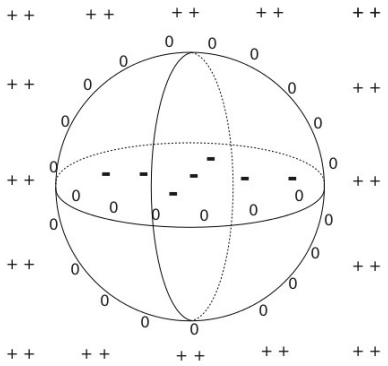
# 1 TP : Fonction à base radiale d'Hermite (HRBF).

L'objectif de ce TP est d'implémenter un système de reconstruction de surface implicite interpolant des données d'Hermite (ensemble de points et normales associées). Pour cela nous utiliserons des fonctions à base radiale.

## 1.1 Rappels :

### 1.1.1 Surface implicite

une surface implicite est définie par une fonction potentiel $f:\mathbb{R}^3\rightarrow\mathbb{R}$. Cette fonction associe à tout point de l'espace $P(x,y,z)$ un scalaire (i.e un nombre unique). Prenons un exemple simple, l'équation cartesienne de la sphère centrée en zéro :



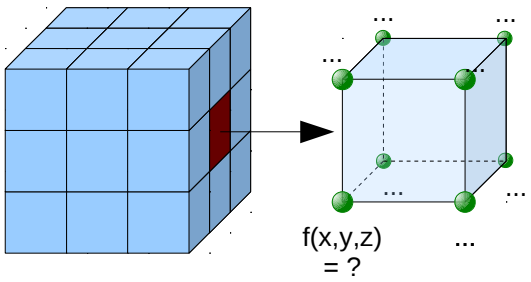
$$f(x,y,z)=x^2+y^2+z^2-R^2=0$$

Pour un point $P$ à l'intérieur de la sphère le potentiel $p=f(P)$ est négatif et à l'extérieur positif. La surface est définie pour $f(P)=0$. La fonction $f$ définit en réalité une infinité d'iso-surfaces selon le potentiel choisi (0, 1 ou encore 0.01 etc. ).

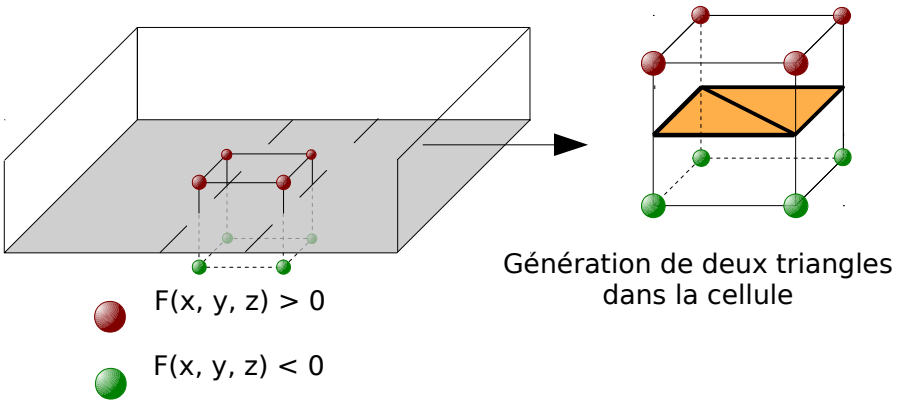
Dans ce TP nous visualiserons les surfaces implicites en utilisant un algorithme de triangulation : le marching cubes.

### 1.1.2 Le marching cubes

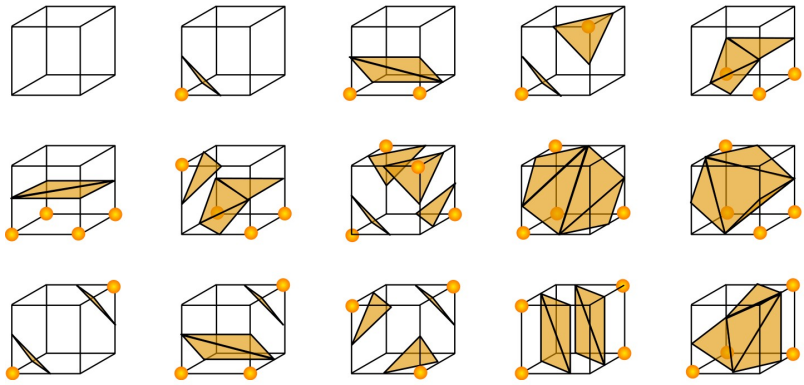
Contrairement aux représentations de surface explicite comme les surfaces paramétriques, on ne peut pas extraire de façon directe un point sur la surface définie par $f$. Il faut discrétiser $f$ sur son domaine à la recherche de la valeur de la surface. Le marching cubes définie une grille 3D sur laquelle nous calculons les différentes valeurs de $f$ :




Les valeurs de $f$ dans chaque cellule de la grille nous permettent alors de générer un ou plusieurs triangles approximant la surface implicite. Ci dessous un exemple de cellule intersectant une surface implicite plane (en gris) :




Voici la liste des 15 configurations possibles et supporté par le marching cube :



La limitation principale du marching cube réside dans son incapacité à reproduire les arrêtes franches (dessinez un carré ; plongez le dans une grille ; et reliez les points d'intersections de la grille avec le carré par des segments). De plus certaines configurations sont ambiguës et peuvent donner lieu à des maillages non 2-manifold (i.e. de variété 2). Pour en savoir plus vous pouvez vous rendre sur le site http://paulbourke.net/geometry/polygonise/ (le sujet utilise le code présenté sur cette page).

### 1.1.3 Normale d'une surface implicite

Après triangulation, il est important de calculer la normale à la surface implicite pour un calcul d'éclairage correct. La calculer à partir du maillage serait une erreur car celui si est une approximation de la surface implicite. Nous pouvons calculer la normale à l'iso-surface de façon plus précise en utilisant directement $f$. En tout point $P$ de l'espace, le gradient de $f$ est orthogonal à l'iso-surface passant par $P$ :

$$\nabla f(x,y,z)=(\frac{\partial f}{\partial x},\frac{\partial f}{\partial y},\frac{\partial f}{\partial z})$$

### 1.1.4 Fonctions à base radiale

Une fonction à base radiale est une fonction dont la valeur dépend de la distance du point évalué $x$ à son centre $p$ (appelé parfois nœud). Nous la notons $\phi(\|p-x\|)$. Voici quelques exemples :

- $\phi(\|p-x\|)=\|p-x\|^3$ (polyharmonique de degrés trois),

- $\phi(\|p-x\|)=e^{-\|p-x\|^2}$ (gaussienne).

Il est possible d'effectuer une combinaison linéaire de fonctions $\phi_i$ afin d'interpoler un nuage de points pour reconstruire une surface implicite.

## 1.2 Sujet

Vous pouvez télécharger et compiler le code associé à ce TP. Pour générer les makefiles avec cmake tapez :

$ cmake -DCMAKE_BUILD_TYPE=Debug .

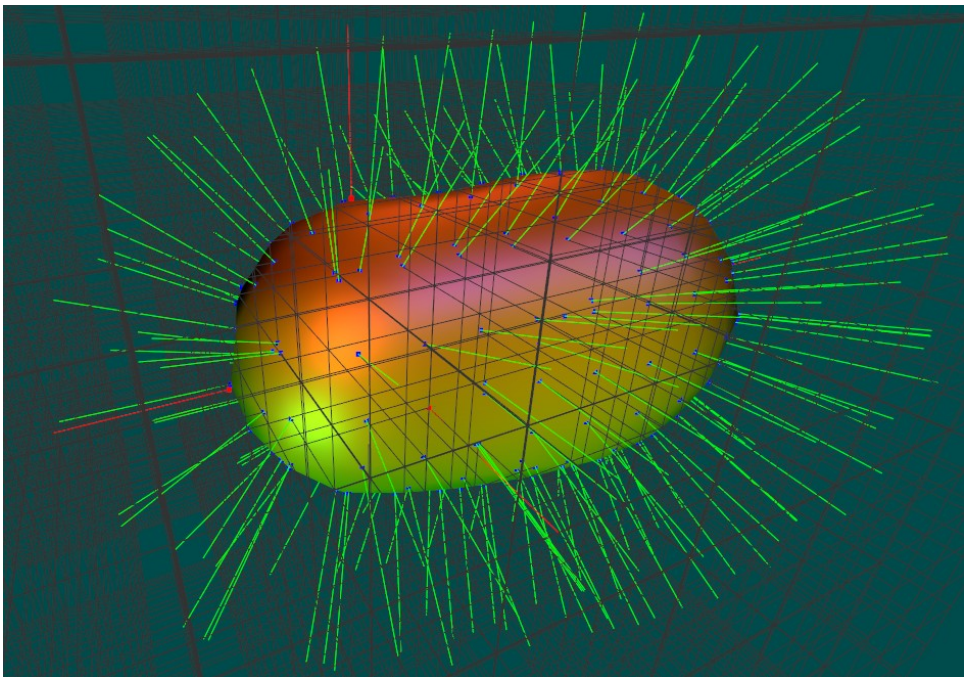
L'option '-DCMAKE_BUILD_TYPE=Debug' permet de compiler en mode debug. Pour compiler tapez simplement le traditionnel :

$ make

L'executable se situe dans ***./bin/*** est doit être executé à partir de celui-ci pour que les shaders soit trouvés et chargés correctement.

Le fichier '***./src/rendersystem/render.cpp***' contient tous les algorithmes de rendu et mise en place des états OpenGL. La scène se compose pour le moment de la grille du marching cubes et du résultat de celui-ci sur une fonction nulle en tout point de l'espace $f(p)=0$. Aucun maillage n'est visible car le marching cubes ne peut reconstruire la surface d'une fonction constante. Pour parcourir le code nous vous conseillons de lancer $ qtcreator proj.pro & .

Vous devez compléter la classe HRBF_fit qui représente une surface implicite afin d'obtenir le maillage suivant :



Des fonctions utilitaires sont mises à votre dispostion pour pouvoir visualiser les points et les normales des maillages (en bleu et vert). Afficher les points et normales utilisés pour interpoler la surface implicite (échantillions en rouge). Et dessinner la grille du marching cubes.

La classe HRBF_fit définit une surface implicite à partir de données d'Hermite (points et normales associées). Vous devez calculer :

- La reconstruction de $f$ (méthode *hermit_fit()* )
- L'évaluation de la fonction de champ $f$ (méthode *eval()* )
- L'évaluation du gradient $\nabla f$. (méthode *grad()* )

Pour ce faire voici l'expression de la HRBF $f$ :

$$f(x)=\sum_i^N \alpha_i \phi_i(\|x-p_i\|)+\sum_i^N \beta_i \nabla \phi_i(\|x-p_i\|)$$

Ici $f$ est une combinaison linéaire de fonctions à base radiale $\phi_i$. Dans le cadre de ce sujet nous prendrons $\phi_i(\|x-p_i\|)=(\|x-p_i\|)^3$. $f$ interpole $N$ points $p_i$. Le problème ici et de trouver les inconnues $\alpha_i$ (qui sont des scalaires) et $\beta_i$ (des vecteurs) en fonction des points $p_i$ et normales $n_i$ associées. Ceci se fait en résolvant le système d'équation suivant :

$$\begin{pmatrix} f(p_i)=0 \\ \nabla f(p_i)=n_i \end{pmatrix}$$

Ce sytème trouve une solution telle que l'iso-surface de potentiel 0 passe par tout les points $p_i$ interpolés et que les gradients de $f$ en ces points $p_i$ soit égaux aux normales $n_i$.

Un premier travail consiste donc à exprimer le problème sous forme matricielle. Puis de le résoudre en utilisant la bibliothèque d'algèbre linéaire **Eigen** (les **#include** sont déjà près) avec une décomposition LU. Par *chance* **Eigen** est très bien documentée (http://eigen.tuxfamily.org/dox/).

**Note :** Nous vous conseillons d'utiliser **wxmaxima** pour vous aider dans vos calculs, notament celui du gradient de $f$ !

**Aide :**

- vous cherchez un systéme du type $Ax=b$**,** avec $x$ le vecteur des inconnues qui incorpore donc les termes $\alpha_i$ et $\beta_i$.
- Il vous faut calculer $\nabla f$ avec **wxmaxima**. Ecrivez explicitement les variables même pour la norme :
  f(x,y,z):= … sqrt( (x-px)^2 + (y-py)^2 + (z-pz)^2) …
- Charger le module pdiff en tapant
  $ *load(pdiff)*
  dans la ligne de commande de **wxmaxima** pour avoir accès aux dérivés de fonctions composées.

## 1.3 Bonus

- Reconstruisez un maillage à partir d'un fichier .obj
- Composez le maillage avec une sphère et l'opérateur de mélange de Pasko pour primitives à support globale :

$$g(f_1,f_2)=f_1+f_2+\sqrt{(f_1^2+f_2^2)}(f_1^2+f_2^2)^{m/2}+\frac{a_0}{1+(f_1/a_1)^2+(f_2/a_2)^2}$$

$m$ correspond à la continuité du mélange (1 devrait suffir). Nous voulons laissons le soins de régler les paramètres $a_i$ .