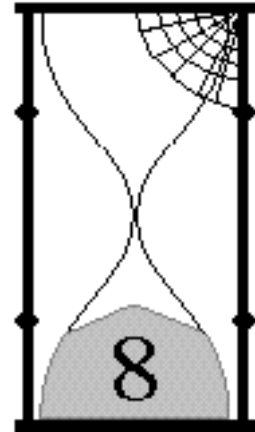


*Dans un grain de sable voir un monde
et dans chaque fleur des champs le Paradis,
faire tenir l'infini dans la paume de la main
et l'Éternité dans une heure.*

*William Blake
Auguries of Innocence*



Techniques applicables à trois dimensions

8.1. Introduction



fin de modéliser de manière encore plus réaliste les grains, on pense évidemment tout de suite à travailler en trois dimensions. Cette idée est louable, mais il faut bien se rendre compte que tout y est plus compliqué. Si on laisse de côté le cas facile des sphères, la détection du ou des points de contact entre deux polyèdres n'est ni simple ni rapide.

Quant aux modèles physiques, l'école des corps indéformables s'applique pour le moment seulement aux sphères, mais le modèle de Cundall 3D et d'autres modèles similaires ont été adaptés à toutes les formes.

Ce chapitre est un peu spécial, puisque que l'on va parler de choses à ne pas faire, et de choses que l'on pourrait essayer ou qui ont été faites par d'autres. Les triangulations sont très délicates à implémenter en 3D, et il faut être prêt à fournir un gros effort de programmation. À notre avis, le jeu n'en vaut pas la chandelle. Après avoir expliqué pourquoi sans trop insister, nous verrons quelques alternatives.

8.2. Pourquoi les triangulations 3D ne sont pas bien adaptées



n 3D, on devrait plutôt parler de tétraédrisation puisque les triangles en 2D correspondent aux tétraèdres en 3D. Mais comme ce mot est horrible et difficile à prononcer, on utilisera encore le terme de triangulation, tout en gardant à l'esprit que l'on se place maintenant en trois dimensions. On va voir que les triangulations ne sont pas bien adaptées à la simulation des milieux granulaires en trois dimensions, et carrément pas

applicables du tout dans le cas de grains polyédriques.

8.2.1. Quand les grains sont des sphères

La triangulation de Delaunay 3D est connue depuis longtemps¹ et peut être maintenue par des opérations locales lorsque les points sont dynamiques². En 2D, le flip était une opération simple consistant à échanger les deux diagonales d'un quadrilatère convexe. En 3D, la transformation analogue est plus compliquée. En effet, le correspondant 3D de l'arête est le triangle et celui du quadrilatère est l'hexaèdre (polyèdre formé de six facettes triangulaires). Un hexaèdre peut être décomposé en tétraèdres de deux façons, comme on peut le voir sur la fig. 8.1.

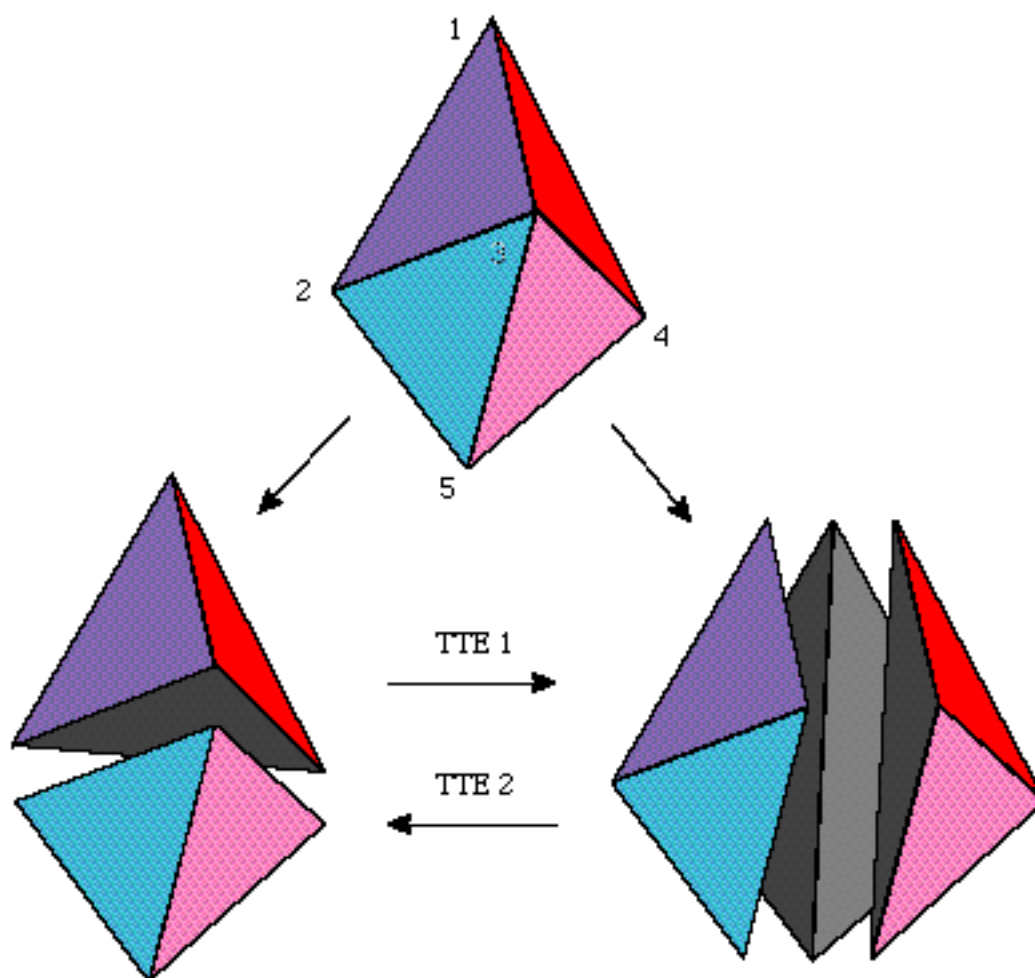


Figure 8.1. Décomposition d'un hexaèdre en deux ou trois tétraèdres

La première décomposition consiste à couper l'hexaèdre en deux par la facette interne (2,3,4). On obtient ainsi les deux tétraèdres (1,2,3,4) et (2,3,4,5). La seconde fait passer une arête centrale par les deux sommets de part et d'autre de cette facette interne; les trois tétraèdres s'obtiennent en trois coups de couteau en faisant en sorte que la pointe du couteau suive l'arête centrale et la lame les arêtes du bord. Les trois tétraèdres sont (1,2,5,3), (1,2,5,4) et (1,3,5,4).

¹ voir 5-[Joe89], 5-[Joe91], 6-[Ten93], 5-[Cig94]. Rappelons que le Delaunay est le graphe dual du Voronoï.

² voir 5-[Alb92], 6-[Roo94]

La transformation correspondant au flip, que l'on appelle en 3D un *swap*, consiste à passer d'une décomposition à l'autre. Cette opération n'est possible que si l'hexaèdre est convexe. On appellera TTE 1 (TTE = Transformation Topologique Élémentaire) la transformation passant de deux à trois tétraèdres et TTE 2 la transformation inverse. Appelons TDP(t_0) la triangulation de Delaunay pondérée non dégénérée au temps t_0 . Il faut encore deux autres transformations pour pouvoir construire n'importe quelle TDP(t_1) à partir d'une TDP(t_0). TTE 3 consiste à insérer un sommet dans un tétraèdre, TTE 4 à enlever un sommet de la triangulation¹.

	nbr de sommets	nbr d'arêtes	nbr de triangles	nbr de tétraèdres
TTE 1	+ 0	+ 1	+ 2	+ 1
TTE 2	+ 0	- 1	- 2	- 1
TTE 3	+ 1	+ 4	+ 6	+ 4
TTE 4	- 1	- 4	- 6	- 4

Tableau 8.1. Effets des quatre TTE sur la triangulation

Quand la TDP est dynamique, des triangles deviennent illégaux. Le test de légalité est semblable à celui fait en deux dimensions (voir formule (4.2)). Un triangle est légal si :

$$\begin{vmatrix} x_1 & y_1 & z_1 & (x_1^2 + y_1^2 + z_1^2) - r_1^2 & 1 \\ x_2 & y_2 & z_2 & (x_2^2 + y_2^2 + z_2^2) - r_2^2 & 1 \\ x_3 & y_3 & z_3 & (x_3^2 + y_3^2 + z_3^2) - r_3^2 & 1 \\ x_4 & y_4 & z_4 & (x_4^2 + y_4^2 + z_4^2) - r_4^2 & 1 \\ x_5 & y_5 & z_5 & (x_5^2 + y_5^2 + z_5^2) - r_5^2 & 1 \end{vmatrix} > 0 \quad (8.1)$$

où (x_i, y_i, z_i) sont les coordonnées du centre de la sphère i de rayon r_i . Les points 1 à 5 sont les sommets d'un hexaèdre et (1,2,3,4) et (2,3,4,5) les deux tétraèdres adjacents à la facette interne (2,3,4). On teste donc les facettes internes de tous les hexaèdres et on a bien une TDP si toutes ces facettes sont légales.

Comme en 2D, on utilise un échéancier. Il permettra ici de déterminer quelle transformation est nécessaire pour maintenir une TDP dynamique. En effet, en 3D plusieurs triangles peuvent être illégaux exactement en même temps (moyennant quelques astuces informatiques pour éviter les problèmes de précision). En supposant que l'on n'ajoute pas de sphères en cours de simulation, le nombre de triangles illégaux simultanément est un critère simple pour savoir quelle TTE appliquer, comme le montre le tableau 8.2.

Nombre de triangles illégaux simultanés	1	3	6
Type de transformation	TTE 1	TTE 2	TTE 4

Tableau 8.2. TTE à effectuer en fonction du nombre de triangles illégaux

¹on peut trouver dans 7-[Rig92], p.103 une preuve que ces quatre opérations sont suffisantes

On le voit, les transformations sont plus compliquées qu'en 2D et certaines belles propriétés disparaissent, comme l'invariance du nombre d'arêtes et de triangles. C'est ennuyeux, mais pas catastrophique, tant que l'on ne doit pas appliquer la TTE 4; cela voudrait dire que des grains devraient disparaître pour maintenir une TDP. Cette propriété peut être utile, par exemple pour modéliser la croissance des polycristaux¹, mais dans notre cas ce serait désastreux.

Il se trouve que lors de nos essais, cela s'est produit ! Lorsque les sphères avaient des rayons différents, il est survenu des configurations telles que l'on devait faire une transformation de type 4, sans qu'il y ait le moindre contact ! L'origine du problème n'a pas été déterminée avec précision, mais ce contretemps nous a permis de constater que déverminer un programme manipulant des triangulations 3D tenait de la gageure... et nous laissons à nos successeurs éventuels le soin de régler ce problème. On a quand même réussi à construire la TDP statique en 3D, dont on peut voir un petit exemple sur la fig. 8.2. Les deux anneaux entourant les sphères permettent de connaître l'orientation de celles-ci.

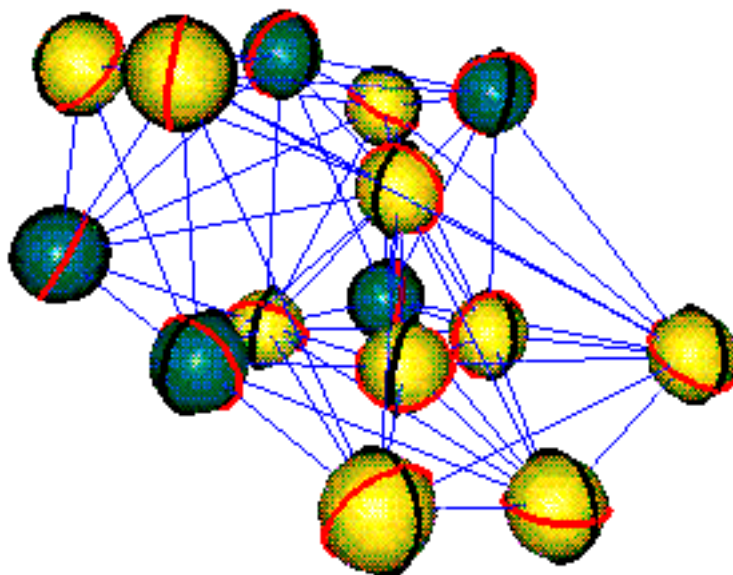


Figure 8.2. Une triangulation de Delaunay pondérée 3D

Cet exemple, qui ne met pourtant en jeu que quinze sphères, nous fait prendre conscience de la difficulté de s'y retrouver dans cette “meule de foin”. Quand on sait qu'un milieu granulaire est composé de plusieurs centaines des grains...

8.2.2. Quand les grains sont des polyèdres

La triangulation des interstices n'est pas généralisable à trois dimensions. Le problème vient du fait que l'on ne peut pas toujours construire une triangulation contrainte en 3D. En 1928 déjà, Schonhart a donné un premier exemple de polyèdre que l'on ne peut pas tétraédriquer dans

¹voir 7-[Tel89], 7-[Rig92], 7-[Tel93], 7-[Xue95]

Mathematische Annalen 98:309-312, exemple repris par Bern et Eppstein dans 5-[Du92], pp. 23-90. Ruppert et Seidel montrent dans les *Proceedings of the Fifth Annual Symposium on Computational Geometry (1989)*, pp. 380-393, qu'il est NP-dur* de déterminer si un polyèdre donné peut être tétraédrisé ou non. Construire une triangulation contrainte est équivalent à trianguler un polyèdre; on peut donc abandonner l'idée de détecter les collisions comme en 2D, et cela d'autant plus que l'école des corps indéformables n'a pas encore été étendue à 3D. De plus, on a vu au chapitre 6 que les triangulations des interstices ne fonctionnaient pas avec des polygones déformables; même s'il était possible de construire ces triangulations, elles ne serviraient à rien.

8.2.3. Synthèse

Nous avons abandonné la technique des triangulations 3D car elle est à notre avis trop compliquée, trop restrictive et sans doute pas très performante par rapport aux techniques classiques. Trop compliquée car elle met en jeu des structures de données très complexes et difficiles à maîtriser¹, trop restrictive car on aimerait pouvoir utiliser autre chose que des sphères, et peu performante car le maintien de la triangulation est délicat et le nombre d'arêtes important. De plus, la phase de mise au point est absolument infernale à cause des structures de données très sophistiquées et parce que l'on a beaucoup de mal à visualiser une triangulation 3D.

8.3. Alors que faire ?



Deux problèmes sont à distinguer : la définition d'un voisinage et la détection des contacts. Rappelons que nos triangulations faisaient les deux choses à la fois en 2D. Les techniques standard de voisinage que l'on a montrées au chapitre 1 - à savoir les quadrillages, les quadrees (qui en 3D s'appellent des octrees), le tri et le halo - sont facilement généralisables à trois dimensions, exception faite des octrees qui demandent la délicate gestion d'un arbre dynamique. Le gros problème en 3D est la détection des contacts, car cela demande beaucoup de temps. De nombreuses méthodes ont déjà été testées en infographie et en robotique pour trouver les points de contact entre des objets polyédriques. Nous présentons ci-dessous une petite sélection de références traitant du sujet.

Dans 5-[Dobk85], Dobkin et Kirkpatrick proposent un algorithme de complexité linéaire pour déterminer la *séparation* de polyèdres convexes. La séparation de deux polyèdres est définie comme la distance minimale entre un point (pas nécessairement un sommet) d'un polyèdre et un point de l'autre. L'algorithme présenté construit la paire de points qui réalise la séparation. Cet algorithme est basé sur une description hiérarchique des polyèdres. Le résultat fournit un algorithme de complexité linéaire pour la détection des intersections de polyèdres convexes.

¹les arêtes-facettes, voir 5-[Dobk89]

Dans 5-[Can86], Canny considère le problème de la détection des collisions d'un objet solide tridimensionnel se déplaçant parmi des obstacles polyédriques. L'espace de configuration de ce problème est de dimension six et la représentation traditionnelle de cet espace utilise trois paramètres de translation et trois angles (typiquement les angles d'Euler). Les contraintes entre les objets font intervenir des fonctions trigonométriques toujours "longues" à calculer sur un ordinateur. L'auteur montre que la représentation des rotations par des quaternions¹ fournit des contraintes purement algébriques dans un espace à sept dimensions. Par de simples manipulations, ces contraintes peuvent être projetées dans un espace à six dimensions sans augmentation de la complexité. La forme algébrique simplifie et accélère grandement les calculs des points de collisions et permet de construire un test efficace d'intersection pour un objet se déplaçant parmi des obstacles.

5-[Moo88] discute de la détection des collisions et leurs conséquences en général, présente deux algorithmes de détection et décrit la modélisation de collisions d'objets arbitraires qui utilise des ressorts, modélisation grossière qui est une simplification du modèle de Cundall.

Dans 5-[Gil88], Gilbert, Johnson et Keerthi décrivent un algorithme pour calculer la distance euclidienne entre une paire de points d'un ensemble convexe dans IR^m . Cet algorithme est approximativement en $O(n)$, où n est le nombre de sommets décrivant les deux polyèdres. Une extension de cette méthodologie à d'autres formes que des polyèdres est décrite dans 5-[Gil90], mais les objets doivent toujours être convexes.

Les références ci-dessus décrivent comment détecter les contacts entre des objets 3D, mais les exemples fournis ne mettent en jeu que peu d'éléments, quelques dizaines au plus. Avec les milieux granulaires, on se trouve devant le problème que, par définition, on doit gérer plusieurs centaines ou milliers d'objets pour avoir des résultats crédibles. Il existe déjà des programmes 3D simulant des milieux granulaires, comme on l'a vu dans le premier chapitre, mais ils utilisent soit des sphères, soit un petit nombre de polyèdres.

La marche à suivre pour minimiser le nombre de tests est d'adopter une attitude hiérarchique, en allant du plus grossier au plus fin, et d'utiliser une représentation aussi adéquate que possible. Dans sa thèse, 3-[O'Co96], dont nous avons déjà parlé dans le chapitre du parallélisme, O'Connor applique ces deux principes à merveille. Sa méthode, qui s'applique aussi bien à deux dimensions qu'à trois, est un peu longue à expliquer en détail, mais en résumé on peut dire les choses suivantes. Le voisinage est déterminé par la méthode du tri selon les axes de coordonnées. Les grains sont enfermés dans des boîtes parallélépipédiques; si les boîtes ne s'intersectent pas, il est inutile de continuer à tester cette paire de grains (principe de hiérarchie). Il faut ensuite avoir une représentation qui repère très vite la zone du polyèdre où le contact peut se produire afin d'éviter de perdre du temps à examiner le polyèdre entier. On peut pour cela discrétiser la surface, comme le fait O'Connor. Une fois la région critique délimitée, il ne reste

¹ quantité de la forme $S + ia + jb + kc$ où i, j, k sont des symboles dont la multiplication obéit à des lois non commutatives. Voir 7-[Ham69].

plus qu'à tester s'il y a contact ou non.

Une autre possibilité pour simuler des milieux granulaires est d'utiliser des automates cellulaires dont on a vu la rapidité mais aussi les inconvénients au chapitre 2.

Enfin, il existe une troisième voie que l'on avait empruntée en 2D, puis abandonnée peut-être un peu tôt parce qu'elle posait des problèmes de visualisation. L'idée consiste à utiliser des sphères mais à déformer la surface de manière probabiliste. En effet, on peut se demander s'il faut à tout prix avoir une représentation précise du grain pour obtenir une simulation réaliste. En utilisant une surface "floue", on pourrait accélérer grandement la vitesse d'exécution en remplaçant les niveaux les plus hauts de la hiérarchie (les plus gourmands en temps de calcul) par des opérations plus simples. Ainsi, on pourrait dire qu'il y a contact lorsque la distance entre les deux surfaces des grains se trouve dans un certain intervalle et avec une certaine probabilité, et que les forces de répulsions agissent dans des directions aléatoires elles aussi. Évidemment, ce serait en réalité plus compliqué si l'on veut garder une certaine cohérence, mais c'est peut-être une idée à approfondir.

8.4. Conclusion



En guise de conclusion, disons que l'étude des techniques informatiques efficaces pour simuler les milieux granulaires en 3D composés d'éléments non sphériques pourraient fournir un travail assez important pour une autre thèse. La technique la plus subtile dont nous ayons connaissance (et semble-t-il la plus efficace) se trouve dans 3-[O'Co96].

Quant aux triangulations, on a renoncé à les utiliser pour les raisons que l'on a vues. Il est cependant certain que l'on doit persévérer dans la voie du 3D, car les simulations 2D ne sont que des approximations des phénomènes réels. Même si l'on parvient à recréer ces phénomènes en 2D, il est toujours possible que l'on induise des effets de bord sans s'en rendre compte. La compréhension profonde des propriétés propres aux milieux granulaires passera nécessairement par des simulations en 3D, même si elles sont elles aussi approximatives. Comme les temps de calcul sont bien plus élevés qu'en 2D, il paraît inévitable que l'on devra aussi avoir recours au parallélisme pour gérer un grand nombre d'objets. Les nouvelles techniques informatiques devront donc à court terme être orientées dans ce sens.

