**DIAGRAMMES DE VORONOI / TRIANGULATIONS DE DELAUNAY**

ouvrir <http://www.cgal.org/>

puis Documentation / overview / Triangulations et Delaunay Triangulations / 2D Triangulations / Delaunay triangulations

L'application fonctionne par évènements, et permet d'insérer interactivement des points dans une triangulation de Delaunay.

1) parcourir le code de l’application pour comprendre les mécanismes d'évènements.

Remarque: la class DT dérive de la triangulation de Delaunay, et est déjà enrichie avec deux fonctions de rendu et une fonction de génération aléatoire.

**ECHAUFFEMENTS**

2) Compléter l'option qui permet de visualiser les arêtes du diagramme de Voronoi (voir cours/CGAL/triangulation\_2).

3) Ajouter l'option qui met en évidence par des arêtes rouges les arêtes de l'enveloppe convexe (sans utiliser convex\_hull()) en ajoutant une fonction draw\_convex\_hull() dans dt.h. Voir encore le cours, il y a un exemple de circulateur autour d'un sommet particulier. Attention, au début de cette fonction et dans toutes les fonctions qui suivent vérifier avant toute chose la dimension de la triangulation, si Dt2 ::dimension() < 2 alors return car la triangulation n’est pas encore de dimension 2.

4) Mettre en évidence la face survolée par la souris en utilisant la fonction locate (Dt2::locate) de la triangulation, en ajoutant une fonction draw\_face\_under\_mouse(const Point& p) dans dt.h. Penser à assembler une primitive OpenGL de type GL\_TRIANGLES. La variable m\_mouse\_pos de la scene doit etre utilisée, elle est déjà mise à jour dans la fonction mouseMoveEvent dans le fichier glviewer.cpp.

5) A partir d'une ligne définie par deux points (choisis arbitrairement), visualiser les faces traversées par cette ligne à l'aide du Line\_face\_circulator. Ce dernier s’utilise comme suit:

*Line\_face\_circulator circ = Dt2::line\_walk(p, q);*  
*Line\_face\_circulator end = circ;*  
CGAL\_For\_all(circ, end)  
{  
// ... draw face here  
}

Pensez à modulariser votre code en créant une fonction gl\_draw\_face(Face\_handle f) membre de la triangulation, qui ne fait que tracer une face, éventuellement avec option de couleurs.

6) Mettre en évidence le sommet le plus proche du point survolé par la souris avec la fonctionnalité nearest\_vertex.

**ALGORITHME DE LLOYD** "simplifié" (Sans regarder ce qu'il se passe au bord)

L'algorithme de Lloyd consiste à alterner un déplacement des générateurs au centre de masse de leur cellule de Voronoi, avec une mise à jour de la partition de Voronoi.

7) calculer les centres de masse des cellules de Voronoi finies.

quelques conseils:

Mettre au point dans un premier temps une fonction Point centroid\_voronoi\_cell(Vertex\_handle v) qui circule sur les faces voisines de v (Face\_circulator) et appelle la fonction Point dual(Face\_handle f) si f est finie. si f est infinie (bool is\_infinite(f)) alors sortir simplement v->point() plutôt que le centre de masse qui n'est pas défini.

Pour calculer le centre de masse, dessiner un polygone convexe et réfléchir à la formule à base de triangulation de ce polygone (arbitraire). notez que l'on peut trianguler ce polygone a partir de v! donc calculer le centre de masse uniquement pendant la phase de circulation sur les faces incidentes a v. aussi, notez que le circulateur peut aller en avant (++) et en arrière (--).

Notez que CGAL adopte une notation algébrique de telle sorte que la différence entre deux points est un vecteur, et que l'on ne peut pas ajouter deux points, etc. Supposons que l'on veuille calculer le barycentre. Il faut utiliser un vecteur comme accumulateur avant de convertir en un point comme suit:

Vector vec = CGAL::NULL\_VECTOR;

for each point pi

vec = vec + (pi - CGAL::ORIGIN);

Point centroid = CGAL::ORIGIN + vec / sum\_areas;

L’itérateur sur les sommets s’utilise comme suit :

Finite\_vertices\_iterator v ;

for (v = finite\_vertices\_begin(); v != finite\_vertices\_end(); v++)

//… do something with v

Avant de lancer LLoyd insérer un ensemble uniforme de points sur le bord du cercle unité, puis aléatoire à l'intérieur.

8) ajouter une option pour afficher les centres de masse des cellules de Voronoi.

9) ajouter une fonction void lloyd() de la classe DT qui effectue une itération de Lloyd.

Idée : Les nouveaux points générateurs sont les centres de masse.

>ALGORITHME DE LLOYD "Le Vrai" (Regarder ce qu'il se passe au bord)

10) gérer l'intersection des cellules avec le bord du domaine.