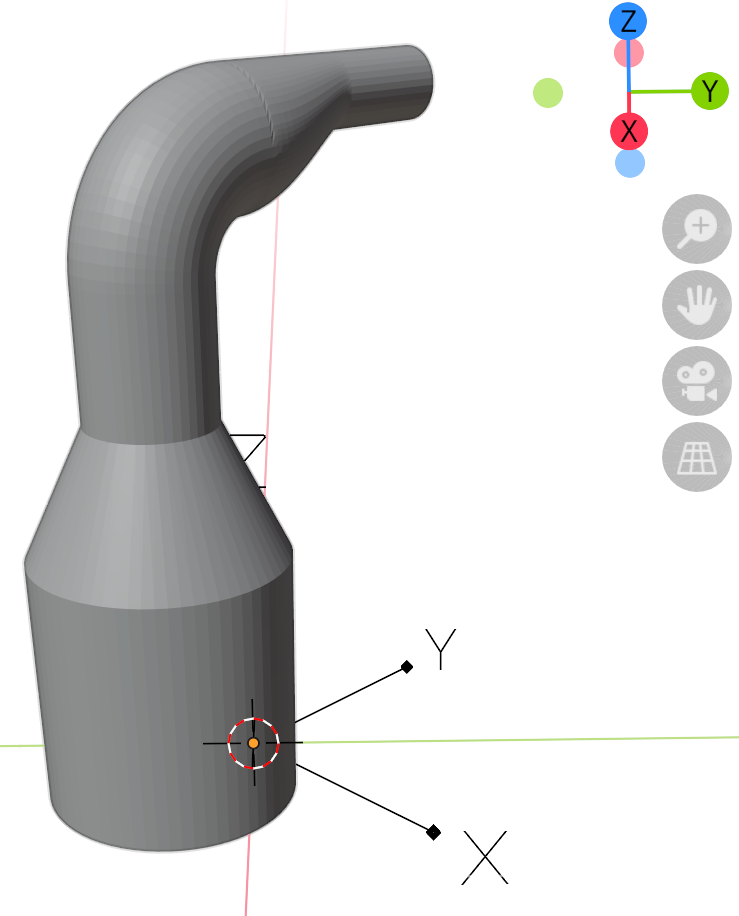
 a  b

Figure 15 : Axe global et local sans rotation (a) et avec une rotation de 45° en z (b)

Les figures 15 a et b montre un objet sans aucune rotation et avec rotation. Sur les figures, le système d’axes global est en haut à gauche et le système d’axes local 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 13a et donc la matrice de transformation aura la forme suivante :

S’il y a maintenant une rotation de 45° sur l’axe z, il va y avoir une différence entre les axes globaux et les axes locaux (Figure 13b). L’axe local x pointe comme la bissectrice des axes x et y globaux, l’axe local y pointe comme la bissectrice des axes y et -x globaux et le l’axe z local pointe toujours comme l’axe z global. Cela donne la matrice de transformation suivante :

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

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 :

De plus, quand une matrice identité est multiplié à une matrice A, on obtient comme résultat la matrice A : .

Une matrice inverse existe pour deux matrices carrées de même taille A et B possèdent la propriété , on dit alors que A est l’inverse de B et que B est l’inverse de A. Cela s’écrit : .

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 :

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

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 :

## Génération

### 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. Vu qu’il a été décidé que les supports doivent partir du plateau et qu’ils soient seulement verticaux, seul les faces qui ont un angle inférieur à une variable angle max (entre 0° et 90°) par rapport au vecteur descendant (Figure 13) et qui ne possède pas de faces en dessous d’elle peuvent avoir besoin de support.

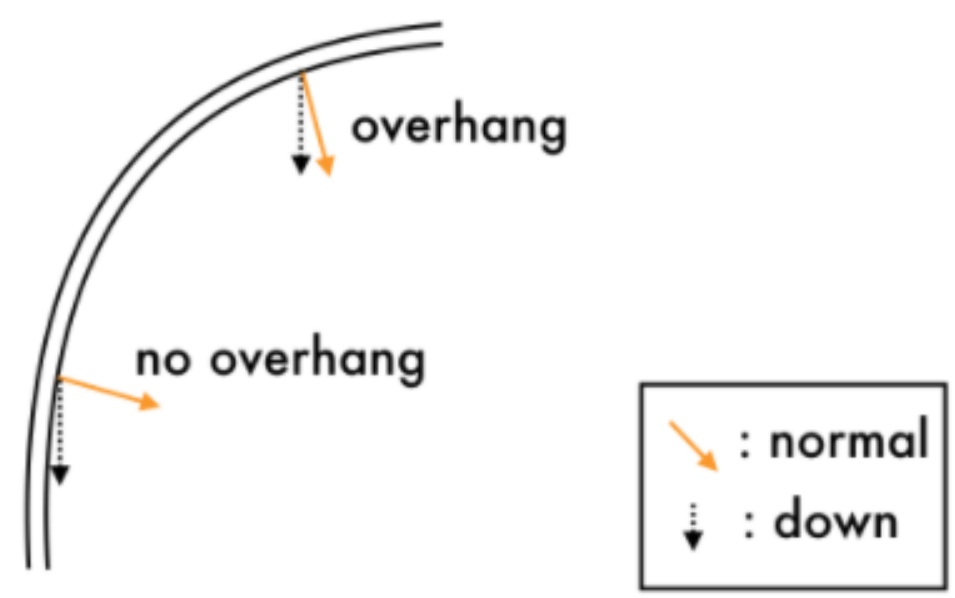


Figure 16 : Principe des faces qui ont besoin de support en fonction de l'angle  
Source : <https://help.autodesk.com/view/MSHMXR/2019/ENU/?guid=GUID-E507C608-8856-47C4-B0B8-BC570DAFD18A>

#### 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 support. 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 est de placer la caméra en dessous de la pièce puis, la caméra est zoomée pour que la taille de l’objet dans l’objectif soit maximale (Figure 13a). La caméra prend ensuite une photo et toutes les faces visibles sur cette photo sont sélectionnées (Figure 13b). Les faces dont l’angle entre le vecteur normal et le vecteur descendant est plus grand que l’angle max sont finalement désélectionnées.

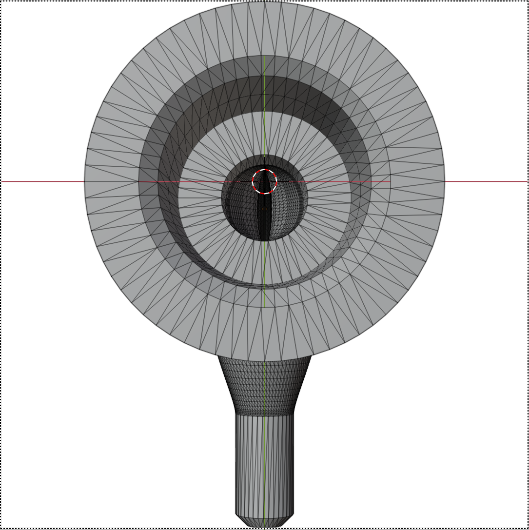
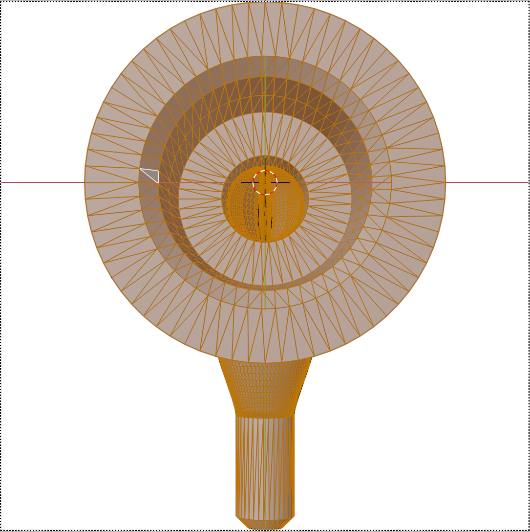
 a  b

Figure 17 : La vue de la caméra après le zoom (a) et la 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 focal de la caméra et de la taille et l’inclinaison de certaines faces (Figure 14). Cette incapacité à sélectionner toutes les faces peut amener à créer un support assez morcelé.

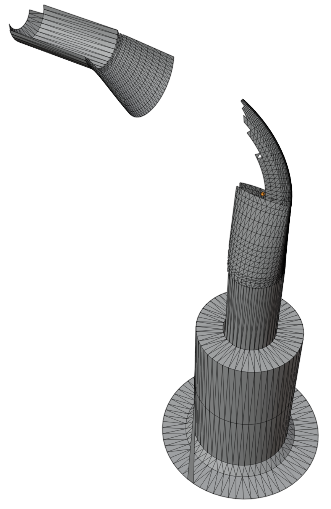
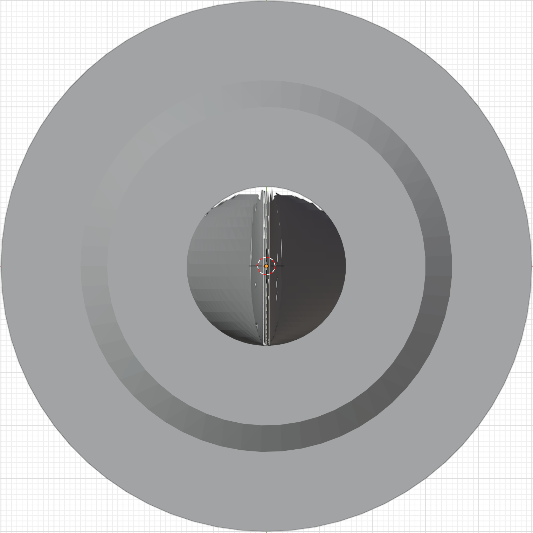
 a  b

Figure 18 : Les faces sélectionnées 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.

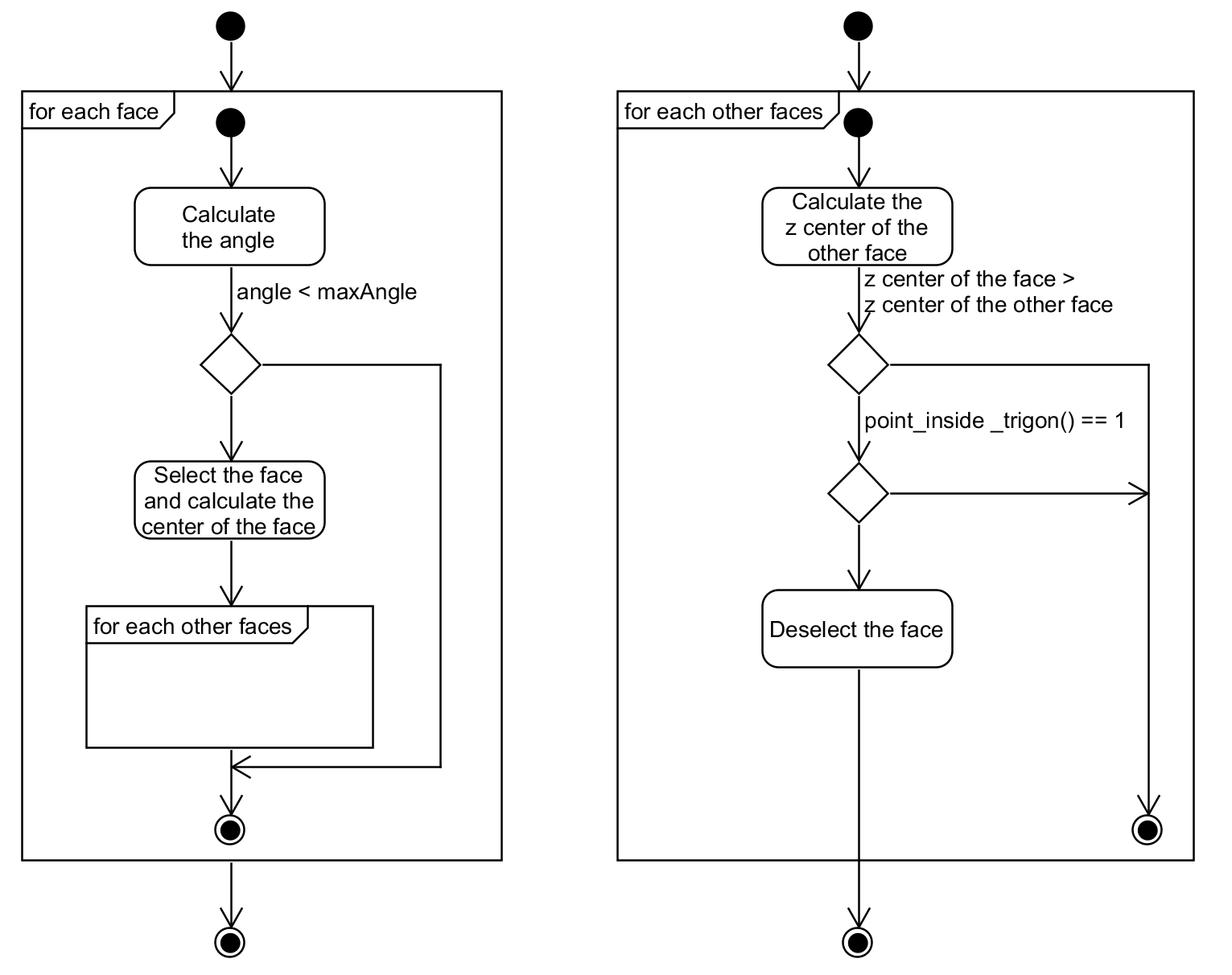


Figure 19 : Diagramme d'états de la méthode du calcul des faces

Cette méthode va sélectionner 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 ait pas de faces en dessous d’elle en parcourant toutes les faces de l’objet et en contrôlant que pour toutes les faces qui sont plus basses qu’elle et dont le centre de la face ne se projette pas dans l’aire des autres faces (méthode point\_inside\_trigon()).

Vu qu’il y a une projection, la méthode point\_inside\_trigon() 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 et ( x ) est le vecteur où :

* x est perpendiculaire à et .
* Le résultat de x vaut : . Dans le cas d’un problème 2D, a3 et b3 sont nuls : .
* Si on place le centre de la main droite au point de départ commun de et avec le pouce sur , l’index sur et le majeur perpendiculaire à la paume, le majeur pointe dans la direction de x (règle du tire-bouchon), donc .

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 suivante.

Tout d’abord, il faut regarder de quel côté du vecteur se trouve le point S avec le produit vectoriel (). Puis, il faut faire pareil avec le vecteur (). 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 20a). Sinon, le point S se trouve entre les deux demi-plans AB et AC (Figure 20b).

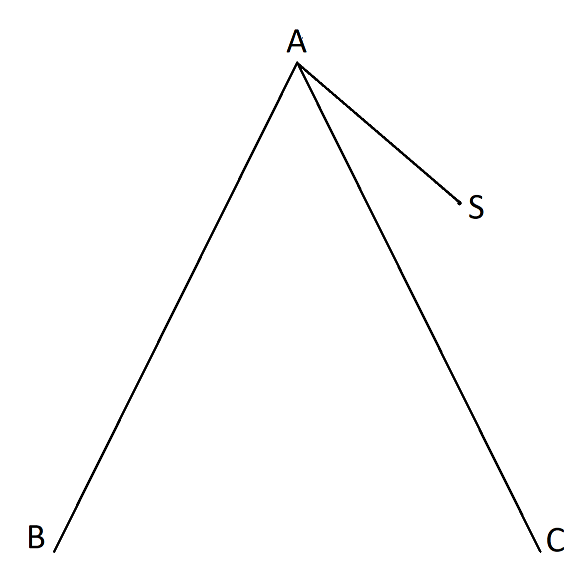
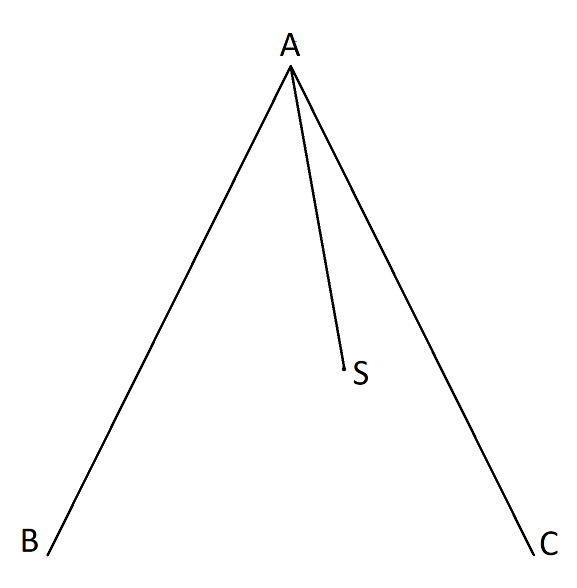
 a  b

Figure 20 : Schéma pour déterminer si un point est dans les demi-plans AB et AC

Il reste finalement a contrôlé de quel côté du vecteur se trouve le point S avec le même principe qu’avant () pour déterminer si le point est dans le triangle (Figure 21).

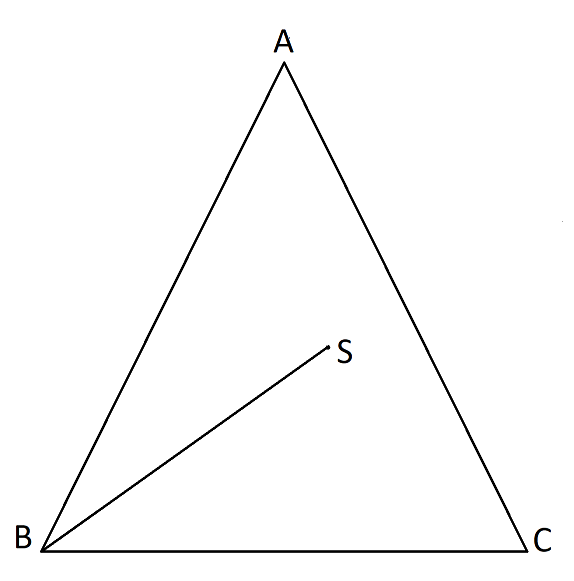


Figure 21 : Schéma pour déterminer si un point est dans le triangle

#### 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 le 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 est 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

Vu les 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 de calcul des faces avait quelques bugs et qu’elle devait être amélioré. 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 quand seulement une partie de la face est couvert par une autre sans que son centre soit couvert (Figure 20a). 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 20b, il y a des grandes et des petites faces, selon l’angle, des faces qui devraient être sélectionné ne le sont pas.

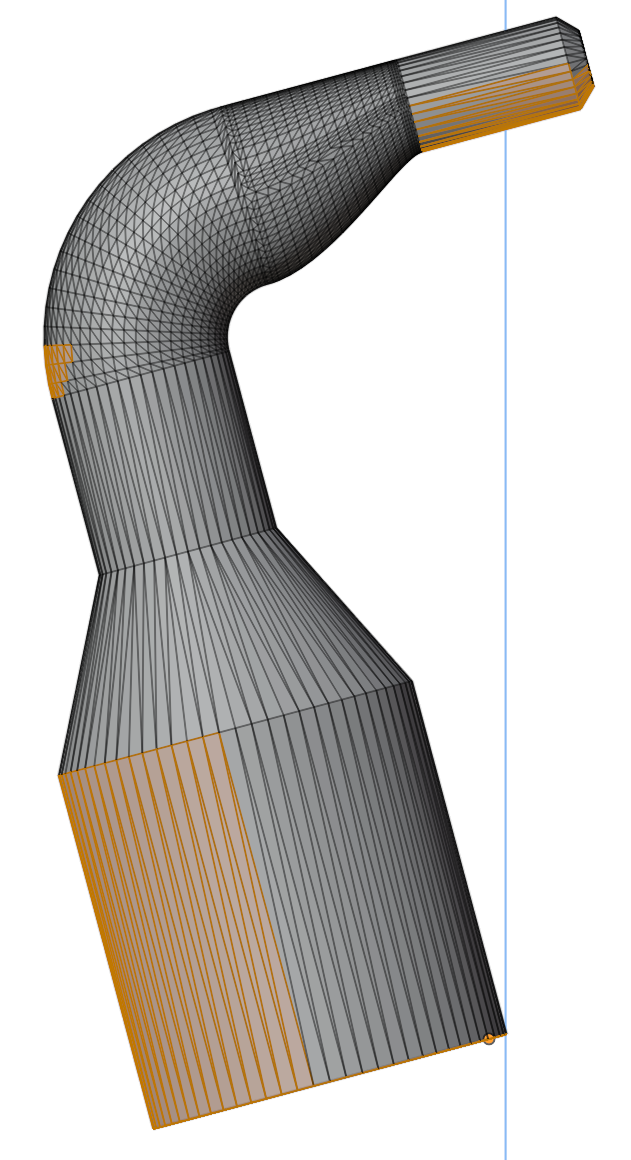
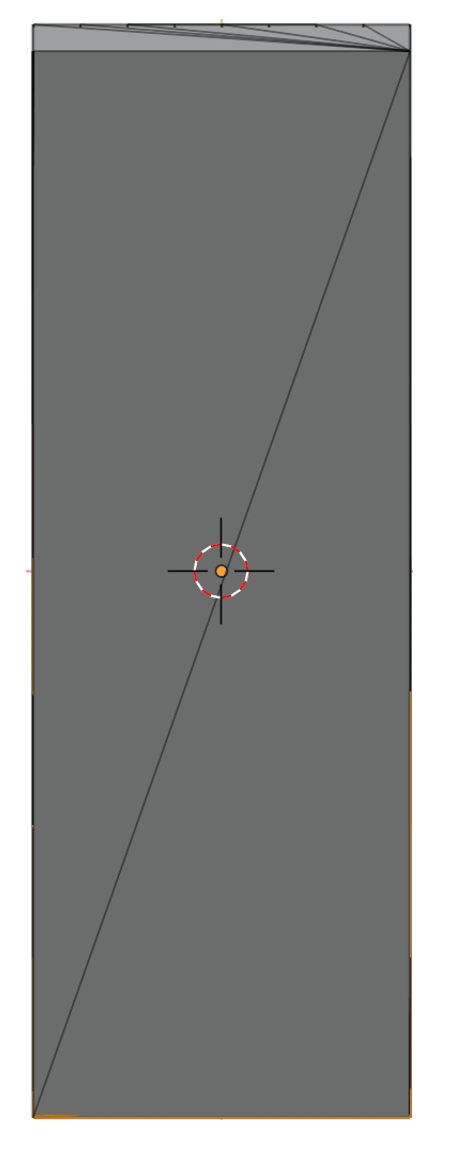
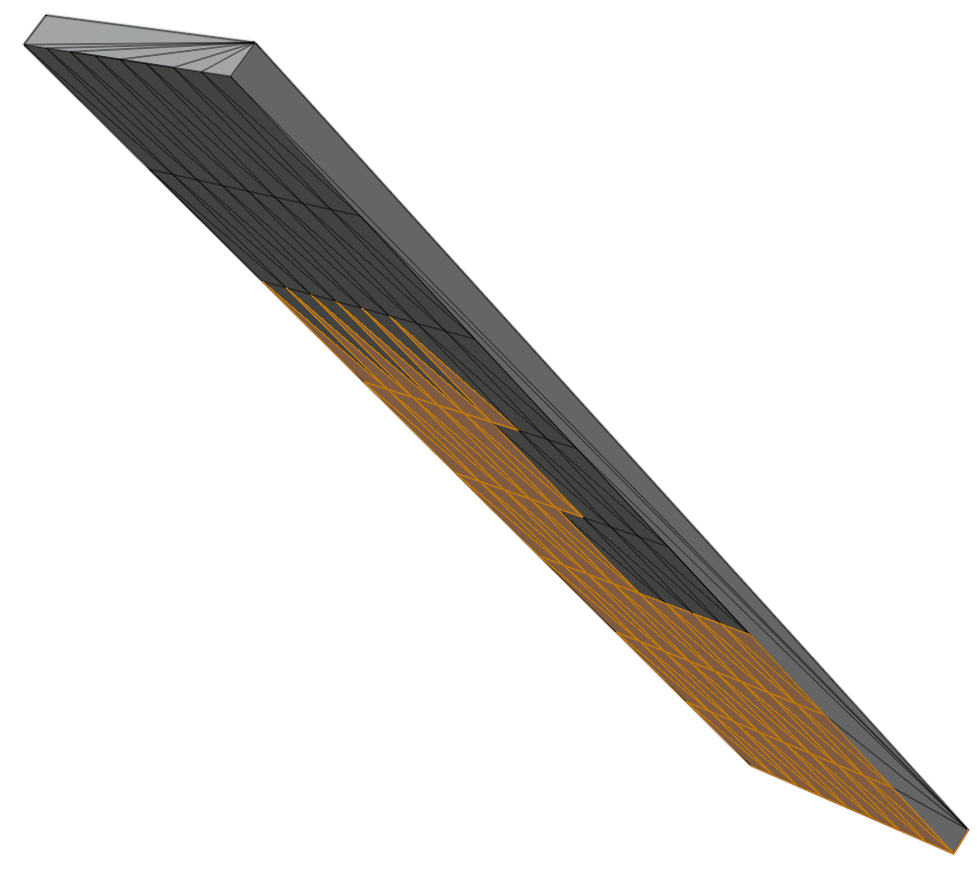
a  b

Figure 22 : 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, vu que les faces sont reliées entre elles et que donc elles possèdent des sommets communs, la méthode point\_inside\_trigon() ne considère donc pas que les points qui se trouvent sur le périmètre et les sommets de la face testé soient dans cette face. Un point est considéré comme étant dans une face que s’il se trouve à l’intérieur de cette dernière.

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 calculée à l’aide de la norme et du produit scalaire. La norme d’un vecteur est sa longueur et s’écrit . Le produit scalaire de deux vecteurs et est le nombre . Le résultat du produit scalaire vaut :

Le produit scalaire permet de faire la projection orthogonale d’un vecteur sur l’autre. La projection de sur est donc le vecteur de longueur .

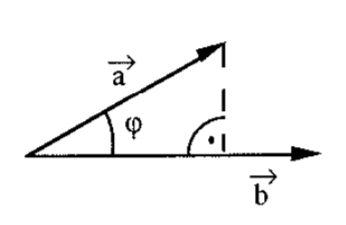


Figure 23 : Projection orthogonale de sur

Donc, pour calculer la distance perpendiculaire d’un point S au plan formé par le triangle ABC, il faut connaître le vecteur normal et la position du point S et d’un des sommets du triangle (A).



Figure 24 : Distance perpendiculaire du point P au plan formé par le triangle ABC

Puis, il faut faire la projection du vecteur sur le vecteur , ce qui donne comme longueur :

Vu 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 du triangle va inverser 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 :

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 25a). 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 25b). Sinon, le point P n’est pas dessus la face.

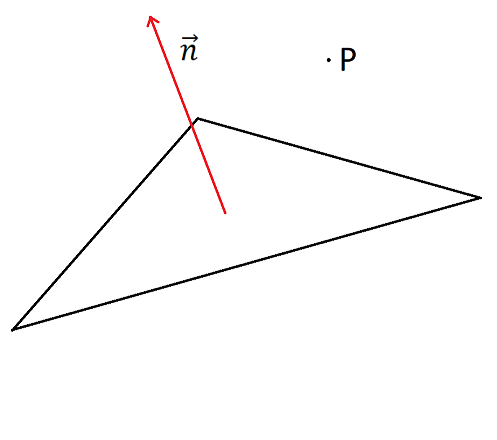
 a  b

Figure 26 : Cas pour déterminer quand le point P est dessus la face

Ces différents changements font que la sélection des faces est la suivante :

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 comparé avec toutes les autres faces. Pour ce contrôle, la distance perpendiculaire entre chacun des trois sommets de la faces 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).

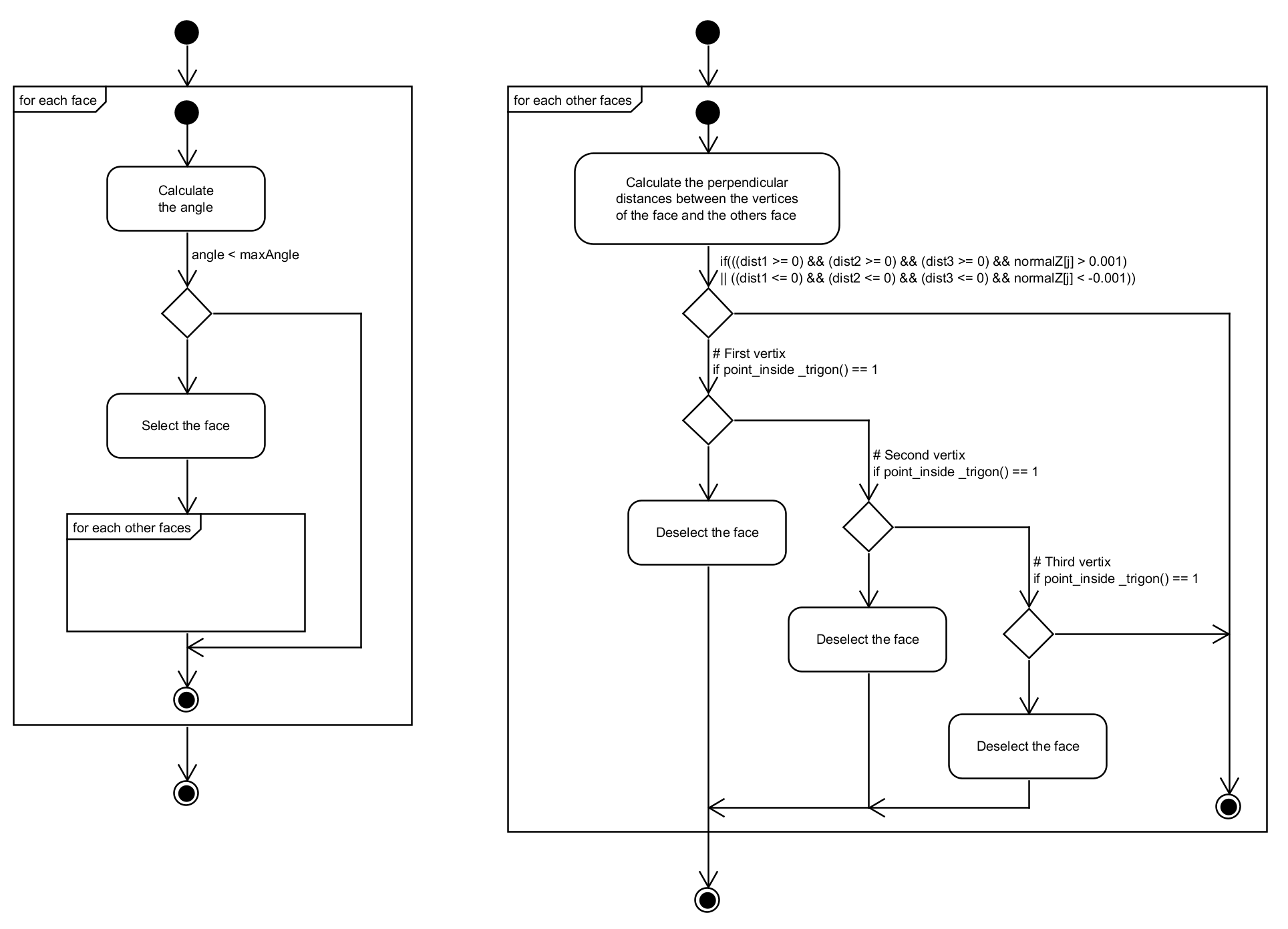


Figure 27 : Diagramme d'état de la version modifiée du calcul des faces

### 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 28).

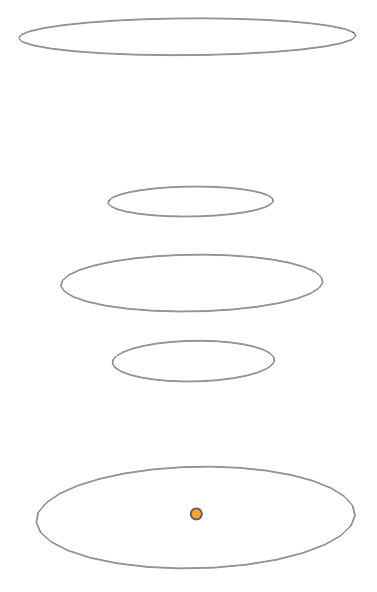
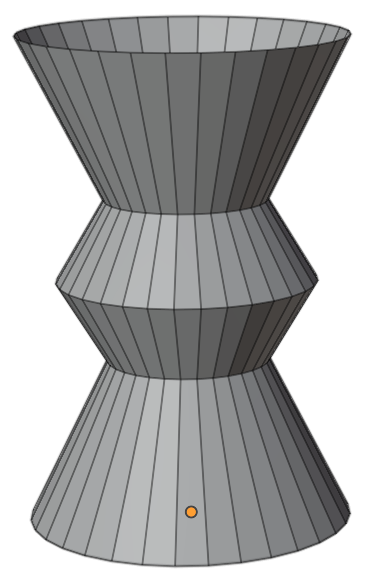
 a  b

Figure 28 : Boucles d'arêtes (a) et création des faces avec l'outil "Bridge Edge Loop" (b)

Cependant, vu la difficulté à sélectionner les arêtes nécessaires et vu que l’outil ne générait pas toujours les faces comme attendues, notamment avec des formes complexes, l’outil « Bridge Edge Loop » a très vite était abandonné. Il a aussi été réfléchi à 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 et donc le problème des faces internes sera résolu après la génération des supports.

Finalement, sur les différentes méthodes testées, la seule qui a donné de bons résultats, peu importe la forme et la complexité de la pièce qui a besoin de supports est la suivante.

Après avoir sélectionner 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, finalement, l’objet de base est supprimé pour qu’il ne reste que le support.

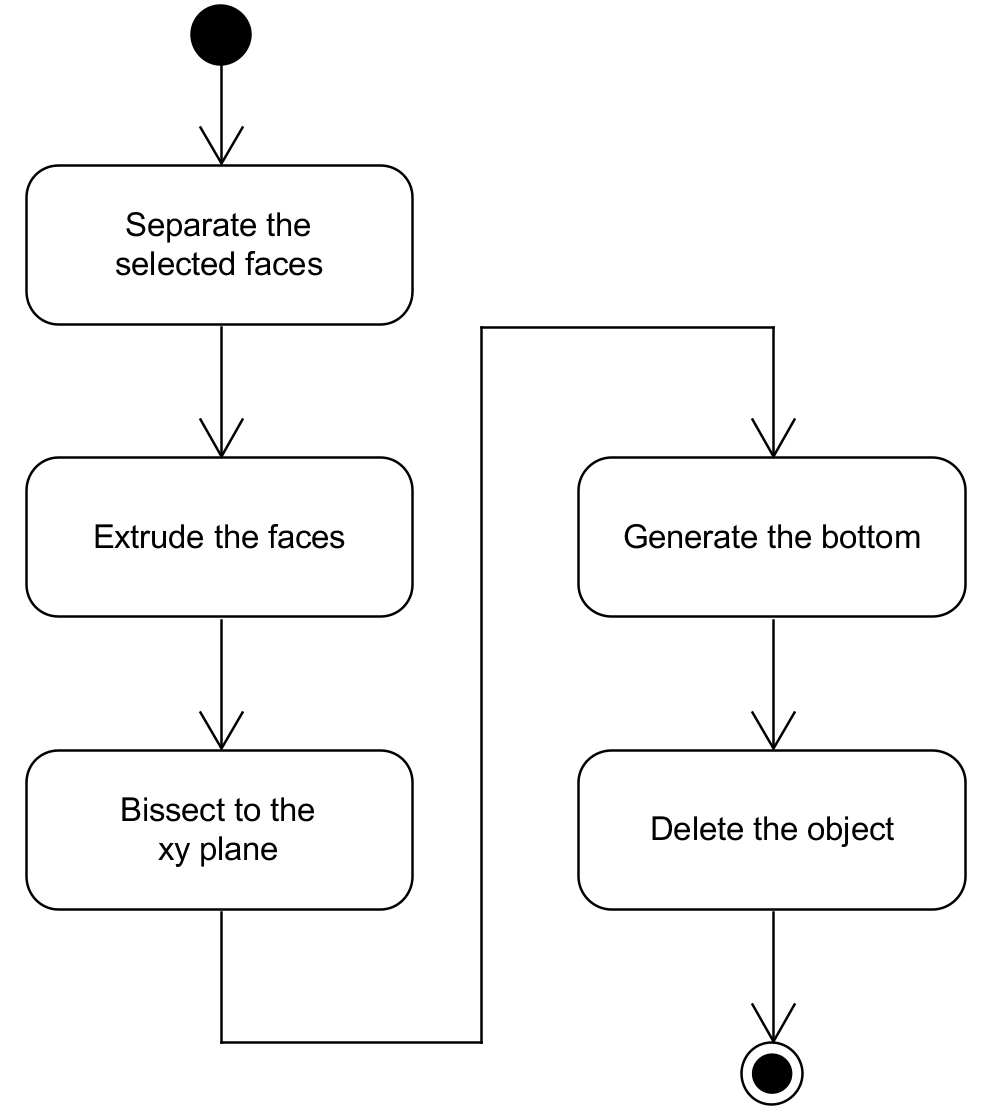
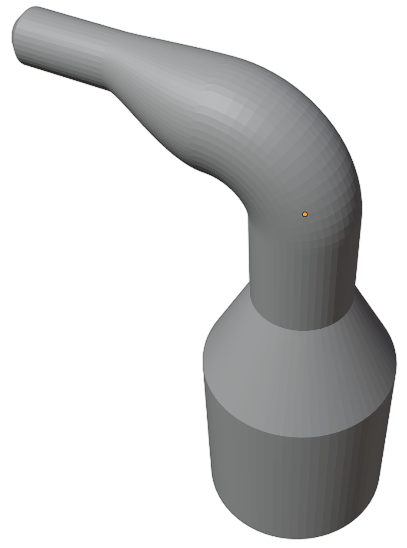
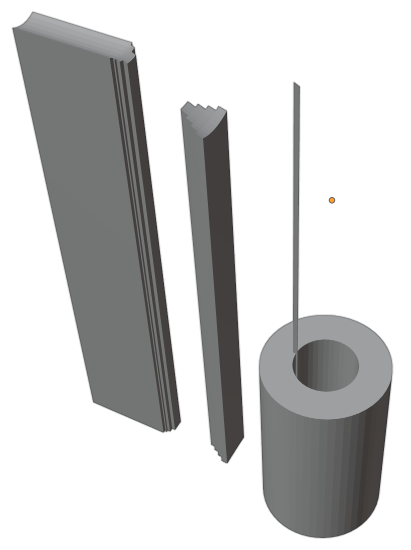


Figure 29 : Diagramme d'état de l'automatisation de la génération des supports

Cette méthode permet de générer facilement des supports verticaux des faces sélectionnées. Le résultat de la génération change en fonction des faces sélectionnées. Plus l’angle maximal choisie 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 30). Avec un angle maximal à 90°, toutes les faces visibles depuis dessous la pièce ont besoin de supports.

 a  b

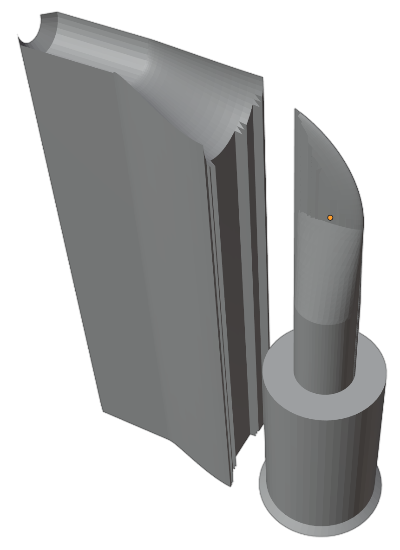
 c  d

Figure 30 : L'objet de base (a) et les supports créés avec le calcul des faces avec un angle max de 30° (b), 60° (c), 90° (d) (mettre à jour !!!!!£ !!!!)

## Options de génération

Plusieurs options ont été développés pour la génération des supports. Ces différentes options permettent à l’utilisateur de choisir différentes options dans le but de mieux générer les supports.

### Aire minimum

Lors de la sélection des faces, ils arrivent souvent que les faces sélectionnées soient réparties en plusieurs groupes de faces. Le problème est que, quand 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 est de ne pas générer de support pour les groupes qui ont une aire inférieure à une valeur minimale choisie.

Pour faire cela, il faut d’abord séparer les faces sélectionnées qui ont besoin de support et supprimer les autres faces qui ne sont plus utiles. Ensuite, Toutes les faces 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 stockées ailleurs. Les autres groupes de faces sont séparés avec la même méthode jusqu’à quand 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 minimum requise.

(L’area min indique l’aire minimale que doit avoir les faces interconnectées pour pouvoir générer les supports. Cela évite que des supports trop petits soient générés. La figure 11 montre que la sélection des faces a créé quatre groupes. L’aire de ces faces est calculée et est comparée à l’area min. La comparaison montre que pour ce cas le groupe 3 a une aire trop petite est qu’il n’y aura pas de support (Figure 12b).)

(Ajouter graphique image etc)

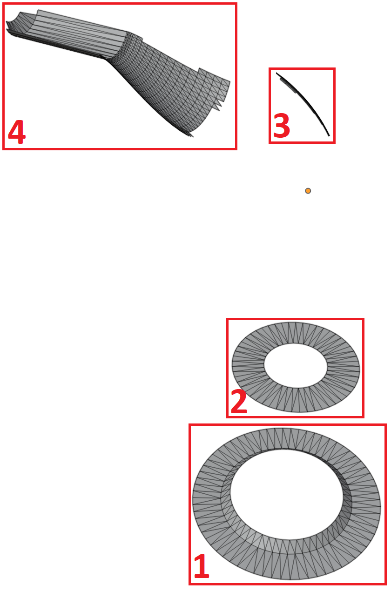


Figure 31 : Faces sélectionnées pour le calcul des faces à 60°

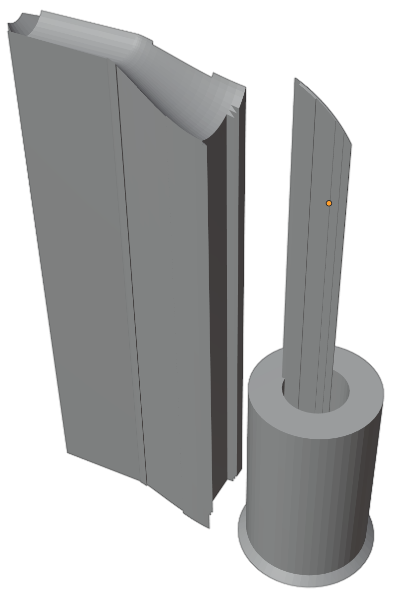
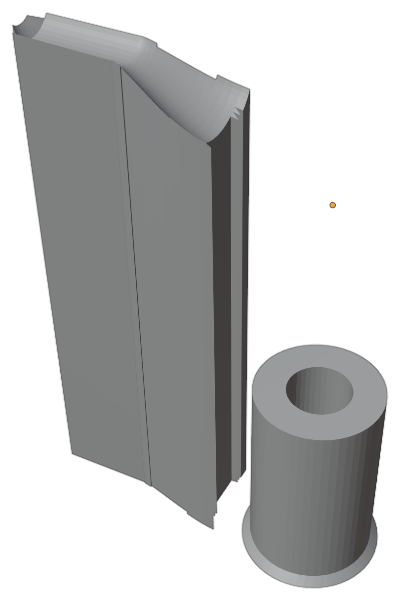
 a  b

Figure 32 : Supports générés avec le calcul des faces à 60° sans le area min (a) et avec un area min (b)

### Génération du fond

Comme expliqué dans la génération des supports, il faut fermer 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().

(Photo non-fermé fermé)

Cependant, comme le montre la figure QW, il arrive parfois que la méthode pour générer le fond ne ferme pas complètement le fichier STL, ce qui pose un problème pour fabriquer la pièce.

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

Un objet de type « Plan » est ajouté dans Blender. Il fait la taille en x et y de l’objet 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. Il reste alors à supprimer les quatre sommets d’origine du plan. Faire cela va permettre de supprimer les faces à l’extérieur du support pour ne garder que le fond du support. Comme le montre la figure QW, le fond de l’objet qui n’avait pas été généré correctement avec la première méthode est correctement généré avec la deuxième.

### 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 ensembles. L’utilisation d’un socle est utile pour ne pas avoir plusieurs supports pour permettre de facilement placer la pièce sur les supports lors du frittage.

La méthode 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 33). D’autres méthodes, plus économe en matière, ont donc été développé.

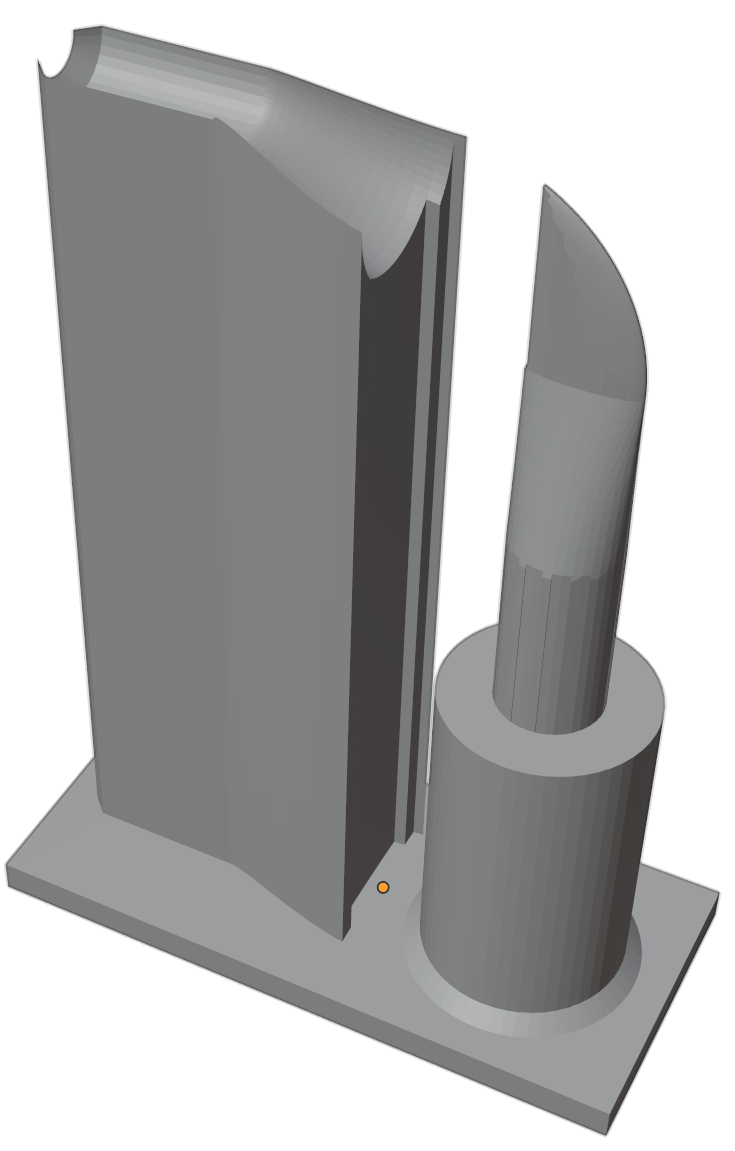


Figure 33 : Création du socle en ajoutant un parallélépipède rectangle dessous le support

(Méthode du resize, méthode extrude along normal)

## Sélection et redimensionnement

Comme cela a été remarqué lors de la première impression plastique en 3D (Figure 12), les supports interne des supports 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 afin de pouvoir redimensionner les parties sélectionnées. Pour ne pas avoir besoin de sélectionner manuellement toutes les faces qui doivent être redimensionner, plusieurs méthodes de sélection ont été développées.

La première méthode sélectionne toutes les faces qui sont connectées aux faces sélectionnées.

La deuxième méthode sélectionne toutes les faces qui sont connectées aux faces sélectionnées et dont l’angle entre leur normal et le vecteur descendant est compris entre une valeur minimum et maximum choisie.

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 trouve à l’intérieur du « Treillis ».

# Validation

# Conclusion

# 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)