**TP ENVELOPPE CONVEXE**

Le bon réflexe à avoir:

ouvrir <http://www.cgal.org/Manual/last/doc_html/cgal_manual/contents.html>

et voir la section Convex hulls

<http://www.cgal.org/Manual/latest/doc_html/cgal_manual/Convex_hull_2/Chapter_main.html>

et <http://www.cgal.org/Manual/latest/doc_html/cgal_manual/Convex_hull_2_ref/Chapter_intro.html>

**QT**

Le point de départ est une application Qt/OpenGL 2D basée sur une simple fenetre.

L'application fonctionne par évènements, et permet d'insérer interactivement des points dans une liste

de points STL. Les données (les points) sont stockées dans la scene (fichier scene.h).

Un noyau CGAL est utilisé pour paramétrer des points 2D.

1) parcourir le code de l’application pour comprendre les évènements utilisés (menu clear).

2) compléter la fonction Scene::convex\_hull() pour calculer l'enveloppe convexe de la liste de points. Le résultat doit être stocké dans une liste STL (nommee m\_hull).

Notez depuis par exemple, cette page

<http://www.cgal.org/Manual/latest/doc_html/cgal_manual/Convex_hull_2_ref/Function_convex_hull_2.html#Cross_link_anchor_961> que l'include (#include <CGAL/convex\_hull\_2.h>) a le même nom que la fonction.

3) noter les fonctions de rendu dans la fonction render() pour tracer l'enveloppe convexe sous la forme d'une ligne polygonale brisée fermée (voir GL\_LINE\_LOOP depuis http://www.opengl.org/sdk/docs/man2/xhtml/glBegin.xml).

Remarque: notez les appels à convex\_hull() dans le code pour un affichage correct à chaque ajout de point.

4) écrire deux fonctions lower\_hull() et upper\_hull() et mettre la fonction render() à jour pour distinguer les deux enveloppes associées en bleu et vert. Utiliser cette fois-ci une ligne brisée ouverte avec GL\_LINE\_STRIP.