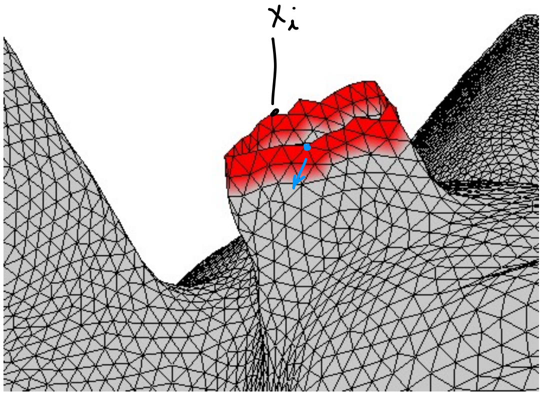
TP réparation de maillages par surface implicite.

Dans ce TP, on souhaite utiliser des surfaces implicites RBF pour réparer des maillages.

Principe

On identifie tout d'abord le bord du trou. Rien de plus simple, ce sont les arêtes qui ne touchent qu'une face (is_boundary le donne) et leurs voisines.



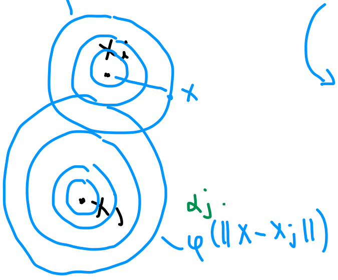
↳ on obtient un ensemble de sommets $(x_i)_{i=1\dots N}$ (on connaît la normale à ces sommets : $\vec{n}_i$).

ce sont les sommets du bord élargi du trou.

La fonction approximante serait naïvement :

$$f(x) = \sum_{i=1}^{N} \underbrace{\alpha_i}_{\text{c.l.}} \cdot \varphi(\|x - x_i\|)$$

où φ sera ici la RBF de plaque mince $\varphi(r) = r^2 \cdot \log(r)$.



$\alpha_i, \alpha_j \in \mathbb{R}$

ce qu'on attend, c'est que la surface associée passe par les points...

↓

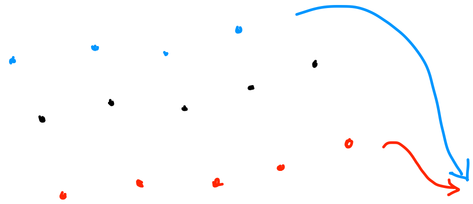
Donc on cherche les α_i tq

$$\forall j \quad \lambda \cdot f(x_j) = 0$$

$$\lambda \cdot f(x_j) = \sum_i \lambda \alpha_i \cdot \varphi(\|x_i - x_j\|)$$

Pb... si on trouve des α_i solution, $(2\alpha_i)_{i=1\dots n}$, $(3\alpha_i)_i$, $\vdots$, $(\lambda\alpha_i)_i$ sont aussi solution...

C'est LE pb. classique



pts de données ←

↓

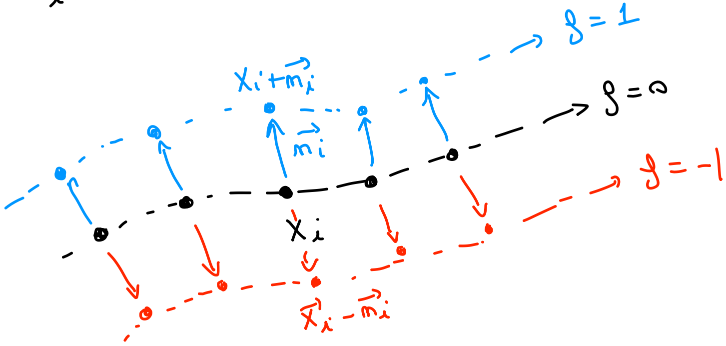
f doit valoir 0

Il faut ajouter des pts "artificiels" où on va imposer

Une solution simple, comme on connait les normales est d'imposer que f vaut 1 aux pts $X_i + \vec{m}_i$ et -1 en $X_i - \vec{m}_i$

une valeur de $f \neq 0$

↳ par exemple $+1$ ou -1



On va donc tripler les centres (où on cale des "gaussiennes")

$(c_i)_{i=1..3N}$ → les pts X_i → où on pose $v_i = 0$

→ les pts $X_i + \vec{m}_i$ → " $v_i = 1$

$\underline{m = 3N}$ → les pts $X_i - \vec{m}_i$ → " $v_i = -1$

pour que le syst. ait une sol. garantie

Au final:

→ modèle
$$f(X) = \sum_{i=0}^{m-1} \alpha_i \cdot \varphi(\|X - c_i\|) + \underbrace{\sum_{j=0}^{9} \beta_j \cdot P_j(X)}$$

question technique, on doit ajouter en plus un polynôme de degré ≤ 2 (en x, y, z)

↓

$a_0 x^2 + a_1 y^2 + a_2 z^2 + a_3 xy + a_4 yz + a_5 xz + a_6 x + a_7 y + a_8 z + a_9$

10 coefs..

↳ ce qu'on veut qu'il vérifie :

(1) $f(c_j) = v_j \quad \forall j = 1 \dots m$

→ m équations $(m = 3.N)$

ms et comme on a $m + d$ inconnues ($d = 10$) à cause du polynôme, on doit ajouter d équations pour que le syst. soit carré. On pose :

(2) $$\sum_{i=0}^{m-1} \alpha_i \cdot P_k(X_i) = 0 \quad \forall k = 0 \dots 9$$

Les inconnues sont les α_i / β_j.

On réécrit ces équations :

$$\begin{cases} (1)\quad \forall j \in 0\ldots m-1 \quad f(c_j) = v_j \quad \begin{matrix} 0 \\ 1 \\ -1 \end{matrix} \\ \qquad\qquad \sum_{i=0}^{m-1} \alpha_i \cdot \varphi(\|c_i - c_j\|) + \sum_{k=0}^{9} \beta_k \cdot P_k(c_j) = v_j \\ (2)\quad \forall k \in 0\ldots 9 \quad \sum_{i=0}^{m-1} \alpha_i \cdot P_k(x_i) = 0 \end{cases}$$

Linéaire / inconnues !

$\Downarrow$

c'est un simple système linéaire à résoudre !

$m+d$ éq.

Mais ... il faut faire apparaître la matrice / vecteur de ce système :

$$A = \begin{array}{c} \\ m \\ d \end{array} \begin{array}{c} \begin{array}{cc} \overleftrightarrow{m} & \overleftrightarrow{d} \end{array} \\ \left(\begin{array}{c|c} \Phi & P \\ \hline P^t & 0 \end{array} \right) \end{array} \qquad B = \left(\begin{array}{c} \vdots \\ v_i \\ \vdots \\ \hline 0 \end{array} \right) \begin{array}{c} m \\ d \end{array}$$

avec $\Phi(i,j) = \varphi(\|c_i - c_j\|)$ — sym $\Phi(i,j) = \Phi(j,i)$

$P(i,j) = P_j(x_i)$

$$\begin{cases} m = 3.N \\ d = 10 \end{cases}$$

$AX = B$ $\qquad X = \left(\begin{array}{c} \vdots \\ \alpha_i \\ \vdots \\ \hline \beta_j \end{array} \right) \begin{array}{c} m \\ d \end{array}$

En résolvant le système, on obtient les coefs α, β déterminant la fonction f :

$\longrightarrow$ qui vaut $v_i = 0$ aux pts de données

$\longrightarrow$ " $v_i = 1$ " " $x_i + \vec{n}_i$

$\longrightarrow$ " $v_i = -1$ " " $x_i - \vec{n}_i$

$\lfloor\!\!\longrightarrow$ en utilisant un polygonaliseur (marching cubes ...) pour mailler $f(x) = 0$, on obtient un maillage qui rebouche le trou.

TP

- Récupérez le matériel de TP
- Allez dans le sous-répertoire src
- Y créer un répertoire « build »
- Aller dans « build » et taper
  cmake ..
  make
- Normalement vous obtenez un exécutable hole_filling qui ne fait rien pour l'instant …

Travail attendu :

1) complétez la fonction « fill_hole » (toutes les fonctions sont déjà pré-implémentées, il s'agit de les assembler …). Le résultat est exporté dans un fichier obj (situé dans le répertoire « build »).

2) tester sur les fichiers test.

3) cet algorithme bouche UN trou - comment le modifier pour boucher TOUS les trous du maillage ?

4) expérimentez :

- en calculant et utilisant la boîte englobante du maillage, choisissez une échelle lambda adaptée pour les normales de telle sorte que les points X_i+lambda*n_i et X_i-lambda*n_i soient sur des isosurfaces « cohérentes » par rapport à la taille du maillage … (si la BB du maillage est de diagonale 0.1, regarder les points X_i+n_i est très loin …)

- dans cette version on ajoutera donc tous les points X_i+lambda*n_i et X_i-lambda*n_i pour « échantillonner » les surfaces +1 / -1, que se passe-t-il si on en met moins (seulement 70% ? 50 % … est-ce que le résultat se déforme beaucoup ? Car par contre, la matrice à inverser est d'autant plus petite …)

4) voire … arrivez-vous à utiliser CGAL pour remplacer ce vilain marching cubes ?