Rapport de stage de M2

Learning the topology of shapes given by point clouds

Mattéo Clémot

Encadrante :
Julie Digne
Équipe Origami (LIRIS)

Février – juillet 2022

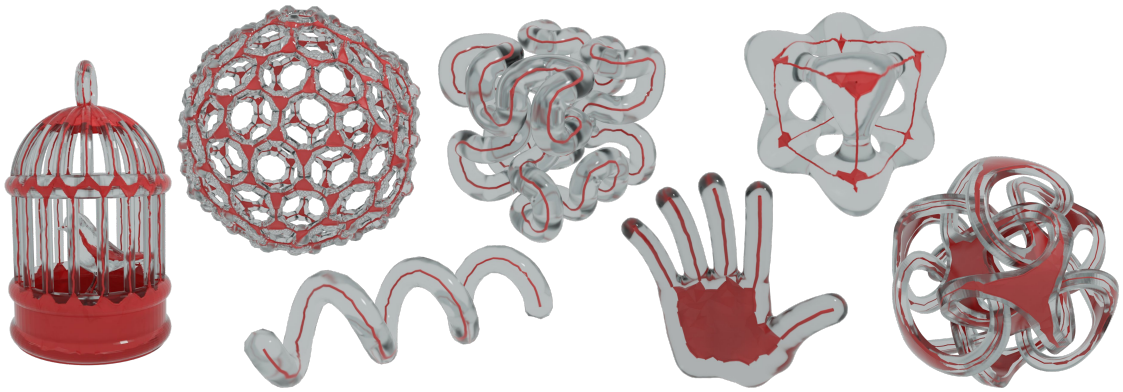


Table des matières

Introduction

Contexte

Il existe différentes manières de représenter un objet tridimensionnel : si le plus courant est probablement d'utiliser un maillage surfacique, on peut également les représenter de façon implicite comme la surface de niveau 0 d'une fonction scalaire définie sur $\mathbb{R}^3$. Un regain d'intérêt est apparu ces dernières années pour cette approche avec l'introduction des *Implicit Neural Representations* (INR) où cette fonction est donnée par un réseau de neurone dont les poids peuvent être optimisés de façon automatique par des méthodes d'apprentissage.

Par ailleurs, l'extraction du squelette d'une forme consiste informellement en l'obtention d'une structure "fine" en forme de graphe, situé au "centre" de la forme, qui permet de capturer à la fois sa topologie et sa géométrie. C'est un problème bien étudié qui peut être approché de différentes manières. L'axe médian est une façon de formaliser la notion de squelette, en le définissant comme l'ensemble des points de l'espace ayant au moins deux plus proches points sur la surface. Il a l'intérêt de donner un squelette qui a du sens dans les régions non tubulaires, à l'inverse des squelettes unidimensionnels.

Objectif

Ce stage s'est concentré sur la possibilité d'extraire un squelette d'une forme à partir d'un échantillonage de points sur sa surface, en utilisant des *Implicit Neural Representations*. En effet, utiliser des méthodes d'apprentissage pourrait permettre une meilleure résilience au bruit dans les données d'entrée en entrée ou à d'éventuelles données manquantes.

Le premier objectif est de maîtriser l'apprentissage d'une fonction de distance signée à la forme, y compris loin de celle-ci alors que les techniques d'INR se concentrent souvent au voisinage de la forme, dans un objectif de reconstruction de surface.

Le second est de parvenir à extraire un squelette qui serait une approximation de l'axe médian de la forme, à partir d'une estimation de sa fonction de distance signée préalablement apprise.

Plan

La Section 1 est consacrée à la formalisation mathématique du problème. Elle pose le problème, définit les notions d'axe médian et de fonction de distance signée, et en donne quelques propriétés utiles. Elle introduit aussi la théorie des solutions de viscosité et présente quelques méthodes récentes de calcul de squelette.

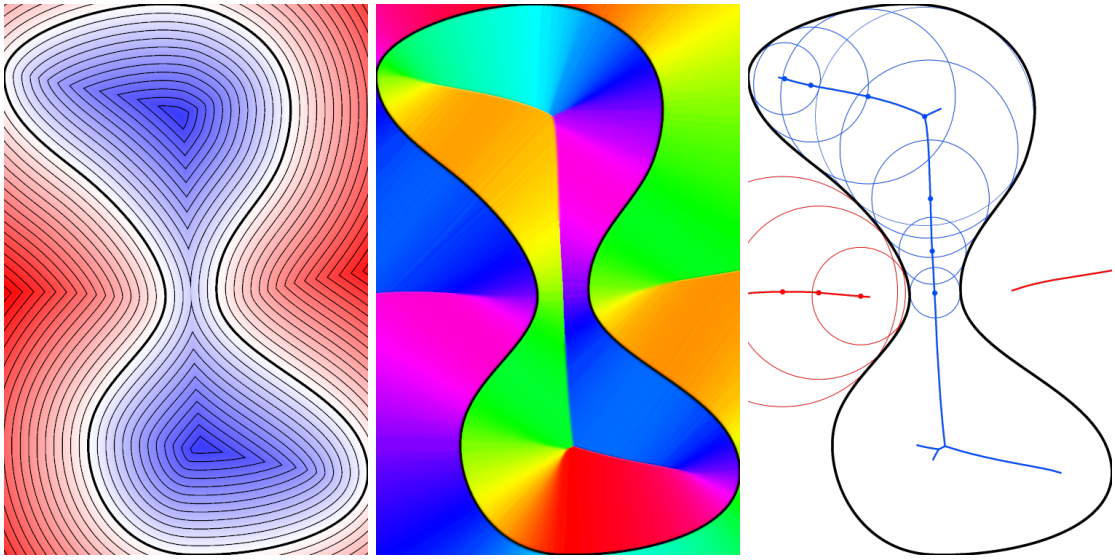
La Section 2 est consacrée à l'apprentissage d'une fonction de distance signée représentée par un réseau de neurone. Elle introduit un nouveau terme de variation totale de la norme du gradient dans la fonction de coût que l'on retrouve dans la littérature, et en discute l'intérêt. Elle y introduit également une méthode d'optimisation inspirée de la théorie des solutions de viscosité.

La Section 3 s'intéresse quant à elle à une alternative à la SDF : on y étudie la possibilité d'apprendre une projection orthogonale sur une surface ou un squelette.

La Section 4 est consacrée à l'extraction d'un squelette à partir de la SDF apprise. Elle se fait en deux temps : d'abord, l'extraction de points situés sur le squelette ; ensuite, la détermination de segments et de triangles entre ces points pour former le squelette.

La Section 5 s'intéresse à l'apprentissage d'un espace de formes, c'est-à-dire à la possibilité d'encoder plusieurs formes à la fois, représentées chacune par un code latent, dans un même réseau. On s'intéresse particulièrement à une approche utilisant un auto-décodeur.

Enfin, la Section 6 donne des détails sur le code écrit pendant le stage et les outils qui ont été utilisés.



(a) (b) (c)

FIGURE 1 – À gauche : SDF d'une courbe fermée $\partial\Omega$ (en noir épais) dans le plan ; l'intérieur (valeurs négatives) est en bleu et l'extérieur (valeurs positives) en rouge. Au centre : la teinte représente la direction du gradient ∇u_Ω de la SDF. Les discontinuités de ∇u_Ω sont ainsi mises en évidences par les changements brusques de teinte. À droite : approximation de l'axe médian (intérieur en bleu et extérieur en rouge) de la courbe $\partial\Omega$, et quelques cercles bitangents à $\partial\Omega$.

1 Formalisation du problème

Dans cette section, on formalise le problème de la squelettisation : la squelettisation consiste à obtenir, à partir d'une forme, une structure simplifiée qui décrit idéalement à la fois la topologie et la géométrie de cette forme. Il existe différente façon de définir cette notion, et de nombreuses façons de la calculer en pratique.

Une formalisation possible est celle de l'*axe médian*. On commence par le définir et préciser ses liens étroits avec la fonction de distance signée. On en donne ensuite quelques propriétés. On s'appuie pour cela entre autres sur les rapports de l'état de l'art d'Attali et al. de 2009 ([ABE09]) et celui de Tagliasacchi et al. de 2016 ([TDS+16]).

1.1 Définitions

Dans les définitions générales qui suivent, on se place en dimension d quelconque. En pratique nous utiliserons $d = 3$.

Définition (Fonction de distance signée)**.** Soit $\Omega \subseteq \mathbb{R}^d$ un ouvert connexe borné non vide. La *fonction de distance signée* (ou SDF pour Signed Distance Function) à Ω est définie par

$$u_\Omega(x) = \begin{cases} -d(x, \partial\Omega) & \text{si } x \in \Omega \\ 0 & \text{si } x \in \partial\Omega \\ d(x, \partial\Omega) & \text{si } x \in \overline{\Omega}^C \end{cases}$$

où $d(x, \partial\Omega) = \min_{y \in \partial\Omega} \|x - y\|$ est la distance euclidienne usuelle de x à $\partial\Omega$.

La SDF u_Ω étant 1-lipschitzienne, une conséquence du théorème de Rademacher est que u_Ω est presque partout différentiable.

Définition (Axe médian)**.** L'axe médian de $\partial\Omega$, noté $\mathcal{M}(\partial\Omega)$, est l'ensemble des points de $\mathbb{R}^d$ tels que la distance $d(x, \partial\Omega)$ est atteinte au moins deux fois. Autrement dit,

$$\mathcal{M}(\partial\Omega) = \left\{ x \in \mathbb{R}^d \mid \#\left\{ y \in \partial\Omega \mid \|x - y\| = d(x, \partial\Omega) \right\} \geq 2 \right\}.$$

De façon équivalente, $\mathcal{M}(\partial\Omega)$ est le lieu des centres des $(d-1)$-sphères qui sont bitangents à $\partial\Omega$, c'est-à-dire qui y sont tangents en au moins deux points (voir Figure 1c).

Lemme 1 (voir par exemple [DF12])**.** Soit $\Omega \subseteq \mathbb{R}^d$ avec une frontière $\partial\Omega$ de classe $\mathcal{C}^1$. Alors pour tout $x \in \mathbb{R}^d$:

— si $x \in \partial\Omega$, u_Ω est différentiable en x et

$$\nabla u_\Omega(x) = n(x), \text{ le vecteur unité normal à } \partial\Omega \text{ en } x$$

— si $x \in \mathbb{R}^d \setminus \partial\Omega$, u_Ω est différentiable en x si et seulement si $x \notin \mathcal{M}(\partial\Omega)$. De plus, si c'est le cas, il existe un unique point $p_x \in \partial\Omega$ tel que $d(x, \partial\Omega) = \|p_x - x\|$, et on a :

$$\nabla u_\Omega(x) = \frac{p_x - x}{u_\Omega(x)}$$

Corollaire 1. En particulier, en tout point x où elle est différentiable, u_Ω vérifie l'*équation eikonale*

$$\|\nabla u_\Omega(x)\| = 1.$$

Corollaire 2. L'axe médian $\mathcal{M}(\partial\Omega)$ de $\partial\Omega$ correspond aux points où la SDF u_Ω n'est pas différentiable (voir Figure 1).

Le lemme suivant justifie le fait de s'intéresser à la fonction de distance signée d'une surface dans l'objectif d'obtenir son axe médian. Il justifiera une méthode d'extraction en Section 4.

Lemme 2. Soit $x \in \partial\Omega$. Il existe $t > 0$ tel que

$$x - t\nabla u_\Omega(x) \in \mathcal{M}(\partial\Omega).$$

Démonstration. Soit $n = \nabla u_\Omega(x)$ le vecteur unité normal à $\partial\Omega$ en x. On note l'ensemble

$$\Pi_{\partial\Omega}(z) = \arg\min_{y \in \partial\Omega} d(z, y).$$

Soit

$$\mathcal{T}_x = \{t > 0 \mid \Pi_{\partial\Omega}(x - tn) \neq \{x\}\}$$

c'est-à-dire les points de la demi-droite $\{x - tn \mid t > 0\}$ dont x n'est pas l'unique projeté sur $\partial\Omega$. Déjà, $\mathcal{T}_x$ n'est pas vide puisque Ω est borné, cette demi-droite intersecte donc $\partial\Omega$ en un point distinct de x. Soit alors $t_0 = \inf \mathcal{T}_x$. On a $t_0 > 0$, puisque n est le vecteur normal à $\partial\Omega$ en x.

— pour tout $\varepsilon > 0$, $\Pi_{\partial\Omega}(x - (t_0 - \varepsilon)n) = \{x\}$ donc $u_\Omega(x - (t_0 - \varepsilon)n) = t_0 - \varepsilon$, donc par continuité de la SDF, $u_\Omega(x - t_0 n) = t_0$, donc $x \in \Pi_{\partial\Omega}(x - t_0 n)$;

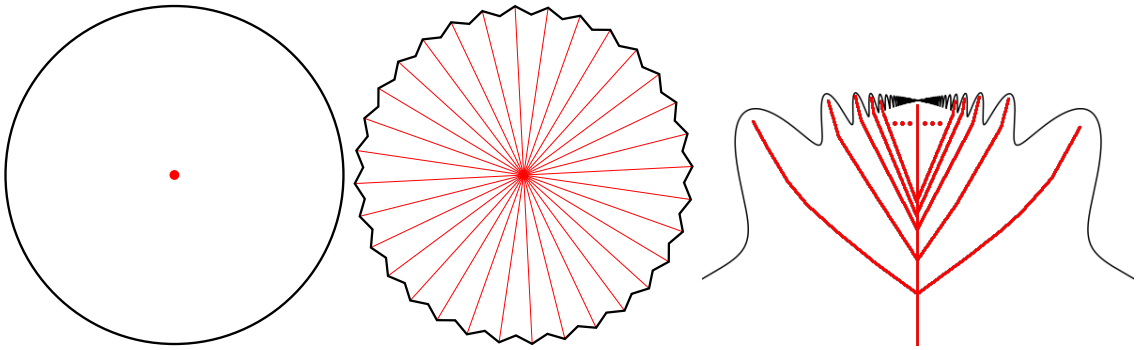
— $\mathcal{T}_x^C = \{t > 0 \mid \Pi_{\partial\Omega}(x - tn) = \{x\}\}$ est un segment ouvert à droite (en effet, si $t \in \mathcal{T}_x^C$, $x - tn \notin \mathcal{M}(\partial\Omega)$ et $t + d(x - tn, \mathcal{M}(\partial\Omega))/2 \in \mathcal{T}_x^C$). Donc $\mathcal{T}_x$ est fermé à gauche, donc $t_0 = \min \mathcal{T}_x$ et donc $\Pi_{\partial\Omega}(x - t_0 n) \neq \{x\}$.

Ainsi $x \in \Pi_{\partial\Omega}(x - t_0 n)$ mais $\Pi_{\partial\Omega}(x - t_0 n) \neq \{x\}$, donc $|\Pi_{\partial\Omega}(x - t_0 n)| \geq 2$ et donc $x - t_0 n = x - t_0 \nabla u_\Omega(x) \in \mathcal{M}(\partial\Omega)$. □

1.2 Propriétés de l'axe médian

Topologie Un résultat d'André Lieutier de 2004 ([Lie04]) affirme que tout ouvert borné de $\mathbb{R}^n$ a le même type d'homotopie que son axe médian. C'est une justification pour prendre l'axe médian comme squelette puisqu'il capture bien la topologie de la forme.

Structure L'axe médian est toujours de mesure n-dimensionnelle nulle (ce résultat a été prouvé par Paul Erdös dès 1946, voir [Erd46]). Ainsi, en deux dimension, l'axe médian est une réunion de courbes de longueur finie. En trois dimensions, l'axe médian est une réunion de variétés à bord, de dimension au plus 2, c'est-à-dire de courbes et de surfaces de mesure finie.



(a) (b) (c)

FIGURE 2 – À gauche et au centre : axe médian de respectivement un cercle et une étoile à 32 branches, illustrant l'instabilité de l'axe médian. À droite : exemple de forme donnant un axe médian avec une infinité de branches.

Finitude Dans la mesure où l'objectif est de le calculer en pratique, il est pertinent de s'interroger sur la finitude de l'axe médian. En l'occurrence, même si la forme $\partial\Omega$ est de classe $\mathcal{C}^\infty$, l'axe médian peut avoir une infinité de branches (voir Figure 2c pour une forme $\mathcal{C}^0$ dont l'axe médian a une infinité de branches). Il existe tout de même des résultats de finitude avec des hypothèses plus fortes, en particulier d'analyticité de la frontière.

Instabilité Malheureusement, la transformation $S \mapsto \mathcal{M}(S)$ n'est pas continue pour la distance de Hausdorff : une petite perturbation de l'entrée peut donner un axe médian très différent, au sens de cette distance (voir Figures 2a et 2b). C'est ainsi une notion instable, ce qui rend son calcul difficile puisque l'on n'a souvent qu'une approximation de la forme en entrée. Cette transformation vérifie cependant une propriété de semi-continuité : pour tout ouvert borné $X \in \mathbb{R}^n$ et $\delta > 0$, il existe $\varepsilon > 0$ tel que pour tout ouvert $Y \in \mathbb{R}^n$,

$$d_H(X^c, Y^c) < \varepsilon \implies \sup_{x \in \mathcal{M}(\partial X)} d(x, \mathcal{M}(\partial Y)) < \delta.$$

Informellement, cela signifie que pour tout Ω, son axe médian $\mathcal{M}(\partial\Omega)$ sera proche d'un sous-ensemble de l'axe médian $\mathcal{M}(\partial\Omega')$ de tout Ω' suffisamment proche de Ω.

1.3 Solutions de viscosité

Les solutions de viscosité, introduites par Pierre-Louis Lions et Michael Crandall, sont un formalisme qui permet de généraliser la notion de solution à une équation aux dérivées partielles (EDP). Cette sous-section est basée sur les définitions et résultats dans l'introduction de Guy Barles ([Bar13]).

En considérant à nouveau le problème de la SDF, on peut remarquer qu'il existe une infinité de solutions continues presque partout différentiables, vérifiant l'équation eikonale là où elles le sont, et vérifiant les conditions aux limites ($u_{|\partial\Omega} = 0$). Il suffit d'observer que c'est déjà le cas en dimension 1, avec une infinité de solutions pour l'équation suivante (voir Figure 3a) :

$$\begin{cases} |u'(x)| = 1 \\ u(1) = u(-1) = 0 \end{cases}$$

Le cadre théorique des solutions de viscosité permettent de définir une solution unique à ce genre de problème, cette solution unique étant u_Ω pour le problème de la SDF. Pour l'équation en dimension 1 ci-dessus, c'est $u : x \mapsto 1 - |x|$ ou son opposée.

Définition (Solution de viscosité pour une EDP du premier ordre)**.** Soit F une fonction continue $\mathcal{D} \times \mathbb{R} \times \mathbb{R}^n \to \mathbb{R}$. u est une solution de viscosité de $F(x, u, \nabla u) = 0$ si et seulement si pour toute fonction $\phi \in \mathcal{C}^1(\mathcal{D})$:

— pour tout maximum local x_0 de $u-\phi$, $F(x_0, u(x_0), \nabla\phi(x_0)) \leq 0$, c'est-à-dire que u est une sous-solution de viscosité ;
— pour tout minimum local x_0 de $u-\phi$, $F(x_0, u(x_0), \nabla\phi(x_0)) \geq 0$, c'est-à-dire que u est une sur-solution de viscosité.

Remarque. Il y a une rupture de symétrie dans la définition d'une solution de viscosité, au sens où une solution de viscosité pour $F(x, u, \nabla u) = 0$ ne l'est pas forcément pour $-F(x, u, \nabla u) = 0$.

Remarque. Les solutions de viscosité peuvent être définies pour les équation non linéaires du second ordre de la forme $F(x, u, \nabla u, \nabla\nabla u) = 0$ où F est une fonction continue $\mathcal{D} \times \mathbb{R} \times \mathbb{R}^n \times \mathbb{S}(n) \to \mathbb{R}$ vérifiant une condition dégénérée d'ellipticité de la forme $F(y, r, p, M_1) \leq F(y, r, p, M_2)$ pour toutes $M_1 \leq M_2$. Cette condition est trivialement vérifiée pour les EDP du premier ordre, donc en particulier pour l'équation eikonale avec $F(x, u, \nabla u) = \|\nabla u\| - 1$.

Théorème 1. L'unique solution de viscosité à l'équation

$$\begin{cases} \|\nabla u\| = 1 \\ u_{|\partial\Omega} = 0 \end{cases}$$

est u_Ω, la fonction de distance signée à $\partial\Omega$.

Vanishing Viscosity Method De façon générale, la *Vanishing Viscosity Method* consiste à, pour trouver la solution de viscosité, prendre la limite d'une famille de solutions $(u_\varepsilon)_{\varepsilon>0}$ à la même équation où l'on a ajouté un terme correspondant au laplacien, pondéré par un coefficient ε que l'on fait tendre vers 0 :

$$\varepsilon\Delta u_\varepsilon + F(\cdot, u_\varepsilon, \nabla u_\varepsilon) = 0 \tag{1}$$

On considère alors la limite uniforme de cette famille de fonctions lorsque $\varepsilon \to 0$:

$$u(x) = \lim_{\varepsilon \to 0^+} u_\varepsilon(x).$$

La propriété qui justifie cette méthode est que cette limite u est alors une solution de viscosité de $F(\cdot, u, \nabla u) = 0$. Historiquement, c'est d'ailleurs ce qui a donné leur nom aux solutions de viscosité.

Cette méthode est utilisée en Section 2.3 avec l'équation eikonale pour forcer la convergence vers la SDF et éviter de "mauvaises" solutions comme en Figure 3b.

1.4 État de l'art en calcul de squelette

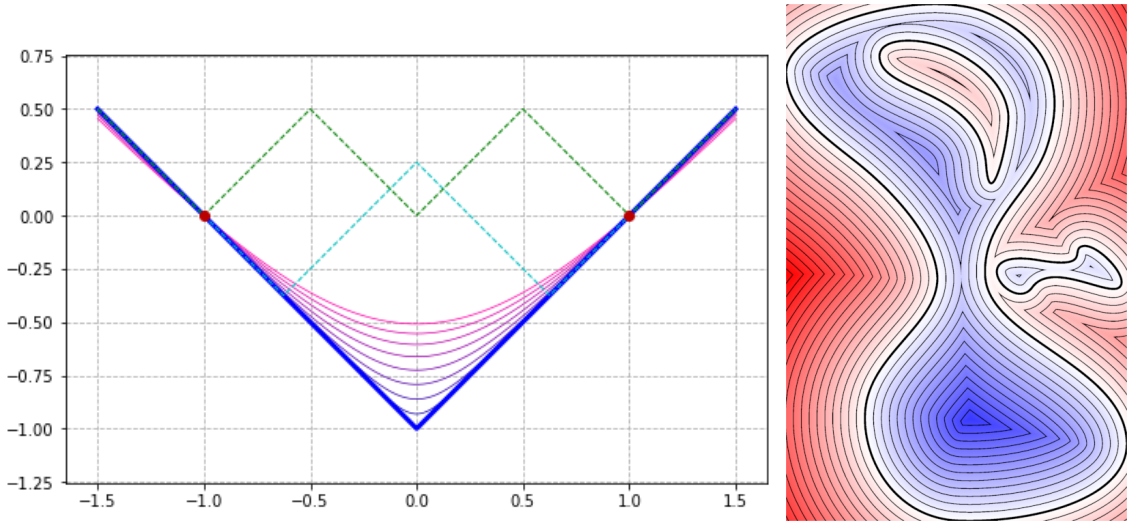
La squelettisation est une question bien étudiée et de très nombreuses méthodes existent (voir [TDS+16]). Dans ce qui suit, on présente rapidement la méthode du diagramme de Voronoï et quelques autres méthodes récentes, représentant la diversité des approches possibles.

Diagramme de Voronoï Le diagramme de Voronoï (voir Section 4 pour une définition) d'un échantillonage $\mathbb{S}$ de points sur une courbe ou une surface $\mathcal{S}$ peut servir à estimer son axe médian. Plus précisément, pour chaque cellule de Voronoï V_p associée à un échantillon $p \in \mathbb{S}$, on peut prendre, pour chacun des deux côtés de $\mathcal{S}$, le sommet de la cellule V_p le plus éloigné de p. Ces sommets sont appelés *pôles* et approchent l'axe médian lorsque la densité de l'échantillonnage augmente. Cependant, la sensibilité au bruit est élevée.

Coverage Axis Dou et al. [DLX+21] proposent d'extraire l'axe médian d'un maillage ou d'un nuage de point en formulant le problème comme un problème de couverture d'ensemble :

1. génération de points intérieurs (p_i, r_i) à partir des sommets du diagramme de Voronoï de l'entrée, où r_i est la distance à la surface ;
2. sélection du plus petit sous-ensemble I_0 de ces points tels que les boules $B(p_i, r_i + \delta)_{i \in I_0}$ couvrent un échantillonage (s_j) de la surface :

$$\begin{array}{ll} \min & \|\mathbf{v}\| \\ s.t. & \mathbf{D}\mathbf{v} \geq \mathbb{1} \\ & \mathbf{v} \in \{0,1\}^{n\times 1} \end{array} \quad \text{où } D_{ij} = \begin{cases} 1 & si \quad \|p_i - s_j\| \leq r_i + \delta \\ 0 & si \quad \|p_i - s_j\| > r_i + \delta \end{cases} \quad \text{et } \mathbb{1} = (1, \ldots, 1)$$



(a) (b)

FIGURE 3 – À gauche : problème unidimensionnel de la SDF pour $\Omega =]-1, 1[$: $|u'| = 1$ presque partout et $u(-1) = u(1) = 0$. En bleu foncé, unique solution de viscosité du problème, $u_\Omega(x) = |x| - 1$. En nuances de violet, suite de solutions u_ε des équations 1, convergeant vers u_Ω. En vert et en cyan, deux solutions au problème qui ne sont pas la solution de viscosité.
À droite : Exemple de solution à $\|\nabla u\| = 1$ presque partout et $u_{|\partial\Omega} = 0$ pour le même Ω qu'en Figure 1, qui n'est pas la solution de viscosité et SDF u_Ω.

Ce problème d'optimisation est alors résolu avec un solveur MILP (Mixed-Integer Linear Programming).

3. établissement des connexions par le calcul du diagramme de Voronoï pondéré de la réunion des points sélectionnés et de l'échantillonage de la surface.

Point2Skeleton Lin et al. [LLL+21] proposent une méthode fondée sur du deep learning :

1. calcul de sphères squelettales (points squelettaux avec rayon de la sphère inscrite) à l'aide d'un MLP qui renvoie des poids pour effectuer une combinaison convexe des points de l'entrée. Les fonctions de coût utilisées tentent de traduire l'erreur de reconstruction de la forme à partir de ces sphères squelettales ;
2. initialisation d'un maillage squelettal par des règles simples, puis prédiction des liens manquants par auto-encodeur (GAE pour *graph auto-encoder*), afin de créer le maillage de squelette final.

ROSA, L1-Medial ROSA [TZCO09] et L1-Medial Skeleton [HWCO+13] sont deux méthodes qui calculent un squelette uni-dimensionnel, qui n'est donc pas exactement ce qui nous intéresse. Cependant, elles peuvent servir de comparaison pour des objets essentiellement tubulaires. La première est fondée sur l'optimisation de plans de coupe. La seconde, sur le calcul de la L1-médiane, avec une définition de la forme

$$\arg\min_X \sum_{i\in I}\sum_{j\in J} \|p_i - s_j\|\theta(\|p_i - s_j\|) + R(X) \text{ avec } \theta(r) = \mathrm{e}^{-r^2/(h/2)^2}$$

où $R(X)$ est un terme de répulsion évitant aux points de se rassembler aux mêmes endroits.

2 Apprentissage de la SDF

Dans cette section, on s'intéresse à l'apprentissage de la fonction de distance signée à une surface dans $\mathbb{R}^3$. Dans la sous-section 2.1 on présente succinctement l'état de l'art dans ce domaine. Les

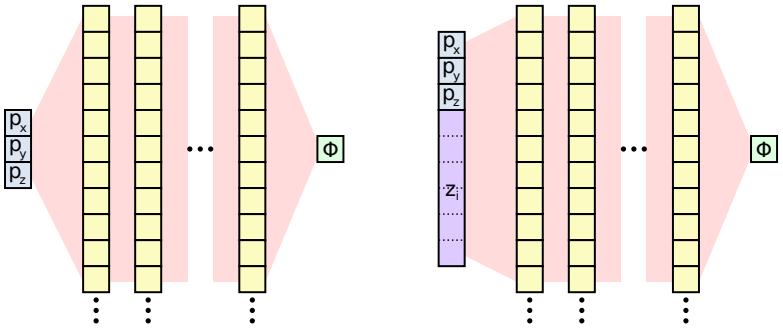


FIGURE 4 – À gauche : architectures d'un MLP employé pour l'apprentissage d'une SDF. À droite : auto-décodeur pour l'apprentissage d'une famille de SDF (Section 5).

sous-sections suivantes présentent mes contributions : 2.2 présente l'architecture générale de la méthode et l'ajout d'un terme de variation totale du gradient à l'architecture SIREN ; 2.3 présente une méthode d'optimisation inspirée de la théorie des solutions de viscosité pour converger vers la bonne solution ; 2.4 présente une technique supplémentaire pour se débarrasser des zones où la SDF est mal apprise.

2.1 État de l'art sur les *Implicit Neural Representations*

Les surfaces implicites, données par la surface de niveau 0 d'une fonction différentiable $f : \mathbb{R}^3 \to \mathbb{R}$, sont une façon de représenter des formes tridimensionnelles. Un regain d'intérêt est apparu pour cette approche avec l'introduction des représentations neuronales implicites (INR, pour *Implicit Neural Representation*) où la fonction f est un réseau de neurones qui peut être optimisé pour construire cette représentation et éviter d'avoir à concevoir la fonction "à la main".

Park et al. [PFS+19] introduisent ainsi une estimation neuronale de la SDF à proximité de la surface, dans un objectif de reconstruction de surface. Ce travail s'intéresse aussi à l'apprentissage d'un espace de formes, en introduisant l'utilisation d'un auto-décodeur capable d'optimiser le code latent d'une forme dans l'espace de forme appris (par Maximum a posteriori) et d'estimer sa SDF. Mescheder et al. [MON+19] s'intéresse aussi à la reconstruction et aux espaces de formes mais en apprenant une fonction d'occupation $f : \mathbb{R}^3 \to [0, 1]$ plutôt que la SDF.

Gropp et al. [GYH+20] proposent une nouvelle façon d'optimiser la SDF, en favorisant la vérification de l'équation eikonale et l'annulation de la SDF sur l'échantillonnage de la surface.

Sitzman et al. [SMB+20] étudient l'intérêt de changer les fonctions d'activation classiques ReLU pour des fonctions périodique (sinus) en introduisant une formulation très générale appelée SIREN, qui peut s'appliquer à la résolution de différentes équations aux dérivées partielles (Poisson, Helmholtz, eikonale). Dans le cadre de l'apprentissage d'une SDF, elle parvient à apprendre d'avantage de détails et à améliorer la convergence lors de l'optimisation.

2.2 Architecture et optimisation

On représente une SDF par un MLP (*MultiLayer Perceptron*) dont l'entrée est un triplet de coordonnées $\mathbf{p} = (x, y, z) \in \mathbb{R}^3$ et la sortie l'estimation $\Phi_\theta(\mathbf{p}) \in \mathbb{R}$ de la SDF en ces coordonnées (voir Figure 4a). Le réseau est constitué de n couches cachées de m neurones avec un sinus pour fonction d'activation :

$$\Phi(\mathbf{p}) = \mathbf{W}_n(\phi_{n-1} \circ \ldots \circ \phi_0)(\mathbf{p}) + \mathbf{b}_n, \quad \phi_i(\mathbf{x}_i) = \sin(\omega_0 \mathbf{W}_i \mathbf{x}_i + \mathbf{b}_i)$$

En pratique, nous utiliserons $n = 8$ et $m = 128$. Par ailleurs, on adopte la valeur $\omega_0 = 30$ proposée dans [SMB+20].

On adopte les fonctions de coût de [GYH+20] et [SMB+20] à quelques adaptations près. L'équation 2 impose une SDF proche de 0 sur l'échantillonnage de points de la surface, ainsi que des gradients alignés avec les normales données en entrée. L'équation 3 impose la vérification de l'équation eikonale, c'est-à-dire contraint la norme du gradient à valoir 1.

$$\mathcal{L}_{\text{surface}} = \int_{\partial\Omega} \|\Phi(\mathbf{x})\| + |1 - \langle \nabla\Phi(\mathbf{x}), \mathbf{n}(\mathbf{x})\rangle| \mathrm{d}\mathbf{x} \tag{2}$$

$$\mathcal{L}_{\text{eikonal}} = \int_{\mathbb{R}^3} |\|\nabla\Phi(\mathbf{x})\| - 1| \mathrm{d}\mathbf{x} \tag{3}$$

Nous avons ajouté un terme (équation 4) correspondant à la variation totale de la norme du gradient, ce qui constitue un terme de régularisation forçant la norme du gradient à être la plus constante possible. L'idée est de minimiser les lieux où le gradient est de norme différente de 1.

$$\mathcal{L}_{\text{TV}} = \int_{\mathbb{R}^3} \|\nabla\|\nabla\Phi\|(\mathbf{x})\| \mathrm{d}\mathbf{x} \tag{4}$$

Enfin, ces trois termes seuls ne parviennent pas en toute généralité à converger vers la bonne SDF loin de la surface. Il est donc nécessaire d'ajouter des points d'apprentissage au processus d'optimisation. Pour cela, en amont de l'optimisation, un ensemble $\mathcal{G}$ de N_{GT} points sont échantillonnés aléatoirement dans le cube unité, et leur plus proche point du nuage en entrée est calculé à l'aide d'un k-d tree, donnant une distance signée $d(\mathbf{p})$ et une direction normalisée $\nabla d(\mathbf{p})$ que l'on considère être la cible. Le terme de l'équation 5 est alors ajouté.

$$\mathcal{L}_{\text{GT}} = \sum_{\mathbf{p}\in\mathcal{G}} |\Phi(\mathbf{p}) - d(\mathbf{p})| + |1 - \langle \nabla\Phi(\mathbf{p}), \nabla d(\mathbf{p})\rangle| \tag{5}$$

La loss final de l'optimisation est alors de la forme

$$\mathcal{L} = \lambda_{\text{s}}\mathcal{L}_{\text{surface}} + \lambda_{\text{e}}\mathcal{L}_{\text{eikonal}} + \lambda_{\text{GT}}\mathcal{L}_{\text{GT}} + \lambda_{\text{TV}}\mathcal{L}_{\text{TV}} \tag{6}$$

où les coefficients de pondération des différents termes sont réglés manuellement ; les valeurs utilisées sont les suivantes :

$$\lambda_e = 20 \quad \lambda_s = 10 \quad \lambda_l = 300 \quad \lambda_{TV} = 1.$$

Remarque. Dans [SMB+20], un terme de la forme de l'équation 7 est ajouté : il permet de contraindre la SDF à être de valeur absolue élevée, dans l'idée de l'empêcher de prendre une valeur nulle en dehors des zones correspondant aux points donnés en entrée. Nous n'avons pas inclus ce terme dans nos expériences : en effet ce terme est problématique pour des formes dont la surface est particulièrement élevée : il y a beaucoup de points dont la SDF doit être proche de 0, ce qui conduit à une surestimation de la SDF.

$$\mathcal{L}_{\text{far}} = \int_{\mathbb{R}^3} \exp\left(-\alpha\|\Phi(\mathbf{x})\|\right) \mathrm{d}\mathbf{x} \tag{7}$$

On résume ci-dessous le processus d'apprentissage de la SDF associée à un nuage de points orientés :

1. tirage de N_{GT} points $\mathbf{p} \in \mathcal{G}$ aléatoires dans le cube unité et calcul de leur distance et direction à l'entrée
2. à chaque itération de l'optimisation :
   (a) tirage de N_{random} points dans le cube unité et estimation de $\mathcal{L}_{\text{eikonal}}$ et $\mathcal{L}_{\text{TV}}$ sur ces points, les gradients étant calculés par auto-différentiation ;
   (b) tirage de N_{surface} points parmi le nuage de points en entrée et estimation de $\mathcal{L}_{\text{surface}}$ sur ces points, les gradients étant calculés par auto-différentiation ;
   (c) calcul de $\mathcal{L}_{\text{GT}}$ avec les points $\mathcal{P}$, les gradients étant calculés par auto-différentiation ;
   (d) étape d'optimisation de $\mathcal{L}$

L'optimisation du réseau est effectuée à l'aide de l'algorithme ADAM avec un *learning rate* de 2×10^{-5}. On prend en général $N_{\text{random}} = 25000$ et $N_{\text{surface}} = \max\{|\mathcal{P}|, 25000\}$. N_{GT} est quant à lui choisi entre 1000 et 10000 selon la complexité de la forme. Enfin, le réseau est initialisé sur la SDF d'une sphère de rayon 0,5, ce qui un effet notoire sur la durée de l'optimisation dans la mesure où l'on part déjà d'une vraie SDF (équation eikonale vérifiée presque partout) qui donne de grandes

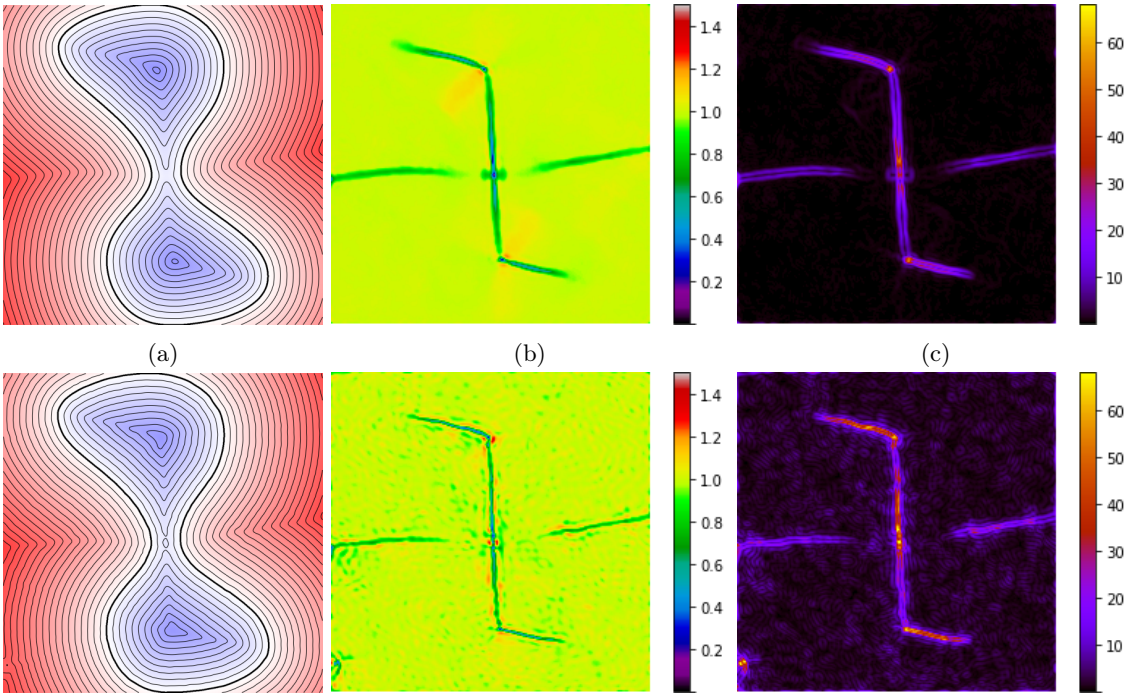


FIGURE 5 – De gauche à droite : SDF Φ, norme du gradient $\|\nabla\Phi\|$, et norme du gradient de la norme du gradient $\|\nabla\|\nabla\Phi\|\|$ de la même forme bidimensionnelle, obtenues avec l'optimisation d'un réseau d'architecture SIREN à 6 couches de 64 neurones, avec (ligne du haut) et sans (ligne du bas) le terme $\mathcal{L}_{\text{TV}}$ de variation totale de la norme du gradient.

valeurs près du bord du cube unité et inversement proche de l'origine, ce qui est souvent le cas pour la SDF cible.

La Figure 5 représente une SDF d'une forme bidimensionnelle obtenue par l'optimisation d'un tel réseau neuronal, avec et sans le terme $\mathcal{L}_{\text{TV}}$ de variation totale de la norme du gradient pour illustrer son intérêt. La Figure 6 représente quelques isosurfaces de trois SDF neuronales.

2.3 Apprentissage inspiré de la *vanishing viscosity method*

Comme vu en Section 1.3, le cadre théorique des solutions de viscosité permet de considérer la *vanishing viscosity method*, où un terme laplacien est ajouté à l'équation eikonale, avec une coefficient tendant vers 0, dans l'objectif de converger vers l'unique solution de viscosité qui est la SDF que l'on souhaite obtenir. Pour cela, on considère la *loss* suivante qui supplante la *loss* eikonale 3 :

$$\mathcal{L}_{\text{VV}} = \int_{\mathbb{R}^3} |\varepsilon\Delta\Phi + \|\nabla\Phi(\mathbf{x})\| - 1|\mathrm{d}\mathbf{x} \tag{8}$$

Remarque. Le laplacien $\Delta\Phi$ est calculé comme la trace de la matrice hessienne. Part conséquent, ce terme est lourd à calculer, et la durée d'optimisation est supérieure à celle de la méthode précédente.

Il faut alors choisir une fonction de décroissance pour ε au cours de l'optimisation du réseau. L'idéal serait d'avoir une transformation quasistatique de ε, ce qui correspondrait à avoir une suite de solutions u_ε. En pratique, j'ai utilisé une fonction logistique de la forme

$$\varepsilon(t) = \varepsilon_0 \left(1 - \frac{1}{1 + \mathrm{e}^{\alpha(-t+t_0)}}\right)$$

dans l'idée d'avoir un plateau à ε_0 pour commencer par converger vers u_{ε_0}, puis une décroissance vers 0 pour déplacer le réseau vers $u_0 = u_\Omega$ qui est l'objectif.

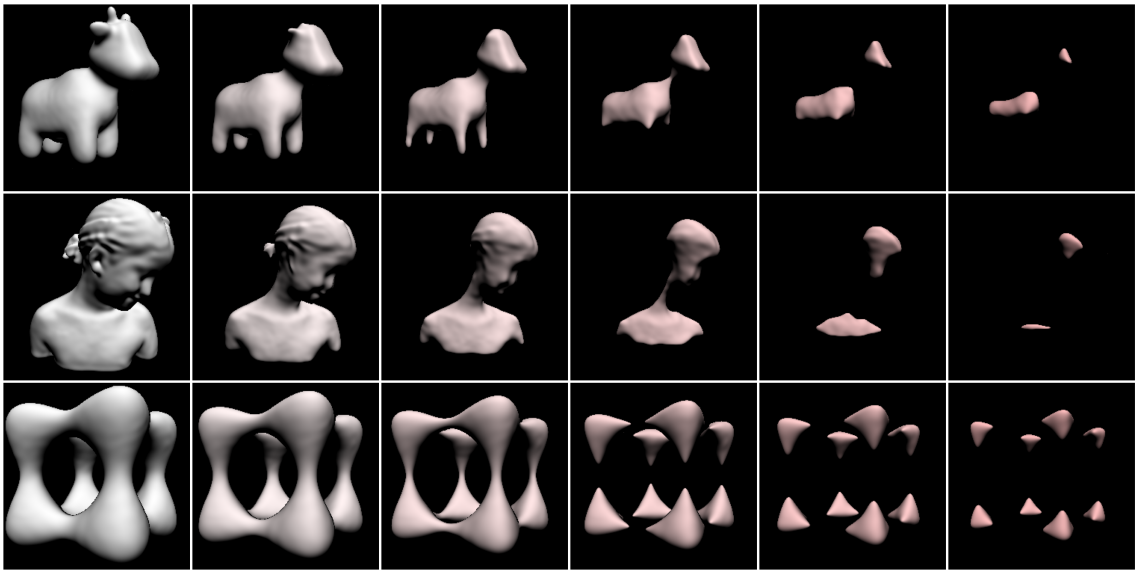


FIGURE 6 – Isosurfaces correspondant à la SDF estimée par un réseau d'architecture SIREN. La colonne de gauche correspond à la surface de niveau 0, et constitue donc une reconstruction de la surface échantillonnée par le nuage de points en entrée.

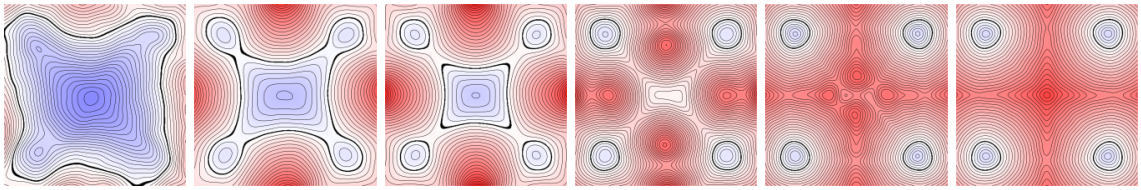


FIGURE 7 – Coupes de la valeur estimée de la SDF pour la forme de la dernière ligne de la Figure 6, à différents itérations de l'optimisation, la valeur de ε dans l'équation 8 n'étant alors pas la même. On voit que malgré l'absence de points d'apprentissage, la bonne solution est trouvée. Sans cette méthode, la SDF estimée serait restée négative au centre du fait de l'initialisation sur une sphère (voir Figure 8).

Cette méthode a donné de très bons résultats sur des formes assez simples (voir Figures 7 et 8), permettant de se passer complètement de points d'apprentissage tout en convergeant vers la bonne SDF, au prix d'un temps d'optimisation plus long. La méthode n'a pas fonctionné pour certaines formes plus complexes, le choix de la courbe de descente de ε semblant responsable.

Remarque. On peut concevoir un mélange de cette méthode avec la précédente : on commence par optimiser avec cette *vanishing viscosity method*, puis une fois que ε est suffisamment petit, on revient à l'équation eikonale "habituelle" et on ajoute le terme de variation totale de la norme du gradient. Les expériences à ce sujet ont donnés de bons résultats : informellement, la première étape permet de converger une solution de bonne qualité *globale*, et la seconde étape permet d'améliorer la qualité *locale* de l'estimation.

2.4 Techniques supplémentaires

Une technique supplémentaire que j'ai essayée pour améliorer la qualité de la SDF est d'ajouter au cours de l'optimisation des points d'apprentissage là où l'estimation peut être considérée comme mauvaise. Plus précisément, tous les n itérations (par exemple $n = 100$), on échantillonne un certain nombre de points de la surface de niveau 0 actuellement représentée par le réseau, en s'inspirant de la méthodologie de [YWOSH21] (voir la Section 4.1 pour plus de détails). On calcule la distance au nuage de point en entrée (en utilisant le k-d tree calculé avant le début de l'optimisation pour les distances des points d'apprentissage au nuage de point) et on ne garde que ceux dont l'estimation

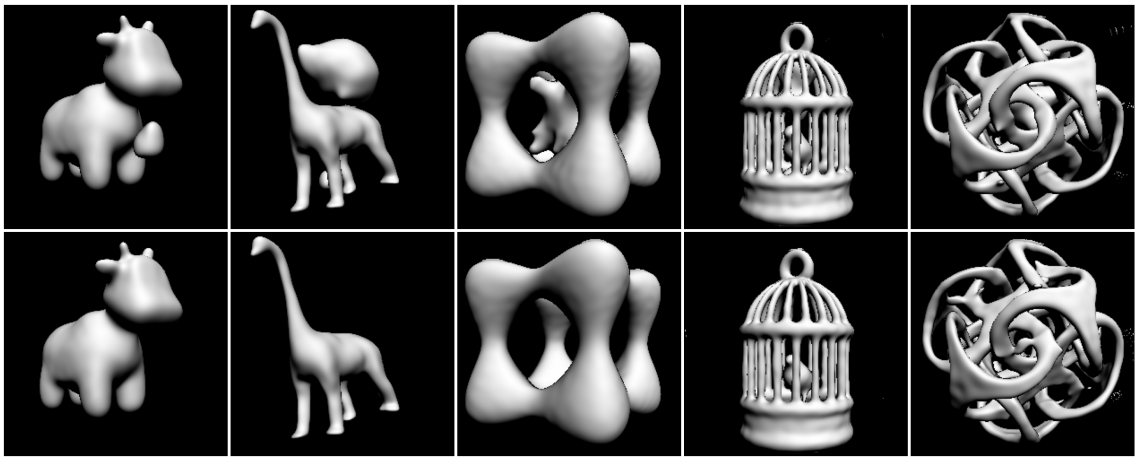


FIGURE 8 – Comparaison de l'isosurface obtenue par optimisation sans points d'apprentissage. Ligne du haut : *loss* eikonale comme dans la Section 2.2 ; ligne du bas : *loss* de l'équation 8 inspirée de la *vanishing viscosity*. Sur ces exemples, à l'exception de metatron (colonne de droite), la *vanishing viscosity method* aide à converger vers la bonne solution, et donc à éviter les "demi-tours" de SDF comme en Figure 3b.

de la SDF est éloignée de cette distance de plus d'un certain seuil. On les ajoute alors aux points d'apprentissage.

On remarque également qu'il ne peut y avoir de maximum local là où la SDF est négative (dans Ω), et pas de minimum local là où elle est positive (dans $\overline{\Omega}^C$). Or on peut obtenir de tels maxima et minima locaux en effectuant une descente ou une montée de gradient, ce qui permet également d'en ajouter aux points d'apprentissage dans le cas où ils sont du "mauvais" côté de $\partial\Omega$.

Remarque. Malheureusement, cette méthode nécessite de ne pas avoir de données manquantes : la SDF ne pourra pas converger vers 0 ailleurs qu'aux alentours du nuage de points.

3 Apprentissage de projections

Cette section présente le travail que j'ai fait concernant l'idée d'apprendre une projection, sur la surface ou sur son squelette.

3.1 Apprentissage d'une projection orthogonale sur la surface

On étudie ici la possibilité d'apprendre, plutôt qu'une fonction de distance, la projection orthogonale sur la surface $S = \partial\Omega$. On veut donc entraîner un réseau $\Pi : \mathbb{R}^3 \to \mathbb{R}^3$ tel que pour tout $\mathbf{p} \in \mathbb{R}^3 \setminus \mathcal{M}(S)$:

$$\Pi(\mathbf{p}) \simeq \arg\min_{\mathbf{q}\in S} d(\mathbf{p}, \mathbf{q}).$$

On s'intéresse aux fonctions de coût suivantes :

$$\mathcal{L}^{\Pi}_{\text{surface}} = \int_S \|\Pi(\mathbf{x}) - \mathbf{x}\| \mathrm{d}\mathbf{x} \tag{9}$$

$$\mathcal{L}^{\Pi}_{\text{idempotence}} = \int_{\mathbb{R}^3} \|\Pi(\Pi(\mathbf{x})) - \Pi(\mathbf{x})\| \mathrm{d}\mathbf{x} \tag{10}$$

$$\mathcal{L}^{\Pi}_{\text{segment}} = \int_{\mathbb{R}^3} \int_0^1 \|\Pi(t\mathbf{x} + (1-t)\Pi(\mathbf{x})) - \Pi(\mathbf{x})\| \mathrm{d}t \mathrm{d}\mathbf{x} \tag{11}$$

$$\mathcal{L}^{\Pi}_{\text{eikonal}} = \int_{\mathbb{R}^3} |\|\nabla\|\Pi - \mathrm{Id}\|(\mathbf{x})\| - 1| \mathrm{d}\mathbf{x} \tag{12}$$

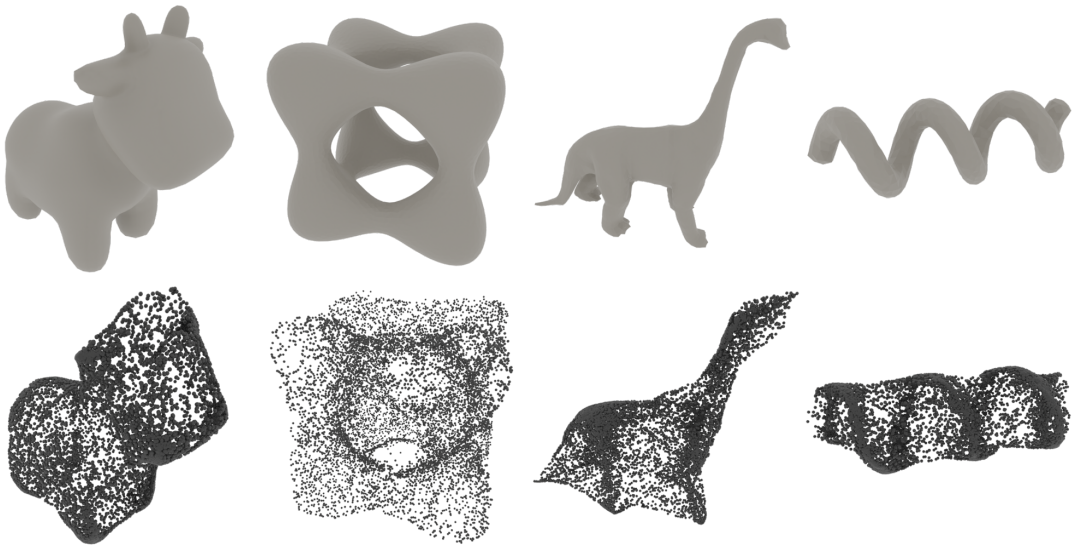


FIGURE 9 – Quelques projections apprises pour différentes formes. L'échantillon de l'image de la projection (ligne du bas) est obtenu en appliquant la projection à 10000 points aléatoires dans le cube unité. On observe la difficulté à faire coïncider cette image aux données d'entrée.

$\mathcal{L}^{\Pi}_{\text{surface}}$ (équation 9) contraint les données d'entrée à être dans l'image de la projection, puisque $\Pi_{|S} = \text{Id}_S$. $\mathcal{L}^{\Pi}_{\text{idempotence}}$ (équation 10) contraint la projection à être idempotente ($\Pi^2 = \Pi$). C'est un cas particulier de la contrainte exprimée avec $\mathcal{L}^{\Pi}_{\text{segment}}$ (équation 11) qui impose Π à prendre la même valeur sur tout le segment de $\mathbf{x}$ à sa projection $\Pi(\mathbf{x})$. Enfin, $\mathcal{L}^{\Pi}_{\text{eikonal}}$ impose la vérification de l'équation eikonale par la fonction de distance non signée u issue de la projection par la relation $u(\mathbf{x}) = \|\Pi(\mathbf{x}) - \mathbf{x}\|$. On ajoute également des points d'apprentissage de façon analogue à la Section 2.2. L'optimisation se déroule de façon très similaire à cette même section. Des résultats sont montrés en Figure 9 : il semble assez difficile de faire coïncider l'image de la projection au nuage de point en entrée.

Remarque. On peut également envisager d'adapter la loss $\mathcal{L}^{\Pi}_{\text{segment}}$ à l'apprentissage d'une SDF. En effet, étant donné $\mathbf{x} \in \mathbb{R}^3 \setminus \mathcal{M}(S)$, $\mathbf{x} - (\text{sign}\,\Phi(\mathbf{x}))\nabla\Phi(\mathbf{x})$ est le projeté orthogonal $\Pi_S(\mathbf{x})$ de $\mathbf{x}$ sur S. Sur le segment entre ces deux points, la SDF doit donc varier linéairement de 0 à $\Phi(\mathbf{p})$. D'où la formulation suivante :

$$\mathcal{L}_{\text{segment}} = \int_{\mathbb{R}^3} \int_0^1 |(\Phi((1-t)\Pi_S(\mathbf{x}) + t\mathbf{x})) - t\Phi(\mathbf{x})|\,\mathrm{d}t\mathrm{d}\mathbf{x} \tag{13}$$

$$\text{où } \Pi_S(\mathbf{x}) = \mathbf{x} - (\text{sign}\,\Phi(\mathbf{x}))\nabla\Phi(\mathbf{x}).$$

Normales Notons que dans ces conditions, on n'utilise pas l'information donnée par les normales. Pour faire cela, une possibilité est d'apprendre en même temps une SDF : on a donc un réseau $(\Pi, \Phi) : \mathbb{R}^3 \to \mathbb{R}^4$ où Φ estime la SDF comme en Section 2.2. On peut alors imposer l'alignement des normales avec $\nabla\Phi$ avec l'équation 2. Il faut alors bien sûr ajouter une *loss* de cohérence entre Π et Φ donnée par l'équation 14, traduisant la nécessité d'avoir $\|\Pi(\mathbf{x}) - \mathbf{x}\| = |\Phi(\mathbf{x})|$:

$$\mathcal{L}^{\Pi}_{\text{SDF}} = \int_{\mathbb{R}^3} ||\Phi(\mathbf{x})| - \|\Pi(\mathbf{x}) - \mathbf{x}\||\mathrm{d}\mathbf{x} \tag{14}$$

Cet ajout semble améliorer les résultats obtenus (voir la Figure 10, qui compare aussi la SDF Φ et la fonction de distance non signée issue de Π).

Le fait que la projection n'est pas continue peut faire s'interroger sur la pertinence d'utiliser un réseau neuronal pour la représenter. Pour autant, on constate que les discontinuités de la projection sont assez bien localisées, en particulier quand on regarde sa direction (voir colonne de droite de

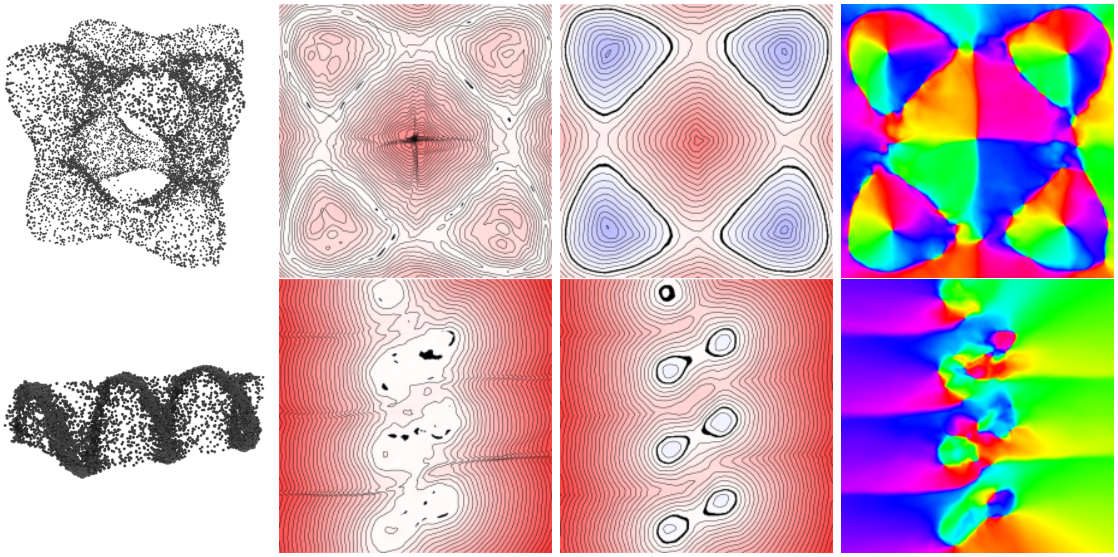


FIGURE 10 – Résultat obtenu avec un réseau neuronal de type (Π, Φ). De gauche à droite : échantillonage de l'image de la projection (application de la projection à 10000 points aléatoires) ; coupe de la distance non signée $\|\mathbf{x} - \Pi(\mathbf{x})\|$; coupe de la SDF $\Phi(\mathbf{x})$; coupe de la direction de $\mathbf{x} - \Pi(\mathbf{x})$ (représentée par la teinte) dans le plan de coupe.

la Figure 10). Dans la mesure où les discontinuités de la projection correspondent à l'axe médian, cela pourrait être intéressant pour l'extraction de ce dernier.

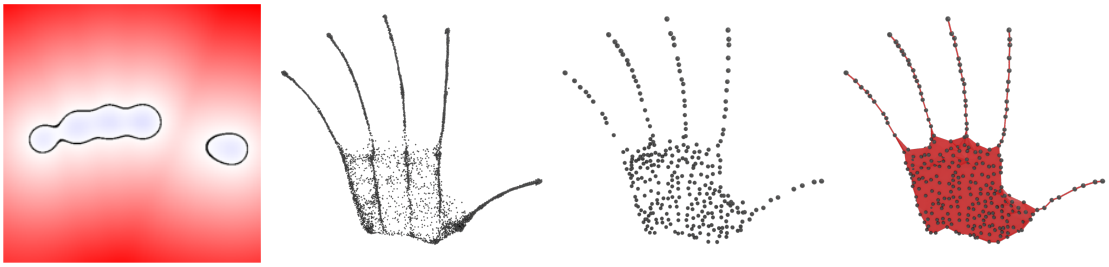
Image de la projection L'optimisation semble avoir parfois du mal à converger vers une projection dont l'image soit bien une surface (ce que l'on souhaite). Par exemple, il existe des zones où l'image de la projection est plutôt "volumique". Pour contraindre davantage l'image de la projection :

- on peut remplacer les termes $\mathcal{L}^{\Pi}_{\text{surface}}$ et $\mathcal{L}^{\Pi}_{\text{idempotence}}$ par la distance de Hausdorff entre le nuage de point et l'image de la projection. Le problème est que cela nécessite d'abandonner la possibilité d'avoir des données manquantes, puisqu'aucune surface ne pourra être complétée là où il n'y a pas de points en entrée ;
- on peut essayer de contraindre algébriquement la projection sur son image. On sait en effet qu'au voisinage de son image, la projection doit être une projection linéaire sur un plan. Plus formellement, pour un point $\mathbf{p} \in \partial\Omega$, la différentielle $\mathrm{d}\Pi(\mathbf{p})$ doit être un projecteur sur un plan. En particulier, son rang doit valoir 2, et ses valeurs propres doivent être $1, 1, 0$. Malheureusement, le rang n'est pas différentiable donc inutilisable pour l'optimisation, et PyTorch ne supporte (actuellement) pas l'auto-différentiation pour le calcul de valeurs propres qui peuvent être complexes (il le supporte avec une hypothèse de caractère hermitien, ce qui n'est pas le cas ici), il est donc difficile d'implémenter cette contrainte sur les valeurs propres de la différentielle.

3.2 Apprentissage d'une projection sur le squelette

On peut aussi imaginer apprendre directement une projection sur le squelette d'une forme. On adopte donc la même architecture $\Pi : \mathbb{R}^3 \to \mathbb{R}^3$ qu'à la section précédente, mais il faut trouver des fonctions de coût caractérisant suffisamment l'axe médian.

Déjà, dans la mesure où l'on apprend encore une projection orthogonale, on peut conserver $\mathcal{L}^{\Pi}_{\text{idempotence}}$, $\mathcal{L}^{\Pi}_{\text{segment}}$ et $\mathcal{L}^{\Pi}_{\text{eikonal}}$ (équation 10, 11,12). Ensuite, pour exprimer le fait que la projection vers le squelette doit se faire vers l'intérieur de la surface, et ne pas rester proche de cette



(a) Coupe de la SDF optimisée (b) Échantillonage du squelette (c) Échantillonage réduit (d) α-complexe

FIGURE 11 – Étapes de la construction du mesh squelettal à partir de la SDF neurale.

surface, on peut ajouter deux fonctions de coût de la forme :

$$\mathcal{L}_{\text{in}}^{\Pi} = \int_{\partial\Omega} \exp\left(-\left\langle \frac{\mathbf{x} - \Pi(\mathbf{x})}{\|\mathbf{x} - \Pi(\mathbf{x})\|}, \mathbf{n}(\mathbf{x}) \right\rangle\right) \mathrm{d}\mathbf{x}$$

$$\mathcal{L}_{\text{far}}^{\Pi} = \int_{\partial\Omega} \exp\left(-\|\mathbf{x} - \Pi(\mathbf{x})\|\right) \mathrm{d}\mathbf{x}$$

Par ailleurs, de façon similaire à la projection sur la surface, la différentielle $\mathrm{d}\Pi(\mathbf{p})$ pour tout point $\mathbf{p}$ de l'image de la projection doit être un projecteur sur une droite ou un plan. En particulier, si on se limite pour simplifier au cas des squelettes unidimensionnels, on pourrait imposer à cette différentielle d'avoir $1, 0, 0$ pour valeurs propres, mais le même problème d'auto-différentiation évoqué plus haut se pose.

Dans tous les cas, les expérimentations sur cette idée de projeter sur un ensemble qui formerait un squelette n'ont pas été concluantes, principalement car il semble difficile de contraindre la structure de l'image de la projection apprise.

4 Extraction du squelette

Cette section présente la méthode que j'ai implémentée pour calculer un ensemble de triangles et de segments représentant le squelette d'une surface, à partir d'une estimation neuronale de sa SDF. L'approche adoptée consiste à d'abord échantillonner un certain nombre de points sur l'axe médian (sous-sections 4.1 et 4.2), puis à les relier entre eux pour former un squelette (sous-section 4.3). La Figure 11 illustre ces étapes successives.

4.1 Extraction de points squelettaux par line search

Dans cette section, on présente la principale approche que nous avons explorée pour l'extraction de points sur l'axe médian à partir d'une SDF neuronale. Elle est justifiée par le Lemme 2. Le tout est résumé dans l'algorithme 1. Tout est implémenté en tenseurs de façon à permettre l'exécution sur GPU, et les tous les gradients sont calculés par auto-différentiation.

Échantillonage de la surface On commence par échantillonner N points sur la surface implicite associée à la SDF neuronale, en s'inspirant de la méthodologie de Yifan et al. [YWOSH21]. Plus précisément, on tire aléatoirement N points $(\mathbf{p}_i^0)_{1 \leq i \leq N}$ dans le cube unité, et on leur applique un certain nombre d'itérations de la méthode de Newton :

$$\mathbf{p}_i^{t+1} = \mathbf{p}_i^t + \frac{\nabla u\left(\mathbf{p}_i^t\right)}{\|\nabla u\left(\mathbf{p}_i^t\right)\|^2} u\left(\mathbf{p}_i^t\right)$$

Comme le gradient est de norme proche de 1, peu d'itérations sont nécessaires. En pratique, on en fait moins de 10.

La méthodologie de Yifan et al. comporte des étapes supplémentaires pour améliorer l'échantillonnage de la surface, avec un resampling uniforme en premier lieu pour améliorer l'uniformité de l'échantillonnage. La méthode permet aussi d'avoir une densité accrue dans les zones à haute courbure par exemple.

Projection sur le squelette Pour trouver des points proche de l'axe médian, le lemme 2 justifie de, pour chaque point $\mathbf{p}$ sur la surface implicite issu de l'étape précédente, calculer le gradient $\nabla u(\mathbf{p})$ qui donne une estimation de la normale $n(\mathbf{p})$, et d'échantillonner régulièrement des points sur un segment partant de $\mathbf{p}$ dans la direction $-\nabla u(\mathbf{p})$ jusqu'à une distance fixée h. On garde le premier point $\mathbf{p}_0$ avec une SDF estimée positive, correspondant au moment où la ligne est sortie de la forme. On ré-échantillonne alors le segment $\mathbf{pp}_0$ régulièrement, et on prend le point $\mathbf{p}'$ où la norme du gradient $\|\nabla u(\mathbf{p}')\|$ est la plus faible, dans l'idée que les endroits où la norme du gradient de la SDF neuronale est faible correspondent aux endroits où la vraie SDF est non-différentiable, c'est-à-dire à l'axe médian.

Algorithm 1: Skeleton sampling

Data: An estimated SDF u, N and n two integer parameters, h maximum search distance.
Result: A set of skeletal points
Sample N points (p_i) on the surface using Newton's method;
for $i = 1 \cdots N$ **do**
  Sample n points $q_j = p_i - t\frac{h}{n}\nabla u(p_i)$ $(t = 1 \cdots n)$;
  Find i_0 the smallest index such that $u(q_{i_0}) > 0$;
  Sample n points r_j between p and q_{i_0};
  Find r_j with lowest $\|\nabla u(r_j)\|$ value;

4.2 Extraction de points squelettaux par minimisation d'énergie

Échantillonner des points sur l'axe médian revient à essayer d'avoir un nuage de points aussi éloigné de la surface que possible, sans pour autant que tous les points soient dans des minima locaux. On peut donc penser à une minimisation d'énergie qui serait constituée d'un terme de SDF, pour contraindre les points à être "loin à l'intérieur", et un terme de répulsion, pour éviter les rassemblements dans les minima locaux :

$$\mathcal{E} = c_{\text{SDF}} \sum_{\mathbf{p}\in\mathcal{S}} \Phi(\mathbf{p}) + c_{\text{rep}} \sum_{\substack{\mathbf{p},\mathbf{q}\in\mathcal{S} \\ \mathbf{p}\neq\mathbf{q}}} \frac{1}{\|\mathbf{p}-\mathbf{q}\|}.$$

En pratique, on peut se limiter à calculer le terme de répulsion pour les plus k proches voisins en utilisant un k-d tree.

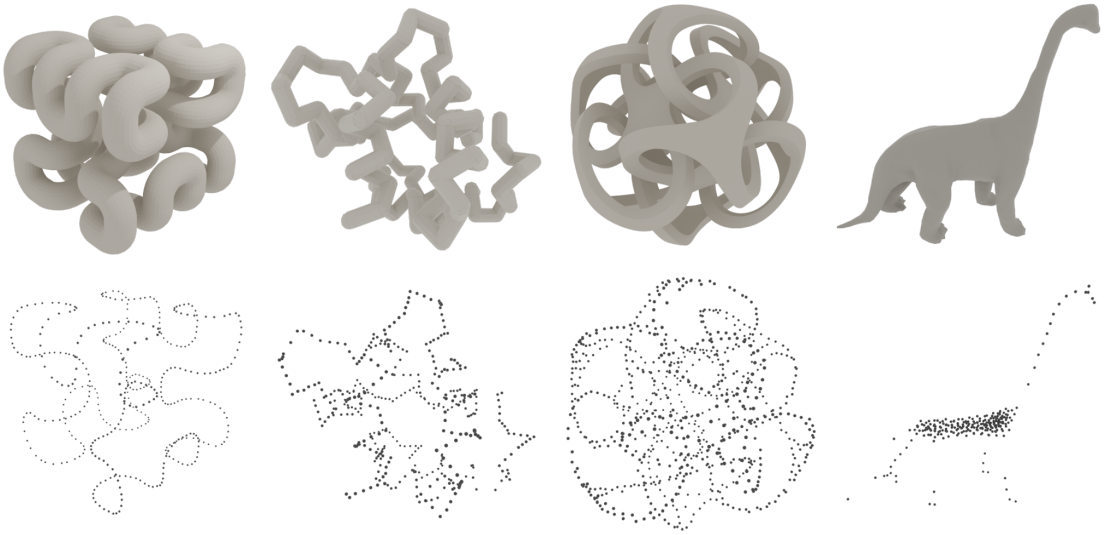
Cette méthode donne de bons résultats lorsque la largeur locale (distance à l'axe médian) de la surface est relativement constante (voir Figure 12). Dans le cas contraire, les points s'accumulent trop dans les zones à grande largeur locale. On pourrait imaginer y remédier en modifiant le terme de répulsion pour le rendre plus fort pour les points ayant une faible SDF, mais je n'ai pas expérimenté davantage cette idée.

Remarque. On peut aussi imaginer ajouter un terme d'élasticité avec les deux plus proches voisins, afin d'encourager les points à être régulièrement espacés dans les zones tubulaires, de la forme

$$c_{\text{elas}} \sum_{\mathbf{p}\in\mathcal{P}} \sum_{\mathbf{q}\in N(\mathbf{p},2)} (\|\mathbf{p}-\mathbf{q}\| - \ell_0)^2.$$

4.3 Construction du squelette

On se donne un nuage de points supposé échantillonner le squelette d'une forme, obtenu que ce soit par *line-search* (Section 4.1), par minimisation d'énergie, ou par une autre méthode encore.



$\lvert\mathcal{S}\rvert = 400$	$\lvert\mathcal{S}\rvert = 400$	$\lvert\mathcal{S}\rvert = 750$	$\lvert\mathcal{S}\rvert = 250$
$c_{\text{SDF}} = 100$	$c_{\text{SDF}} = 100$	$c_{\text{SDF}} = 100$	$c_{\text{SDF}} = 100$
$c_{\text{rep}} = 3$	$c_{\text{rep}} = 3$	$c_{\text{rep}} = 3$	$c_{\text{rep}} = 3$

FIGURE 12 – Quelques exemples d'ensemble de points squelettaux obtenus par minimisation d'énergie. On constate que cette méthode marche particulièrement bien tant que l'épaisseur de la forme est à peu près constante. Dans le cas contraire, les résultats ne sont pas convaincants.

On veut désormais utiliser ces points comme base pour obtenir un ensemble de lignes (dans les régions tubulaires) et de triangles (dans les zones planes) représentant le squelette.

La méthode par *line-search* (Section 4.1) donne un échantillonage a priori non uniforme, et potentiellement beaucoup plus dense que nécessaire. La première étape que nous adoptons consiste donc à réduire la densité de cet échantillonage par une procédure simple, qui consiste à prendre les points un par un et à supprimer tous les autres points qui sont lui sont distants de moins d'un certain rayon fixé.

On calcule alors l'α-complexe de cet échantillonage réduit. Il s'agit d'un sous-ensemble de leur complexe de Delaunay dans lequel on garde uniquement les simplexes dont la plus petite boule sphère circonscrite est de rayon inférieur à α. Un exemple visuel de l'α-complexe d'un même nuage de points dans le plan pour différentes valeurs de α est donné en Figure 13. On donne la définition formelle des notions utilisées ci-dessous.

Définition (Diagramme de Voronoï)**.** Soit P un ensemble de points de $\mathbb{R}^d$ appelés les *sites*. La *cellule de Voronoï* V_u d'un site $u \in P$ est l'ensemble des points de $\mathbb{R}^d$ qui sont plus proches de u que de n'importe quel autre site $v \in P$:

$$V_u = \{x \in \mathbb{R}^d \mid \forall v \in P, \|x - u\| \leq \|x - v\|\}.$$

Définition (Complexe de Delaunay)**.** Le complexe de Delaunay (souvent appelé triangulation de Delaunay en dimension 2) d'un ensemble de points P dans $\mathbb{R}^d$ est le dual de l'ensemble des cellules de Voronoï. Autrement dit, ce sont les simplexes dont les cellules de Voronoï associées à chacun des points sont d'intersection non vide :

$$\text{Del}(P) = \left\{ \sigma \subset P \mid \bigcap_{u \in \sigma} V_u \neq \varnothing \right\}.$$

De façon équivalent, c'est l'ensemble des simplexes possédant une sphère circonscrite vide (au sens où la boule ouverte associée ne contient aucun point de P).

Définition (α-complexe)**.** L'α-complexe $K_\alpha(P)$ d'un ensemble de points P est un sous-ensemble du complexe de Delaunay de P. On ne garde que les simplexes σ dont les cellules de Voronoï

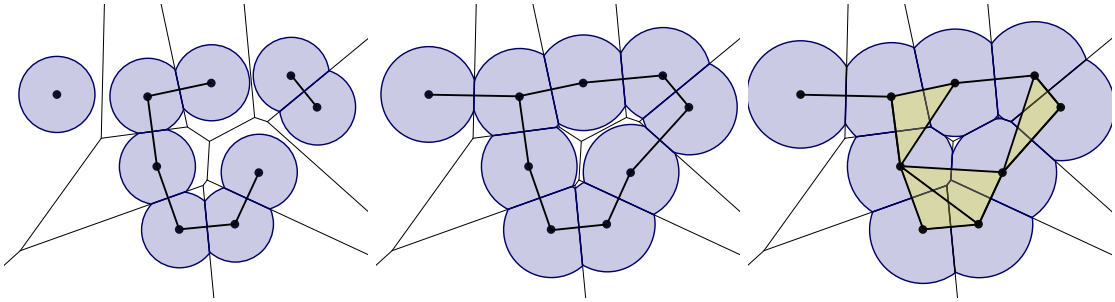


FIGURE 13 – α-complexe du même ensemble de points dans le plan, pour trois valeurs différentes de α. Le diagramme de Voronoï est en noir, et les boules de rayon α en bleu.

intersectées avec une boule de rayon α, c'est-à-dire les $V_u \cap B(u, \alpha)$ associé à chaque $u \in \sigma$, ont une intersection non vide.

$$K_\alpha(P) = \left\{ \sigma \subset P \mid \bigcap_{u \in \sigma} (V_u \cap B(u, \alpha)) \neq \varnothing \right\}.$$

De façon équivalente, c'est l'ensemble des simplexes $\sigma \subset P$ qui ont une sphère circonscrite vide de rayon inférieur à α.

Une fois le calcul de l'α-complexe (calculé en pratique avec la bibliothèque GUDHI), on se retrouve avec un ensemble de segments, triangles, et tétraèdres entre les points de l'échantillonnage réduit. On n'en garde que les segments et les triangles pour obtenir une estimation du squelette.

À propos de la topologie L'α-complexe de P a le même type d'homotopie que l'union des boules de rayon α centrées en les points de P. Ainsi si l'échantillonnage du squelette est correct topologiquement (au sens où il existe α tel que cette union de boules de rayon α a le même type d'homotopie que l'axe médian, et donc que la forme Ω d'après une des propriétés présentée en Section 1), alors le squelette obtenu aura le même type d'homotopie que la forme Ω et capturera bien sa topologie.

Contrainte topologique J'ai aussi réfléchi à contraindre la caractéristique d'Euler χ de l'α-complexe que l'on calcule. En effet, elle doit être égale à la caractéristique d'Euler $\chi(\partial\Omega)$ de la surface, que l'on peut calculer à partir de la SDF en utilisant la formule de Gauss-Bonnet, qui dit que la courbure de Gauss intégrée sur la surface donne $2\pi\chi(\partial\Omega)$. Je n'ai pas encore réussi à faire ce calcul de façon fiable du fait de la difficulté à calculer la surface de l'isosurface de niveau 0.

Résultats L'annexe A présente les résultats de cette méthode de construction du squelette combinée avec l'extraction de points squelettaux par *line-search*. En particulier, les Figures 20 et 21 présentent la résistance du squelette obtenu au bruit sur les données d'entrée et aux données manquantes, et illustre le bénéfice du terme de variation totale de la norme du gradient sur la qualité du squelette.

5 Espace de formes

Dans cette section, on s'intéresse à l'apprentissage d'un espace de formes. Plus précisément, on se donne N formes dont la SDF est notée $(u_i)_{1 \leq i \leq N}$. On souhaite optimiser un réseau Φ_θ (voir Figure 4b) tel que pour un certain code latent $\mathbf{z}_i$ et un triplet de coordonnées $\mathbf{x}$,

$$\Phi_\theta(\mathbf{z}_i, \mathbf{x}) = u_i(\mathbf{x}).$$

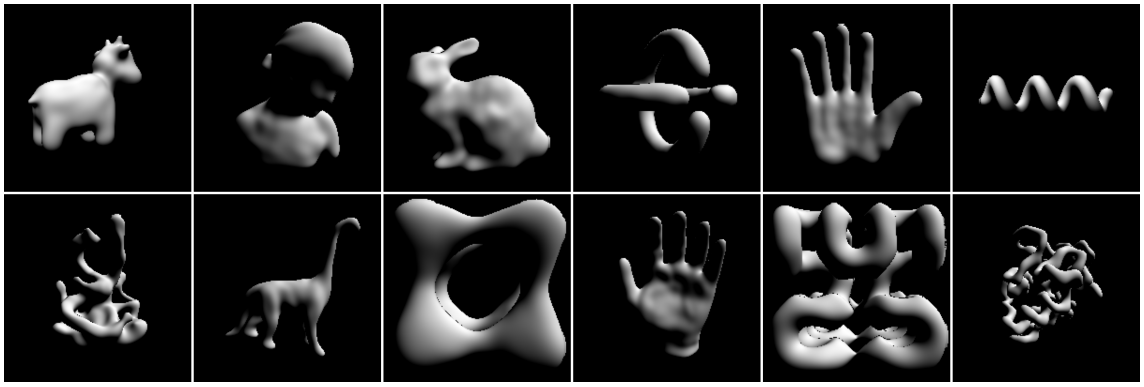


FIGURE 14 – Différentes surfaces implicites stockées dans le même réseau, leur code latent ayant été appris lors de l'optimisation (approche auto-décodeur).

5.1 Auto-décodeur

L'approche la plus naturelle pour obtenir des codes latents est sans doute l'utilisation d'un réseau auto-encodeur. Cependant, dans notre cadre, la nature des données d'entrée, à savoir un nuage de points désordonné, pose problème. En outre, les auteurs de DeepSDF [PFS+19] argumentent que la partie encodeur n'est pas forcément une bonne utilisation des ressources de calcul, et qu'il faudrait concevoir différents auto-encodeurs pour les différents types de données d'entrée (nuage de points, maillage...). On s'intéresse donc à l'approche auto-décodeur, où les codes latents associés aux formes d'apprentissage sont apprises pendant l'optimisation, et où l'obtention d'un code latent pour une nouvelle forme inconnue se fait par Maximum a posteriori (MAP).

On se donne en entrée N nuages de points orientés $(\mathcal{P}^i, \mathcal{N}^i)_{1 \leq i \leq N}$. On note $\mathcal{L}((\mathcal{P}, \mathcal{N}), \Phi)$ la fonction de coût de la Section 2.2, mesurant la qualité de l'estimation $\Phi : \mathbb{R}^3 \to \mathbb{R}$ de la SDF de la forme dont est échantillonné le nuage de points orientés $(\mathcal{P}, \mathcal{N})$. Les codes latents $\{\mathbf{z}_i\}_{i=1}^N$ associés aux données d'entraînement sont alors optimisés en même temps que les paramètres θ du réseau, c'est-à-dire qu'on cherche

$$\underset{\theta, \{\mathbf{z}_i\}_{i=1}^N}{\arg\min} \sum_{i=1}^{N} \mathcal{L}((\mathcal{P}^i, \mathcal{N}^i), \Phi_\theta(\mathbf{z}_i, \cdot)) + \frac{1}{\sigma^2} \|\mathbf{z}_i\|^2.$$

[PFS+19] propose d'initialiser les codes latents $(\mathbf{z}_i)$ aléatoirement selon $\mathcal{N}(0, 0.01^2)$. [GYH+20] les initialise plutôt à $\mathbf{0}$ et propose de les prendre dans $\mathbb{R}^{256}$.

À l'inférence, les paramètres θ du réseau ont été fixés. Le calcul du code latent $\hat{\mathbf{z}}$ d'une nouvelle entrée $(\mathcal{P}, \mathcal{N})$ s'estime par Maximum a posteriori :

$$\hat{\mathbf{z}} = \underset{\mathbf{z}}{\arg\min} \left(\mathcal{L}((\mathcal{P}, \mathcal{N}), \Phi_\theta(\mathbf{z}, \cdot)) + \frac{1}{\sigma^2} \|\mathbf{z}\|^2 \right).$$

Par rapport à un auto-encodeur, cette méthode a le désavantage de nécessiter des itérations d'optimisation pour trouver un nouveau code latent. En revanche, elle peut très facilement s'adapter à d'autres type de données d'entrée.

5.2 Résultats

Afin de tester la méthode, j'ai entraîné un unique réseau stockant 12 formes distinctes (voir Figure 14). Il est constitué de 8 couches de 128 neurones, et les codes latents associés à chaque forme et optimisés en même temps que les poids du réseau sont pris dans $\mathbb{R}^{128}$. Prendre moins, par exemple 64, pour la taille des couches ou même pour la dimension des codes latents n'a pas permis d'apprendre de façon raisonnable ces 12 formes.

Afin d'explorer l'espace de formes ainsi appris, la Figure 15 montre les formes obtenues lorsqu'on interpole linéairement les codes latents appris pour deux formes distinctes, ici un dinosaure et une main. On remarque que l'interpolation obtenue est raisonnable, alors qu'à priori, $\Phi_\theta(\mathbf{z}, \cdot)$ pour un

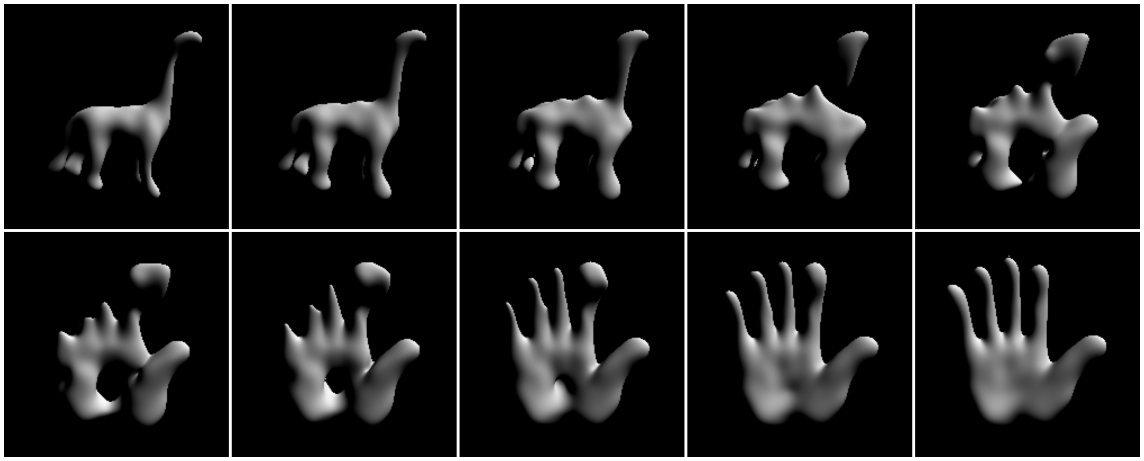


FIGURE 15 – Surfaces implicites correspondant à une interpolation linéaire des codes latents entre un diplodocus et une main.

code interpolé $\mathbf{z}$ n'a pas reçu de contrainte lors de l'optimisation pour être l'approximation d'une SDF. En revanche, l'inférence ne fonctionne pas pour une nouvelle forme ou une des formes stockée légèrement modifiée, ce qui est attendu, puisque 12 formes semblent clairement insuffisantes pour cette utilisation.

5.3 Squelette et espace de formes

On peut imaginer apprendre un espace de forme beaucoup moins varié que ci-dessus, par exemple des humains scannés dans diverses positions, comme le dataset DFAUST ([BRPMB17]). On s'attend alors à ce que l'inférence par Maximum a posteriori fonctionne bien, donnant la possibilité d'obtenir rapidement (car en optimisant seulement le code latent) la SDF d'un nouveau nuage de points du même type, puis d'en déduire le squelette, par exemple avec l'approche proposée en Section 4.

6 Implémentations et expériences

6.1 Outils utilisés

J'ai utilisé un certains nombre de bibliothèques, logiciels et datasets, parmi lesquels :

- PyTorch, bibliothèque permettant de faire du calcul tensoriel principalement destiné à effectuer de l'apprentissage profond. Il permet de tirer profit de la puissance de calcul des cartes graphiques via CUDA ;
- GUDHI (Geometry Understanding in Higher Dimensions) ([The22]), bibliothèque spécialisée en topologie algorithmique et en analyse topologique des données. Elle dispose d'un module pour le calcul de l'α-complexe ;
- PyGEL, binding Python de la bibliothèque GEL dont je me suis principalement servi pour la lecture et l'enregistrement de maillages ;
- pour l'analyse en composantes principales, j'ai utilisé l'implémentation de scikit-learn. Pour les k-d trees et les complexes de Delaunay, j'ai utilisé les implémentations de Scipy ;
- Blender, logiciel open source de modélisation, d'animation et de rendu 3D, pour faire les rendus de squelettes avec le maillage ou le nuage de points dont ils sont issus ;
- beaucoup d'expériences ont été réalisées sur des formes issues du dataset Thingi10K ([ZJ16]), qui comprend des formes très variées.

La plupart des expériences ont été faites sur un ordinateur portable équipé d'une carte graphique GTX 1060 avec 6 Go de mémoire. Certaines expériences, en particulier pour l'apprentissage d'un espace de forme, ont été effectuée sur le service Google Colab qui met à disposition des GPU avec 16 Go de mémoire (avec une limitation en temps de calcul).

6.2 Code écrit

Le code écrit durant le stage l'a été en Python. Il comprend, entre autres :
- concernant l'entraînement de réseaux :
  - l'apprentissage d'une SDF ;
  - l'apprentissage d'un espace de forme par l'approche auto-décodeur, et inférence ;
  - l'apprentissage d'une projection orthogonale sur la surface ;
- concernant l'extraction de squelette :
  - la projection sur la surface de niveau 0 et le resampling uniforme décrit dans [YWOSH21] ;
  - l'extraction de points squelettaux par line-search ;
  - l'extraction de points squelettaux par minimisation d'énergie ;
  - la sélection de points squelettaux par une méthode fondée sur un problème de recouvrement inspirée de [DLX+21] ;
  - le calcul de l'α-complexe avec GUDHI et sa conversion en mesh squelettal ;
- un rendu de surface implicite par la méthode du *sphere-tracing* ([Har96]) ;
- l'échantillonnage uniforme de points sur un mesh en utilisant la méthode d'échantillonnage de triangle de [OFCD02] ;
- la génération de nuages de points d'axe médian connu pour une étude quantitative de la résistance au bruit et aux données manquantes ; calcul des normales par analyse en composantes principales puis orientation.

Le code est disponible sur le dépôt `https://github.com/MClemot/Skeleton`.

6.3 Article

En mai, un article a été soumis à la conférence SIGGRAPH Asia 2022. Il aborde le terme de variation totale dans l'apprentissage de la SDF, la technique d'extraction de squelette décrite dans la Section 4, et la résistance de cette méthode au bruit et aux données manquantes, en comparaison des méthodes existantes.

Conclusion

Ce stage aura finalement été assez expérimental et a permis l'obtention de résultats plutôt encourageants. En particulier, l'introduction du terme de variation totale du gradient a permis d'améliorer la qualité des SDF estimées au niveau local. D'autre part, le lien qui a été fait avec la théorie des solutions de viscosité, fournissant la méthode théorique de la *vanishing viscosity*, a permis de concevoir une méthode d'optimisation des réseaux permettant d'envisager la possibilité de s'affranchir des points d'apprentissage grâce à la convergence de cette méthode vers la bonne solution. Si du travail reste à faire sur les paramètres et si elle est assez coûteuse en temps, cette méthode a fonctionné sur différents modèles. Enfin, ce stage a montré qu'il était possible d'utiliser ces estimations de SDF pour extraire une approximation de l'axe médian d'une forme. De plus, cette méthode d'extraction est plutôt résistante au bruit et aux données manquantes ce qui conforte l'utilisation des réseaux neuronaux pour cette application.

Du travail reste à faire sur un certain nombre d'aspects autour de ce stage. Premièrement, l'extraction du squelette à partir d'une estimation SDF reste assez basique et pourrait certainement être améliorée. Notamment, on pourrait imaginer une simplification du complexe simplicial obtenu ; on pourrait aussi imaginer l'ajout de contraintes sur la topologie du squelette. Deuxièmement, il serait sans doute intéressant d'explorer la possibilité de l'apprentissage d'une projection sur une surface, et de voir si cette approche donne de bons résultats en reconstruction de surface, par exemple. Troisièmement, il serait pertinent de tester l'approche auto-décodeur de DeepSDF pour apprendre un espace de forme dans le but d'extraire le squelette d'une nouvelle forme dans une famille de forme.

Références

[ABE09] Dominique Attali, Jean-Daniel Boissonnat, and Herbert Edelsbrunner. Stability and computation of medial axes-a state-of-the-art report. In *Mathematical foundations of scientific visualization, computer graphics, and massive data exploration*, pages 109–125. Springer, 2009.

[Bar13] Guy Barles. An introduction to the theory of viscosity solutions for first-order hamilton–jacobi equations and applications. In *Hamilton-Jacobi equations : approximations, numerical analysis and applications*, pages 49–109. Springer, 2013.

[BRPMB17] Federica Bogo, Javier Romero, Gerard Pons-Moll, and Michael J. Black. Dynamic FAUST : Registering human bodies in motion. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

[DF12] Charles Dapogny and Pascal Frey. Computation of the signed distance function to a discrete contour on adapted triangulation. *Calcolo*, 49(3) :193–219, 2012.

[DLX+21] Zhiyang Dou, Cheng Lin, Rui Xu, Lei Yang, Shiqing Xin, Taku Komura, and Wenping Wang. Coverage axis : Inner point selection for 3d shape skeletonization. *arXiv preprint arXiv :2110.00965*, 2021.

[Erd46] Paul Erdös. On the hausdorff dimension of some sets in euclidean space. *Bulletin of the American Mathematical Society*, 52(2) :107–109, 1946.

[GYH+20] Amos Gropp, Lior Yariv, Niv Haim, Matan Atzmon, and Yaron Lipman. Implicit geometric regularization for learning shapes. In *International Conference on Machine Learning*, pages 3789–3799. PMLR, 2020.

[Har96] John C Hart. Sphere tracing : A geometric method for the antialiased ray tracing of implicit surfaces. *The Visual Computer*, 12(10) :527–545, 1996.

[HWCO+13] Hui Huang, Shihao Wu, Daniel Cohen-Or, Minglun Gong, Hao Zhang, Guiqing Li, and Baoquan Chen. L1-medial skeleton of point cloud. *ACM Trans. Graph.*, 32(4) :65–1, 2013.

[Lie04] André Lieutier. Any open bounded subset of rn has the same homotopy type as its medial axis. *Computer-Aided Design*, 36(11) :1029–1046, 2004.

[LLL+21] Cheng Lin, Changjian Li, Yuan Liu, Nenglun Chen, Yi-King Choi, and Wenping Wang. Point2skeleton : Learning skeletal representations from point clouds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4275–4284. IEEE Computer Society, 2021.

[MON+19] Lars Mescheder, Michael Oechsle, Michael Niemeyer, Sebastian Nowozin, and Andreas Geiger. Occupancy networks : Learning 3d reconstruction in function space. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4460–4470, 2019.

[OFCD02] Robert Osada, Thomas Funkhouser, Bernard Chazelle, and David Dobkin. Shape distributions. *ACM Transactions on Graphics (TOG)*, 21(4) :807–832, 2002.

[PFS+19] Jeong Joon Park, Peter Florence, Julian Straub, Richard Newcombe, and Steven Lovegrove. Deepsdf : Learning continuous signed distance functions for shape representation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 165–174, 2019.

[SMB+20] Vincent Sitzmann, Julien Martel, Alexander Bergman, David Lindell, and Gordon Wetzstein. Implicit neural representations with periodic activation functions. *Advances in Neural Information Processing Systems*, 33 :7462–7473, 2020.

[TDS+16] Andrea Tagliasacchi, Thomas Delame, Michela Spagnuolo, Nina Amenta, and Alexandru Telea. 3d skeletons : A state-of-the-art report. In *Computer Graphics Forum*, volume 35, pages 573–597. Wiley Online Library, 2016.

[The22] The GUDHI Project. *GUDHI User and Reference Manual*. GUDHI Editorial Board, 3.5.0 edition, 2022.

[TZCO09] Andrea Tagliasacchi, Hao Zhang, and Daniel Cohen-Or. Curve skeleton extraction from incomplete point cloud. In *ACM SIGGRAPH 2009 papers*, pages 1–9. 2009.

[YWOSH21] Wang Yifan, Shihao Wu, Cengiz Oztireli, and Olga Sorkine-Hornung. Iso-points : Optimizing neural implicit surfaces with hybrid representations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 374–383, 2021.

[ZJ16] Qingnan Zhou and Alec Jacobson. Thingi10k : A dataset of 10,000 3d-printing models. *arXiv preprint arXiv :1605.04797*, 2016.

A Résultats

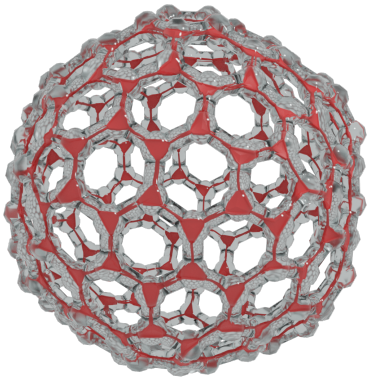
Résultats à partir de maillages La Figure 16 présente le squelette calculé à partir de plusieurs maillages. Plus précisément, un nuage de 100000 points est tiré uniformément à la surface du maillage, en suivant la méthode de [OFCD02] pour échantillonner uniformément un triangle, puis passé en entrée à la méthode décrite en Section 2 pour obtenir une estimation de SDF, permettant d'extraire un squelette avec la méthode décrite en Section 4.

Données manquantes La Figure 17 montre un résultat pour un nuage de point avec des données manquantes. Par exemple, les bras ne sont pas "fermés", mais la méthode parvient à construire un squelette unidimensionnel dans ces régions.

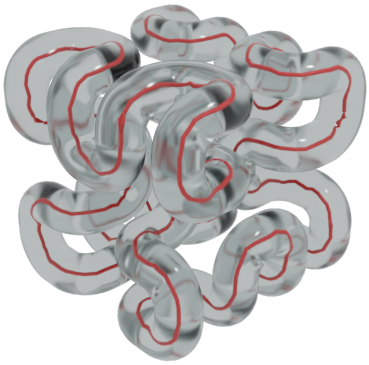
Résistance quantitative au bruit La Figure 20 présente le squelette calculé pour un nuage de point sur un tore à petit rayon variable, dont l'axe médian est connu (c'est un cercle de rayon le grand rayon du tore), auquel j'ai ajouté un bruit gaussien, pour plusieurs écart-type. Les normales sont calculées par une analyse en composante principale des plus proches voisins, et orientées grâce à la connaissance de l'axe médian original. La Table 1 donne une mesure quantitative de l'écart des points squelettaux calculés à cet axe médian. Les résultats sont donnés avec et sans le terme de régularisation de variation totale du gradient pour mettre en évidence son intérêt. Le résultat de L1 Medial Skeleton [HWCO$^+$13] est aussi donné pour comparaison.

Résistance quantitative aux données manquantes La Figure 21 et la Table 2 présentent aussi le squelette et l'écart des points squelettaux calculés au vrai axe médian d'un tore à petit rayon variable, mais dont on a ôté une certaine partie des données. Les résultats sont données avec et sans le terme de régularisation de variation totale du gradient, et en comparaison avec les résultats de L1 Medial Skeleton.

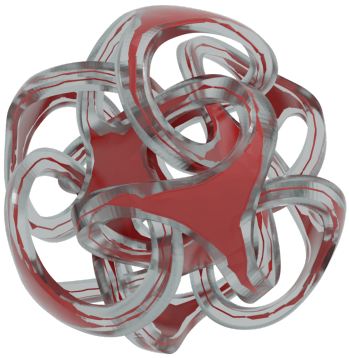
(a) Cage à oiseaux (Thingi10K)



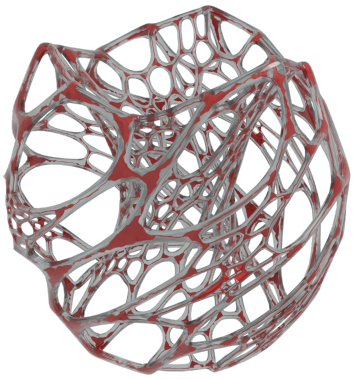
(b) Molécule sphérique "buckminsterfullerène" (Thingi10K)



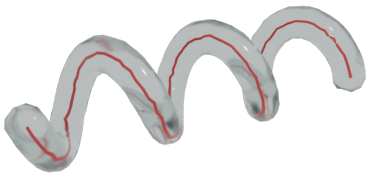
(c) Cube de Hilbert (Thingi10K)



(d) Metatron (Thingi10K)

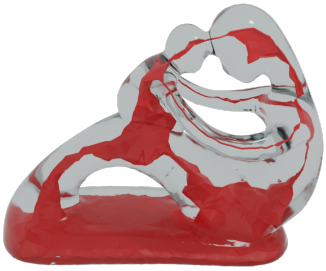


(e) Vase organique (Thingi10K)

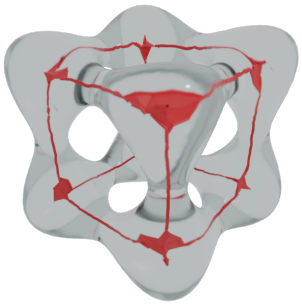


(f) Hélice

(g) Main

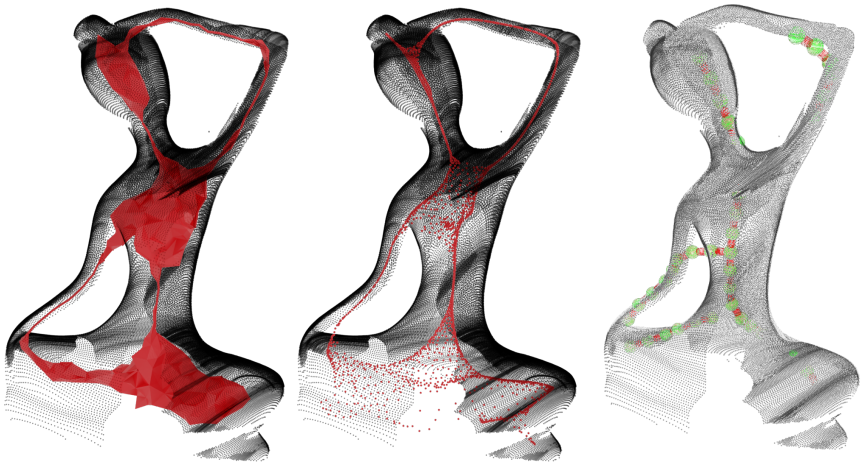


(h) Fertilité



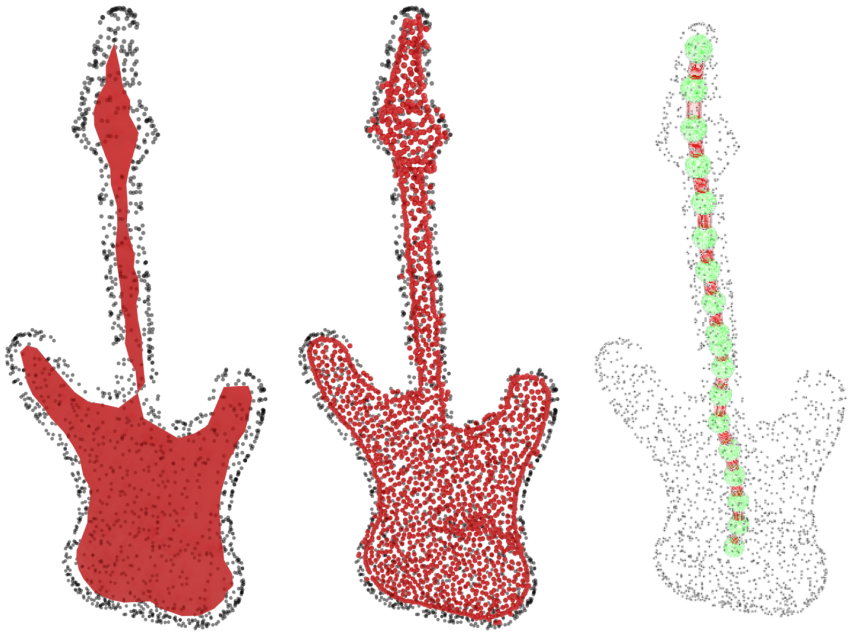
(i) Cube troué

FIGURE 16 – Résultats obtenus pour différentes formes. Le mesh de départ en représenté par du verre et le squelette calculé est en rouge.



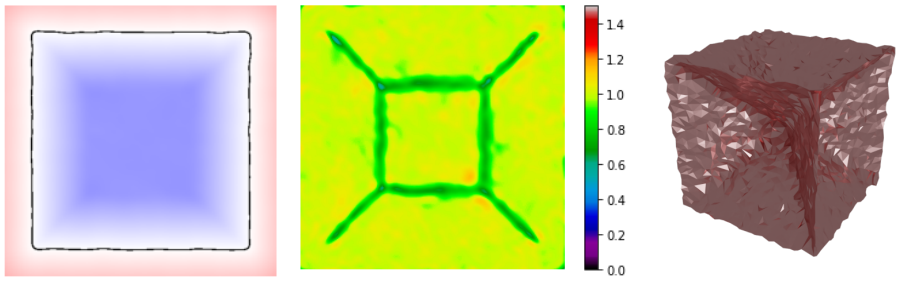
(a) Neural Skeleton (b) Deep Point Consolidation (c) L1 Medial Skeleton

FIGURE 17 – Extraction de squelette pour un nuage de points avec des données manquantes, comparée avec Deep Point Consolidation et L1 Medial Skeleton

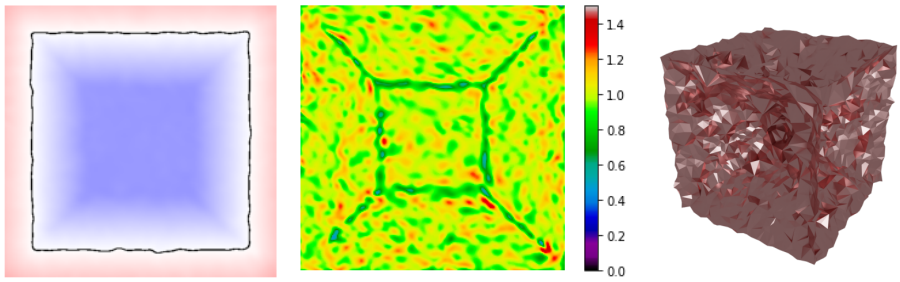


(a) Neural Skeleton (b) Point2Skeleton (c) L1 Medial Axis

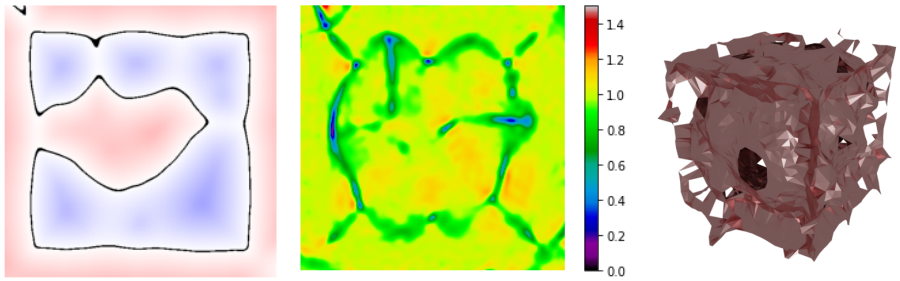
FIGURE 18 – Extraction de squelette d'un nuage de point représentant une guitare, comparé avec Point2Skeleton et L1 Medial Axis



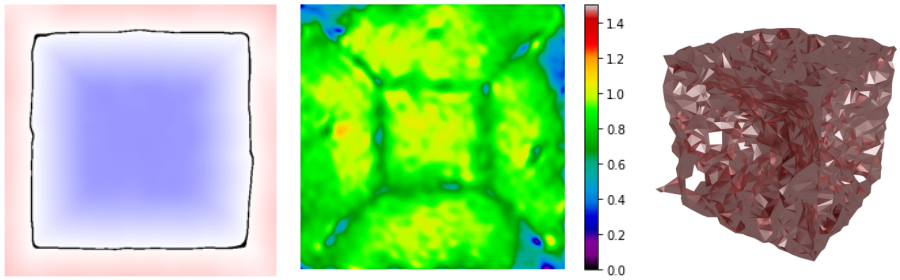
(a) Avec tous les termes



(b) Sans le terme de régularisation de variation totale du gradient



(c) Sans les points d'apprentissage



(d) Sans la loss pour l'équation eikonale

FIGURE 19 – Résultats obtenus en supprimant différentes termes de la loss de la Section 2.2 (équation 6), pour un cube. Colonne de gauche : coupe de la SDF ; colonne du milieu : coupe de la norme du gradient ; colonne de droite : estimation du squelette associée. Pour cette étude d'ablation, l'initialisation ne s'est pas faite avec un réseau pré-entraîné sur une sphère (comme habituellement), mais avec un réseau à poids aléatoires, pour ne pas donner d'avantage à l'expérience sans points d'apprentissage (puisque le cube est assez proche de la sphère).

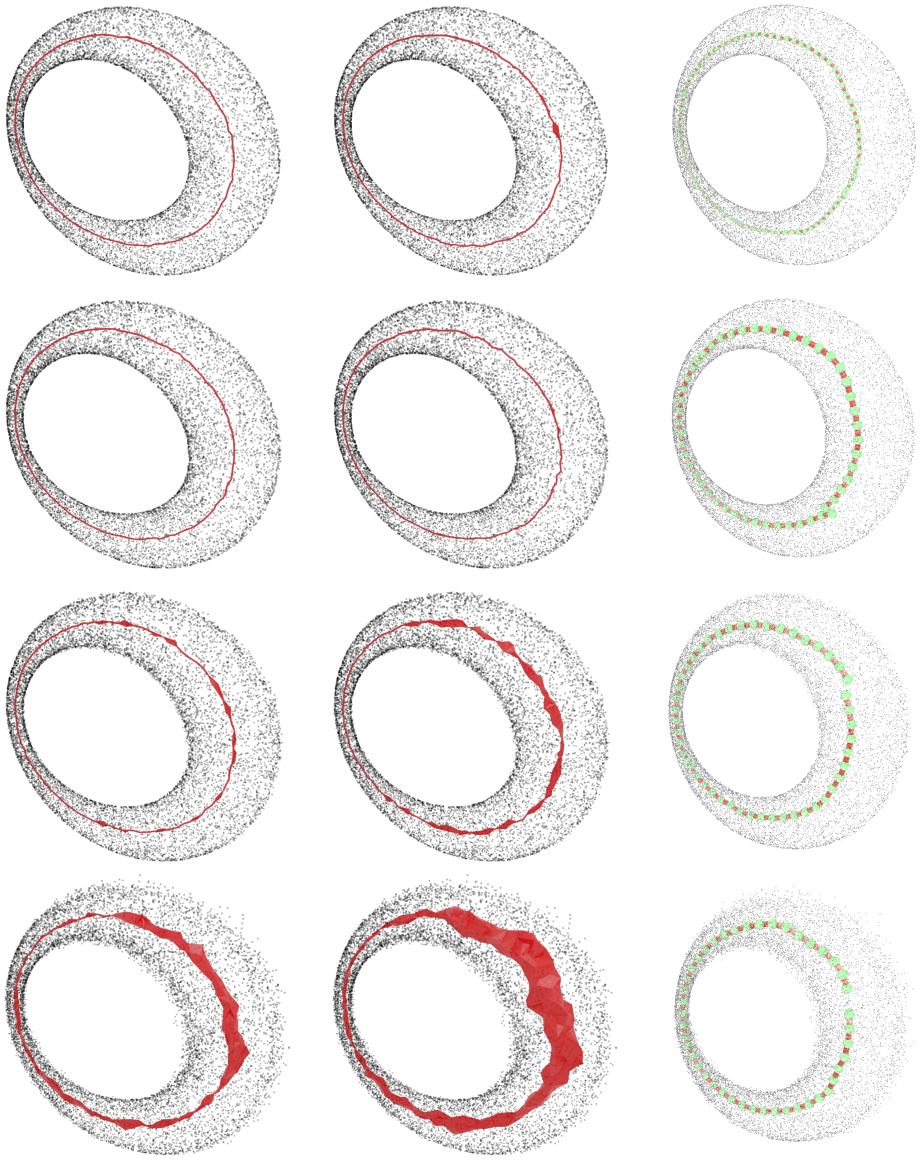


FIGURE 20 – Résistance au bruit de notre méthode. On ajoute un bruit gaussien aux points en entrée puis on recalcule les normales par analyse en composantes principales. Le bruit est centré, avec un écart-type exprimé en pourcentage de la diagonale de la forme. De haut en bas : sans bruit, $\sigma = 0.003$, $\sigma = 0.01$, $\sigma = 0.03$. Voir la Table 1 pour les mesures quantitatives.

Bruit	Avec $\mathcal{L}_{\text{TV}}$		Sans $\mathcal{L}_{\text{TV}}$	
	Erreur moyenne	Erreur max	Erreur moyenne	Erreur max
0	0.0031	0.0139	0.0038	0.0186
0.003	0.0032	0.0154	0.0041	0.0214
0.01	0.0051	0.0268	0.0076	0.0555
0.03	0.0151	0.0902	0.0218	0.1475

TABLE 1 – Résistance au bruit de notre méthode. Mesures quantitatives de la qualité du squelette pour les expériences de la Figure 20.

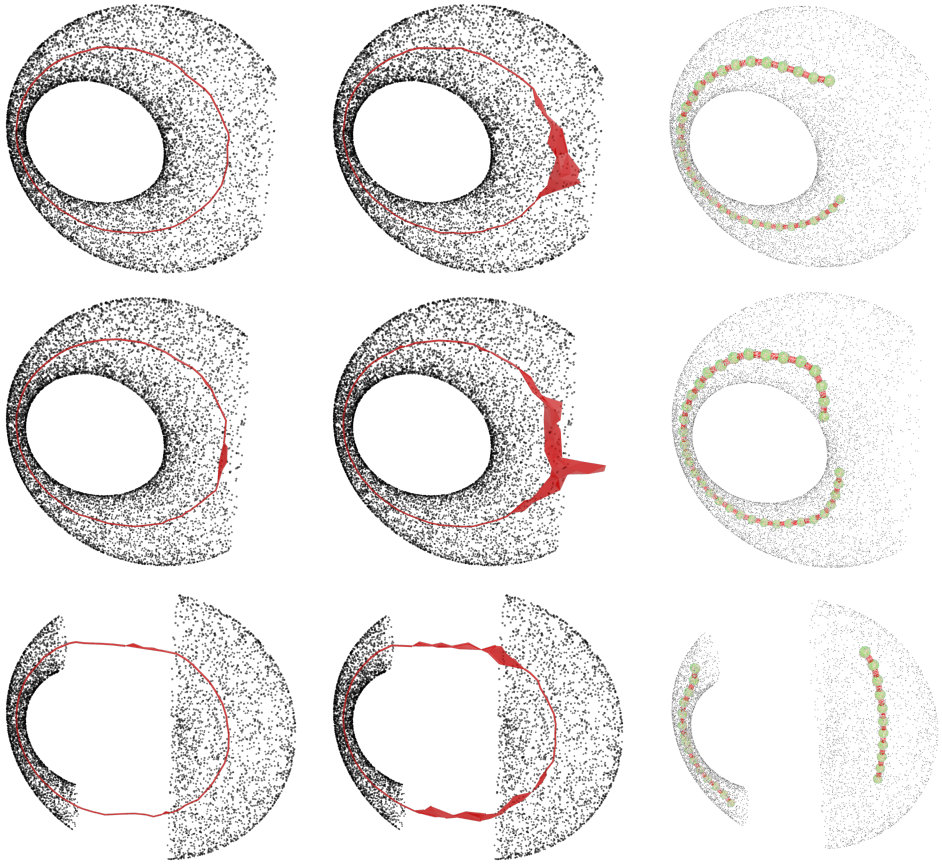


FIGURE 21 – Résistance aux données manquantes de notre méthode. Un tore à petit rayon variable est échantillonné et diverses zones lui sont ôtées. La régularisation avec $\mathcal{L}_{\text{TV}}$ permet de mieux récupérer le squelette. Voir Table 2 pour les mesures quantitatives.

Exp	Avec $\mathcal{L}_{\text{TV}}$		Sans $\mathcal{L}_{\text{TV}}$	
	Erreur moyenne	Erreur max	Erreur moyenne	Erreur max
1	0.0033	0.0137	0.0244	0.1159
2	0.0053	0.0340	0.0387	0.2906
3	0.0116	0.0607	0.0220	0.0795

TABLE 2 – Résistance aux données manquantes de notre méthode. Mesures quantitatives de la qualité du squelette pour les expériences de la Figure 21.