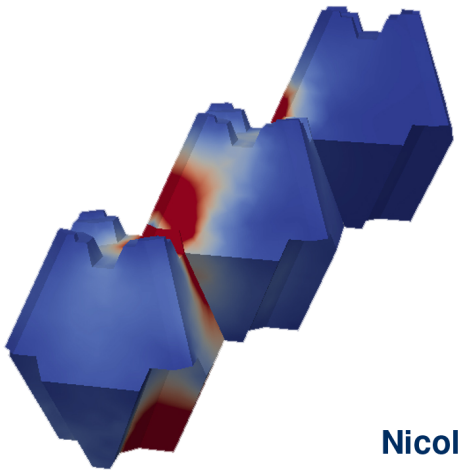
UCL Université catholique de Louvain

École polytechnique de Louvain (EPL)

epl

# Développement de structures composées de droxels



Mémoire présenté par
**Mathieu MULLIER**

en vue de l'obtention du grade de Master
**Ingénieur civil des constructions**

Promoteur(s)
**Pierre LATTEUR**

Lecteur(s)
**Nicolas DOCQUIER, Sébastien GOESSENS , Benoît PARDOEN**

Année académique 2017-2018

# Remerciements

L'accomplissement du travail de recherche et d'écriture lié à ce travail de fin d'étude n'aurait pu avoir lieu sans l'aide de certaines personnes que je tiens à remercier sincèrement pour leur aide précieuse, leurs conseils ou leurs encouragements. Je tiens particulièrement à remercier :

— Monsieur Pierre Latteur pour m'avoir permis de réaliser mon mémoire dans un sujet qui m'intéresse réellement et pour sa confiance en mon autonomie et ma capacité à lui rendre un bon travail. Il était toujours disponible en cas de besoin et il a pu m'introduire aux bonnes personnes qui ont pu m'aider à avancer dans mes travaux.

— Monsieur Sébastien Goessens, assistant de Monsieur Latteur. Il a supervisé et accompagné la totalité de mon travail. Il a pu me prodiguer de très bons conseils, m'orienter dans mes recherches avec bon sens. Il m'a également aidé de manière très directe en imprimant en 3D certaines pièces pour moi ou encore en m'aidant à couler les premiers blocs en bétons.

— L'UCL, plus particulièrement l'EPL, pour m'avoir donné une formation me permettant d'effectuer une recherche utile et efficace et d'en faire un rapport, également pour m'avoir fourni les infrastructures et les outils nécessaires au bon déroulement de mon travail.

— L'équipe du Laboratoire de Mécanique et de Génie Civil de l'Université de Montpellier, plus particulièrement Monsieur Frédéric Dubois et Monsieur Rémi Mozul. Ils m'ont très chaleureusement accueilli et m'ont consacré énormément de leur temps pour m'aider à comprendre leur programme de simulation numérique LMGC90. Ils ont su faire de mon déplacement chez eux une réussite, tant sur le point de vue scientifique qu'humain. Ils m'ont également aidé après mon déplacement en implémentant certaines fonctionnalités dont j'ai eu besoin et en répondant à mes innombrables questions. Je ne saurais les remercier suffisamment pour leur aide.

— L'équipe du Laboratoire d'Essais Mécanique, Structures et Génie Civil de Louvain-la-Neuve pour leur aide précieuse lors de mes différents essais, leur disponibilité, leur bienveillance et leurs conseils reflétant leur bon sens et leur expérience.

— Le reste du personnel de l'UCL, enseignants ou autres, avec qui j'ai eu des contacts lors de ce travail, qui ont tous d'une certaine manière contribué à l'aboutissement de ce projet.

— Les autres étudiants en Master 2 d'ingénieur civil en construction à l'UCL avec qui j'ai suivi tous ces cours qui m'ont finalement permis de réaliser ce travail. L'entraide entre étudiants n'est pas toujours un standard suivant les études et l'université mais je pense vraiment être bien tombé.

— Ma famille et mes amis qui m'ont soutenu lorsque j'en avais besoin et qui ont pu me motiver quand j'avais du mal à avancer dans mon travail. Plus particulièrement mes parents, qui ont toujours tout fait pour me mettre dans de bonnes conditions pour réussir mes études, je pense notamment à leur disponibilité sans failles, leur support financier qui m'a permis de me concentrer à 100% sur mes études. Je pense aussi à leurs nombreux mots d'encouragement qui ont toujours su booster mon moral. Je tiens également à mentionner mes quatre grands-parents qui m'ont toujours accueilli lorsque je devais étudier, l'environnement calme et adapté à l'étude qu'ils m'ont fourni a sans conteste joué un rôle dans la réussite de mes différents examens à l'université et sur la qualité de ce travail.

Enfin, je tiens à vous remercier vous, cher lecteur, pour votre intérêt envers mon travail. J'espère que celui-ci vous intéressera, vous inspirera ou répondra à certaines de vos interrogations concernant le développement de structures composées de droxels.

Bonne lecture ...

# Table des matières


**1 Introduction** **7**
- 1.1 Origine et légitimité du droxel . . . 7
- 1.2 Utilisations du droxel . . . 9
- 1.3 Objectif du travail . . . 11

**2 Géométrie** **13**
- 2.1 Géométrie existante . . . 13
- 2.2 Influence de la géométrie sur les structures . . . 15
- 2.3 Adaptation et choix final de géométrie . . . 17

**3 Blocs et propriétés mécaniques** **19**
- 3.1 Fabrications de blocs . . . 19
  - 3.1.1 Méthode mise en oeuvre . . . 19
  - 3.1.2 Composition de béton utilisée . . . 21
- 3.2 Résistance du béton utilisé . . . 25
- 3.3 Coefficient de frottement . . . 26
- 3.4 Armatures . . . 29

**4 Calcul de structures** **31**
- 4.1 Stabilité globale . . . 32
  - 4.1.1 Besoins . . . 32
  - 4.1.2 Introduction à LMGC90 . . . 32
  - 4.1.3 Application aux droxels - Calculs en rigides . . . 32
  - 4.1.4 Différents cas testés numériquement . . . 35
  - 4.1.5 Tests en laboratoire . . . 40
  - 4.1.6 Analyse des résultats . . . 48
- 4.2 Résistance aux ruptures internes . . . 49
  - 4.2.1 Besoins . . . 49
  - 4.2.2 Calculs en déformables . . . 49
  - 4.2.3 Différents cas testés (laboratoire et numérique) . . . 51
  - 4.2.4 Analyse des résultats . . . 54

**5 Applications** **55**
5.1 Limites dans la réalisation . . . . . 55
5.1.1 Réaction d'une colonne face au vent . . . . . 55
5.1.2 Porte-à-faux vs Arche . . . . . 56
5.2 Quelques structures . . . . . 57
5.2.1 Passerelle piétonne . . . . . 57
5.2.2 Phare . . . . . 57
5.2.3 Abri couloir . . . . . 57
5.3 Cas pratique et note de calcul . . . . . 60
5.3.1 Structure étudiée . . . . . 60
5.3.2 Simulations . . . . . 60

**A Guide d'installation de LMGC90** **75**
A.1 Avant-propos . . . . . 75
A.2 Pré-requis . . . . . 76
A.2.1 MacPorts . . . . . 76
A.2.2 Eléments à installer à partir du terminal . . . . . 76
A.2.3 Paraview . . . . . 77
A.3 Installation et compilation de LMGC90 . . . . . 78
A.4 Exemple d'utilisation . . . . . 79

**B Scripts LMGC90** **81**
B.1 Script de génération, Section 4.1, cas 1, calcul en rigides . . . . . 81
B.2 Script de calcul, Section 4.1, cas 1, calcul en rigides . . . . . 86
B.3 Script de génération, Section 4.2, cas 3, calcul en déformable . . . . . 89
B.4 Script de calcul, Section 4.2, cas 3, calcul en déformable . . . . . 94
B.5 Script de génération, Section 5.3, structure doublement renforcée, calcul en rigides . . . . . 98

# Chapitre 1

# Introduction

## 1.1 Origine et légitimité du droxel

À l'inverse de la plupart des secteurs industriels de nos jours, celui de la construction est resté très peu automatisé et emploie des hommes pour la presque totalité des travaux sur chantier. L'utilisation d'éléments préfabriqués et les moyens de fabrication en usine de ces éléments ont subi une belle évolution ces dernières années, il serait dès lors intéressant d'avoir également une évolution des méthodes et moyens de construction mis en oeuvre sur le chantier. Et cela pour plusieurs raisons.

On citera par exemple le fait que le coût de la main d'oeuvre augmente de plus en plus, surtout la main d'oeuvre locale. Cela pousse les entrepreneurs à engager de la main d'oeuvre étrangère, souvent moins chère, ce qui induit souvent la présence de plusieurs langues sur un chantier ou l'absence d'une langue commune, et donc des erreurs de communication. On pourrait ajouter le manque de qualification de certains ouvriers ou même tout simplement les imprécisions et erreurs liées au travail manuel, allant parfois jusqu'à compromettre la sécurité sur un chantier. Sachant qu'un accident sur chantier entraine parfois des pertes humaines et également de lourdes sanctions financières, les éviter est une priorité. En 2012 aux USA, l'OSHA a relevé que pas loin de 20% des accidents mortels dans l'industrie privée proviennent du secteur de la construction[1]. Ces accidents sont très souvent dus à une erreur humaine. Cette erreur peut être la non-identification d'une situation dangereuse, la non-considération d'une situation dangereuse ou encore l'exécution d'activités particulières sans formation adéquate.

Ensuite, on notera le fait que les architectes et ingénieurs utilisent de plus en plus des modèles BIM et CAD pour la conception de leurs ouvrages. Ces outils ont énormément d'avantages par rapport aux plans papiers, notamment le fait de pouvoir réaliser des ouvrages avec des formes artistiques et novatrices, même si très complexes. Cependant, la réalisation concrète de ces ouvrages se révèle parfois compliquée. En effet, les

FIGURE 1.1 – Siège de Vodafone, Porto

méthodes de construction actuelle ne permettent pas toujours de réaliser des géométries compliquées sans dépenser de grandes sommes d'argent. On devra par exemple souvent créer des coffrages très spécifiques et à utilisation unique, comme par exemple pour la structure en béton du bâtiment Vodafone. (Figure 1.1)

En considérant le fait que les drones sont de plus en plus performants et ont un coût de plus en plus abordable, la réflexion a été faite qu'ils pourraient constituer un bon moyen de construction. En effet, on aurait alors une automatisation de la construction sur base du modèle numérique, des risques revus à la baisse, un facteur d'incertitude très faible et on se passerait même de l'utilisation de grues. Tout cela ayant bien évidemment une conséquence économique positive pour le chantier.

Ces drones ne pouvant porter qu'une charge limitée et ne pouvant effectuer que des mouvements limités également, il a fallu imaginer des éléments de maçonnerie posables facilement et avec lesquels on pourrait effectuer différentes géométries. Est alors arrivé, parmi d'autres idées, le droxel.

FIGURE 1.2 – Droxelisation d'une géométrie

FIGURE 1.3 – Blocs de béton assemblables

## 1.2 Utilisations du droxel

Le nom *droxel* vient de la contraction drone-voxel. En effet, l'idée est de pouvoir *droxeliser* un modèle 3D, c'est-à-dire composer une structure de géométrie équivalente sur base de droxels. (Figure 1.2)[2]

Mais le droxel n'a pas besoin d'être assemblé par des drones pour trouver son utilité. En effet, on a là un élément de maçonnerie ne nécessitant pas de ciment. Il pourrait dès lors trouver son utilité dans des structures éphémères ou des structures devant souvent être démontées et remontées. Il existe déjà des blocs de béton empilables et facilement démontables mais ils ne permettent que la construction d'éléments en deux dimensions comme des murs et ne tolèrent pas les porte-à-faux. (Figure 1.3)

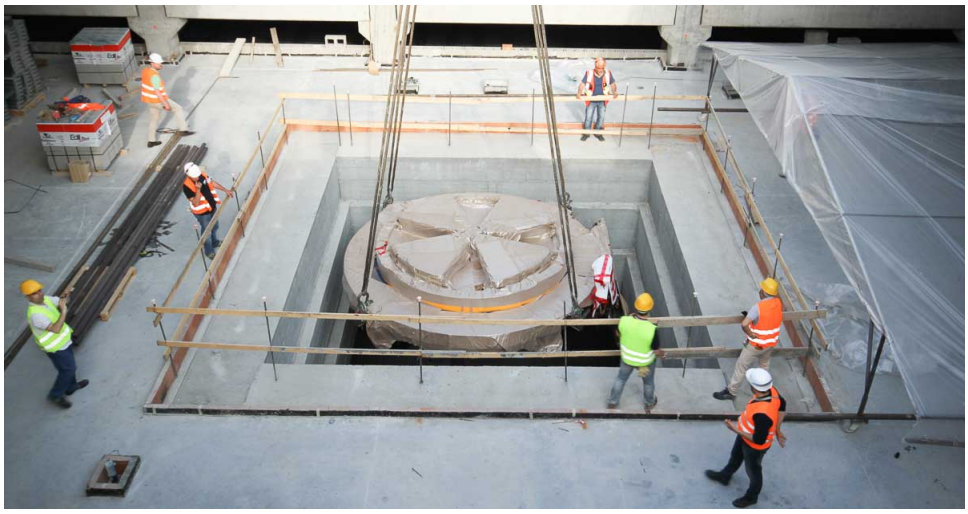


FIGURE 1.4 – Cyclotron, Louvain-la-Neuve

Les droxels pourraient donc par exemple se révéler utiles dans la construction de cages en béton de protection autour des cyclotrons (figure 1.4). En effet, ces protections doivent souvent être retirées lors de travaux sur le cyclotron ou tout simplement lors du démantèlement. De plus, le béton se révèle souvent contaminé par les radiations nucléaires, même si des projets de bétons à faible activation voient progressivement le jour. Les droxels pourraient fournir une alternative aux solutions actuelles et on pourrait même envisager la réutilisation des blocs déjà contaminés et la combinaison avec les futures solutions de béton à faible activation en coulant les blocs avec ces nouveaux bétons.

Les droxels pourraient également être utilisés à des fins d'innovation en design ou à des fins artistiques. En effet, un ouvrage réalisé avec ces blocs se démarquera certainement par son originalité, ce qui peut être une fin en soi.

## 1.3 Objectif du travail

L'objectif de ce travail est d'étudier les structures composées de droxels. Pour ce faire, il faut choisir une géométrie, trouver un support numérique pour calculer les structures et ensuite vérifier la pertinence de ces résultats théoriques en faisant des essais sur des micro-structures réelles en laboratoire. Pour réaliser ces essais, il faut d'abord fabriquer un moule pour ensuite fabriquer des droxels en béton.

Tous ces éléments sont détaillés dans les chapitres qui suivent. Certaines théamatiques trouveront une réponse claire et d'autres ne seront que partiellement résolues. Dans ce dernier cas, des pistes seront fournies pour aller plus loin.

# Chapitre 2

# Géométrie

## 2.1 Géométrie existante

La géométrie du droxel a été développée afin de remplir 3 critères majeurs : la tolérance de pose, la possibilité de réaliser des structures en 3 dimensions avec ceux-ci et le souci de limiter le plus possible les instabilités du bloc lors du transport aérien par drone.

La géométrie proposée par Jean-Sébastien Breton et Justin Leplat dans leur mémoire[2] est représentée à la figure 2.1.

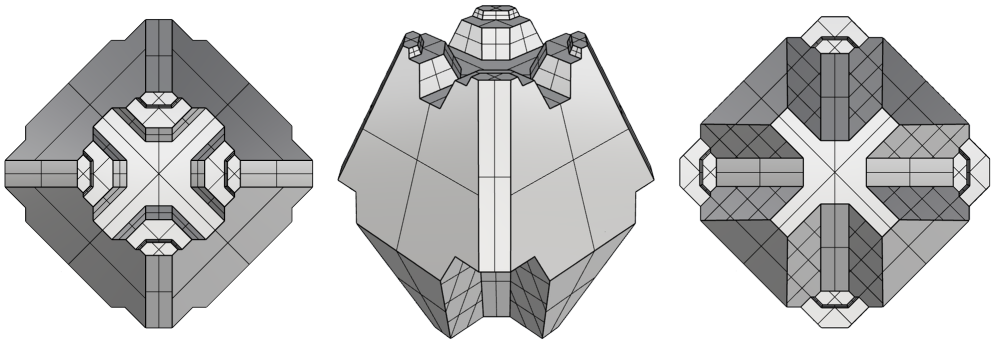


FIGURE 2.1 – Vue de haut, de bas et en perspective d'un droxel

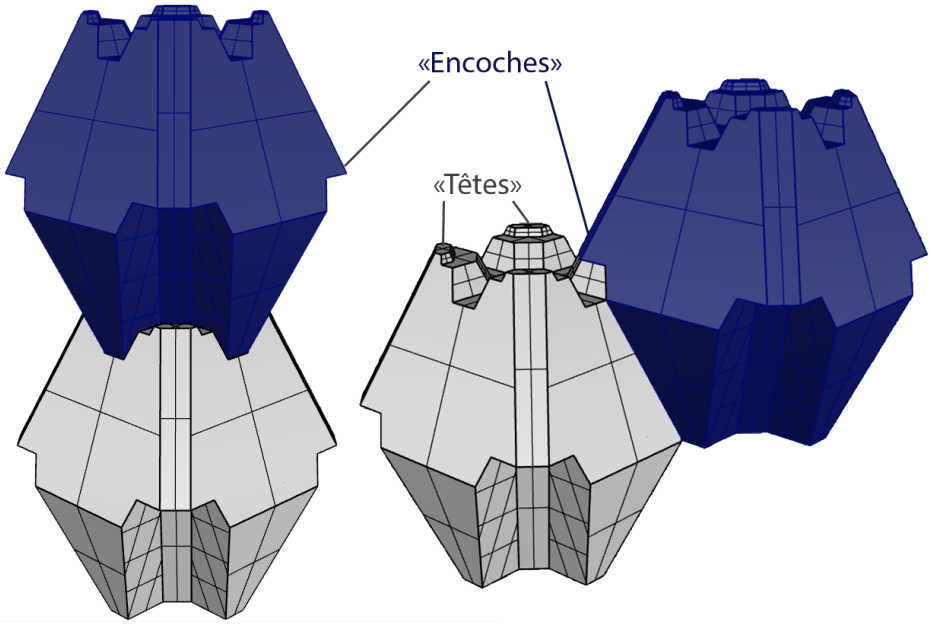



FIGURE 2.2 – Possibilités d'emboîtements de droxels

Les blocs sont empilables soit en posant la croix de la base d'un droxel dans l'encoche en forme de croix sur la tête d'un autre droxel, soit en posant une encoche du coin à mi-hauteur d'un droxel sur une tête d'un coin de la face supérieure d'un autre droxel, comme représenté à la figure 2.2. Dans la suite de ce travail, il sera fait référence, par les termes "encoches" et "têtes", aux éléments géométriques tels que désignés sur cette figure.

Ces 2 types d'emboîtements, liés à la double symétrie de la géométrie, permettent d'établir des configurations très variées. Un exemple représentant bien les possibilités de configurations découlant de la géométrie du droxel est représenté à la figure 2.3.

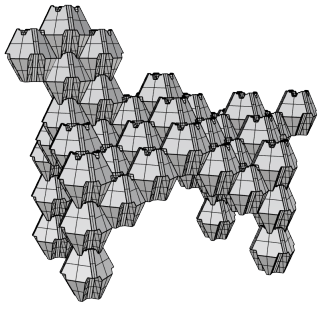


FIGURE 2.3 – Structure en forme de chien composée de droxels

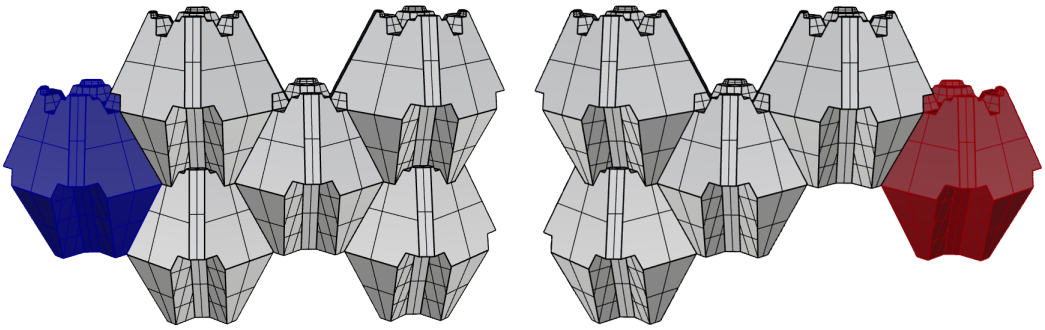


FIGURE 2.4 – Stabilité d'un droxel en fonction des appuis

## 2.2 Influence de la géométrie sur les structures

La contrainte majeure dans la composition d'une structure composée de droxels est qu'un droxel ne peut flotter dans le vide, il doit toujours avoir un appui inférieur. Comme illustré sur la figure 2.4, le droxel bleu est stable mais le rouge ne l'est pas.

Il est donc impossible de réaliser l'équivalent d'une poutre horizontale plate avec des droxels, on est limité à un angle minimum par rapport à l'horizontale. Cet angle varie en fonction de la géométrie du droxel.

Plus on se rapproche d'une géométrie similaire à celle d'un parallélépipède à base carrée de petite hauteur et de grande base, plus on arrive à un angle minimum faible, cependant, on perd en tolérance de pose. À l'inverse, en rétrécissant la tête et la base par rapport à la largeur à mi-hauteur, on gagne en tolérance de pose. On perd par contre ici en résistance car l'aire de la section à la base faiblit. Deux cas sont représentés sur la figure 2.5

À gauche, les mesures du droxel sont les mesures standards que nous garderons car c'est un bon compromis entre tolérance de pose et angle d'élévation. On observe pour cette géométrie un pas en hauteur entre 2 droxels de 25 cm, une tolérance de pose [1] de 8,19 cm et un angle d'élévation de 33,61°. À droite, pour la deuxième géométrie présentée, on observe un pas de 18,78 cm, une tolérance de 3 cm et un angle d'élévation de 21,44°.

1. Cette tolérance de pose est prise comme la distance diagonale maximale à laquelle on peut poser le droxel suivant sans avoir de problème d'emboitement dans le cas de l'ajout d'un droxel supérieur entre deux droxels préalablement posés en diagonale. Elle est exprimée ici dans un but comparatif et ne peut être considérée comme tolérance de pose absolue car celle-ci dépend de la direction, de l'imprécision et du nombre de droxels en appuis inférieurs.

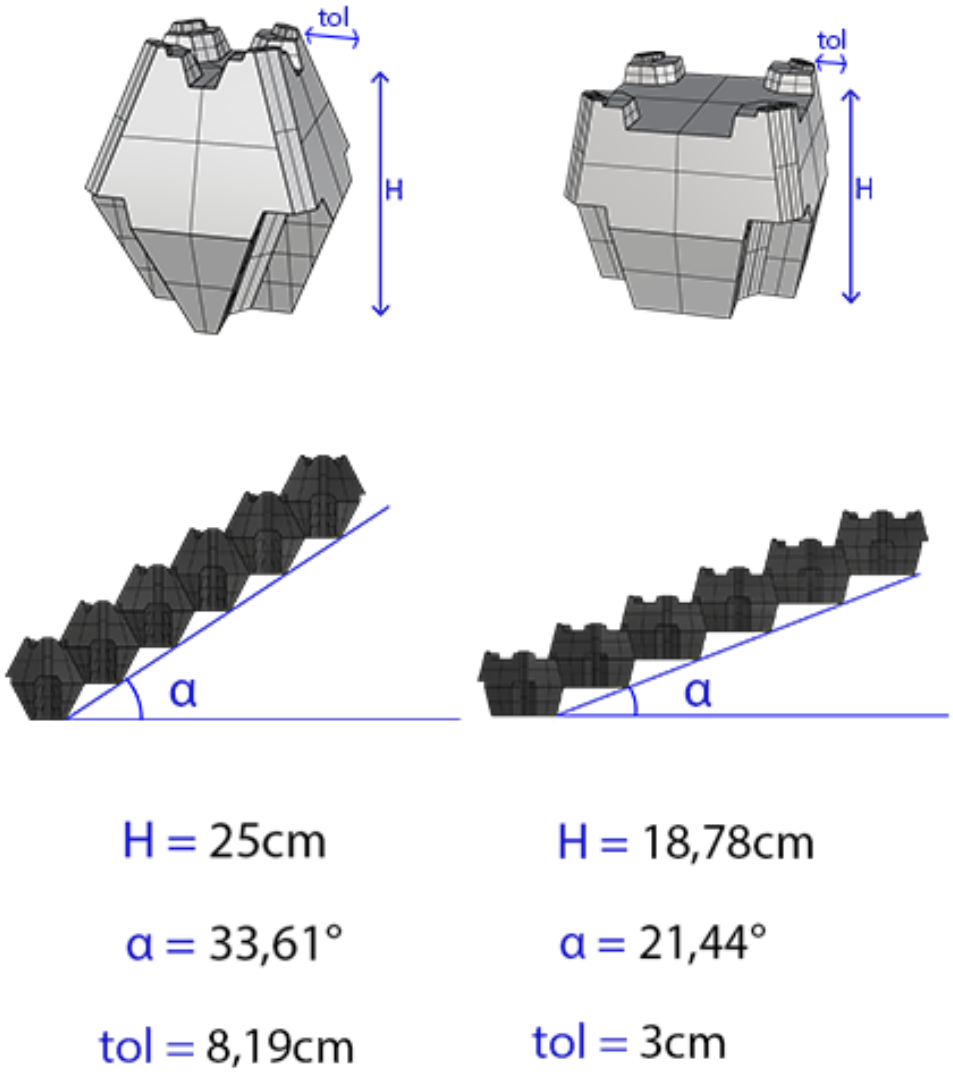



FIGURE 2.5 – Influence de la géométrie des droxels sur la tolérance de pose et l'angle de montée

## 2.3 Adaptation et choix final de géométrie

La géométrie présentée précédemment est la géométrie idéale, qui entraine un emboitement parfait sans aucun jeu entre les droxels. Cependant, lors des simulations numériques qui seront abordées dans le chapitre suivant, il a été constaté que les droxels ne s'emboitaient pas correctement à cause du frottement du béton. Il a donc fallu adapter cette géométrie de sorte que les droxels puissent glisser et s'emboiter correctement, le tout sous l'unique action de leur poids propre. La modification qui a été faite consiste à créer un jeu de 0.1 mm sur chaque contact entre droxel en renfonçant une des deux faces en contact. La moitié des faces de chaque droxel ont donc été renfoncées de 0.1 mm. On a donc pour un droxel de 25 cm de hauteur les mesures présentées sur la figure 2.6 (on parle ici bien du pas entre 2 droxels ; la hauteur totale d'un droxel étant, quant à elle, en réalité supérieure à 25 cm. En effet, il faut ajouter la hauteurs des têtes de la face supérieure puisque les droxel situé au-dessus viendra se loger entre ces 4 têtes).

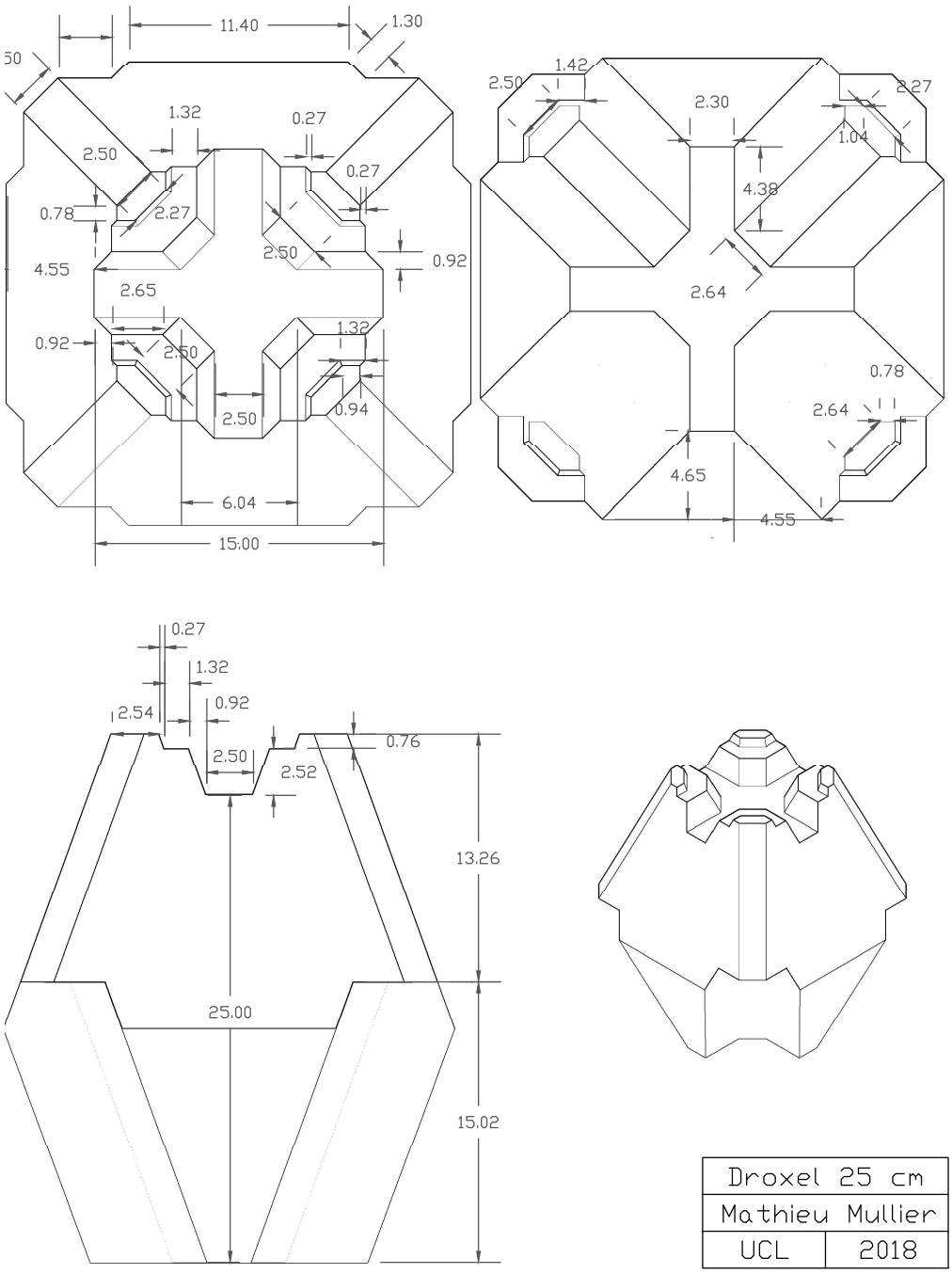



FIGURE 2.6 – Plan d'un droxel

# Chapitre 3

# Blocs et propriétés mécaniques

## 3.1 Fabrications de blocs

### 3.1.1 Méthode mise en oeuvre

Etant donné que les blocs doivent avoir des dimensions très précises pour permettre un bon emboitement, la méthode du moule en bois a été rejetée. On a donc préféré réaliser un moule à l'imprimante 3D. Vu la présence de renfoncements et la diversité des faces et de leurs orientations, il a fallu optimiser le moule pour avoir un démoulage facile. On a donc décidé de réaliser un moule en 8 parties formant un gros cube, de sorte à pouvoir retirer chacune de ces parties en biais pour minimiser le frottement sur la pièce de béton fraichement réalisée. Le fichier numérique du moule est représenté sur la figure 3.1.

Le bloc est coulé à l'envers ; l'orifice pour insérer le béton se situant à la base de celui-ci. Les 8 parties du moule sont maintenues en place et parfaitement alignées grâce à un système de serrage comme représenté sur la figure 3.2. On utilise également du papier collant quelconque pour empêcher le moule de perdre de l'eau par les fentes entre ses différentes parties et une huile de coffrage pour diminuer le frottement béton-moule lors du démoulage. Le béton est donc coulé dans le moule et ensuite vibré à l'aide d'une table vibrante pour s'assurer qu'il remplisse bien tous les orifices.

On notera un temps d'impression de 2 jours pour une des 8 parties ainsi que l'importance du choix du matériau pour le moule réalisé à l'imprimante 3D. En effet, les 2 matériaux principaux, l'ABS et le PLA, ont tous les deux étés utilisés pour le moule et on voit très clairement la différence de durabilité entre les parties fabriquées avec un ou l'autre matériau.

L'ABS peut donner des difficultés à l'impression car il est fort sensible aux changements de température et d'environnement au cours de celle-ci et on peut voir apparaitre des fentes entre les couches si on a un changement de ces conditions d'impression. Ce-

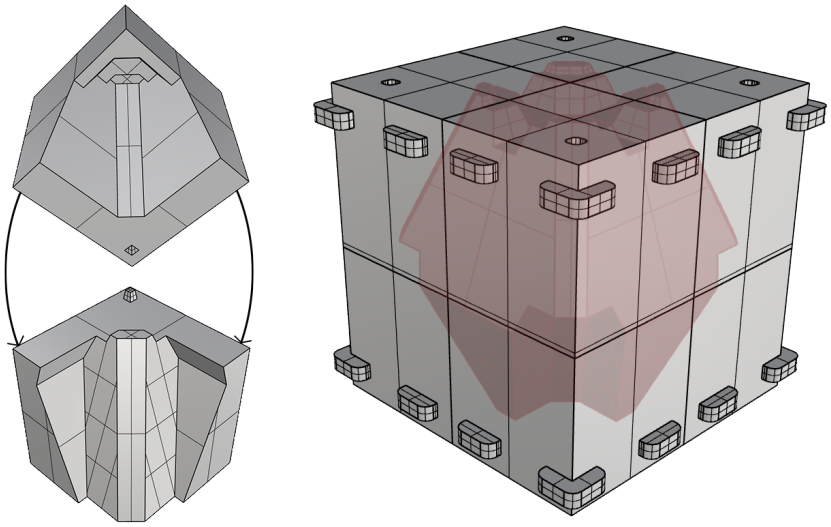
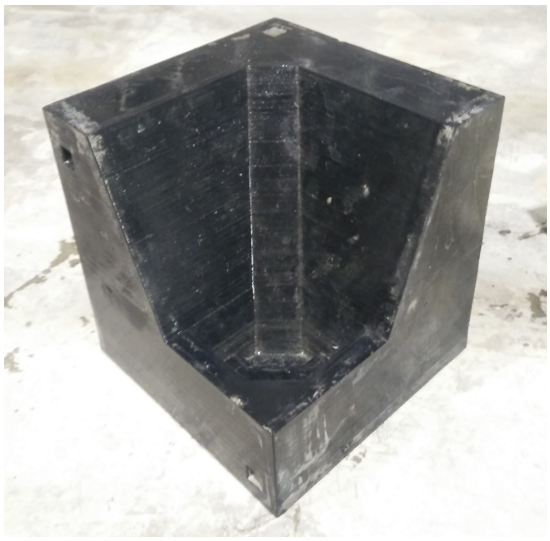


FIGURE 3.1 – Moule de droxel (fichier 3D)

FIGURE 3.2 – Moule et système de serrage



(a) Pièce en ABS

(b) Pièce en PLA

FIGURE 3.3 – ABS vs PLA après 14 utilisations

pendant, il est très durable et résiste bien aux différentes utilisations. Si bien que l'on n'a constaté aucune détérioration des pièces qui en sont composées.

À l'inverse, le PLA est simple à imprimer et donne facilement un très bon fini mais il est beaucoup plus sensible aux contraintes et utilisations. On a pu observer une dégradation avancée de la surface de certaines pièces.

Au final, on préfèrera donc l'ABS car il a une bien meilleure durabilité. L'apparition de fentes à l'impression peut être évitée au moyen d'un bon environnement. De plus, les fentes peuvent être corrigées en passant de l'acétone au pinceau. Un exemple de pièce pour chaque matériau est donné à la figure 3.3 (après 14 utilisations).

### 3.1.2 Composition de béton utilisée

La composition du béton joue un grand rôle sur plusieurs paramètres comme l'uniformité du bloc produit, sa finition ou son poids. Deux compositions principales ont été retenues suites aux différents essais.

#### Composition 1 : Blocs légers

Cette composition est le résultat de plusieurs tests plus ou moins fructueux. L'objectif était d'obtenir un béton avec une assez bonne plasticité, de sorte à pouvoir remplir uniformément le moule, même dans les angles difficilement atteignables. Un autre critère, d'une importance moindre mais non-négligeable était le poids du béton. La ré-

FIGURE 3.4 – Granulométrie de l'argex concassé

sistance du béton, quant à elle, a été reléguée au second plan car le but était d'avoir des blocs uniformes plus que des blocs résistants. En effet, l'uniformité des faces entraine des contacts plus sains entre les blocs empilés, ce qui est important puisqu'on va essayer de faire coller les résultats de tests en laboratoire avec un modèle numérique, celui-ci utilisant une géométrie parfaite.

La composition par litre de béton est décrite dans le tableau ci-dessous. Le ciment utilisé est un ciment 52,5 (ciment à prise rapide) pour avoir un meilleur rendement (1 bloc par jour). Il convient également de préciser que l'Argex a été concassé de sorte à avoir des granulats aussi bien que des particules plus fines. (Figure 3.4)

| Composants | Quantités [kg/L] |
|---|---|
| Ciment 52,5 | 0.495 |
| Argex concassé | 0.486 |
| Eau | 0.349 |
| Entraineur d'air | 15 ml |
| Plastifiant | 15 ml |

On obtient un béton de masse volumique approximative de 1,494 kg/litre. Les droxels précédemment décrits faisant un volume de 7,696 litres, ils ont donc un poids variant de 11,17 kg à 11,8 kg.

**Composition 2 : Blocs résistants**

Il a fallu mettre au point une seconde composition pour 2 raisons. Premièrement, les blocs de la première composition ne permettaient pas de réaliser certains tests de la section 4.1 car on observait une rupture par cisaillement locale sur certains blocs avant

(a) Composition 1 (b) Composition 2

FIGURE 3.5 – Finition obtenue pour les 2 compositions

le glissement que l'on cherchait justement à quantifier. La deuxième raison est qu'il était important de vérifier si les facteurs de sécurité observés en comparant les tests au modèle numérique dépendaient de la composition du béton. En effet, même si la résistance ou la densité du béton entre en tant que paramètre dans le modèle numérique, il se peut que l'on observe des comportements différents.

On a donc réalisé des blocs de composition différente, avec comme buts principaux la résistance et l'uniformité du béton. La composition finale retenue est indiquée au tableau suivant.

| Composants | Quantités [kg/L] |
|---|---|
| Graviers | 1.078 |
| Sable | 0.733 |
| Ciment 52,5 | 0.394 |
| Eau | 0.178 |

On obtient un béton de masse volumique approximative de 2,417 kg/litre et donc des droxels de 18,6 kg approximativement.

La finition obtenue est plus ou moins similaire pour les 2 compositions. Un exemple pour chaque composition est donné à la figure 3.5.

**Note**

Dans la suite de ce travail, par soucis de clarté, le terme composition 1 sera utilisé pour parler des blocs légers tandis que le terme composition 2 sera utilisé pour parler des blocs résistants.

## 3.2 Résistance du béton utilisé

La résistance du béton est importante car elle est déterminante dans le cas d'une rupture au sein d'un bloc. Les 2 compositions ont donc été testées en compression, chacune sur plusieurs cubes (essai de compression de cube standard).

| Composition | Echantillon | Résistance mesurée [MPa] |
|---|---|---|
| 1 : Blocs légers | 1 | 21.8 |
| | 2 | 21.5 |
| | 3 | 17.1 |
| 2 : Blocs résistants | 1 | 52.4 |
| | 2 | 55.1 |

À titre informatif, un cube de composition 2 après écrasement est montré à la figure 3.6.

FIGURE 3.6 – Blocs de composition 2 écrasé

## 3.3 Coefficient de frottement

Le coefficient de frottement est un facteur qui sera déterminant dans les calculs de stabilité des structures composées de droxels en béton. Le coefficient nous intéressant est bien évidemment le coefficient de frottement statique, car le coefficient de frottement dynamique n'a de l'intérêt que si un mouvement a lieu. Or, un mouvement dans la structure découlerait d'une instabilité. Les valeurs données par les différents documents scientifiques à ce sujet[3] sont de l'ordre de 0,4-0,9. Cette fourchette très large découle de la diversité des bétons.

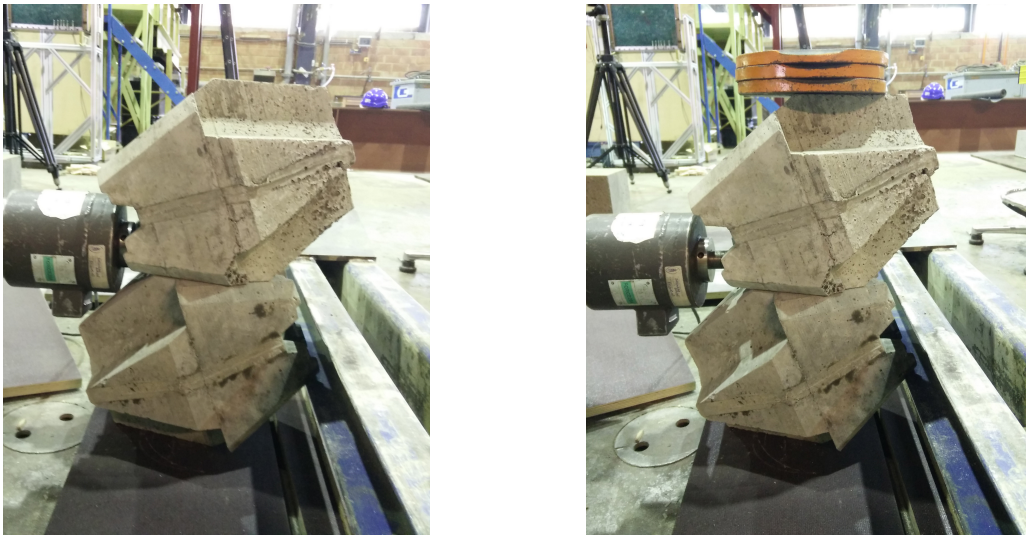
Il est dès lors judicieux d'effectuer des tests sur nos blocs pour avoir une valeur plus précise de ce coefficient.

Le test est réalisé de la manière suivante : 2 éléments sont posés l'un sur l'autre, l'élément inférieur est bloqué dans sa translation tandis que celui du dessus est poussé par un vérin en vitesse constante. Le pic de force enregistré correspond alors à la force nécessaire pour faire passer l'élément de l'état statique à l'état dynamique. On peut ensuite effectuer des tests en surchargeant l'élément supérieur avec des poids normalisés pour obtenir différentes valeurs de force résistante.

Le test a ici d'abord été réalisé avec un des cubes de compostion 1 destinés à la compression. Cependant, les moules de droxels imprimés grâce à l'imprimante 3D présentent des stries dues au mode d'opération de cette machine. En effet, celle-ci imprime couche par couche en partant de la base et on observe donc de très petites rainures entre ces couches. Ce schéma de rainures est donc répliqué sur la surface des droxels coulés dans ces moules, ce qui leur confère un coefficient de frottement plus élevé, du moins dans le sens perpendiculaire aux rainures. On a donc également réalisé des tests avec les droxels plutôt que les cubes et on a observé, comme suspecté, une différence dans les coefficients. Pour la compostion 2, on a directement testé les droxels sans passer par les cubes.

Le montage est représenté à la figure 3.7. Le graphe de la force enregistrée dans le vérin en fonction du déplacement pour le test du droxel de composition 1 surchargé de 18 kg est représenté à titre d'exemple à la figure 3.8.

Les résultats de tous ces tests sont repris dans les tableaux 3.1 et 3.2. On peut voir des coefficients de frottement assez différents d'un test à l'autre. Pour la suite de ce document, on considérera un coefficient de 0.6 car c'est une approche sécuritaire. En effet, moins il y a de frottement, plus on risque le glissement d'un bloc sous une charge latérale comme le vent.



(a) Droxel seul (b) Droxel surchargé

FIGURE 3.7 – Dispositif du test de frottement

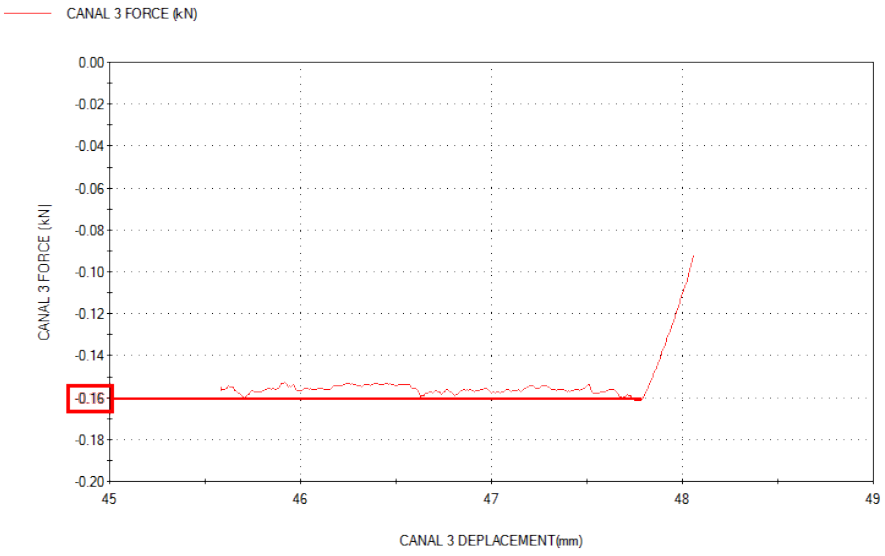



FIGURE 3.8 – Force vs déplacement pour un droxel de compositttion 1 surchargé de 18 kg

| Cube | à vide | +2 kg | +4 kg | +6 kg |
|---|---|---|---|---|
| F vert [N] | 51.375 | 70.999 | 90.615 | 110.235 |
| Fmax hor [N] | 20 | 31 | 45 | 57 |
| Coefficient de frottement | 0.389 | 0.437 | 0.497 | 0.517 |
| Droxel | à vide | +6 kg | +12 kg | +18 kg |
| F vert [N] | 113.129 | 171.98 | 230.849 | 289.709 |
| Fmax hor [N] | 80 | 140 | 150 | 160 |
| Coefficient de frottement | 0.707 | 0.814 | 0.65 | 0.586 |

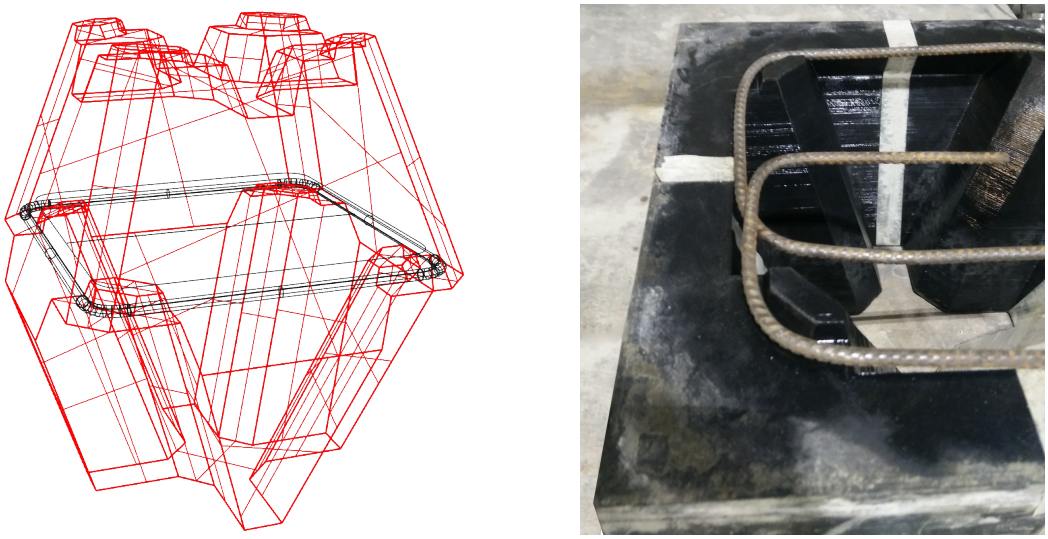
TABLE 3.1 – Résultats du test de frottement (Composition 1)

| Droxel | à vide | +6 kg | +12 kg | +18 kg |
|---|---|---|---|---|
| F vert [N] | 181.485 | 240.345 | 299.205 | 358,065 |
| Fmax hor [N] | 115 | 125 | 206 | 220 |
| Coefficient de frottement | 0.634 | 0.52 | 0.688 | 0.614 |

TABLE 3.2 – Résultats du test de frottement (Composition 2)

## 3.4 Armatures

Après avoir réalisé tous les tests du chapitre 4 [1], il a été décidé qu'il serait intéressant de placer une armature au niveau des encoches pour effectuer quelques tests supplémentaires. On a donc placé une armature de 10 mm sur la partie du moule avec les renfoncements pour les encoches. L'armature a été pliée de sorte à tourner autour des renfoncements pour les encoches tout en laissant un peu d'espace pour que le béton puisse venir l'enrober. Les 2 extrémités de l'armatures se recoupent et rentrent ensuite dans le bloc. (Figure 3.9)



(a) Placement dans le droxel (b) Placement dans le moule

FIGURE 3.9 – Dispositif de renforcement des encoches

1. Même si ce nouveau droxel découle de resultats d'un chapitre postérieur, il semblait judicieux d'en expliquer la fabrication dans ce chapitre-ci

# Chapitre 4

# Calcul de structures

Dans ce chapitre, nous allons détailler comment il est possible de calculer numériquement une structure composée de droxels. Le modèle final permettra de déterminer si une structure potentielle serait stable face aux efforts de vent, poids propre ou autres. Dans le cas d'une stabilité insuffisante, on pourra grâce au modèle redimensionner pour avoir la stabilité nécessaire.

Le chapitre est divisé en deux grandes sections. La première est l'étude du comportement de la structure en supposant les droxels comme rigides. On a alors un mode de rupture qui correspond à un déséquilibre dans la structure. La deuxième est l'étude de la rupture par destruction d'un bloc à cause d'une contrainte trop importante au sein de celui-ci. Les droxels sont alors considérés comme déformables.

## 4.1 Stabilité globale

### 4.1.1 Besoins

Les programmes classiques utilisés dans le secteur de la construction ne permettent pas de calculer une structure composée de droxels. En effet, ceux-ci sont très efficaces pour calculer des efforts et contraintes dans des voiles ou des poutres mais ils sont incapables d'effectuer une bonne analyse d'un élément de construction composé d'un grand nombre de sous-éléments posés les uns sur les autres sans être attachés et de donner la réaction sur chacun de ces sous-éléments.

Il nous faut donc un logiciel qui permettrait de calculer les réactions dans une structure quelconque composée de droxels et de montrer les modes de rupture en cas d'instabilité.

### 4.1.2 Introduction à LMGC90

Etant donné l'incapacité des programmes commerciaux actuels à fournir ce genre de résultats, il a fallu se tourner vers un programme de recherche. Le choix a été fait de travailler avec LMGC90 (Logiciel de Mécanique Gérant le Contact en Fortran90). Ce programme est développé par le Laboratoire de Mécanique et Génie Civil de l'Université de Montpellier. C'est un logiciel gratuit et open source, destiné aux simulations physiques de matériaux et structures discrètes.

Le software fonctionne principalement grâce à des scripts en Python permettant de réaliser des tâches divisées en 3 grandes catégories :

— La construction d'un modèle numérique (*pre-processing*)
— Le calcul (*computation*)
— La visualisation et l'analyse des résultats (*displaying* et *post-processing*)

La compréhension du fonctionnement de ce programme n'est pas aisée et il m'a fallu me rendre à Montpellier pour apprendre à l'utiliser correctement. J'ai réalisé suite à ce déplacement un guide d'aide à l'installation. Celui-ci se trouve dans l'annexe A.

### 4.1.3 Application aux droxels - Calculs en rigides

L'utilisation de LMGC90 est divisée en 3 parties comme expliqué précédemment. La partie pre-processing se concrétise par un script de génération du modèle numérique. Elle permet de définir la géométrie, les efforts, etc et de rassembler toutes les informations nécessaires au calcul dans des fichiers organisés dans un dossier nommé *DATBOX*.

Pour la partie calcul, on a le choix entre 3 différents types de calcul : les calculs en rigides, en solides déformables linéaires et en solides déformables non-linéaires. Dans cette section, on cherche juste à étudier la stabilité sans rupture dans un bloc. On peut donc utiliser le script de calcul pour les solides rigides.

### Théorie du calcul en rigides

De manière générale, LMGC90 se base sur l'équation de mouvement d'un élément pour la réalisation des calculs :

$$\mathbb{M}(\boldsymbol{q},t)\dot{\boldsymbol{V}} = \boldsymbol{F}_{ext}(\boldsymbol{q},\boldsymbol{V},t) + \boldsymbol{F}_{int}(t) + \boldsymbol{R} \tag{4.1}$$

avec :
- $\boldsymbol{q} \in \mathbb{R}^n$ est le vecteur des degrés de liberté généralisés
- $\boldsymbol{V} \in \mathbb{R}^n$ est le vecteur des vitesses généralisées
- $\boldsymbol{R} \in \mathbb{R}^n$ est le vecteur des résultantes des forces de contact
- $\boldsymbol{F}_{ext} \in \mathbb{R}^n$ est le vecteur des forces externes
- $\boldsymbol{F}_{int}(t) \in \mathbb{R}^n$ est le vecteur des forces internes (solides déformables) et des termes d'inertie non-linéaires (centrifuges et gyroscopiques)
- $\mathbb{M}(\boldsymbol{q},t) : \mathbb{R}^n \longmapsto \mathcal{M}^{n\times n}$ est le vecteur des moments externes

Si on considère des solides rigides, on peut utiliser le système d'équations de Newton-Euler en introduisant $\boldsymbol{v}$ la vitesse de translation du centre de masse et $\boldsymbol{\omega}$ la vitesse de rotation du centre de masse :

$$\begin{cases} \mathbb{M}_R\dot{\boldsymbol{v}} = \boldsymbol{F}_{ext}(t) + \boldsymbol{R} \\ \mathbb{J}_R(\boldsymbol{q})\dot{\boldsymbol{\omega}} = \boldsymbol{M}_{ext}(t) + \boldsymbol{M}_R \end{cases} \tag{4.2}$$

avec :
- $\boldsymbol{F}_{ext}$ et $\boldsymbol{R}$ sont les vecteurs de résultantes repectivement des forces extérieures et des forces de contact
- $\boldsymbol{M}_{ext}$ et $\boldsymbol{M}_R$ sont les vecteurs de moments dûs repectivement aux forces extérieures et aux forces de contact

### Algorithme pour le calcul

Le script de calcul contient une boucle temporelle qui fera un nombre défini d'itérations. Le schéma simplifié est représenté à la figure 4.1 [1]. Pour chacune de ces itérations, les actions suivantes sont effectuées.

1. Issue de la page Git de LMGC90

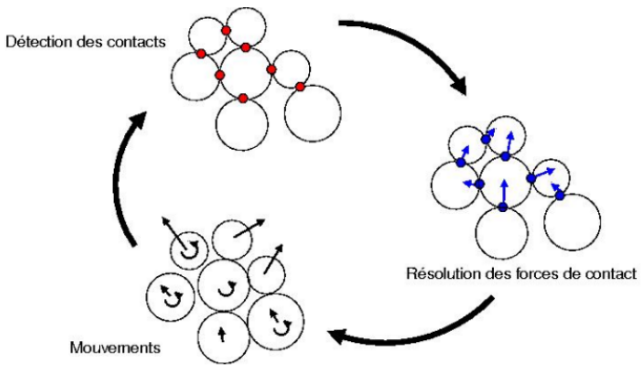



FIGURE 4.1 – Fonctionnement algorithme

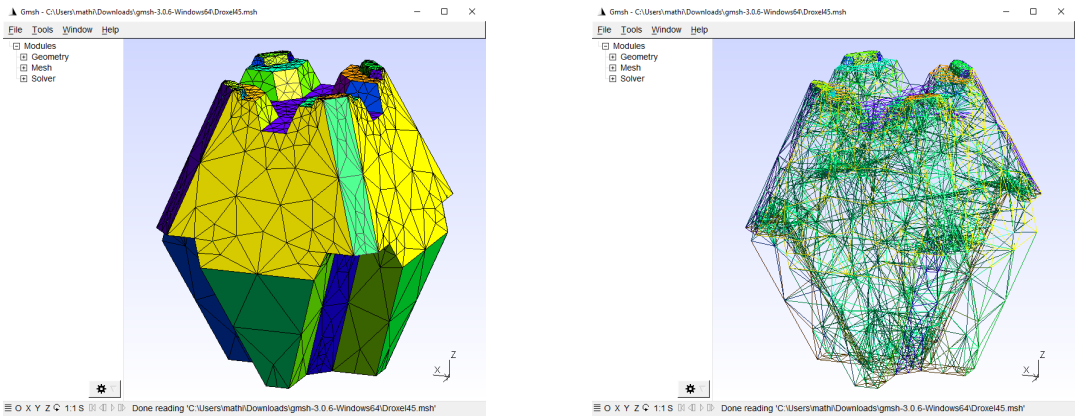
— On laisse le système avancer en vitesse libre (on ne tient pas compte des forces de contact) jusqu'à une configuration intermédiaire. On détecte alors les nouveaux contacts
— Pour l'ensemble des contacts, on obtient les forces de contact sur base des vitesses de translation et de rotation et des coordonnées du point de contact et du centre d'inertie. Cette résolution de système est effectuée avec un solveur de contact comme Gauss-Seidel. Ce solveur effectue un calcul itératif sur le système des contacts. Etant donné que le coût en calcul pour déterminer si le solveur converge à une itération $n$ est identique au coût nécessaire pour effectuer une nouvelle itération, on définit en tant que paramètre le nombre d'itérations entre chaque test de convergence.
— Sur base des forces de contact précédemment déterminées ainsi que des coordonnées, poids, etc des corps, on calcule les nouvelles vitesses de translation et de rotation.
— On stocke ces informations dans une matrice qui servira à la première étape de la prochaine itération

### Maillage

Les corps soumis dans les scripts de génération sur LMGC90 doivent être maillés au préalable. On utilise pour cela Gmsh [2]. Ce programme nous permet d'importer une géométrie au format STEP ou IGES [3] et de la mailler, en choisissant la taille des éléments etc. Le maillage d'un droxel fourni par ce programme est représenté à la figure 4.2. Le maillage joue un rôle important dans l'optimisation. En effet, un maillage trop serré entrainera des calculs trop lourds pour un gain en précision négligeable tan-

2. Disponible sur : http ://gmsh.info/
3. Ces formats sont des formats d'exportation de données graphiques très communs. Un peu à la manière du format PDF.

dis qu'un maillage trop large donnera des imprécisions non-négligeables. Il faut donc mailler consciencieusement. Dans notre cas, le maillage du droxel de 25 cm contient 1171 noeuds et 3661 tétraèdres.




(a) Surfaces visibles (b) Surfaces invisibles

FIGURE 4.2 – Maillage d'un droxel dans Gmsh

### 4.1.4 Différents cas testés numériquement

Grâce à LMGC90 on va pouvoir tester différentes configurations sous différents cas de charge, et ce pour les 2 compositions de béton. Le but étant de valider le modèle numérique via de tests en laboratoire, on va chercher à investiguer tous les types de contacts possibles.

#### Cas 1

Colonne de 3 droxels. Le droxel de base est ancré tandis que le droxel supérieur se voit appliquer une force sur son centre de masse. Cette force est dirigée selon l'axe X ou Y.

Pour modéliser cela sur LMGC90, on crée 3 avatars de droxel que l'on dispose verticalement. Etant donné que le contact doit être calculé et définit par le programme lors des itérations, on laisse un petit espace (ici de l'ordre de 2-3 mm) entre les droxels. Ceux-ci tomberont alors sous l'effet de la gravité, combleront ces quelques mm de vide et s'emboiteront.

Concernant l'application de la force, il faut l'appliquer après que les droxels se soient emboités. Il convient donc de réaliser au préalable une simulation sans cette force pour voir le temps à partir duquel on peut appliquer celle-ci. On crée alors une fonction

qui définira la force comme nulle jusqu'au temps précédemment définit, puis comme non-nulle pour les pas de temps suivants. Etant donné que le programme ne peut pas calculer une force limite pour laquelle il y a une chute de la structure, il faut procéder par essais et erreurs. On va donc devoir définir une force $F_0$ que l'on évalue comme proche de la force limite avant la chute. On peut évaluer très grossièrement cette force avec des calculs manuscrits dans ce cas-ci car la configuration n'est pas très compliquée. On effectue alors la simulation avec cette force et on observe le résultat. En cas de chute, on la diminue. Dans le cas contraire, on l'augmente et on relance la simulation. On itère cette procédure jusqu'à approcher la force de rupture de manière plus précise.

Les paramètres utilisés pour cette simulation sont les suivants :
- matériau : rigide de densité 1494 kg/m$^3$ pour la composition 1 et 2417 kg/m$^3$ pour la composition 2
- force : non-nulle à partir de 0,1 sec et de valeur itérée
- coefficient de friction : 0,6
- delta temps : 0,01
- nombre de pas de temps : 200 (On a donc 2 secondes qui sont calculées)
- theta : 0.5
- frequ_display : 5
- frequ_detect : 20 car les contacts ne changent pas au cours de la simulation

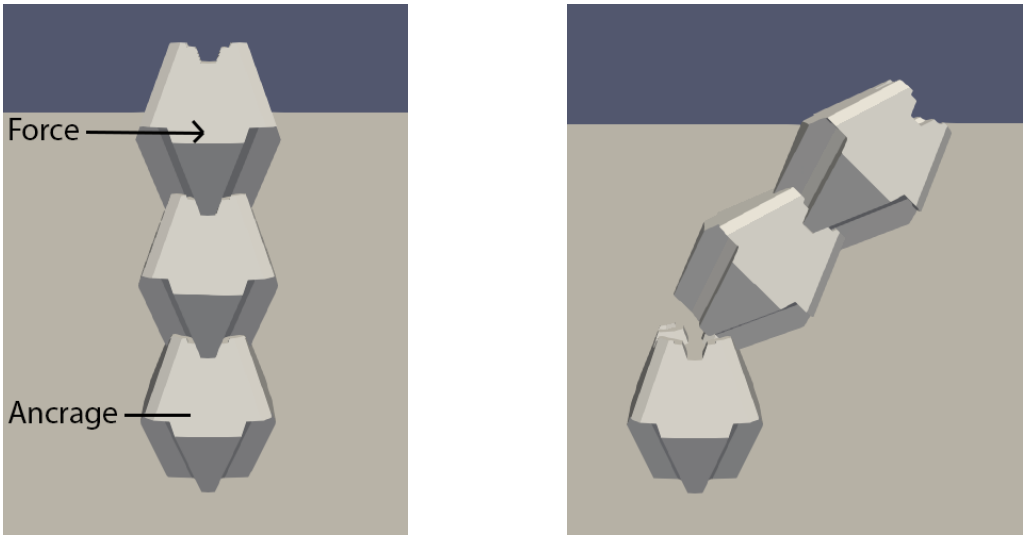
Dans le cas de la composition 1, on constate après itération que la structure est stable pour une force de 32 N et qu'elle ne l'est pas pour une force de 33 N. On peut donc en déduire que la force de rupture se situe dans cet intervalle. Pour la composition 2, ces nombres deviennent 50 et 51 N. Une capture d'écran à $t = 1.35s$ des tests à 32 et 33 N sur la composition 1 est présentée à la figure 4.3

**Cas 2**

Colonne de 3 droxels. Comme dans le cas 1, le droxel inférieur est ancré tandis que le droxel supérieur se voit appliquer une force sur son centre de masse. Par contre, la force est cette fois-ci dirigée à 45° entre les axes X et Y (Figure 4.4). Les paramètres utilisés pour la simulation sont les mêmes que dans le cas 1. Seule la direction de la force change. [4]

Pour la composition 1, on observe une structure stable pour une force de 29 N et une structure instable pour une force de 30 N. Pour la composition 2, on a 46 et 47 N.

4. La condition de force précédente se traduisait par un argument *component* valant [1] et une intensité $F$. La nouvelle condition de force se traduit par un argument *component* valant [1 2] et une intensité valant $F * cos(45)$




(a) 32 N, t=1.35 sec (b) 33 N, t=1.35 sec

FIGURE 4.3 – Extraits de résultats du cas 1 pour la composition 1 (Visualisation dans Paraview)

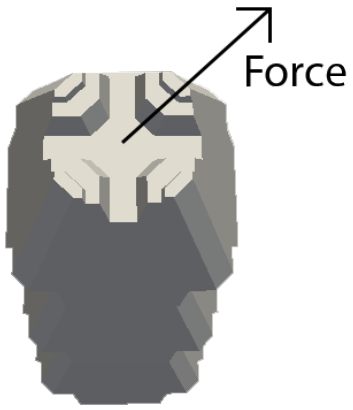
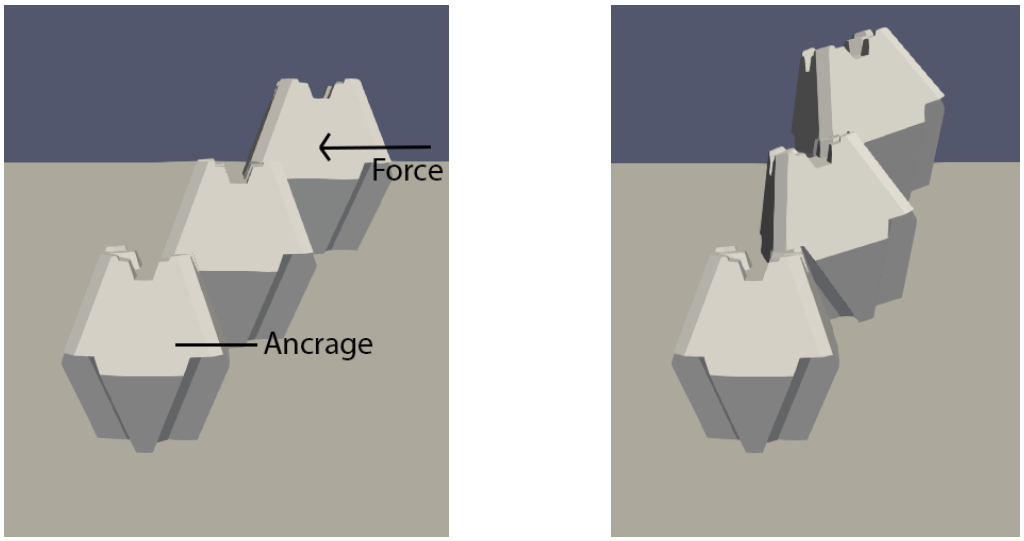



FIGURE 4.4 – Orientation de la force pour les cas 2 et 4




(a) 350 N, t=0.5 sec (b) 375 N, t=0.5 sec

FIGURE 4.5 – Extraits de résultats du cas 3 pour la composition 1 (Visualisation dans Paraview)

**Cas 3**

Porte-à-faux de 2 droxels empilés en diagonale sur un droxel ancré. On applique une force orientée selon l'axe X ou Y sur le droxel supérieur.

On observe une structure stable pour une force de 350 N et une structure instable pour une force de 375 N. Pour la composition 2, on a 575 et 600 N. Le test sur la composition 1 est représenté à la figure 4.5.

**Cas 4**

Porte-à-faux de 2 droxels empilés en diagonale sur un droxel ancré. On applique cette fois une force orientée à 45° entre les axes X et Y sur le droxel supérieur.

On observe une structure stable pour une force de 400 N et une structure instable pour une force de 425 N. Pour la composition 2, on a 675 et 700 N.

**Cas 5**

2 droxels empilés en diagonale dans 2 directions différentes sur un droxel ancré. On applique une force sur le droxel supérieur. La configuration est représentée sur la figure 4.6. La force appliquée ici est représentée en bleu.

On observe une structure stable pour une force de 400 N et une structure instable pour une force de 425 N. Pour la composition 2, on a 475 et 500 N.

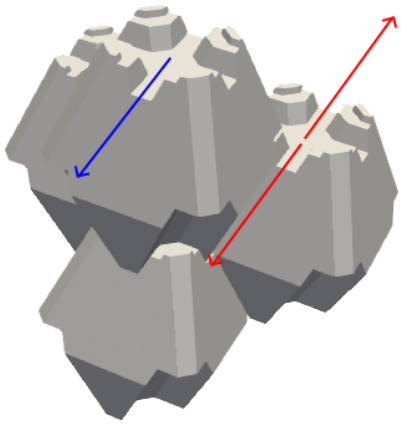


FIGURE 4.6 – Configuration du cas 5

Si on applique en revanche cette force à un des droxels intermédiaires (quelle que soit l'orientation de la force) comme représenté en rouge sur la figure 4.6, on observe une augmentation drastique de la force de rupture. On a pour la composition 1 des valeurs de l'ordre de 350 000 N. Ces résultats tendent à prouver qu'il n'est pas nécessaire d'étudier la stabilité au renversement/glissement d'un droxel non supérieur dans une structure car il y aura une rupture au sein d'un droxel bien avant une instabilité due à un glissement ou un renversement. Il convient bien évidemment de vérifier cette conjecture lors de tests en laboratoire comme au chapitre suivant.

### Influence du coefficient de frottement

Dans les cas 1 et 2, le coefficient de frottement n'a que très peu d'influence puisqu'on a une rupture par renversement et non par glissement. Augmenter le coefficient ne se traduira pas par une augmentation de la force de rupture. Cependant, on peut vérifier pour le cas 3 l'influence du coefficient de frottement sur cette force. On réitère donc les tests du cas 3 avec des coefficients de frottement différents. Les résultats sont repris dans le tableau 4.1.

On observe une augmentation (non-linéaire) considérable de la force de rupture avec le coefficient de frottement. On a donc bien fait de préférer l'approche sécuritaire en prenant un petit coefficient pour les simulations.

| Coeff de frottement | Test stable [N] | Test instable [N] |
|---|---|---|
| 0.5 | 300 | 325 |
| 0.6 | 350 | 375 |
| 0.7 | 425 | 450 |
| 0.8 | 550 | 575 |

TABLE 4.1 – Influence du coefficient de frottement sur la force rupture (Configuration du cas 3 et composition 1)

### 4.1.5 Tests en laboratoire

Avant d'entamer cette section, voici un petit rappel des résultats des tests numériques pour les 5 cas et les 2 compositions de béton. La force de rupture a été prise comme la moyenne entre la force pour laquelle on observait la stabilité de la structure et celle pour laquelle on avait une instabilité. Elle est donc plus précise lorsque cette fourchette est fine et on peut gagner en précision si l'on effectue un grand nombre de tests pour réduire la largeur de cette fourchette.

| Cas | Composition | Force de rupture [N] |
|---|---|---|
| 1 | 1 : Blocs légers | 32.5 |
| | 2 : Blocs résistants | 50.5 |
| 2 | 1 : Blocs légers | 29.5 |
| | 2 : Blocs résistants | 46.5 |
| 3 | 1 : Blocs légers | 362.5 |
| | 2 : Blocs résistants | 587.5 |
| 4 | 1 : Blocs légers | 412.5 |
| | 2 : Blocs résistants | 687.5 |
| 5 | 1 : Blocs légers | 412.5 |
| | 2 : Blocs résistants | 487.5 |

TABLE 4.2 – Récapitulatif des tests numériques

#### Méthode

Les tests ont été réalisés de la manière suivante. On a disposé les blocs de composition de béton adéquate dans la configuration de chaque cas. Pour les tests sur les blocs de composition 1, l'ancrage du droxel inférieur a été matérialisé par une dalle de béton coulée autour de celui-ci, suffisamment lourde pour empêcher tout mouvement. Pour les tests sur les blocs de composition 2, on a fixé dans un droxel deux tiges filetées qui permettent d'attacher celui-ci à une poutre (Figure 4.7) tout en pouvant déplacer

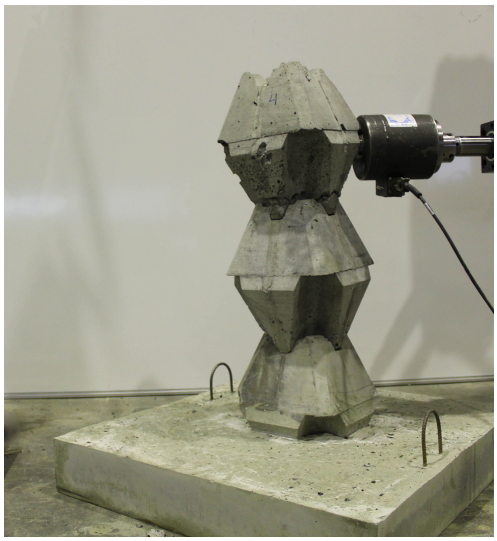
FIGURE 4.7 – Ancrage pour la composition 2

latéralement ce point de fixation en fonction des besoins du tests (le verrin n'étant pas déplaçable, il faut donc déplacer l'ancrage d'un test à l'autre). Cette poutre est elle-même fixée au plancher d'essai grâce à des attaches métalliques.

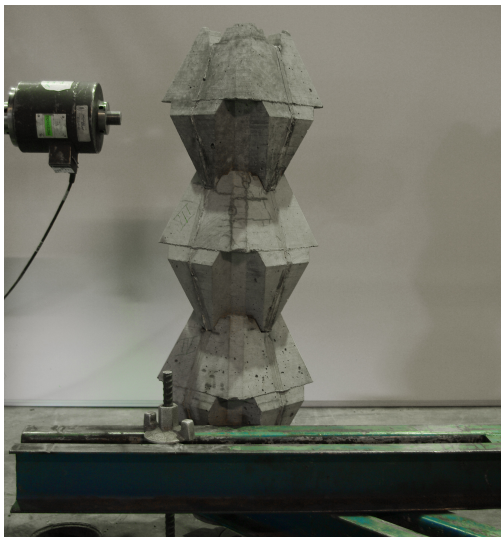
Les 2 ancrages ainsi que le reste de la configuration des tests sont représentés à la figure 4.8

La force est appliquée au moyen d'un vérin et en déplacement constant. On prend mesure de la force appliquée en fonction du déplacement et on considère comme force de rupture le pic de force appliquée par le vérin atteint au cours de l'essai. On a cependant de très nombreux pics qui rendent le graphe moins net et moins précis. On a donc choisi de lisser ces pics en affichant la courbe de la moyenne mobile sur 5 points. A titre d'exemple, la figure 4.9 montre le résultat d'un test sur le cas 1 avec la composition 1 [5]. On y voit en bleu le graphe des données brutes et en rouge le graphe de la moyenne mobile. On y observe un pic de force de 39 N. On observe également une perturbation dans la montée en force. Celle-ci est probablement due à une irrégularité sur une surface de béton.

5. le vérin commence à un position de 80mm et le déplacement se matérialise par une diminution de cette valeur



(a) ancrage composition 1



(b) ancrage composition 2

FIGURE 4.8 – Exemple de configuration (cas 2) pour les tests sur les 2 compositions

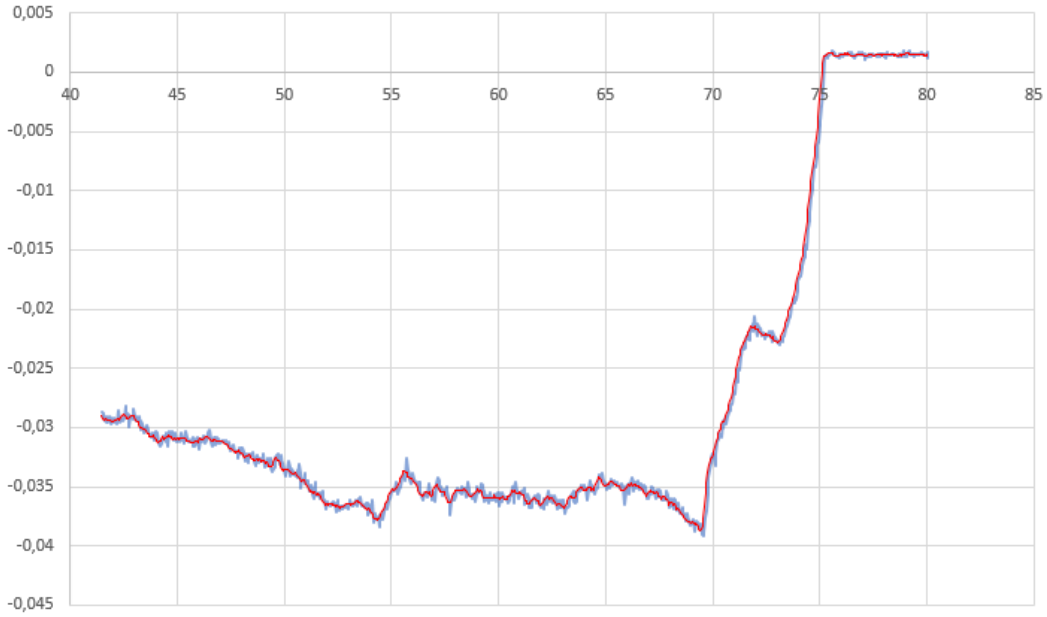



FIGURE 4.9 – Force [kN] en fonction du déplacement [mm] pour le test 4 sur le cas 1

**Cas 1**

Pour cette configuration, on a réalisé 10 tests avec les blocs de composition 1 en changeant à chaque fois les 2 blocs supérieurs utilisés pour le test ainsi que leurs orientations. On a également réalisé 4 tests avec les blocs de composition 2 (le nombre de blocs de cette composition disponibles pour les tests était plus limité). Les résultats sont repris dans le tableau 4.3.

| Composition | Test | Force de rupture [N] | Moyenne [N] | F.S. |
|---|---|---|---|---|
| 1 : Blocs légers | 1 | 43 | 50.1 | 1.54 |
| | 2 | 59 | | |
| | 3 | 39 | | |
| | 4 | 51 | | |
| | 5 | 51 | | |
| | 6 | 52 | | |
| | 7 | 52 | | |
| | 8 | 50 | | |
| | 9 | 52 | | |
| | 10 | 52 | | |
| 2 : Blocs résistants | 1 | 65 | 71.67 | 1.42 |
| | 2 | 86 | | |
| | 3 | 67 | | |
| | 4 | 62 | | |

TABLE 4.3 – Résultats des tests en laboratoire sur le cas 1

Pour la composition 1, on observe une bonne constance des résultats. La moyenne est de 50,1 N. On notera cependant la présence de tests avec une résistance près de 20% plus faible que cette moyenne. Le résultat numérique étant de 32,5 N, on a un facteur de sécurité de $50,1/32,5 = 1,54$.

Pour la composition 2, les résultats montrent une force de rupture de 71,67 N en moyenne. Le résultat numérique étant de 50,5 N, on a un facteur de sécurité de 1,42.

**Cas 2**

Pour cette configuration, on a réalisé 4 tests avec les blocs de composition 1 en changeant à chaque fois les 2 blocs supérieurs utilisés pour le test ainsi que leurs orientations. On a également réalisé 4 tests avec les blocs de composition 2. Les résultats sont repris dans le tableau 4.4.

| Composition | Test | Force de rupture [N] | Moyenne [N] | F.S. |
|---|---|---|---|---|
| 1 : Blocs légers | 1 | 22 | 22.33 | 0.76 |
| | 2 | 19 | | |
| | 3 | 22 | | |
| | 4 | 26 | | |
| 2 : Blocs résistants | 1 | 113 | 83.33 | 1.79 |
| | 2 | 76 | | |
| | 3 | 52 | | |
| | 4 | 122 | | |

TABLE 4.4 – Résultats des tests en laboratoire sur le cas 2

Pour la composition 1, la moyenne est de 22,33 N. Le résultat numérique étant de 29,5 N, on a un coefficient de sécurité de 0,76, ce qui est inférieur à 1. On a donc pour ce cas une sous-évaluation du risque au renversement avec la simulation numérique. Après observation minutieuse des droxels suite au test, on se rend compte que les parties de la base du droxel supérieur en contact avec le droxel central s'effritent, on perd donc la résistance due au frottement prévue par le calcul numérique en rigides.

Pour la composition 2, les droxels ne s'effritent pas lors du test et l'on observe des résultats collant mieux à la simulation numérique. En effet, on a une force de rupture moyenne de 83,33 N et donc un facteur de sécurité de 1,79 car le modèle numérique prévoit une rupture à 46,5 N. Il est cependant important de noter un éparpillement plus important des résultats en termes de force de rupture et plus particulièrement un test ayant rompu à 52 N, ce qui donne un facteur de sécurité singulier de 1,118.

**Cas 3, 4 et 5**

Pour ces 3 configurations, la presque totalité des tests avec les 2 compositions de béton ont mené à une rupture des encoches avant de mener à un glissement (Figure 4.10). On n'a donc pas pu quantifier le glissement ici.

Sauf dans le cas 4 avec la composition 2. En effet, on observe une montée en charge jusque 1300 N et une cinématique de rupture très similaire à celle fournie par la simulation numérique correspondante. La comparaison peut se faire en observant les figures 4.11 et 4.12 qui montrent respectivement le mode de rupture de la simulation numérique et le mode de rupture du test en laboratoire.

FIGURE 4.10 – Exemple de rupture d'encoche (cas 3, 4 et 5)

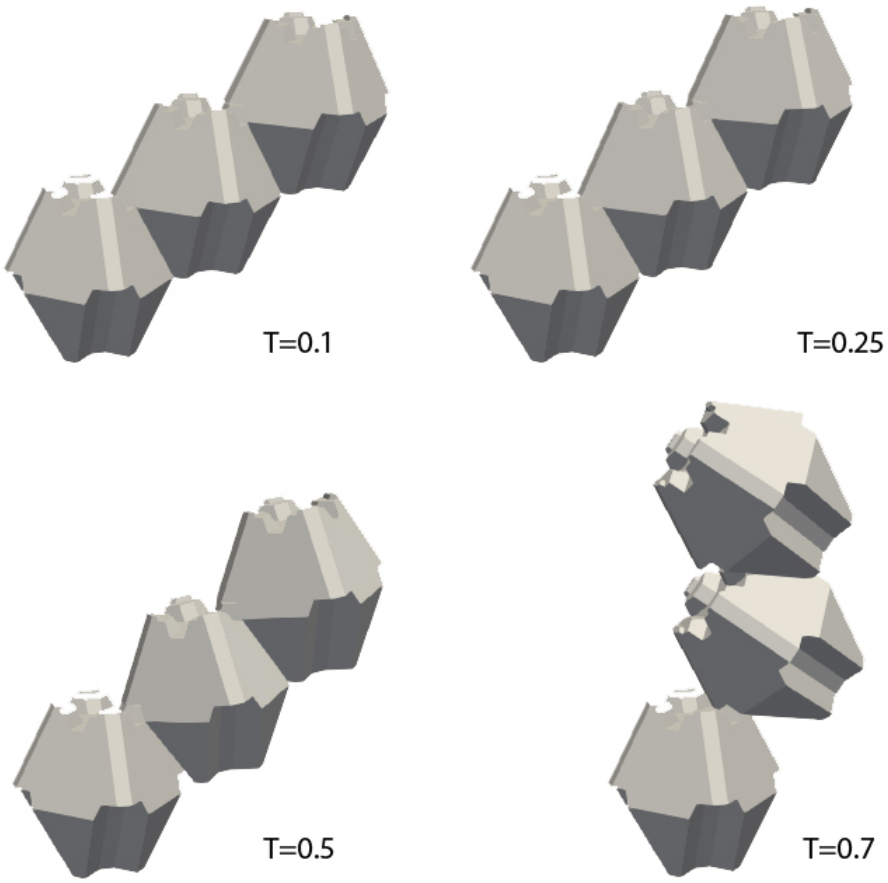



FIGURE 4.11 – Mode de rupture de la simulation numérique du cas 4

(a) Vue générale

(b) Zoom

FIGURE 4.12 – Mode de rupture du cas 4 en laboratoire

### 4.1.6 Analyse des résultats

De manière générale, si l'on exclut bien tous les tests ayant donné lieu à une rupture d'encoche ou micro rupture dans les blocs, les résultats expérimentaux correspondent aux simulations numériques avec un facteur de sécurité moyen de plus ou moins 1,5. Aucun test n'a donné lieu à une sous-évaluation numérique de la résistance d'une configuration et la plus grande surévaluation s'est traduite par un facteur de sécurité singulier de 2,62. On observe cependant un écart-type systématiquement plus important dans les tests sur la composition 2.

Les tests entrepris tendent donc à démontrer que LMGC90 est adapté pour calculer numériquement la stabilité globale (sans rupture au sein des blocs) de structures composées de droxels. Il serait bien-entendu judicieux d'effectuer plus de tests en faisant varier la taille des droxels, les configurations et les compositions de béton avant de pouvoir affirmer cela avec certitude.

Il serait également intéressant de simuler un séisme en faisant varier de manière sinusoïdale la vélocité du plan d'appui dans la simulation numérique. Cependant, vérifier les résultats en laboratoire s'avérerait extrêmement laborieux car il faudrait ancrer les droxels dans un plan qui puisse imiter ces déplacements.

## 4.2 Résistance aux ruptures internes

### 4.2.1 Besoins

Dans la section précédente, on a pu observer que les instabilités par glissement ou renversement n'avaient lieu que dans les cas de droxels empilés croix sur croix. Dans la presque totalité des tests qui impliquaient des droxels empilés encoche sur tête, on observait une rupture au sein d'un droxel avant toute autre forme d'instabilité. Il est donc important d'étudier ce type de comportement.

On va donc utiliser le calcul en déformable sur LMGC90 pour prédire ce genre de comportement.

### 4.2.2 Calculs en déformables

En déformables, l'équation descriptive générale (4.1) ne peut être transformée pour arriver aux équations de Newton-Euler (4.2) comme dans le cas des rigides. Le programme utilisera donc cette équation pour calculer les vitesses de translation et rotation sur base des forces de contact. La résolution des contacts quant à elle se fait de la même manière.

Le coût des calculs en déformable est beaucoup plus important. À titre d'exemple, le calcul en rigide du cas 1 a nécessité 1 minute de calcul [6], alors que le calcul en déformable du même cas a nécessité 7 minutes.

Dans certains cas avec un nombre plus important de corps, le calcul nécessite une quantité de RAM excédant celle de la machine. Le calcul est alors infiniment long et il faut donc considérer l'utilisation d'un supercalculateur. Même si cela n'a pas été fait dans le cadre de ce mémoire, LMGC90 s'adapte très facilement à ce genre d'utilisation car il nécessite juste Linux pour fonctionner et produit des résultats sous forme de fichiers dans des dossiers que l'on peut facilement récupérer et visualiser sur une autre machine.

#### Aide pour la détection des contacts

En effectuant les premières simulations numériques, on s'est rendu compte que le logiciel ne détectait pas correctement les contacts entre les différents droxels. Il a donc fallu lui simplifier la tâche en définissant des *physical groups* dans le maillage. Ces groupes ont en fait permis de dire quel groupe de surface devait être en contact avec quel autre groupe de surface, ce qui aide le logiciel à détecter les bons contacts entre

6. Machine utilisée pour le calcul : MacBook Air 11" 2013 (1.7 GHZ Intel Core i7, 8 Go 1600 MHz DDR3, Intel HD Graphics 5000 1536 Mo)

les droxels.

Le fichier STEP utilisé par Gmsh pour le maillage contient les informations nécessaires à l'établissement de ces groupes. Si on l'ouvre dans Gmsh et que l'on active, dans les options, la visibilité des labels de surface, on obtient les numéros associés à chaque surfaces (Figure 4.13). On groupe donc les surface par 4, 8 ou 12 et on les met dans un fichier *.geo* qui pourra ensuite être lu par Gmsh pour créer un maillage tout en ajoutant ces *physical groups*. Ce fichier est repris ci-dessous.

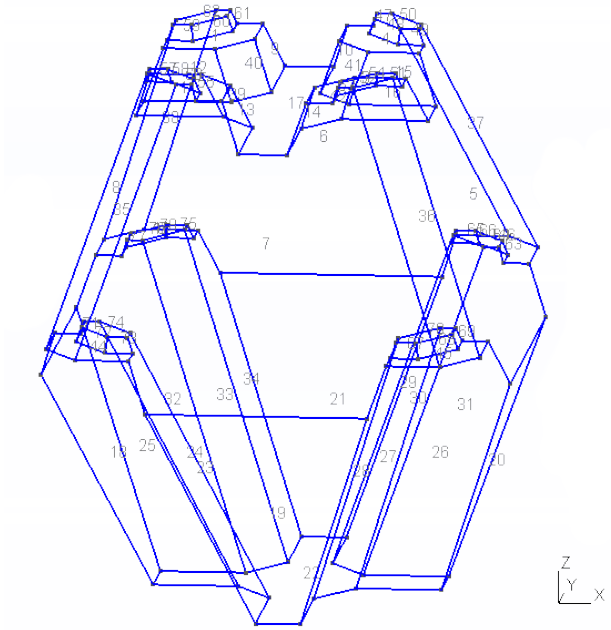



FIGURE 4.13 – Numéros des surfaces dans Gmsh

```
1  Merge "Droxel25ajuste.stp";
2  //+
3  Physical Surface("down") = {22};
4  //+
5  Physical Volume("droxel") = {1};
6  //+
7  Physical Surface("mid_p") = {45, 44, 46, 43};
8  //+
9  Physical Surface("top_p") = {3, 2, 4, 1};
10 //+
11 Physical Surface("top") = {17};
12 //+
13 Physical Surface("top_v") = {41, 40, 42, 39, 10, 14, 16, 15, 12,
       11, 13, 9};
14 //+
```

```
15 Physical Surface("mid_e") = {72, 73, 71, 67, 68, 69, 63, 64, 65,
       75, 76, 77};
16 //+
17 Physical Surface("top_e") = {47, 48, 49, 51, 52, 53, 55, 56, 57,
       61, 60, 59};
18 //+
19 Physical Surface("down_v") = {31, 29, 28, 26, 23, 25, 32, 34};
20 //+
21 Physical Surface("top_ev") = {35, 36, 37, 38};
22 //+
23 Physical Surface("down_ev") = {24, 27, 30, 33};
24 //+
25 Physical Surface("top_top") = {54, 58, 62, 50};
26 //+
27 Physical Surface("encoches") = {66, 78, 74, 70};
28 //+
29 Physical Surface("oblique_top") = {5, 6, 7, 8};
30 //+
31 Physical Surface("oblique_bot") = {18, 19, 20, 21};
```

La première ligne est une commande demandant d'utiliser la géométrie du fichier STEP. Les lignes suivantes correspondent à l'ajout de groupes. Ceux-ci seront utilisés dans le script de génération de la géométrie pour définir des paires candidat/antagoniste. (Annexe B.3 lignes 34 à 50 et 128 à 161)

### 4.2.3 Différents cas testés (laboratoire et numérique)

Dans les cas 3,4 et 5 testés en laboratoire dans le but de quantifier le glissement, on a observé une rupture d'encoche avant un glissement. On va donc dans cette section, à l'aide du calcul en déformable de LMGC90, essayer d'analyser ces ruptures numériquement pour ensuite comparer ces résultats numériques à ce que l'on avait obtenu précédemment, et éventuellement, réaliser de nouveaux tests en laboratoire pour compléter les précédents.

En effet, les calculs en déformable étant très couteux, on ne procèdera pas par itération pour trouver la force limite avant instabilité comme pour les calculs en rigides. On va plutôt effectuer un seul calcul par cas, en appliquant une force d'une valeur égale à la rupture observée au laboratoire et ensuite comparer les contraintes observées dans le droxel. On va voir si les zones les plus sollicitées dans les résultats numériques correspondent à la zone de rupture observée en laboratoire et on va comparer les contraintes principales fournies par le programme avec les contraintes de rupture obtenues dans la section 3.2 dans le but d'éventuellement dégager un coefficient de sécurité.

### Cas 3

Pour ce cas, on avait observé lors des 3 tests réalisés en laboratoire avec la composition 1 des ruptures d'encoches à 17, 18 et 17,5 N. On va donc appliquer cette force sur le droxel supérieur. On va s'intéresser à 2 types de résultats.

Premièrement, on va analyser la contrainte de Von Mises. En effet, même si le critère de Von Mises n'est utilisé que pour les aciers ductiles, il a l'avantage de permettre une bonne visualisation de la transmission de la contrainte tout en étant représenté par un simple scalaire. De plus, l'affichage de cette contrainte est implémenté d'origine dans le script de calcul et est peu coûteux en ressources. La figure 4.14 représente cette contrainte après 2 secondes de simulation. On accordera peu d'importance aux valeurs de ces contraintes, on s'intéresse surtout à leur répartition. On peut effectivement observer que les encoches sont sujettes à une grande concentration de contraintes, plus spécifiquement celle du bloc intermédiaire car elle subit les actions due aux 2 droxels supérieurs.

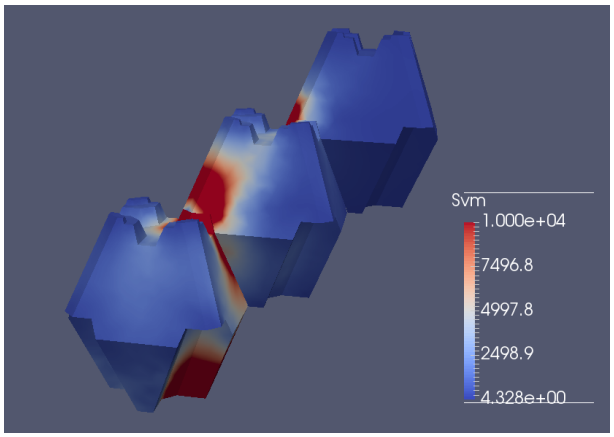



FIGURE 4.14 – Contrainte de Von Mises pour le cas 3, composition 1, t = 2 sec

Ensuite, on va s'intéresser aux contraintes principales. Ce sont elles qui sont plus susceptibles d'indiquer la rupture. Cependant, elles sont récupérables sous forme de vecteurs et l'affichage est donc beaucoup plus lourd. La figure 4.15 montre la contrainte principale la plus importante en t = 2 secondes. On observe une concentration très importante de cette contrainte dans l'encoche faisant le lien entre le droxel ancré et les 2 droxels en porte-à-faux, ce qui est justement l'endroit où l'on a pu observer une rupture en laboratoire. On peut aussi voir une grande contrainte en compression à la base du droxel ancré, ce qui est tout à fait plausible. Cependant, la faiblesse du béton est sa faible résistance à la traction.

Dans le cas de notre composition 1, on a une résistance moyenne avoisinant les 20

MPa pour un cube. On va donc assimiler ce béton à un béton de classe C16/20. On sait que pour ce béton on a $f_{ctm} = 1,76$ MPa. Il est difficile d'établir et de quantifier un lien de proportionnalité entre le résultat numérique et la résistance attendue du béton. On peut cependant retenir qu'une contrainte principale de $1.923 * 10^4$ a mené à une rupture dans cette configuration.

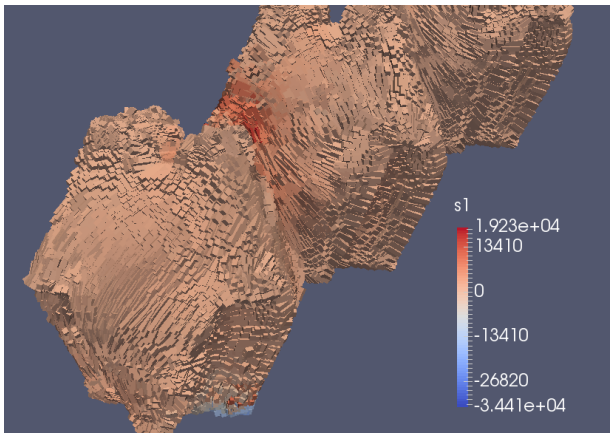



FIGURE 4.15 – Première contrainte principale pour le cas 3, composition 1, t = 2 sec

### Cas 5

La simulation numérique du cas 5 nous donne une concentration des contraintes qui correspond également au mode de rupture observé en laboratoire. Les valeurs de contraintes principales obtenue via la simulation sont, comme pour le cas précédent, difficilement reliables à la valeur de rupture réelle.

Dans le but d'observer des interactions rigide - déformable et pour simuler la présence d'un bloc en béton armé, on a réitéré le test avec comme bloc supérieur un rigide pour observer les contraintes dans les blocs intermédiaires. Les contraintes de Von Mises de ces 2 blocs pour cette simulation sont représentées à la figure 4.16.

### Influence des armatures

Comme indiqué à la section 3.4. On a réalisé un droxel à encoches renforcées. Certains tests ont été faits avec ce droxel et les résultats ont toujours été les mêmes. On observait non plus une rupture d'encoche mais plutôt une rupture dans la tête complémentaire à cette encoche. Ce changement de mode de rupture s'accompagnait bien sûr d'un gain en résistance de la structure.

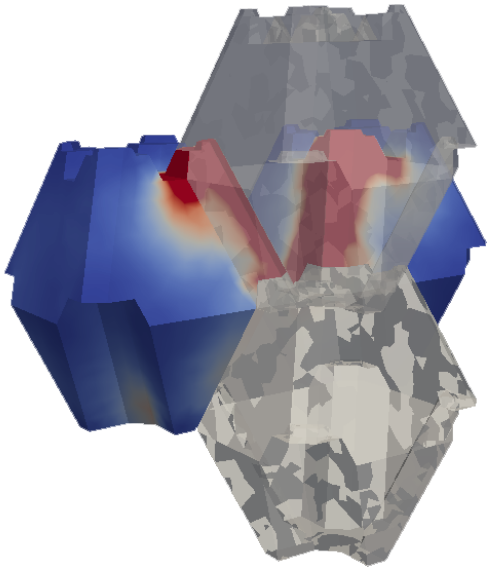


FIGURE 4.16 – Contrainte de Von Mises pour le cas 5 avec un droxel supérieur rigide, composition 1, t = 1 sec

### 4.2.4 Analyse des résultats

Le calcul en déformable sur LMGC90 nous fournit une très bonne idée des faiblesses internes liées aux différentes configurations. Il permet d'avoir indication de l'endroit de la rupture ainsi que du mode de rupture.

Cependant, le calcul ne nous permet pas encore de définir s'il y a rupture ou non pour une force donnée. Les droxels qui ont été testés ne faisant que 25 cm de hauteur, leurs encoches présentaient des sections assez petites et on a donc des irrégularités au niveau de la composition du béton à cet endroit (parfois il y a un gros gravier en travers de cette section, parfois il n'y en a aucun). De plus, même si la simulation pouvait déterminer si la contrainte entraine une rupture, les calculs demandent beaucoup de ressources et il serait donc difficile de procéder par simples itérations pour arriver à une force de rupture précise. On pourrait cependant effectuer la simulation avec la charge en service ou à l'état limite ultime pour obtenir la contrainte maximale dans la structure et en déduire un coefficient de sécurité par rapport à la résistance du béton. Il faudrait pour cela développer plus en profondeur le calcul de ces structures en déformables sur LMGC90.

# Chapitre 5

# Applications

## 5.1 Limites dans la réalisation

Au regard des valeurs des forces de rupture obtenues dans les différentes configurations testées, on pourrait déduire qu'il est utile d'utiliser au maximum les emboitements tête sur encoche et d'éviter si possible les colonnes non chargées. On a aussi pu observer que les porte-à-faux pouvaient entrainer une concentration de contraintes très élevée dans les encoches. On va détailler dans cette section le comportement de deux éléments et les risques qui en découlent.

### 5.1.1 Réaction d'une colonne face au vent

Le premier élément intéressant à analyser est la colonne simple de droxels. En effet, elle fort sujette au renversement. On va ici analyser la hauteur maximale acceptable avant que l'on ai un risque de renversement trop important. On prend comme force due au vent une pression de 2000 $N/m^2$ que l'on applique à chaque droxel surplombant le droxel d'ancrage. Le sens critique d'application de cet effort est celui du cas 2 de la section 4.15 et non pas du cas 1. En effet, d'une part, on voit dans les résultats de ces tests que la colonne résiste moins bien à cette orientation de force et d'autre part, on a une aire exposée plus importante. L'aire prise en compte pour calculer la force de vent est la projection du droxel sur un plan perpendiculaire à la direction du vent. Cela nous met du coté de la sécurité car, les faces étant en fait pour la plupart orientées à 45° par rapprot à cette direction, le vent ne les heurte pas de face et la force est donc moindre.

On a donc réalisé des nouveaux tests numériques (calcul en rigides) avec comme paramètres le composition de béton, le nombre et la taille des droxels. Plus la taille des droxels est élevée, plus la structure est stable. En effet, le moment stabilisant est du au poids du droxel et donc au cube de la hauteur tandis que le moment déstabilisant du au vent dépend de l'aire exposée et donc du carré de la hauteur. Les résultats de ces tests montrent que même dans le cas le moins critique (colonne de 2 droxels de 45 cm

de composition 2), la structure est instable. Il convient de rappeler que l'hypothèse de l'aire prise en compte pour l'action du vent est très sécuritaire. Cependant, ces résultats nous montrent qu'il faut éviter de laisser une colonne non-chargée, ce dont il faudra tenir compte lors du pré-design de structures.

### 5.1.2 Porte-à-faux vs Arche

Un autre élément important que l'on a pu observer dans le chapitre 4 est que les porte-à-faux simples résistent moins à la charge que les arches. Les calculs en déformables ne nous permettant pas encore de quantifier de manière précise les contraintes au sein des droxels, on ne peut pas comparer de manière efficace ces 2 configurations mais il est certain qu'une arche sera toujours plus sécuritaire qu'un porte-à-faux.

De la même manière, le fait de relier 2 porte-à-faux opposés comme sur la figure 5.1 permet de ramener de l'effort vers le centre sans le faire passer par le porte-à-faux. On allège donc la contrainte dans la zone critique (encoches des premiers droxels en porte-à-faux).

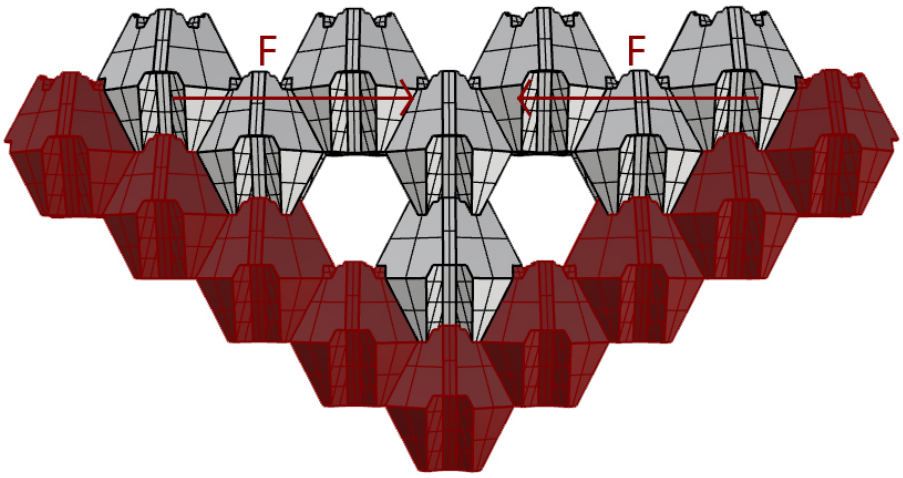



FIGURE 5.1 – Transmission d'effort dans 2 porte-à-faux

## 5.2 Quelques structures

Cette section reprend quelques exemples de structures de droxels. On limitera à peu près à 100 kg le poids maximum d'un bloc, ce qui se traduira par une hauteur maximale entre deux droxels de 45cm (à la place des 25cm comme dans les sections précédentes). Le coût des calculs étant relativement élevé et les démarches encore assez lourdes, on ne procèdera au dimensionnement que de la dernière structure.

### 5.2.1 Passerelle piétonne

On a ici une petite passerelle piétonne constituée de bloc de 45 cm de hauteur. La structure est relativement lourde puisqu'elle est constituée de 314 blocs de 100 kg. Elle fait donc 31 tonnes, ce qui est largement supérieur à ce que l'on attend pour ce type d'ouvrage. Il serait donc intéressant d'essayer de creuser les colonnes et l'alléger également de manière réfléchie la partie surplombant ces colonnes. (Figure 5.2)

### 5.2.2 Phare

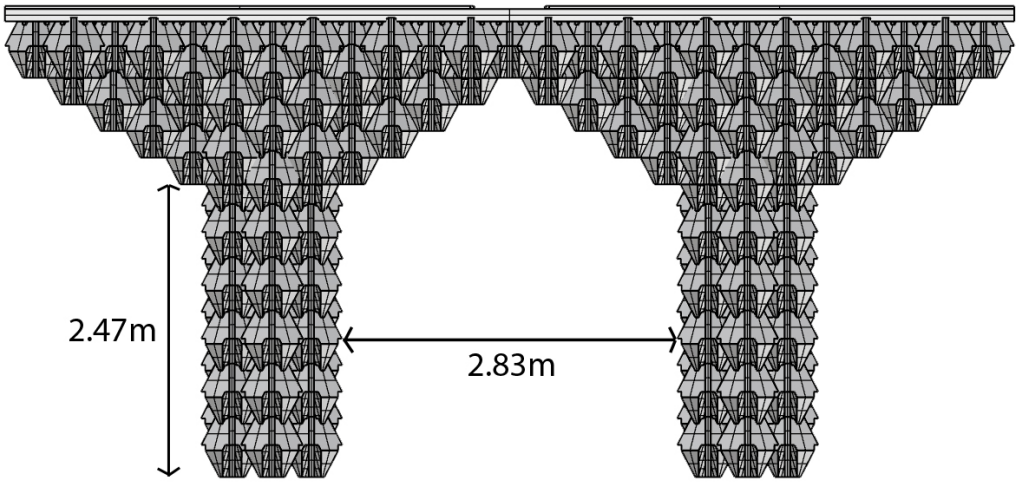
Il est facilement concevable que l'on doive construire un phare dans une zone difficilement accessible pour des camions ou autres. Il serait alors difficile d'acheminer des éléments préfabriqués, du matériel ou même de la main d'oeuvre sur le site du chantier. Les droxels montreraient alors toute leur utilité puisqu'ils peuvent être acheminés sur le chantier via un drone. Bien sûr, il faudrait quand même réaliser des essais géotechniques, préparer des ancrages et superviser la construction, ce qui implique quand-même qu'un accès au chantier soit possible. Cependant, les droxels pourraient simplifier la phase de construction.

Un exemple de phare constitué de droxels est donné à la figure 5.3. La structure comporte 477 blocs.

### 5.2.3 Abri couloir

Cette structure est présentée et dimensionnée dans la section suivante.

(a) Vue 3D




(b) Vue en plan

FIGURE 5.2 – Passerelle piétonne

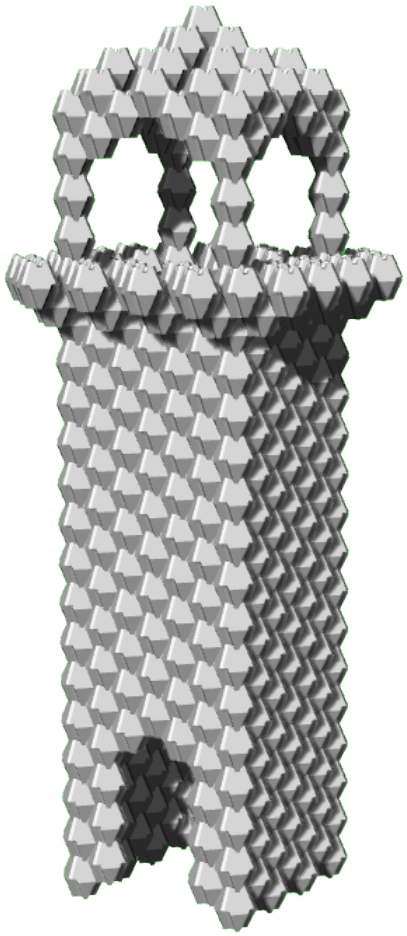


FIGURE 5.3 – Phare constitué de droxels

## 5.3 Cas pratique et note de calcul

### 5.3.1 Structure étudiée

Dans le but de pouvoir mettre en pratique les outils développés jusque qu'ici, on a décidé de réaliser une structure et d'analyser sa stabilité. La structure choisie est un abri longitudinal destiné à faire le lien entre 2 bâtiments. La structure est réalisée avec des droxels du pas vertical de 45 cm car il faut laisser une ouverture de minimum 2m et cela permet de limiter le nombre de blocs (jusqu'ici on utilisait des droxels avec un pas vertical de 25 cm).

La structure est composée de colonnes de 4 droxels sur lesquelles reposent des voutes auxquelles on a retiré l'élément central pour leur donner une forme plus circulaire. La structure est représentée à la figure 5.4 et la géométrie est détaillée à la figure 5.5.

### 5.3.2 Simulations

#### Stabilité face au poids propre

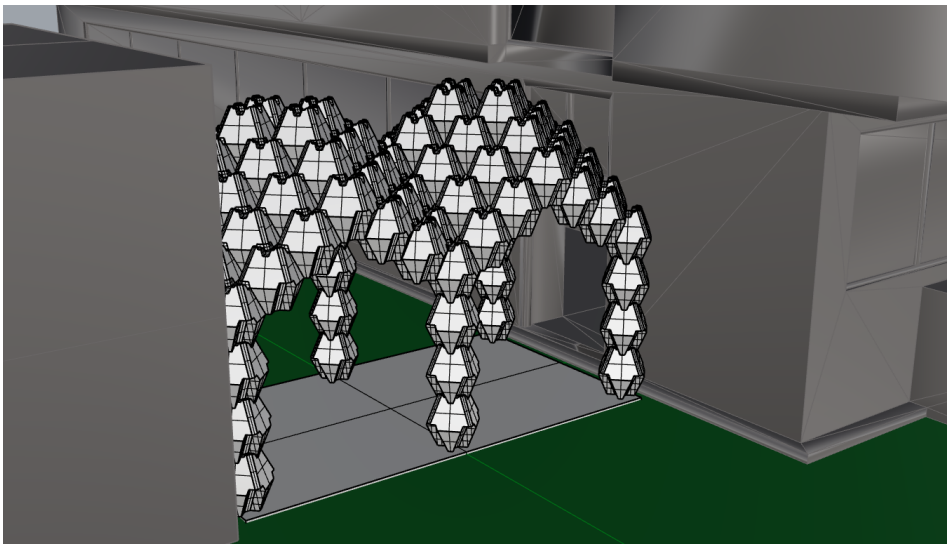
La première étape est de voir si la structure est stable face à son poids propre. On lance donc une simulation avec comme seul condition limite l'ancrage des droxels à la base de chaque pied.

On observe que la structure est instable. La figure 5.6 montre l'évolution de la structure. Les colonnes se dérobent sous la poussée latérale exercée par les voûtes.
Il faut donc envisager un changement de géométrie ou de condition aux appuis

#### Alternative 1 : Ancrage du sommet des colonnes

La première alternative étudiée est de supposer que les colonnes sont ancrées jusqu'au 4ème droxel en hauteur (totalité de la colonne) à la place de l'ancrage précédent qui concernait seulement le premier droxel. Cet ancrage pourrait se matérialiser par une colonne cylindrique classique en béton plein qui remplacerait les 3 premiers droxels de chaque colonne sur laquelle on viendrait ancrer le 4ème droxel à la hauteur originale. On pourrait aussi ancrer les droxels supérieurs des 4 colonnes extérieures aux bâtiments voisins que l'on peut voir sur la figure 5.4a.

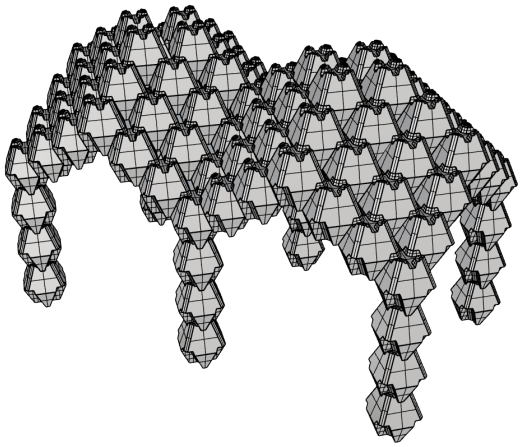
On lance donc une nouvelle simulation avec comme nouvelle condition d'appui l'ancrage des droxels liant les colonnes aux voûtes. Sans surprise, les appuis étant bloqués, on observe une stabilité parfaite de la structure dans ces nouvelles conditions. Cette alternative est la plus évidente mais pas la plus intéressante à étudier on va donc se pencher sur un autre cas.



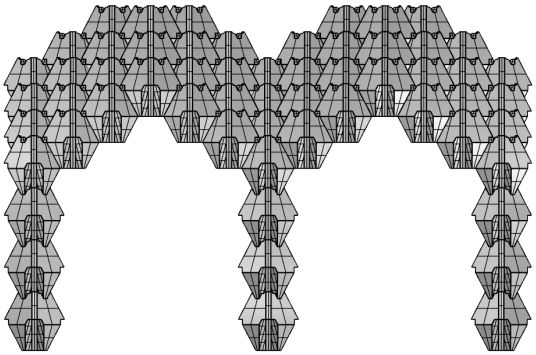
(a) Modèle

(b) Analogie avec la réalité (Croix du Sud, Louvain-la-Neuve)

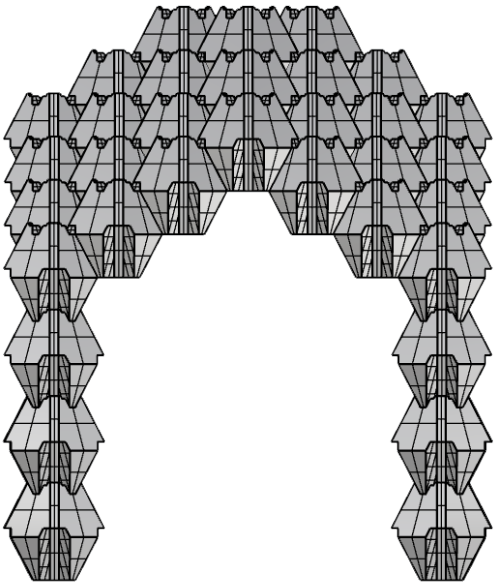
FIGURE 5.4 – Structure étudiée



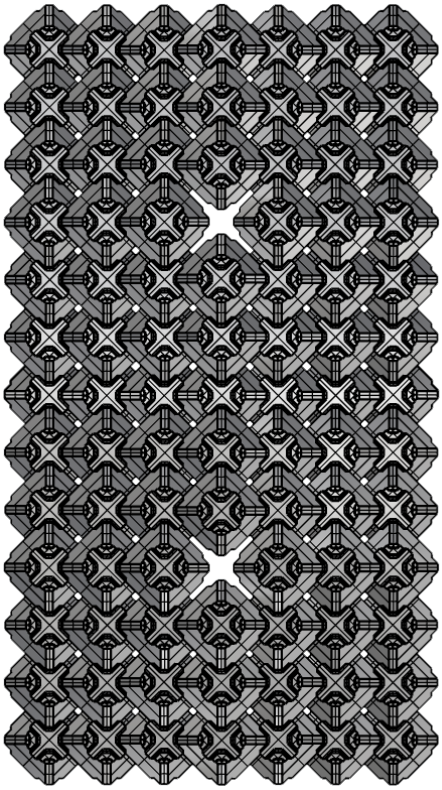
(a) Vue en perspective



(b) Vue de coté

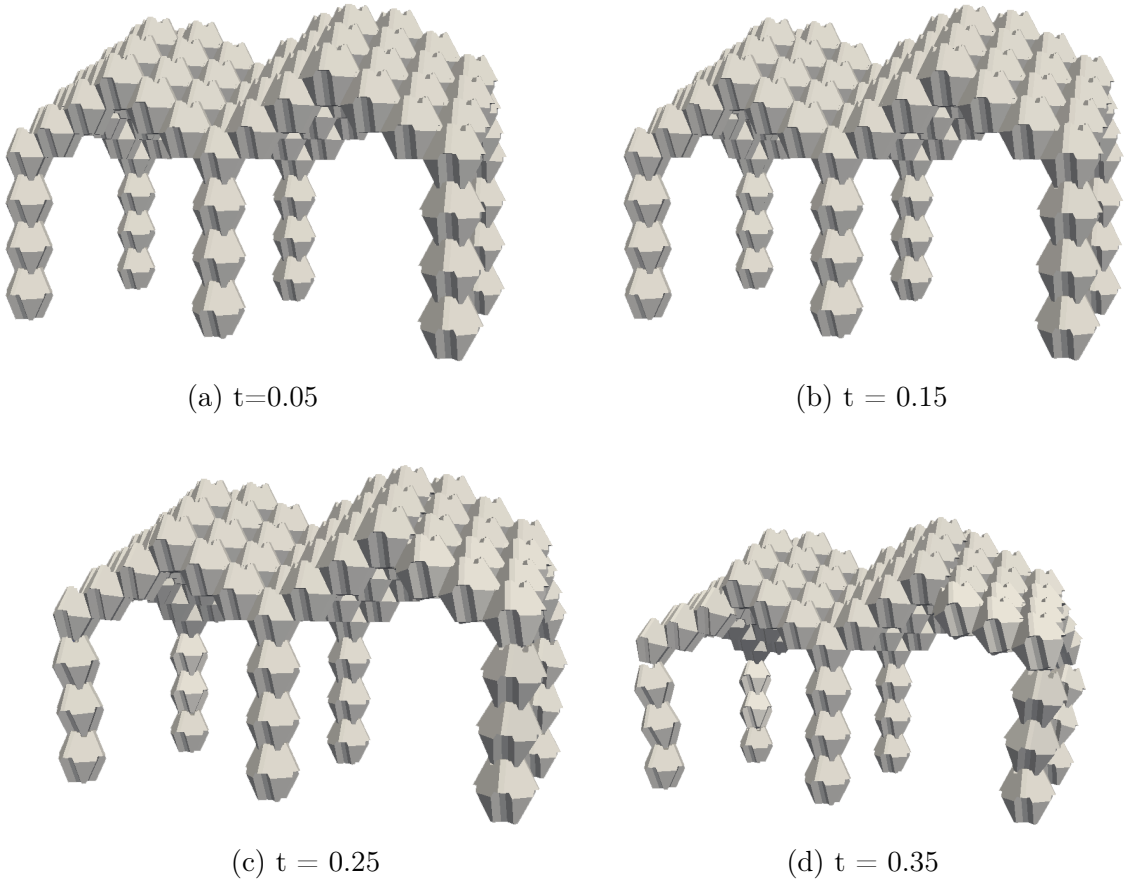


(c) Vue de front



(d) Vue d'en haut

FIGURE 5.5 – Géométrie de la structure



(a) t=0.05 (b) t = 0.15

(c) t = 0.25 (d) t = 0.35

FIGURE 5.6 – Stabilité face au poids propre

**Alternative 2 : Renforcement des colonnes**

En observant le mode de rupture de la structure de base fourni par la simulation numérique (Figure 5.6), on se rend compte que la rupture se matérialise par un déplacement latéral du dessus des colonnes dans le sens longitudinal de la structure. On peut donc envisager de renforcer ces colonnes dans cette direction en intercalant des droxels intermédiaires. On a donc des emboitements encoches sur têtes qui, comme observé dans nos simulations dans le chapitre 4, sont beaucoup plus stables et renforceront donc la colonne face au renversement dont elle était originellement l'objet. La structure renforcée est présentée sur la figure 5.7.

Comme espéré, la simulation montre une stabilité parfaite face au poids propre pour cette nouvelle structure. On va donc garder cette géométrie et effectuer plus de tests.

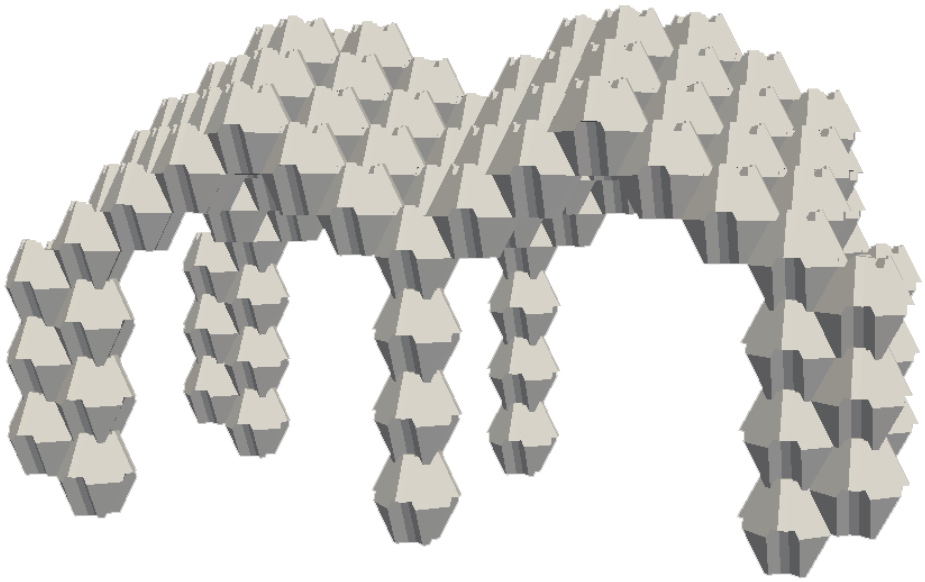


FIGURE 5.7 – Structure avec colonnes renforcées

### Stabilité face au vent

On va maintenant étudier le comportement de la structure face au vent. On va pour cela appliquer une charge sur le centre de masse des droxels concernés, dirigée selon l'axe faible de la structure. La projection d'un droxel dans un plan perpendiculaire à cette direction donne une aire de 0,1884 $m^2$ et on prend une charge due au vent de 2 $kN/m^2$. On a donc une charge de 376,8 $kN$ pour un droxel complètement exposé. Pour nous mettre du côté de la sécurité, on va négliger la réduction d'exposition des droxels dans la voute (due au droxel inférieur faisant partiellement barrière au vent). On applique donc cette charge à la moitié de la structure située face au vent. Etant donné les grandes ouvertures dans la structure, on va également supposer que les colonnes de la moitié non-exposée de la structure sont aussi l'objet de cette force. La structure ainsi que les droxels chargés sont représentés à la figure 5.8.

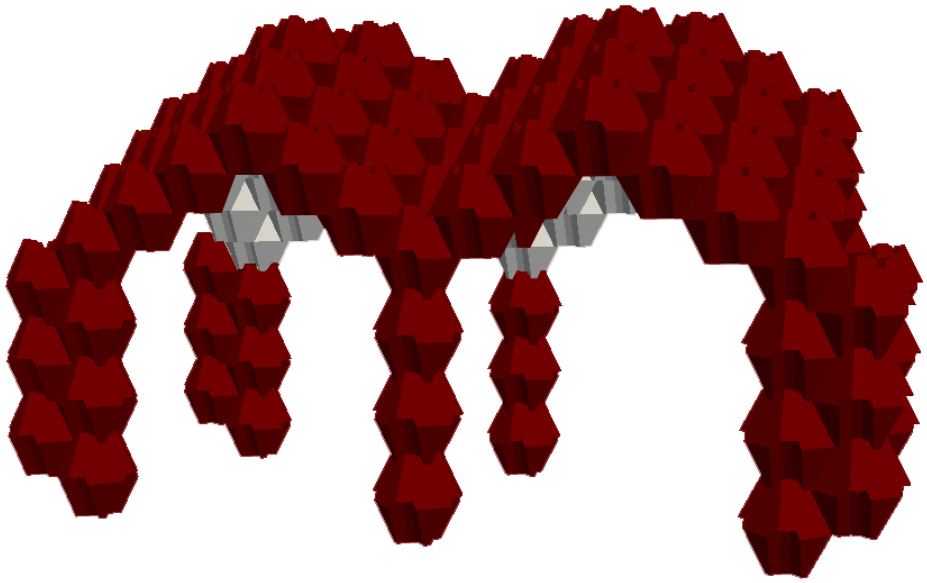
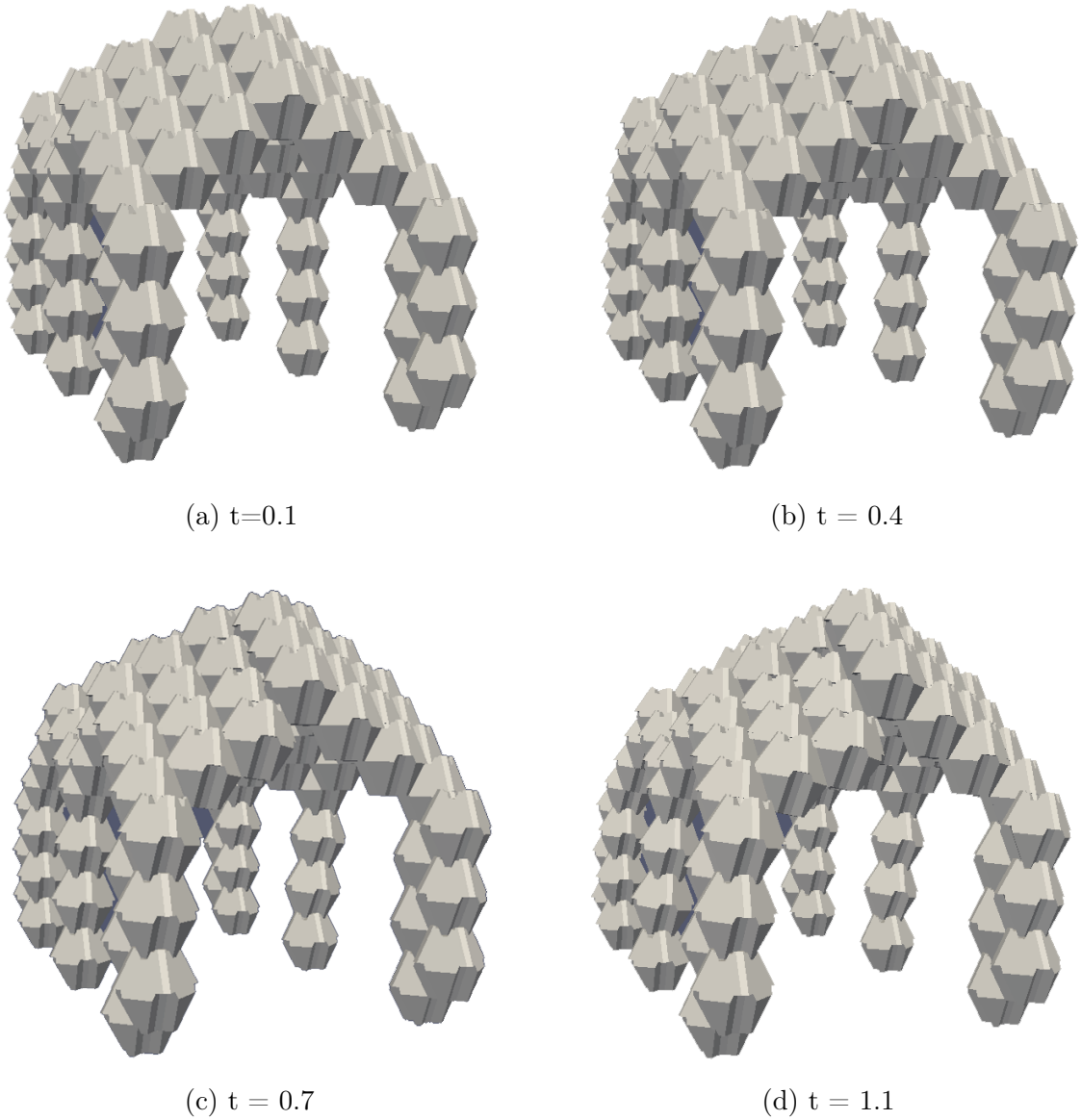


FIGURE 5.8 – Exposition de la structure face à un vent perpendiculaire à l'axe longitudinal

Les résultats de la simulation montrent une instabilité face au vent, la structure toute entière bascule de coté comme indiqué à la figure 5.9. Il va donc également falloir procéder à un renforcement dans cette direction. De la même manière que dans le cas précédent, on va intercaler des droxels entre les droxels de la colonne, cette fois-ci dans la direction perpendiculaire à l'axe longitudinal de la structure. La simulation numérique montre que la structure subit un très léger déplacement pendant 0,5 secondes après le chargement puis se stabilise. La figure 5.10 montre en gris opaque la structure avant application de la charge et en bleu transparent la structure après stabilisation

suite à l'application de la charge. Il faut alors juger si ce déplacement est critique ou non. Dans notre cas, sachant qu'il se stabilise après 0,5 secondes et que les charges de vent prises en compte ont été surestimées en prenant une surface d'exposition complète même pour les droxels protégés, on va juger ce déplacement comme négligeable.

Le script de génération de la géométrie et des forces utilisé pour cette simulation est joint en annexe.



(a) t=0.1 (b) t = 0.4

(c) t = 0.7 (d) t = 1.1

FIGURE 5.9 – Stabilité face au vent

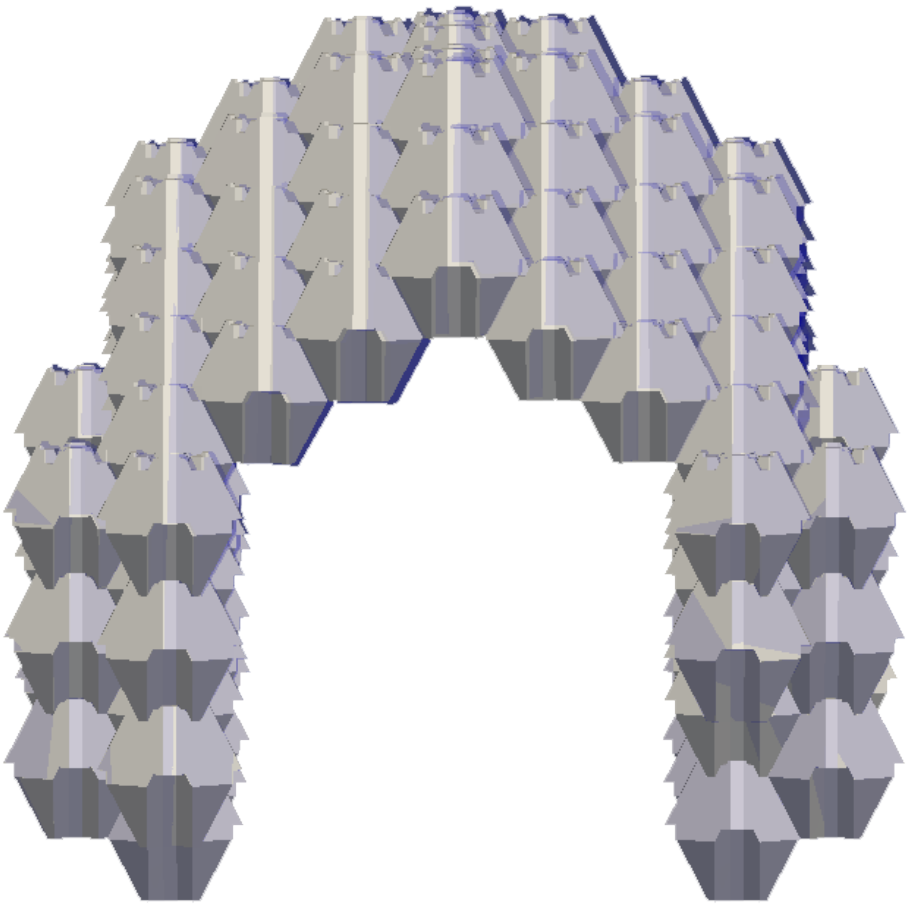


FIGURE 5.10 – Réaction de la structure doublement renforcée face à un vent perpendiculaire

# Conclusion

## Travail réalisé

Au fil de ce travail, on a pu établir différents outils permettant de développer des structures composées de droxels et on a pu prendre du recul face à certains aspects de ces structures. Les recherches ont pris une tournure expérimentale plus importante que prévue car il a fallu confirmer chaque petite avancée du coté numérique par plusieurs essais en laboratoire et ceux-ci ne se passaient pas toujours comme prévu. Malgré cela, on peut citer plusieurs outils et avancées.

On a défini a une géométrie de bloc qui permet de réaliser une grande variété de configuration de structures, cela grâce à sa capacité à s'emboiter en porte-à-faux ou verticalement. Ces structures pourraient très bien s'immiscer dans un contexte architectural fonctionnel aussi bien qu'artistique, grâce à leur caractère innovant.

On a ensuite une méthode de production expérimentale de ces blocs qui fonctionne. Grâce à une imprimante 3D et au modèle de moule développé, on peut produire ces blocs assez facilement en laboratoire.

Toujours concernant la fabrication des blocs, on a développé 2 compositions de béton qui permettent d'avoir une bonne finition et qui s'inscrivent dans un contexte de construction via des drones. Pour ces 2 compositions, on a analysé le coefficient de frottement et la résistance à la compression afin de les caractériser au mieux et de pouvoir corréler ces résultats avec les expériences que l'on a par la suite réalisées sur des microstructures. Dans le but d'aller plus loin, on a aussi renforcé un droxel avec une armature en acier.

On a alors développé une méthode de calcul de la stabilité au glissement et renversement de ces structures en utilisant le calcul en rigide de LMGC90. Les simulations ainsi réalisées permettent de quantifier la sécurité d'un ouvrage face à une instabilité de ce type. Ces simulations sont facilement réalisables sur n'importe quelle structure, il suffit de récupérer les codes donnés en annexe dans ce travail et d'introduire le maillage correspondant à la structure que l'on souhaite étudier ainsi que les conditions limites et les forces en action.

Dans cette continuité, on a développé un autre aspect de simulation numérique, le calcul en déformables. Il nous permet de prédire le lieu de rupture au sein des droxels en cas de contraintes trop importantes. Ce mode de calcul permet également d'obtenir l'orientation des contraintes principales en tout endroit de la structure.

## Pistes pour un travail postérieur

Il existe de nombreuses pistes d'amélioration qui méritent d'être explorées lors de futures recherches sur le sujet. Voici une liste non-exhaustive :

— Réaliser un nouveau moule (en ABS uniquement) avec la même géométrie mais de taille différente, par exemple 45 cm de haut et fabriquer des blocs de cette taille. On peut également investiguer de nouvelles compositions de béton qui limiteraient les irrégularités dans les encoches. On peut par exemple réduire la taille des graviers de la composition 2 développée dans ce mémoire de sorte à avoir des petites sections plus homogènes.
— Effectuer des tests plus variés sur des microstructures plus complexes (4-5 blocs) pour confirmer le modèle concernant la stabilité au glissement et au renversement établi dans ce travail.
— Approfondir le calcul en déformables sur les structures de droxels avec LMGC90, réaliser plus de tests (notamment avec des forces verticales) et essayer de dégager des informations plus précises. En se plongeant plus profondément dans la documentation de la *postpro* de LMGC90 et en suivant les formations bi-annuelles de Montpellier, il devrait être possible d'obtenir des informations plus exploitables sur la rupture au sein de droxels.
— Fabriquer plus de droxels en béton armé et les tester. En plus de renforcer les encoches, il serait intéressant de renforcer les têtes du droxels (Figure 5.11) avec des pièces métalliques. Cela déplacerait la rupture d'une petite section locale à une grande section englobant tout le droxel. On gagnerait donc énormément en résistance et le droxel n'aurait pas de faiblesse particulière. On pourrait envisager d'isoler avec LMGC90 les droxels nécessitant un renforcement (encoche, tête ou les 2).
— On pourrait également réfléchir à d'autres matériaux que le béton. Utiliser un matériau avec un comportement moins fragile permettrait de diversifier les structures. Il faut cependant faire attention à garder une certaine densité et une résistance en compression acceptable. On pourrait aussi varier les matériaux dans une même structure.

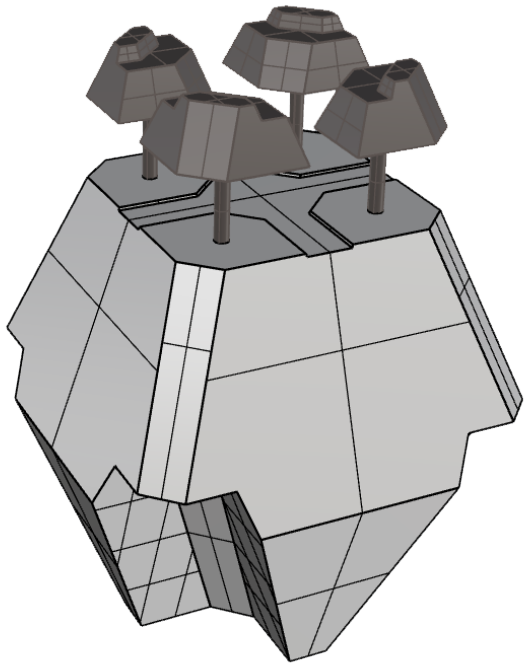


FIGURE 5.11 – Exemple de renforcement des têtes d'un droxel

## Mot de la fin

Comme développé dans l'introduction, il est raisonnable de penser que la construction par drones verra le jour dans un futur pas tellement lointain. Les droxels représentent un medium très intéressant et polyvalent pour ce type de construction car ils permettent de réaliser des géométries en 3D très variées.

On n'est malheureusement pas encore dans la capacité de dimensionner et calculer de façon complète une structure qui en serait composée mais on dispose de quelques outils et pas mal de pistes pour poursuivre cette recherche.

De plus, le logiciel LMGC90 utilisé dans ce travail est constamment en cours de développement et sera probablement de plus en plus accessible. Cela devrait ouvrir la porte à de potentiels nouveaux utilisateurs et permettre un développement plus rapide et plus efficace des travaux come celui-ci.

# Annexe A

# Guide d'installation de LMGC90

## A.1 Avant-propos

Premièrement, les instructions d'installation sont disponibles sur la page GitLab de LMGC90. Cependant, elles sont brèves et peuvent se révéler insuffisantes pour quelqu'un n'ayant pas de connaissances poussées en informatique. Cette annexe vise donc à guider un peu plus précisément le lecteur lors de l'installation du logiciel. Il est très probable que les instructions d'installation évoluent avec le programme (notamment la version de python utilisée changera prochainement), il est donc conseillé de lire ce guide conjointement aux instructions de la page GitLab car certaines lignes de code à insérer dans le terminal pourraient différer.

Deuxièmement, ce guide a été réalisé avec Mac OS, les utilisateurs Windows devront utiliser une partition Linux ou le sous-système Linux comme proposé dans les dernières versions de Windows 10. Les commandes du terminal Linux sont très similaires à celles du terminal Mac OS. Cependant, les dépendances sous Windows ne s'installent pas de la même manière et le lecteur est donc invité à suivre les instructions de la page GitLab.

Enfin, l'équipe des développeurs du logiciel à Montpellier réalise annuellement des formations pendant lesquelles ils aident les intéressés à installer et utiliser leur programme. Le lecteur peut se renseigner directement par mail auprès de l'équipe.

## A.2 Pré-requis

### A.2.1 MacPorts

La première étape est de télécharger MacPorts. C'est un gestionnaire de paquets. Il permet de télécharger le code source d'un programme ainsi que les dépendances de celui-ci et d'ensuite compiler ce code source pour en faire un code binaire utilisable par l'ordinateur.

Un logiciel installé de façon ordinaire via un fichier DMG ou autre fonctionne comme une boite noire, tout ce dont le programme a besoin se trouve dans ce fichier. On a donc parfois plusieurs programmes utilisant les mêmes dépendances et celles-ci sont installées à l'identique dans plusieurs boites. À l'inverse, les logiciels installés via MacPorts peuvent utiliser d'autre logiciels également installés via MacPorts et chaque fonctionnalité ne doit être installé qu'une seule fois. MacPorts se chargent d'effectuer les liens et les mises à jour.

Pour installer MacPorts, il suffit de se rendre sur le site officiel et de télécharger le fichier *.pkg* correspondant à la version du système. Avant d'ouvrir le fichier, il faut s'assurer d'avoir l'outil de développement Xcode téléchargeable sur le site officiel d'Apple.

### A.2.2 Eléments à installer à partir du terminal

Une fois MacPorts installé, on peut commencer à installer les éléments qui permettront le fonctionnement de LMGC90 sur la machine. Il faut donc ouvrir l'application terminal et y entrer certaines commandes.

Premièrement, pour pouvoir compiler LMGC90, il va falloir plusieurs dépendances que l'on peut installer grâce à la commande suivante dans le terminal :

```
sudo port install gcc7 python27 py27-numpy swig-python cmake +gui
```

Ensuite, il va falloir définir les versions de python et de compilateur installées grâce à la ligne précédentes comme versions par défaut sur notre système. Cela se fait comme suit :

```
sudo port select gcc mp-gcc7
sudo port select python python27
```

Enfin, pour visualiser les résultats, construire la documentation et lire les tutoriels, il va falloir les éléments installés grâce aux lignes suivantes :

```
sudo port install vtk +python27
sudo port install doxygen graphviz py27-sphinx py27-numpydoc ...
texlive-latex-extra dvipng ImageMagick
sudo port install py27-jupyter
```

### A.2.3 Paraview

Pour visualiser les résultats fournis par LMGC90, on utilisera Paraview. Celui-ci est facilement téléchargeable sur son site officiel.

## A.3 Installation et compilation de LMGC90

Il faut maintenant se procurer la dernière version du logiciel sur la page GitLab. Elle se présente sous forme d'archive. Celle-ci peut être décompressée dans un répertoire au choix, où l'on souhaite installer le programme.

Il faut ensuite se placer dans le dossier du programme, pour cela il va falloir entrer la commande `cd` suivie du chemin d'accès au dossier. Celui-ci peut être obtenu en faisant glisser le dossier directement du Finder dans le terminal. On peut alors se placer dans le répertoire *build* puis compiler le programme et créer la documentation grâce aux commandes suivantes :

```
cd build
cmake ..
make
make docs
```

Il ne reste alors plus qu'à ajouter à Python le chemin d'accès vers le dossier build. Pour cela, il faut le éditer le fichier *.profile* (par exemple grâce à l'éditeur *vim*). On effectue pour ça la commande : `vim .profile`. Il faut ensuite ajouter à ce fichier les lignes suivantes :

```
if [ -z ${PYTHONPATH} ]; then
  export PYTHONPATH=mysourcepath/build
else
  export PYTHONPATH=${PYTHONPATH}:mysourcepath/build
fi
export PATH=${PATH}:mysourcepath/src/addons
```

Pour sauvegarder et quitter, il faut écrire la ligne `:wq` et appuyer sur la touche Enter. LMGC90 devrait maintenant être opérationnel.

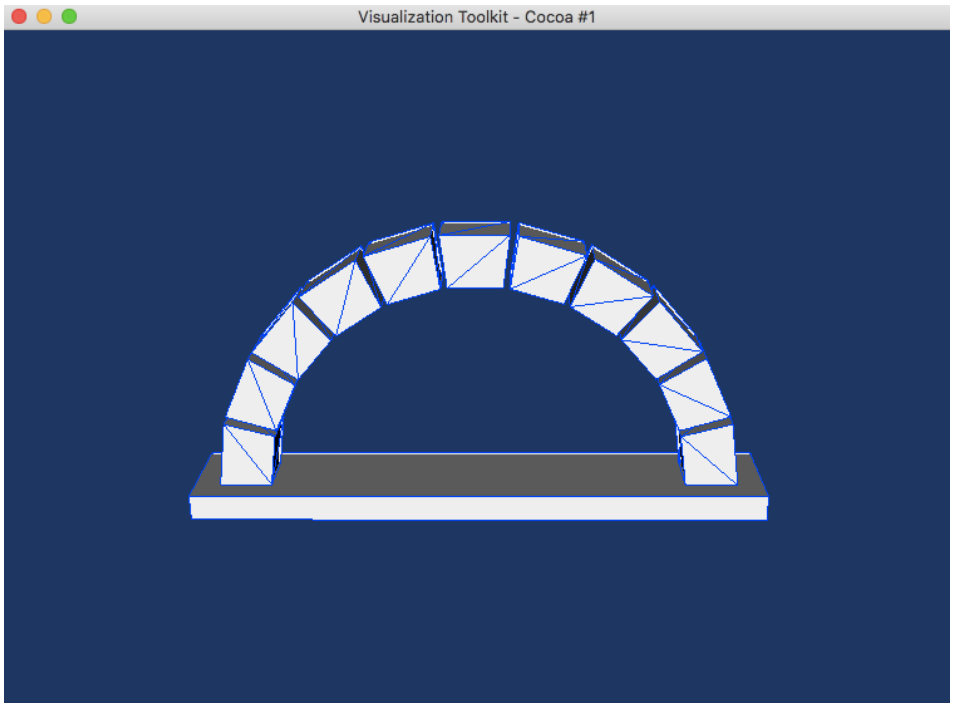



FIGURE A.1 – Représentation de la géométrie

## A.4 Exemple d'utilisation

Pour vérifier que le programme fonctionne bien sans erreur et en comprendre le schéma de fonctionnement, il peut être utile de lancer un exemple. Le dossier *examples* contient une série d'exemples. On peut par exemple prendre dans le dossier *RIGID_3D* l'exemple de l'arche de plein ceintre. Le dossier contient un fichier *gen_sample.py* qui permet de créer la situation et un fichier *command.py* qui permet d'effectuer le calcul.

On va donc ouvrir le terminal, se placer dans le dossier de l'arche en effectuant la commande `cd` suivie du chemin d'accès au dossier. On peut ensuite exécuter le scripte de génération en écrivant la commande

```
python gen_sample.py
```

Ceci devrait ouvrir une fenêtre dans laquelle est représentée l'arche (figure A.1) et créé un dossier nommé *DATBOX*. On peut dès lors lancer le calcul en effectuant la commande

```
python command.py
```

Cela crée alors 3 nouveaux dossiers nommés *DISPLAY*, *OUTBOX* et *POSTPRO*.

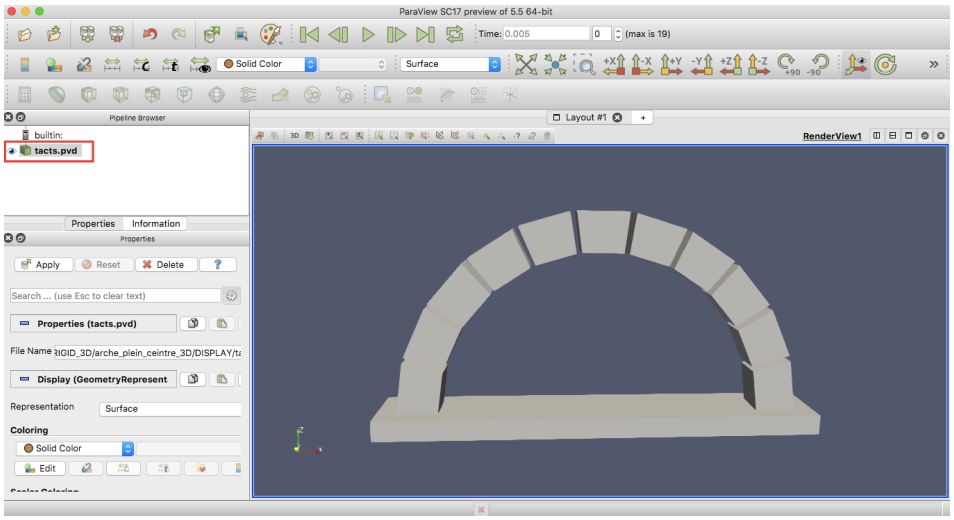



FIGURE A.2 – Visualisation des résultats dans Paraview

On peut finalement visualiser les résultats dans Paraview. On effectue la commande `paraview` depuis le terminal. Une fois Paraview ouvert, on peut aller dans *open*, sélectionner le dossier *DISPLAY*, ouvrir le fichier *tacts.pvd* et activer l'affichage (figure A.2). On devrait alors être capable de visualiser l'arche comme escompté.

# Annexe B

# Scripts LMGC90

## B.1 Script de génération, Section 4.1, cas 1, calcul en rigides

```
1  import os,sys,copy
2  import numpy
3  import math
4  import random
5
6  from pylmgc90 import pre
7
8  if not os.path.isdir('./DATBOX'):
9    os.mkdir('./DATBOX')
10
11 # Attention : En 3D l'axe Z est dirige vers le haut
12 dim = 3
13
14 bodies = pre.avatars()
15 mat    = pre.materials()
16 svs    = pre.see_tables()
17 tacts  = pre.tact_behavs()
18
19 # On cree un materiau
20 tdur = pre.material(name='TDURx',materialType='RIGID',density=1494)
21 mat.addMaterial(tdur)
22
23 # On cree un modele rigide
24 mod = pre.model(name='rigid', physics='MECAx', element='Rxx3D',
       dimension=dim)
25
26 # On cree un plan pour supporter la structure
27 down = pre.rigidPlan(axe1=10. , axe2=10., axe3=0.01, center
       =[0,0,-0.02], material=tdur, model=mod, color='VERTx')
28 bodies += down
29
```

```
30 # Fixation des DDL du plan
31 down.imposeDrivenDof(component=[1, 2, 3, 4, 5, 6], dofty='vlocy')
32
33 # On lit le maillage total dans un fichier, au format gmsh
34 droxel_mesh = pre.lecture(name='Droxel25ajuste.msh', dim=dim)
35
36 # On cree un corps de reference
37 body_ref = pre.volumicMeshToRigid3D(volumic_mesh=droxel_mesh, model
       =mod, material=tdur, color='BLEUx')
38
39 # On centre le corps de reference et on lui donne les bonnes
       dimensions
40 vertices = numpy.matmul(body_ref.bulks[0].axis,body_ref.contactors
       [0].vertices.T).T
41 dx = numpy.max(vertices[:,0]) - numpy.min(vertices[:,0])
42 dz = numpy.max(vertices[:,2]) - numpy.min(vertices[:,2])
43 realdim = [0.25,0.24099] #[h,L]
44 htot = (0.28272/0.25)*realdim[0]
45 ratio = realdim[1]/dx
46 body_ref.contactors[0].vertices[:,:] *= ratio
47 vertices = numpy.matmul(body_ref.bulks[0].axis,body_ref.contactors
       [0].vertices.T).T
48
49 # On definit les espacement dans le maillage de droxels
50 vSpace=(realdim[0]/2)+0.005
51 hSpace=(0.26599/0.25)*realdim[0]
52 meshZSpace=vSpace
53 meshXYSpace=hSpace/2
54
55 # On cree une fonction permettant de placer un droxel a un certain
       nombre d'espaces d'un autre
56 def createAndMove(avatar,mdz=0,mdx=0,mdy=0):
57     new_avatar = copy.deepcopy(avatar)
58     if mdz%2!=0 :
59         if mdx%2!=1 or mdy%2!=1:
60             print("[WARNING] : Bloc at coordinates {},{},{} can't
                   be placed".format(mdx,mdy,mdz))
61             return
62         new_avatar.translate(dz=mdz*vSpace, dx=mdx*hSpace/2, dy=mdy
               *hSpace/2)
63     else :
64         if mdx%2!=0 or mdy%2!=0:
65             print("[WARNING] : Bloc at coordinates {},{},{} can't
                   be placed".format(mdx,mdy,mdz))
66             return
67         new_avatar.translate(dz=mdz*vSpace, dx=mdx*hSpace/2, dy=mdy
               *hSpace/2)
68     bodies.addAvatar(new_avatar)
69
70 # On cree une fonction permettant de creer automatiquement un
       parrallelepipede a base rectangle
71 def createRectangle(length,depth,height):
```

```
72      mesh_bodies = []
73      for i in range(length):
74          for j in range(depth):
75              for k in range(height):
76                  mesh_bodies.append([i,j,k])
77      for i in mesh_bodies:
78                      createAndMove(body_ref, mdz=i[2], mdx=i[0], mdy
                           =i[1])
79
80  # On cree une fonction permettant de creer automatiquement un mur
       en diagonale
81  def createDiag(L, h):
82      mesh_bodies = []
83      for i in range(L):
84          for j in range(h):
85              mesh_bodies.append([i,i,j])
86      for i in mesh_bodies:
87          createAndMove(body_ref, mdz=i[2], mdx=i[0], mdy=i[1])
88
89  # On definit la geometrie (ici : 3 droxels empiles)
90  wallDepth=1
91  wallLength=1
92  wallHeight=5
93  createRectangle(wallLength,wallDepth,wallHeight)
94
95  # On definit l'evolution dans le temps de la force a appliquer au
       droxel superieur
96  def vy(t):
97      if t <= 0.1:
98          return 0
99      else:
100         return  40
101 pre.writeEvolution(f=vy, instants=numpy.linspace(0,2,200) ,path='./
       DATBOX/', name='vy.dat')
102
103 # On liste les conditions que l'on veut imposer (ici : droxel du
       bas ancre et force sur droxel sup)
104 inputDrvDof = {'k1':({'component':1, 'description':'evolution', '
       evolutionFile':'vy.dat','ct':10, 'amp':0., 'omega':0., 'phi':0.,
        'rampi':1., 'ramp':0., 'dofty':'force'},[[0,0,4]]),
105     'k2':({'component':[1, 2, 3, 4, 5, 6], 'dofty':'vlocy'
           },[[0,0,0],[2,2,2]]),
106             }
107
108 # On cree une fonction lisant le dictionnaire des conditions a
       imposer et les appliquant
109 tol=0.0001
110 zCenter=body_ref.nodes[1].coor[2]
111 for value in inputDrvDof.values():
112     for select in value[1]:
113         if len(select)==2:
```

```
114             if (select[0][2]+select[0][0])%2!=0 or (select[0][2]+
                    select[0][1])%2!=0 or (select[1][2]+select[1][0])
                    %2!=0 or (select[1][2]+select[1][1])%2!=0 :
115                 print('[WARNING] : Mesh point of one corner can\'t
                        contain any block')
116             for b in bodies:
117                 mZ=meshZSpace
118                 mXY=meshXYSpace
119                 if  b.nodes[1].coor[0] < mXY*max(select[0][0],
                        select[1][0])+tol and b.nodes[1].coor[0] > mXY*
                        min(select[0][0],select[1][0])-tol and b.nodes
                        [1].coor[1] < mXY*max(select[0][1],select[1][1])
                        +tol and b.nodes[1].coor[1] > mXY*min(select
                        [0][1],select[1][1])-tol and b.nodes[1].coor[2]
                        < mZ*max(select[0][2],select[1][2])+tol+zCenter
                        and b.nodes[1].coor[2] > mZ*min(select[0][2],
                        select[1][2])-tol+zCenter:
120                     b.imposeDrivenDof(**value[0])
121         if len(select)==3:
122             if (select[2]+select[0])%2!=0 or (select[2]+select[1])
                    %2!=0 :
123                 print('[WARNING] : Mesh point can\'t contain any
                        block')
124             for b in bodies:
125                 mZ=meshZSpace
126                 mXY=meshXYSpace
127                 if  b.nodes[1].coor[0] < mXY*select[0]+tol and b.
                        nodes[1].coor[0] > mXY*select[0]-tol and b.nodes
                        [1].coor[1] < mXY*select[1]+tol and b.nodes[1].
                        coor[1] > mXY*select[1]-tol and b.nodes[1].coor
                        [2] < mZ*select[2]+tol+zCenter and b.nodes[1].
                        coor[2] > mZ*select[2]-tol+zCenter:
128                     b.imposeDrivenDof(**value[0])
129
130 # On definit une loi de contact
131 b=pre.tact_behav(name='iqsc0',law='IQS_CLB',fric=0.6)
132 tacts+=b
133
134 # On definit les interactions
135 sv = pre.see_table(CorpsCandidat='RBDY3',candidat='POLYR',
        colorCandidat='BLEUx',behav=b,
136                    CorpsAntagoniste='RBDY3',antagoniste='PLANx',
                           colorAntagoniste='VERTx',alert=.1)
137 svs+=sv
138 sv = pre.see_table(CorpsCandidat='RBDY3',candidat='POLYR',
        colorCandidat='BLEUx',behav=b,
139                    CorpsAntagoniste='RBDY3',antagoniste='POLYR',
                           colorAntagoniste='BLEUx',alert=.1)
140 svs+=sv
141
142 # On ecrit les dossiers et les infos
143 pre.writeBodies(bodies,chemin='DATBOX/')
```

```
144 pre.writeBulkBehav(mat,chemin='DATBOX/',dim=3)
145 pre.writeTactBehav(tacts,svs,chemin='DATBOX/')
146 pre.writeDrvDof(bodies,chemin='DATBOX/')
147 pre.writeDofIni(bodies,chemin='DATBOX/')
148 pre.writeVlocRlocIni(chemin='DATBOX/')
149
150 post=pre.postpro_commands()
151 pre.writePostpro(commands=post, parts=bodies, path='DATBOX/')
152
153 # On demande d'afficher en couleur les blocs soumis a des
      conditions
154 try:
155     pre.visuAvatars(bodies,drvdof_color=[1,0,0])
156 except:
157   pass
```

## B.2 Script de calcul, Section 4.1, cas 1, calcul en rigides

```
from __future__ import print_function
import os,sys

from pylmgc90.chipy import *
from numpy import *

checkDirectories()

# On definit les parametres de calcul
dt = 1e-2
nb_steps = 200

theta = 0.5

freq_display = 5

POLYR_SkipAutomaticReorientation()
POLYR_TopologyAngle(10.)
POLYR_FlatnessAngle(5.)

PRPRx_UseNcDetection(2e-2)
PRPRx_WithNodalContact()

PRPRx_LowSizeArrayPolyr(4000)

freq_detect = 20
norm='Stored_Delassus_Loops          '
tol = 1e-5
relax = 1.0
quad = 'Maxm '
gs_it1 = 200
gs_it2 = 10

nlgs_3D_DiagonalResolution()

SetDimension(3,0)

print('INIT TIME STEPPING')
TimeEvolution_SetTimeStep(dt)
Integrator_InitTheta(theta)

# On lit le modele
print('READ BODIES')
RBDY3_ReadBodies()
PLANx_LoadTactors()
POLYR_LoadTactors()

```

```
48 print('READ BEHAVIOURS')
49 bulk_behav_ReadBehaviours()
50 tact_behav_ReadBehaviours()
51
52 # On lit les charges
53 RBDY3_LoadBehaviours()
54
55 print('READ INI DOF')
56 TimeEvolution_ReadIniDof()
57 RBDY3_ReadIniDof()
58
59 print('READ INI Vloc Rloc')
60 TimeEvolution_ReadIniVlocRloc()
61 PRPLx_ReadIniVlocRloc()
62 PRPRx_ReadIniVlocRloc()
63
64 print('READ DRIVEN DOF')
65 RBDY3_ReadDrivenDof()
66
67 overall_WriteBodies()
68 RBDY3_WriteBodies()
69
70 bulk_behav_WriteBehaviours()
71 tact_behav_WriteBehaviours()
72
73 overall_WriteDrivenDof()
74 RBDY3_WriteDrivenDof()
75
76 # On ouvre les fichier ou l'on va ecrire
77 OpenDisplayFiles()
78 OpenPostproFiles()
79
80 # On calcule la masse
81 RBDY3_ComputeMass()
82
83 for k in range(nb_steps):
84
85     print('INCREMENT STEP')
86     TimeEvolution_IncrementStep()
87     RBDY3_IncrementStep()
88
89     print('DISPLAY TIMES')
90     TimeEvolution_DisplayStep()
91
92     print('COMPUTE Fext')
93     RBDY3_ComputeFext()
94
95     print('COMPUTE Fint')
96     RBDY3_ComputeBulk()
97
98     print('COMPUTE Free Vlocy')
99     RBDY3_ComputeFreeVelocity()
```

```
100
101     print('SELECT PROX TACTORS')
102     overall_SelectProxTactors(freq_detect)
103
104     PRPLx_SelectProxTactors()
105     PRPRx_SelectProxTactors()
106
107     PRPLx_RecupRloc()
108     PRPRx_RecupRloc()
109
110     nlgs_3D_ExSolver(norm,quad, tol, relax, gs_it1, gs_it2)
111
112     PRPLx_StockRloc()
113     PRPRx_StockRloc()
114
115     print('COMPUTE DOF')
116     RBDY3_ComputeDof()
117
118     print('UPDATE DOF')
119     TimeEvolution_UpdateStep()
120     RBDY3_UpdateDof()
121
122     utilities_logMes('WRITE OUT Vloc Rloc')
123     TimeEvolution_WriteOutVlocRloc(freq_display)
124     PRPLx_WriteOutVlocRloc()
125     PRPRx_WriteOutVlocRloc()
126
127     # On ecrit les resultats
128     WriteDisplayFiles(freq=freq_display,ref_radius=2e-2)
129     WritePostproFiles()
130
131     overall_CleanWriteOutFlags()
132
133 CloseDisplayFiles()
134 ClosePostproFiles()
```

## B.3 Script de génération, Section 4.2, cas 3, calcul en déformable

```
import os,sys,copy
import numpy as np
import math
import random
from pylmgc90 import pre
if not os.path.isdir('./DATBOX'):
  os.mkdir('./DATBOX')

dim = 3

bodies = pre.avatars()
mats    = pre.materials()
svs    = pre.see_tables()
tacts  = pre.tact_behavs()
mods   = pre.models()

# mod & mat
mot = pre.model(name='M3D_T',physics='MECAx',element='TE4xx',dimension=dim, external_model='yes__',
                kinematic='small',material='elas_',anisotropy='iso__',mass_storage='coher')
mods+=mot

map = pre.material(name='mou__',materialType='ELAS',elas='standard',
                   young=60e9,nu=0.3,anisotropy='isotropic',
                   density=2417)

mats.addMaterial(map)

# on lit le maillage total dans un fichier, au format gmsh
droxel_mesh = pre.lecture(name='Droxel25.msh', dim=dim) # pas oublier de rentrer dimensions du maillage plus bas

body_ref = pre.buildMeshedAvatar(mesh=droxel_mesh, model=mot, material=map)


#CS/ASp
body_ref.addContactors(group='down', shape='CSpxx', color='BLUEx', quadrature=0)
body_ref.addContactors(group='mid_p', shape='CSpxx', color='REDxx', quadrature=0)
body_ref.addContactors(group='down_v', shape='CSpxx', color='YELLx', quadrature=0)
body_ref.addContactors(group='mid_e', shape='CSpxx', color='GREEx', quadrature=0)
```

```
39 body_ref.addContactors(group='down_ev', shape='CSpxx', color='ORANx
       ', quadrature=0)
40 body_ref.addContactors(group='top_top', shape='CSpxx', color='PURPx
       ', quadrature=0)
41
42
43
44 body_ref.addContactors(group='top', shape='ASpxx', color='BLUEx')
45 body_ref.addContactors(group='top_p', shape='ASpxx', color='REDxx')
46 body_ref.addContactors(group='top_v', shape='ASpxx', color='YELLx')
47 body_ref.addContactors(group='top_e', shape='ASpxx', color='GREEx')
48 body_ref.addContactors(group='top_ev', shape='ASpxx', color='ORANx'
       )
49 body_ref.addContactors(group='oblique_top', shape='ASpxx', color='
       BLACx') # ajoute par moi
50 body_ref.addContactors(group='encoches', shape='ASpxx', color='
       PURPx') # ajoute par moi
51
52 realdim = [0.25,0.24099] #[h,L]
53 htot = (0.28272/0.25)*realdim[0]
54
55 vSpace=(realdim[0]/2)+0.00001
56 hSpace=(0.26599/0.25)*realdim[0]
57 meshZSpace=vSpace
58 meshXYSpace=hSpace/2
59
60 def createAndMove(avatar,mdz=0,mdx=0,mdy=0):
61     new_avatar = copy.deepcopy(avatar)
62     if mdz%2!=0 :
63         if mdx%2!=1 or mdy%2!=1:
64             print("[WARNING] : Bloc at coordinates {},{},{} can't
                    be placed".format(mdx,mdy,mdz))
65             return
66         new_avatar.translate(dz=mdz*vSpace, dx=mdx*hSpace/2, dy=mdy
                *hSpace/2)
67     else :
68         if mdx%2!=0 or mdy%2!=0:
69             print("[WARNING] : Bloc at coordinates {},{},{} can't
                    be placed".format(mdx,mdy,mdz))
70             return
71         new_avatar.translate(dz=mdz*vSpace, dx=mdx*hSpace/2, dy=mdy
                *hSpace/2)
72     bodies.addAvatar(new_avatar)
73
74 mesh_bodies = [[0,0,0],
75                                [1,1,1],
76                [2,2,2]]
77
78
79 for i in mesh_bodies:
80     createAndMove(body_ref, mdz=i[2], mdx=i[0], mdy=i[1])
81
```

```
82 mesh = pre.lecture(name='ground.msh', dim=dim) # pas oublier de
       rentrer dimensions du maillage plus bas
83
84 # create a model of rigid
85 mod = pre.model(name='rigid', physics='MECAx', element='Rxx3D',
       dimension=dim)
86 tdur = pre.material(name='TDURx',materialType='RIGID',density
       =2200.)
87 mats.addMaterial(tdur)
88
89 down = pre.volumicMeshToRigid3D(volumic_mesh=mesh, model=mod,
       material=tdur, color='REDxx')
90 down.translate(dz=-0.1)
91
92 inputDrvDof = {#'k1':({'component':[1,2], 'description':'evolution
       ', 'evolutionFile':'vy.dat','ct':10, 'amp':0., 'omega':0., 'phi
       ':0., 'rampi':1., 'ramp':0., 'dofty':'force'},[[0,0,4]]),
93     'k2':({'component':[1, 2, 3, 4, 5, 6], 'dofty':'vlocy'
          },[[0,0,0]]),
94                 }
95
96
97 tol=0.0001
98 zCenter=body_ref.nodes[1].coor[2]
99 for value in inputDrvDof.values():
100     for select in value[1]:
101         if len(select)==2:
102             if (select[0][2]+select[0][0])%2!=0 or (select[0][2]+
                    select[0][1])%2!=0 or (select[1][2]+select[1][0])
                    %2!=0 or (select[1][2]+select[1][1])%2!=0 :
103                 print('[WARNING] : Mesh point of one corner can\'t
                        contain any block')
104             for b in bodies:
105                 mZ=meshZSpace
106                 mXY=meshXYSpace
107                 if  b.nodes[1].coor[0] < mXY*max(select[0][0],
                        select[1][0])+tol and b.nodes[1].coor[0] > mXY*
                        min(select[0][0],select[1][0])-tol and b.nodes
                        [1].coor[1] < mXY*max(select[0][1],select[1][1])
                        +tol and b.nodes[1].coor[1] > mXY*min(select
                        [0][1],select[1][1])-tol and b.nodes[1].coor[2]
                        < mZ*max(select[0][2],select[1][2])+tol+zCenter
                        and b.nodes[1].coor[2] > mZ*min(select[0][2],
                        select[1][2])-tol+zCenter:
108                     b.imposeDrivenDof(**value[0])
109         if len(select)==3:
110             if (select[2]+select[0])%2!=0 or (select[2]+select[1])
                    %2!=0 :
111                 print('[WARNING] : Mesh point can\'t contain any
                        block')
112             for b in bodies:
113                 mZ=meshZSpace
```

```
114                 mXY=meshXYSpace
115                 if  b.nodes[1].coor[0] < mXY*select[0]+tol and b.
                        nodes[1].coor[0] > mXY*select[0]-tol and b.nodes
                        [1].coor[1] < mXY*select[1]+tol and b.nodes[1].
                        coor[1] > mXY*select[1]-tol and b.nodes[1].coor
                        [2] < mZ*select[2]+tol+zCenter and b.nodes[1].
                        coor[2] > mZ*select[2]-tol+zCenter:
116                     b.imposeDrivenDof(**value[0])
117
118
119 down.imposeDrivenDof(component=[1, 2, 3, 4, 5, 6], dofty='vlocy')
120
121 bodies.addAvatar(down)
122
123 a=pre.tact_behav(name='iqsc0',law='GAP_SGR_CLB',fric=0.6)
124 tacts+=a
125 b=pre.tact_behav(name='iqsc1',law='GAP_SGR_CLB',fric=0.6)
126 tacts+=b
127
128 sv = pre.see_table(CorpsCandidat='MAILx',candidat='CSxxx',
        colorCandidat='BLUEx',behav=a,
129                    CorpsAntagoniste='RBDY3',antagoniste='POLYR',
                           colorAntagoniste='REDxx',alert=1e-3,halo
                           =0.05)
130 svs+=sv
131
132 sv = pre.see_table(CorpsCandidat='MAILx',candidat='CSxxx',
        colorCandidat='BLUEx',behav=b,
133                    CorpsAntagoniste='MAILx',antagoniste='ASpxx',
                           colorAntagoniste='BLUEx',alert=1e-3,halo
                           =0.005)
134 svs+=sv
135
136 sv = pre.see_table(CorpsCandidat='MAILx',candidat='CSxxx',
        colorCandidat='REDxx',behav=b,
137                    CorpsAntagoniste='MAILx',antagoniste='ASpxx',
                           colorAntagoniste='REDxx',alert=1e-3,halo
                           =0.005)
138 svs+=sv
139
140 sv = pre.see_table(CorpsCandidat='MAILx',candidat='CSxxx',
        colorCandidat='YELLx',behav=b,
141                    CorpsAntagoniste='MAILx',antagoniste='ASpxx',
                           colorAntagoniste='YELLx',alert=1e-3,halo
                           =0.005)
142 svs+=sv
143
144 sv = pre.see_table(CorpsCandidat='MAILx',candidat='CSxxx',
        colorCandidat='GREEx',behav=b,
145                    CorpsAntagoniste='MAILx',antagoniste='ASpxx',
                           colorAntagoniste='GREEx',alert=1e-3,halo
                           =0.005)
```

```
146 svs+=sv
147
148 sv = pre.see_table(CorpsCandidat='MAILx',candidat='CSxxx',
    colorCandidat='ORANx',behav=b,
149                  CorpsAntagoniste='MAILx',antagoniste='ASpxx',
                         colorAntagoniste='ORANx',alert=1e-3,halo
                         =0.005)
150
151 svs+=sv
152
153 sv = pre.see_table(CorpsCandidat='MAILx',candidat='CSxxx',
    colorCandidat='PURPx',behav=b,
154                  CorpsAntagoniste='MAILx',antagoniste='ASpxx',
                         colorAntagoniste='PURPx',alert=1e-3,halo
                         =0.005)
155
156 svs+=sv
157
158 sv = pre.see_table(CorpsCandidat='MAILx',candidat='CSxxx',
    colorCandidat='YELLx',behav=b,
159                  CorpsAntagoniste='MAILx',antagoniste='ASpxx',
                         colorAntagoniste='BLACx',alert=1e-3,halo
                         =0.005)
160
161 svs+=sv
162
163 pre.writeBulkBehav(mats,chemin='DATBOX/',dim=3)
164 pre.writeModels(mods,chemin='DATBOX/')
165 pre.writeTactBehav(tacts,svs,chemin='DATBOX/')
166 pre.writeBodies(bodies,chemin='DATBOX/')
167 pre.writeDrvDof(bodies,chemin='DATBOX/')
168 pre.writeDofIni(bodies,chemin='DATBOX/')
169 pre.writeGPVIni(bodies,chemin='DATBOX/')
170 pre.writeVlocRlocIni(chemin='DATBOX/')
171
172 post=pre.postpro_commands()
173 pre.writePostpro(commands=post, parts=bodies, path='DATBOX/')
174
175
176 try:
177   pre.visuAvatars(bodies,drvdof_color=[1,0,0])
178 except:
179   pass
```

## B.4 Script de calcul, Section 4.2, cas 3, calcul en déformable

```
# importing chipy module
from pylmgc90.chipy import *

# Initializing
Initialize()

# checking/creating mandatory subfolders
checkDirectories()


#
# defining some variables
#

# space dimension
dim = 3

# modeling hypothesis ( 1 = plain strain, 2 = plain stress, 3 = axi
    -symmetry)
mhyp = 1

# time evolution parameters
dt = 1e-3
nb_steps = 10 #1000

# theta integrator parameter
theta = 0.5

# deformable  yes=1, no=0
deformable = 1

# interaction parameters
Rloc_tol = 5.e-2

# nlgs parameters
tol = 1e-5
relax = 1.0
norm = 'Quad '
gs_it1 = 400
gs_it2 = 50
type='Stored_Delassus_Loops          '

nlgs_3D_DiagonalResolution()

# write parameter
freq_write   = 50

```

```
47 # display parameters
48 freq_display = 1 #10
49 ref_radius = 0.01
50
51 mecaMAILx_SparseStorage()
52
53 #
54 # read and load
55 #
56
57 # Set space dimension
58 SetDimension(dim,mhyp)
59 #
60 utilities_logMes('INIT TIME STEPPING')
61 TimeEvolution_SetTimeStep(dt)
62 Integrator_InitTheta(theta)
63 #
64 utilities_logMes('READ BEHAVIOURS')
65 ReadBehaviours()
66 if deformable: ReadModels()
67 #
68 utilities_logMes('READ BODIES')
69 ReadBodies()
70 #
71 utilities_logMes('LOAD BEHAVIOURS')
72 LoadBehaviours()
73 if deformable: LoadModels()
74 #
75 utilities_logMes('READ INI DOF')
76 ReadIniDof()
77 #
78 if deformable:
79   utilities_logMes('READ INI GPV')
80   ReadIniGPV()
81 #
82 utilities_logMes('READ DRIVEN DOF')
83 ReadDrivenDof()
84 #
85
86 # pour calculer une premiere config TT
87 mecaMAILx_ComputeContactDetectionConfiguration()
88
89 utilities_logMes('LOAD TACTORS')
90 LoadTactors()
91 #
92 utilities_logMes('READ INI Vloc Rloc')
93 ReadIniVlocRloc()
94
95 #
96 # paranoid writes
97 #
98 utilities_logMes('WRITE BODIES')
```

```
99  WriteBodies()
100 utilities_logMes('WRITE BEHAVIOURS')
101 WriteBehaviours()
102 utilities_logMes('WRITE DRIVEN DOF')
103 WriteDrivenDof()
104
105 #
106 # open display & postpro
107 #
108
109 utilities_logMes('DISPLAY & WRITE')
110 OpenDisplayFiles()
111 OpenPostproFiles()
112
113 #
114 # simulation part ...
115 #
116
117 # ... calls a simulation time loop
118 # since constant compute elementary mass matrices once
119 utilities_logMes('COMPUTE MASS')
120 ComputeMass()
121
122 # since constant compute elementary stiffness matrices once
123 utilities_logMes('COMPUTE STIFFNESS')
124 ComputeBulk()
125
126 mecaMAILx_ComputeRayleighDamping(0.,0.005)
127
128 # precondensation
129 #mecaMAILx_SetPreconAllBodies()
130 #CLxxx_PushPreconNodes()
131 #ALpxx_PushPreconNodes()
132 #mecaMAILx_ComputePreconW()
133
134 # since constant compute iteration matrix once
135 AssembleMechanicalLHS()
136
137 for k in range(1, nb_steps + 1, 1):
138     #
139     utilities_logMes('INCREMENT STEP')
140     IncrementStep()
141     #
142     utilities_logMes('COMPUTE Fext')
143     ComputeFext()
144     #
145     utilities_logMes('COMPUTE Fint')
146     ComputeBulk()
147     #
148     utilities_logMes('ASSEMBLAGE')
149     AssembleMechanicalRHS()
150     #
```

```
151     utilities_logMes('COMPUTE Free Vlocy')
152     ComputeFreeVelocity()
153     #
154     utilities_logMes('SELECT PROX TACTORS')
155     SelectProxTactors()
156     #
157     utilities_logMes('RESOLUTION' )
158     RecupRloc()
159     #
160     ExSolver(type, norm, tol, relax, gs_it1, gs_it2)
161     UpdateTactBehav()
162     #
163     StockRloc()
164     #
165     utilities_logMes('COMPUTE DOF, FIELDS, etc.')
166     ComputeDof()
167     #
168     utilities_logMes('UPDATE DOF, FIELDS')
169     UpdateStep()
170     #
171     utilities_logMes('WRITE OUT DOF')
172     WriteOutDof(freq_write)
173     #
174     utilities_logMes('WRITE OUT GPV')
175     WriteOutGPV(freq_write)
176     #
177     utilities_logMes('WRITE OUT Rloc')
178     WriteOutVlocRloc(freq_write)
179
180     # On demande ici de recuperer les contraintes principales
181     utilities_logMes('VISU & POSTPRO')
182     if k%freq_display == 0:
183                 WriteDisplayFiles(1,ref_radius)
184                 mecaMAILx_GpvVTKDrawAll(ref_radius/2,1)
185
186     WritePostproFiles()
187
188     overall_CleanWriteOutFlags()
189 #
190 # close display & postpro
191 #
192 CloseDisplayFiles()
193 ClosePostproFiles()
194
195 # this is the end
196 Finalize()
```

## B.5 Script de génération, Section 5.3, structure doublement renforcée, calcul en rigides

```
import os,sys,copy
import numpy
import math
import random

from pylmgc90 import pre

if not os.path.isdir('./DATBOX'):
  os.mkdir('./DATBOX')

# Attention : En 3D l'axe Z est dirige vers le haut
dim = 3

bodies = pre.avatars()
mat    = pre.materials()
svs    = pre.see_tables()
tacts  = pre.tact_behavs()

# On cree un materiau
tdur = pre.material(name='TDURx',materialType='RIGID',density=1494)
mat.addMaterial(tdur)

# On cree un modele rigide
mod = pre.model(name='rigid', physics='MECAx', element='Rxx3D',
    dimension=dim)

# on lit le maillage total dans un fichier, au format gmsh
droxel_mesh = pre.lecture(name='Droxel45.msh', dim=dim)

# On cree un corps de reference
body_ref = pre.volumicMeshToRigid3D(volumic_mesh=droxel_mesh, model
    =mod, material=tdur, color='BLEUx')

# On centre le corps de reference et on lui donne les bonnes
    dimensions
vertices = numpy.matmul(body_ref.bulks[0].axis,body_ref.contactors
    [0].vertices.T).T
dx = numpy.max(vertices[:,0]) - numpy.min(vertices[:,0])
dz = numpy.max(vertices[:,2]) - numpy.min(vertices[:,2])
realdim = [0.25*1.8,0.24099*1.8] #[h,L]
htot = (0.28272*1.8/0.25)*realdim[0]
ratio = realdim[1]/dx
body_ref.contactors[0].vertices[:,:] *= ratio
vertices = numpy.matmul(body_ref.bulks[0].axis,body_ref.contactors
    [0].vertices.T).T

# On definit les espacement dans le maillage de droxels
```

```
43 vSpace=(realdim[0]/2)+0.00001
44 hSpace=(0.26599/0.25)*realdim[0]
45 meshZSpace=vSpace
46 meshXYSpace=hSpace/2
47
48 # On cree une fonction permettant de placer un droxel a un certain
       nombre d'espaces d'un autre
49 def createAndMove(avatar,mdz=0,mdx=0,mdy=0):
50     new_avatar = copy.deepcopy(avatar)
51     if mdz%2!=0 :
52         if mdx%2!=1 or mdy%2!=1:
53             print("[WARNING] : Bloc at coordinates {},{},{} can't
                   be placed".format(mdx,mdy,mdz))
54             return
55         new_avatar.translate(dz=mdz*vSpace, dx=mdx*hSpace/2, dy=mdy
               *hSpace/2)
56     else :
57         if mdx%2!=0 or mdy%2!=0:
58             print("[WARNING] : Bloc at coordinates {},{},{} can't
                   be placed".format(mdx,mdy,mdz))
59             return
60         new_avatar.translate(dz=mdz*vSpace, dx=mdx*hSpace/2, dy=mdy
               *hSpace/2)
61     bodies.addAvatar(new_avatar)
62
63
64 # On cree le maillage de droxels correspondant a la structure
65 legs1= [[0,12,0], [-1,13,1], [-1,11,1], [0,12,2], [-1,13,3],
       [-1,11,3], [0,12,4], [-1,13,5], [-1,11,5], [0,12,6], [6,6,0],
       [6,6,2], [6,6,4], [6,6,6], [12,0,0], [13,-1,1], [11,-1,1],
       [12,0,2], [13,-1,3], [11,-1,3], [12,0,4], [13,-1,5], [11,-1,5],
       [12,0,6]]
66 legs1b= [[6,18,0], [5,19,1], [7,19,1], [6,18,2], [5,19,3],
       [7,19,3], [6,18,4], [5,19,5], [7,19,5], [6,18,6], [12,12,0]
       ,[12,12,2], [12,12,4], [12,12,6], [18,6,0], [19,7,1], [18,6,2],
       [19,5,3], [19,7,3], [18,6,4], [19,5,5], [19,7,5], [18,6,6]]
67 arc1= [[1,11,7], [2,10,8], [3,9,9], [4,8,8], [5,7,7], [7,5,7],
       [8,4,8], [9,3,9], [10,2,8], [11,1,7]]
68 arc1b= [[7,17,7], [8,16,8], [9,15,9], [10,14,8], [11,13,7],
       [13,11,7], [14,10,8], [15,9,9], [16,8,8], [17,7,7]]
69 arc2= [[1,13,7], [2,12,8], [3,11,9], [4,10,10], [5,9,9], [6,8,8],
       [7,7,7], [8,6,8], [9,5,9], [10,4,10], [11,3,9], [12,2,8],
       [13,1,7]]
70 arc2b= [[5,17,7], [6,16,8], [7,15,9], [8,14,10], [9,13,9],
       [10,12,8], [11,11,7], [12,10,8], [13,9,9], [14,8,10], [15,7,9],
       [16,6,8], [17,5,7]]
71 arc3= [[2,14,8], [3,13,9], [4,12,10], [5,11,11], [6,10,10],
       [7,9,9], [8,8,8], [9,7,9], [10,6,10], [11,5,11], [12,4,10],
       [13,3,9], [14,2,8]]
72 arc3b= [[4,16,8], [5,15,9], [6,14,10], [7,13,11], [8,12,10],
       [9,11,9], [10,10,8], [11,9,9], [12,8,10], [13,7,11], [14,6,10],
       [15,5,9], [16,4,8]]
```

```
73  arc4= [[3,15,9], [4,14,10], [5,13,11], [7,11,11], [8,10,10],
        [9,9,9], [10,8,10], [11,7,11], [13,5,11], [14,4,10], [15,3,9]]
74
75  totbod = legs1+legs1b+arc1+arc1b+arc2+arc2b+arc3+arc3b+arc4
76
77  # On place un corps a chaque point du maillage precedemment defini
78  for i in totbod:
79      createAndMove(body_ref, mdz=i[2], mdx=i[0], mdy=i[1])
80
81
82
83  # On definit l'evolution dans le temps de la force a appliquer aux
        droxels concernes
84  VentPa = 2000
85  AireDrox = 0.1884
86  ForceDrox = VentPa * AireDrox
87  def vy(t):
88      if t <= 0.15:
89          return 0
90      else:
91          return  ForceDrox
92  pre.writeEvolution(f=vy, instants=numpy.linspace(0,2,200) ,path='./
        DATBOX/', name='vy.dat')
93
94  # On liste les conditions que l'on veut imposer (ici : droxels de
        base ancres et force de vent sur les droxels exposes)
95  inputDrvDof = {'k1':({'component':[1, 2], 'description':'evolution'
        , 'evolutionFile':'vy.dat','ct':10, 'amp':0., 'omega':0., 'phi'
        :0., 'rampi':1., 'ramp':0., 'dofty':'force'},[[0,12,2],
        [0,12,4], [0,12,6], [6,6,2], [6,6,4], [6,6,6], [12,0,2],
        [12,0,4], [12,0,6], [1,11,7], [2,10,8], [3,9,9], [4,8,8],
        [5,7,7], [7,5,7], [8,4,8], [9,3,9], [10,2,8], [11,1,7],
        [1,13,7], [2,12,8], [3,11,9], [4,10,10], [5,9,9], [6,8,8],
        [7,7,7], [8,6,8], [9,5,9], [10,4,10], [11,3,9], [12,2,8],
        [13,1,7], [2,14,8], [3,13,9], [4,12,10], [5,11,11], [6,10,10],
        [7,9,9], [8,8,8], [9,7,9], [10,6,10], [11,5,11], [12,4,10],
        [13,3,9], [14,2,8], [6,18,2], [6,18,4], [12,12,2], [12,12,4],
        [18,6,2], [18,6,4]]),
96       'k2':({'component':[1, 2, 3, 4, 5, 6], 'dofty':'vlocy'
            },[[0,12,0], [6,6,0], [12,0,0], [6,18,0], [12,12,0],
            [18,6,0]]),
97              }
98
99  # On cree une fonction lisant le dictionnaire des conditions a
        imposer et les appliquant
100 tol=0.0001
101 zCenter=body_ref.nodes[1].coor[2]
102 for value in inputDrvDof.values():
103     for select in value[1]:
104         if len(select)==2:
105             if (select[0][2]+select[0][0])%2!=0 or (select[0][2]+
                    select[0][1])%2!=0 or (select[1][2]+select[1][0])
```

```
                %2!=0 or (select[1][2]+select[1][1])%2!=0 :
106                 print('[WARNING] : Mesh point of one corner can\'t
                        contain any block')
107             for b in bodies:
108                 mZ=meshZSpace
109                 mXY=meshXYSpace
110                 if  b.nodes[1].coor[0] < mXY*max(select[0][0],
                        select[1][0])+tol and b.nodes[1].coor[0] > mXY*
                        min(select[0][0],select[1][0])-tol and b.nodes
                        [1].coor[1] < mXY*max(select[0][1],select[1][1])
                        +tol and b.nodes[1].coor[1] > mXY*min(select
                        [0][1],select[1][1])-tol and b.nodes[1].coor[2]
                        < mZ*max(select[0][2],select[1][2])+tol+zCenter
                        and b.nodes[1].coor[2] > mZ*min(select[0][2],
                        select[1][2])-tol+zCenter:
111                     b.imposeDrivenDof(**value[0])
112         if len(select)==3:
113             if (select[2]+select[0])%2!=0 or (select[2]+select[1])
                    %2!=0 :
114                 print('[WARNING] : Mesh point can\'t contain any
                        block')
115             for b in bodies:
116                 mZ=meshZSpace
117                 mXY=meshXYSpace
118                 if  b.nodes[1].coor[0] < mXY*select[0]+tol and b.
                        nodes[1].coor[0] > mXY*select[0]-tol and b.nodes
                        [1].coor[1] < mXY*select[1]+tol and b.nodes[1].
                        coor[1] > mXY*select[1]-tol and b.nodes[1].coor
                        [2] < mZ*select[2]+tol+zCenter and b.nodes[1].
                        coor[2] > mZ*select[2]-tol+zCenter:
119                     b.imposeDrivenDof(**value[0])
120
121
122
123 # On definit une loi de contact
124 b=pre.tact_behav(name='iqsc0',law='IQS_CLB',fric=0.6) # best for
        concrete = ? (0.3 before)
125 tacts+=b
126
127 # On definit les interactions
128 sv = pre.see_table(CorpsCandidat='RBDY3',candidat='POLYR',
        colorCandidat='BLEUx',behav=b,
129                    CorpsAntagoniste='RBDY3',antagoniste='PLANx',
                           colorAntagoniste='VERTx',alert=.1)
130 svs+=sv
131 sv = pre.see_table(CorpsCandidat='RBDY3',candidat='POLYR',
        colorCandidat='BLEUx',behav=b,
132                    CorpsAntagoniste='RBDY3',antagoniste='POLYR',
                           colorAntagoniste='BLEUx',alert=.1)
133 svs+=sv
134
135 # On ecrit les dossiers et les infos
```

```
136 pre.writeBodies(bodies,chemin='DATBOX/')
137 pre.writeBulkBehav(mat,chemin='DATBOX/',dim=3)
138 pre.writeTactBehav(tacts,svs,chemin='DATBOX/')
139 pre.writeDrvDof(bodies,chemin='DATBOX/')
140 pre.writeDofIni(bodies,chemin='DATBOX/')
141 pre.writeVlocRlocIni(chemin='DATBOX/')
142
143 post=pre.postpro_commands()
144 pre.writePostpro(commands=post, parts=bodies, path='DATBOX/')
145
146 # On demande d'afficher en couleur les blocs soumis a des
      conditions
147 try:
148     pre.visuAvatars(bodies,drvdof_color=[1,0,0])
149 except:
150   pass
```

# Table des figures


1.1 Siège de Vodafone, Porto . . . . . 8
1.2 Droxelisation d'une géométrie . . . . . 9
1.3 Blocs de béton assemblables . . . . . 9
1.4 Cyclotron, Louvain-la-Neuve . . . . . 10

2.1 Vue de haut, de bas et en perspective d'un droxel . . . . . 13
2.2 Possibilités d'emboîtements de droxels . . . . . 14
2.3 Structure en forme de chien composée de droxels . . . . . 14
2.4 Stabilité d'un droxel en fonction des appuis . . . . . 15
2.5 Influence de la géométrie des droxels sur la tolérance de pose et l'angle de montée . . . . . 16
2.6 Plan d'un droxel . . . . . 18

3.1 Moule de droxel (fichier 3D) . . . . . 20
3.2 Moule et système de serrage . . . . . 20
3.3 ABS vs PLA après 14 utilisations . . . . . 21
3.4 Granulométrie de l'argex concassé . . . . . 22
3.5 Finition obtenue pour les 2 compositions . . . . . 23
3.6 Blocs de composition 2 écrasé . . . . . 25
3.7 Dispositif du test de frottement . . . . . 27
3.8 Force vs déplacement pour un droxel de compositttion 1 surchargé de 18 kg 27
3.9 Dispositif de renforcement des encoches . . . . . 29

4.1 Fonctionnement algorithme . . . . . 34
4.2 Maillage d'un droxel dans Gmsh . . . . . 35
4.3 Extraits de résultats du cas 1 pour la composition 1 (Visualisation dans Paraview) . . . . . 37
4.4 Orientation de la force pour les cas 2 et 4 . . . . . 37
4.5 Extraits de résultats du cas 3 pour la composition 1 (Visualisation dans Paraview) . . . . . 38
4.6 Configuration du cas 5 . . . . . 39
4.7 Ancrage pour la composition 2 . . . . . 41
4.8 Exemple de configuration (cas 2) pour les tests sur les 2 compositions . 42
4.9 Force [kN] en fonction du déplacement [mm] pour le test 4 sur le cas 1 . 42

4.10 Exemple de rupture d'encoche (cas 3, 4 et 5) . . . . . 45
4.11 Mode de rupture de la simulation numérique du cas 4 . . . . . 46
4.12 Mode de rupture du cas 4 en laboratoire . . . . . 47
4.13 Numéros des surfaces dans Gmsh . . . . . 50
4.14 Contrainte de Von Mises pour le cas 3, composition 1, t = 2 sec . . . . 52
4.15 Première contrainte principale pour le cas 3, composition 1, t = 2 sec . 53
4.16 Contrainte de Von Mises pour le cas 5 avec un droxel supérieur rigide, composition 1, t = 1 sec . . . . . 54

5.1 Transmission d'effort dans 2 porte-à-faux . . . . . 56
5.2 Passerelle piétonne . . . . . 58
5.3 Phare constitué de droxels . . . . . 59
5.4 Structure étudiée . . . . . 61
5.5 Géométrie de la structure . . . . . 62
5.6 Stabilité face au poids propre . . . . . 63
5.7 Structure avec colonnes renforcées . . . . . 64
5.8 Exposition de la structure face à un vent perpendiculaire à l'axe longitudinal . . . . . 65
5.9 Stabilité face au vent . . . . . 66
5.10 Réaction de la structure doublement renforcée face à un vent perpendiculaire . . . . . 67
5.11 Exemple de renforcement des têtes d'un droxel . . . . . 72

A.1 Représentation de la géométrie . . . . . 79
A.2 Visualisation des résultats dans Paraview . . . . . 80

# Liste des tableaux

3.1 Résultats du test de frottement (Composition 1) . . . . . 28
3.2 Résultats du test de frottement (Composition 2) . . . . . 28

4.1 Influence du coefficient de frottement sur la force rupture (Configuration du cas 3 et composition 1) . . . . . 40
4.2 Récapitulatif des tests numériques . . . . . 40
4.3 Résultats des tests en laboratoire sur le cas 1 . . . . . 43
4.4 Résultats des tests en laboratoire sur le cas 2 . . . . . 44

# Bibliographie

[1] T. P. Firm, "What are the causes of accident on construction site ?," apr 2014.

[2] J. S. Breton and J. Leplat, "Feasability study for drone-based manufacturing of architectural structures," 2015. UCL-Université Catholique de Louvain.

[3] G. S. Org, "Appendix h, soil properties." https ://www.globalsecurity.org/.

Rue Archimède, 1 bte L6.11.01, 1348 Louvain-la-Neuve **www.uclouvain.be/epl**