Tatouage des bases de données géographiques

Ammar MECHOUCHE

29 août 2005

Résumé

Le tatouage (watermarking) permet l'insertion robuste et discrète d'information dans un document, comme par exemple l'identité de son propriétaire. Les buts recherchés sont multiples, mais le plus important est la protection du droit d'auteur. On dissimule dans le document des informations relatives à son propriétaire pour que personne ne puisse se l'approprier. Le tatouage est également utilisé pour identifier les utilisateurs malhonnêtes et éviter la piraterie. En effet un propriétaire d'un document qui veut commercialiser son oeuvre (par exemple un logiciel) peut marquer différemment tous les exemplaires autorisés (les différentes copies qu'il a vendues). S'il rencontre une copie illégale, le marquage lui permet de remonter à la personne responsable de la fraude, cette forme de tatouage s'appelle l'empreinte numérique (Fingerprinting). De nombreuses techniques de tatouage existent pour les documents multimédia comme l'image, le son ou la vidéo. Actuellement se développent des techniques permettant de tatouer des données structurées. L'équipe Vertigo a développé un modèle de tatouage pour les bases de données avec contraintes, où l'insertion d'information doit préserver la qualité d'un certain nombre de requêtes déclarées au préalable. Le résultat déjà obtenu est une méthode de tatouage générale préservant n'importe quelle requête SQL (bases de données relationnelles) ou XPath (données XML), et est implantée dans le prototype Watermill[1]. L'objectif de cette étude est d'étendre ces résultats aux bases de données géographiques comportant, en plus des données textuelles habituelles, de l'information géométrique. L'application cible est le tatouage des données utilisées par l'Institut Géographique National IGN[2]. Ce travail s'inscrit dans le cadre de l'ACI Sécurité et Informatique 2007 Tadorne[3].

Mots clés : Tatouage, fingerprinting, bases de données géographiques, contraintes, tadorne

Table des matières

1 Introduction

Aujourd'hui, l'avènement des technologies numériques fait apparaître un souci majeur qui réside dans la facilité avec laquelle les documents numériques peuvent être copiés de façon illicite sans subir de détériorations. Ne pouvant pas faire la différence entre le document d'origine et les copies, il est très difficile, voire impossible, de protéger le droit de propriété intellectuelle. Pour pallier ce problème, des techniques de tatouage des oeuvres numériques peuvent être envisagées. Il s'agit alors de tatouer les oeuvres numériques comme faisaient nos ancêtres sur les corps humains. Le tatouage en informatique est donc l'insertion d'une information dans la donnée à tatouer. Cette information doit être introduite de manière discrète de telle sorte que seul le propriétaire de la donnée sache la détecter, mais aussi doit préserver la sémantique de la donnée en elle même. Le tatouage doit respecter également, lors de son insertion, des contraintes sur les données, comme, la précision par exemple ; il doit également être résistant aux attaques malveillantes qui visent sa destruction : le propriétaire doit pouvoir détecter le tatouage même si ce dernier est attaqué. En conséquence, les algorithmes utilisés dans le tatouage doivent être conçus de manière intelligente et résistants aux différentes attaques.

Dans le cadre du projet Tadorne on s'intéresse particulièrement au tatouage des bases de données contraintes XML, relationnelles et géographiques. Dans ce stage, on s'intéresse au tatouage des bases de données géographiques, et l'application cible étant le tatouage des données utilisées par l'IGN.

La figure 1 illustre le processus de tatouage et de détection des bases de données. Les bases de données géographiques sont les outils opérationnels qui

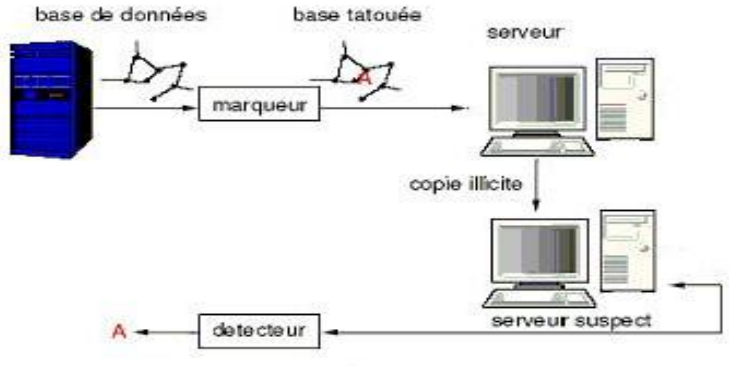

FIG. 1 – Schéma global du tatouage des bases de données

permettent d'organiser et de gérer l'information géographique sous forme numérique. Ce sont des ensembles structurés de fichiers décrivant les objets ou phénomènes

localisés sur la Terre (avec leurs attributs et leurs relations nécessaires à la modélisation de l'espace géographique). Ces ensembles sont munis d'un système de gestion permettant de les tenir à jour, de les archiver et de les diffuser. Les informations géographiques se présentent sous forme de points, de lignes ou de polygones auxquels sont rattachées des données descriptives. Cette association permet d'effectuer des sélections ou des classifications pour personnaliser la représentation et favoriser les analyses spatiales.

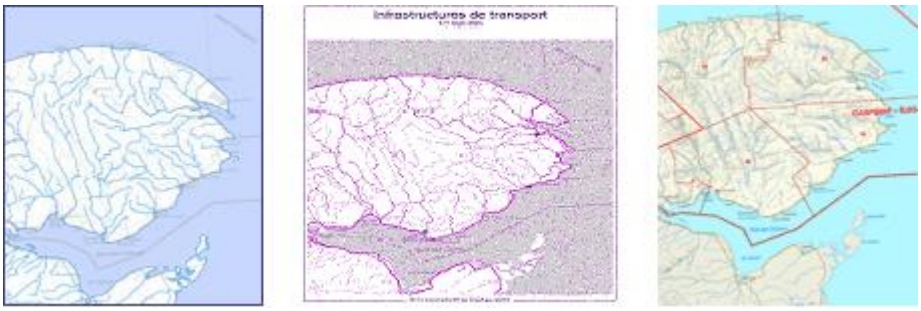

FIG. 2 – Données géographiques[04]

Nous allons justement nous servir des données vectorielles présentes dans les bases de données géographiques pour proposer une solution de tatouage de ces dernières.

Quatre laboratoires de recherche travaillent sur le projet Tadorne :
- Le laboratoire CEDRIC du CNAM de Paris ;
- Le laboratoire COGIT de l'IGN de Paris ;
- Le laboratoire GREYC de l'Université de Caen ;
- Le laboratoire LAMSADE de l'Université de Paris Dauphine.

Objectifs et activités

1. Analyser le problème réel du tatouage des données sous contraintes, basé sur les données géographiques et les services de l'IGN ;
2. Produire des algorithmes efficaces pour le tatouage des données relationnelles et les services web ;
3. Valider les résultats sur une plateforme libre, pour permettre la reproductibilité de ces résultats et la comparaison avec d'autres techniques.

2 Etat de l'art

2.1 Tatouage des bases de données

2.1.1 Watermaking Relational Data Bases(R.Agrawal et J.Kiernan)[05]

Agrawal et Kiernann se sont inspirés des techniques de tatouage développées pour les images, vidéo et audio. Les caractéristiques des données relationnelles diffèrent de celles du multimédia. Un objet multimédia est une grande suite de bits avec des redondances considérables, ce qui facilite la procédure de tatouage. Par contre, les données relationnelles sont des ensembles de n-uplets considérés chacun comme un objet. Le tatouage doit être réalisé sur chacun de ces objets.

Algorithme de tatouage

1. Déterminer les n-uplets à tatouer en utilisant une fonction de hachage basée sur la clé secrète : seul le propriétaire de la relation peut ensuite savoir quels sont les n-uplets tatoués.
2. De même pour les attributs à tatouer et le bit de position à changer.
3. Perturber (tatouer) les bits des attributs et n-uplets déterminés à l'étape précédente.

Algorithme de détection L'algorithme de détection aussi utilise la clé secrète pour déterminer les n-uplets, et les attributs tatoués en utilisant la fonction de hachage.

1. Comparer la valeur du bit retrouvée à celle qui devait y être.
2. Calculer le rapport entre le nombre de bits correspondant aux bits insérés et le nombre de bits de la marque.
3. Si ce rapport dépasse un seuil calculé au préalable(80% par exemple), on considère la donnée comme tatouée.

Pour que le tatouage soit fiable et robuste, il faut bien choisir les paramètres de la fonction de hachage.

Robustesse aux attaques Les expérimentations effectuées ont montré que généralement, si on fixe de bons paramètres pour l'algorithmes de tatouage, on aura une faible probabilité de succès d'une attaque, notamment contre les attaques insérant des bits au hasard(BitFlipping Attack), les attaques consistant à remplacer un ensemble de n-uplets de la relation tatouée par d'autres n-uplets d'une autre relation similaire (Mix and Match Attack), les attaques consistant à tatouer une relation déjà tatouée.

Apport Une adaptation des techniques utilisées pour le tatouage d'images aux bases de données relationnelles. C'est le premier algorithme de tatouage des bases de données relationnelles.

Inconvénients Pour pouvoir appliquer cet algorithme, il faudra trouver des attributs numériques dans la base, supportant des perturbations minimes sans que la base ne soit altérée.

Extension proposée par les auteurs Etendre la technique proposée pour les attributs non numériques. Aussi introduire une empreinte numérique pour pouvoir identifier le coupable dans le cas où il peut y avoir des sources multiples de copies illicites.

2.1.2 Watermarking numerics sets(R.Sion, M. Atallah et S. Prabhakar)[06]

Dans cet article, les auteurs se basent sur une étude effectuée sur le tatouage des ensembles numériques pour présenter une méthode de tatouage des bases de données, en contrôlant les propriétés qui doivent être préservées. Ils définissent une limite de distorsion autorisée sur les données en terme de mesure d'utilisabilité. L'idée est de pouvoir modifier des valeurs d'attributs sans dépasser une certaine limite afin de préserver les propriétés initiales et la structure d'une base de données. Par exemple, pour un ensemble S=s_1,...,s_n de données à marquer, et V=v_1,...,v_n, l'ensemble résultant du tatouage, on voudra vérifier que :

$$(s_i - v_i)^2 < t_i \ i = (1, 2, 3, .., n)$$

L'insertion de la marque se fait en plusieurs étapes : à chaque fois qu'on introduit une distorsion sur un n-uplet, on vérifie si toutes les conditions sont vérifiées, sinon on essaie sur un autre n-uplet. Cette méthode est très générale car elle permet de tatouer en respectant un ensemble quelconque de contraintes. Cependant, le temps de calcul est très important, car pour chaque bit de marque inséré, il faut vérifier toutes les contraintes.

2.1.3 A High Capacity Watermarking System for Digital Maps(G. Schulz, M. Voigt)[07]

Dans cet article, les auteurs proposent une méthode d'insertion de tatouage dans des données vectorielles 2D. Le système consiste à rajouter de l'information dans les cartes numériques. L'information est rajoutée en changeant les coordonnées (x,y) des points dans la limite de tolérance de la donnée. Un code correcteur est utilisé pour reconstruire les données erronées, et la synchronisation sert à détecter l'information lorsque la carte est altérée. L'idée originale de l'algorithme proposé est l'insertion des méta-données dans la donnée elle même, ce qui permet leur indépendance des méta données définies par les formats des fichiers utilisés, qui sont souvent perdues en convertissant le fichier d'origine en d'autres types de fichiers.

Fonctionnement du système (tatouage) Deux bits peuvent être insérés par point en décalant le point verticalement et horizontalement. Si le bit inséré est zéro, on décale le point vers la ligne marquée par zéro. Si le bit inséré

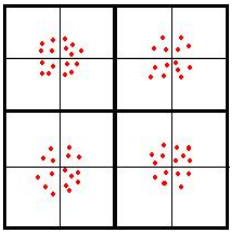

FIG. 3 – La nouvelle distribution des points

est un, alors on décale le point vers la ligne marquée par un. Pour pouvoir rajouter du bruit aléatoire sans pour autant perdre l'information insérée, on s'intéresse seulement à la partie dans laquelle le point a été décalé ; c'est à dire : on n'est pas obligé de décaler le point vers les lignes marquées par un ou zéro avec exactitude. Maintenant une longue suite de bits doit être insérée dans la carte. Avant l'insertion, cette séquence est divisée en deux parties égales : une partie est insérée horizontalement et l'autre verticalement. La carte est divisée en plusieurs bandes (strips) horizontales et verticales. Les bits d'information sont affectés continuellement pour ces bandes jusqu'à atteindre la dernière bande, et tous les points de ces bandes sont décalés soit verticalement ou horizontalement selon le bit inséré. On peut schématiser la distribution des points de la carte dans un seul patch (figure 3). Malgré la taille petite du patch, les points peuvent être distribués avec inégalité aux différents bits. La meilleure façon pour le tatouage d'être robuste et résistant aux attaques est d'affecter les points avec égalité à tous les bits. Pour ce faire, un point peut être décalé dans un patch voisin sous certaines conditions, sans bien sûr dépasser la tolérance maximum.

Performances et robustesse aux attaques du système Cet algorithme est robuste contre les attaques suivantes :

- Simplifications des polylignes[1] (Polyline simplifications) tel l'algorithme de Douglas-Peuker détaillé plus loin ;
- L'ajout et suppression de points ;
- L'ajout d'un bruit aléatoire.

Par contre l'algorithme n'est pas résistant aux attaques telles que la rotation de la carte. Les simulations effectuées ont montré qu'avec plus de six points affectés à chaque bit, l'erreur de reconnaissance du tatouage est très minime : probabilité inférieure à 10^{-5}.

Apport

- Nature de l'information insérée (méta-données)
- Résistance aux attaques de simplification polylignes telle que l'attaque de Douglas Peucker

[1] Un polygone non fermé.

Inconvénients

- Absence d'une clé secrète pour l'insertion du tatouage ;
- Le fait de bouger tous les points de la carte pourrait la rendre inutilisable ;
- Les points après le tatouage sont concentrés dans les mêmes régions, et une attaque qui remplace les points concentrés dans une région par leur barycentre pourrait être envisageable.

Extension proposée par les auteurs Construire le même algorithme basé sur une clé secrète

2.1.4 Watermarking 2D Vector Maps in the Mesh-Spectral Domain(R. Ohbuchi, H. Ueda, S. Endoh)[08]

Des algorithmes existent pour le tatouage des mailles 3D. Ces algorithmes altèrent les coordonnées ou la connectivité des sommets pour le tatouage. Dans cet article les auteurs proposent un algorithme de tatouage de cartes numériques vectorielles. L'algorithme insère le tatouage dans la représentation du domaine fréquentiel des cartes.

Algorithme de tatouage L'algorithme consiste à insérer un message binaire dans la carte en modifiant les coefficients du domaine fréquentiel de la carte comme décrit ci-dessous.

1. On extrait les points des polygones ;
2. Pour le calcul de la représentation du domaine fréquentiel, l'algorithme établit une connectivité entre les sommets de la carte en utilisant la triangulation de Delaunay ;
3. Pour des raisons d' efficacité et de robustesse, la carte est divisée en plusieurs surfaces rectangulaires équilibrées du point de vue nombre de sommets contenus dans chaque zone, la méthode utilisée est, la k-d-tree subdivision[09].
4. La maille est ensuite transformée en domaine fréquentiel en utilisant l'analyse spectrale de maille proposée par Karni et al[10].
5. Un message est inséré dans chacune de ces surfaces, en modulant (modifiant) fréquences ainsi obtenues dans chaque surface.

Les tatouages sont extraits en comparant la carte (éventuellement attaquée) à la carte originale. C'est donc un algorithme non aveugle (informé).

Algorithme de détection

1. D'abord les deux cartes (originale et attaquée), sont alignées en utilisant un processus d'optimisation itératif pour minimiser la distance entre un ensemble de points repères. De cette manière, on supprime les transformations affines (Affine transformation) appliquées à la carte tatouée ;

2. Ensuite, la même subdivision identique à celle utilisée lors de l'insertion est recréée sur la carte originale, et cette subdivision est transférée sur la carte tatouée ;
3. Pour chaque sous-surface correspondante sur la carte tatouée, on applique une analyse spectrale ;
4. La comparaison des coefficients spectraux extrait le tatouage inséré.

Robustesse aux attaques L'algorithme décrit est résistant contre :
- L'ajout d'un bruit aléatoire
- Les transformations globales (Global affine transformation), rotation et translation.
- L'insertions et suppression des sommets
- L'altération de l' ordre des objets dans les fichiers
- Les déformations locales

Apport de l'algorithme Les résultats des expérimentations effectuées ont montré une nette amélioration par rapport au précédent article proposé par Ohbuchi[11], notamment la résistance aux déformations locales de la carte et l'ajout aléatoire de bruit, et aussi le cropping (enlever une partie de la carte). L'analyse spectrale est d'une grande utilité, car elle permet d'obtenir des coefficients qu'on peut perturber pour l'insertion du tatouage, mais aussi ces coefficients sont obtenus en utilisant une clé privée, ce qui renforce la fiabilité et la sécurité de l'algorithme. Aussi, on trouve dans l'algorithme le processus de normalisation qui consiste à éliminer les transformations globales, pour cela l'algorithme compare des chaînes de caractères supposées être dans la carte.

Inconvénients Néanmoins l'algorithme proposé présente un inconvénient relatif au temps de calcul lors de la triangulation qui peut être intolérables dans quelques applications. Un autre inconvénient est le fait que l'algorithme soit non aveugle et nécessite la carte originale pour détecter le tatouage. Le repositionnement (processus de normalisation) peut aussi être considéré comme un inconvénient, du moment qu'il suppose l'existence des chaînes de caractères dans les cartes.

Extensions proposées par les auteurs Les extensions envisagées pour cet algorithme sont :
- L'amélioration de la robustesses aux attaques, en calculant une meilleure représentation du domaine fréquentiel ;
- Enumérer l'ensemble des attaques possibles sur les cartes, ainsi que la perception humaine de la déformation des cartes.

3 Tatouage de bases de données sous contraintes

3.1 Problématique des données IGN

3.1.1 Présentation

Dans les bases de données géographiques, on utilise trois types de modèles[12] :

1. Le modèle topologique : dans ce modèle, on prend en compte les relations topologiques d'un objet avec son voisinage ;
2. Le modèle polygonal : c'est le modèle vectoriel (Shape 2D) où les vecteurs forment des polygones représentant les objets, sans tenir compte de la topologie ;
3. Et le modèle Bitmap : c'est le modèle raster, c'est complètement des données sous formes d'images.

Le modèle qui nous intéresse dans notre étude est le modèle polygonal, qui est constitué d'un ensemble de polygones représentant la forme des objets, et un ensemble d'annotations représentant les caractéristiques de ces objets.

Un prolème majeur se pose lorsqu'on veut tatouer les bases de données géographiques. En effet, le tatouage consiste à modifier les coordonnées numériques des points constituant les polygones de la base, cependant ces modifications ont pour effet de changer la forme et la précision des objets. L'algorithme de tatouage doit donc respecter ces contraintes ou un sous ensemble de ces contraintes selon les exigences du propriétaire de la base de données géographique (l'IGN dans notre cas).

3.2 Contraintes d'utilisabilité

Lors de nos différentes rencontres avec les chercheurs du laboratoire COGIT de l'IGN, nous avons explicité les contraintes d'utilisabilité les plus importantes à respecter lors du tatouage, et qui sont les suivantes :

1. Contraintes de précision :
   - Précision de l'ordre de 1/10mm pour une carte imprimée, soit une précision standard de l'ordre de 2,5m pour une carte 1 :25000. Pour les BD Topo, le niveau de détail est de l'ordre du mètre. Et pour la BD Carto, le niveau de détail est de l'ordre de 10 mètres ;
   - Pour raison de perception par l'utilisateur de la carte, lors de la transformation de la BD topo vers la BD carto, des anamorphoses[2] sont réalisées.
2. Contraintes métriques :
   - Préserver les distances sur le réseau routier ;
   - Virages : garder le même rayon de courbure (important pour les voies ferrées) ;

[2]Zoom local sur un objet pour rendre sa forme perceptible malgré l'échelle

- Cadastre : Les positions ne sont pas importantes, mais la surface doit être légale ;
- Certains angles sont très importants : par exemple les angles droits.

3. Contraintes de simplification :
   - Passage de la BD Topo à la BD Carto :
     - Transformation *polygone* → *point* : le point est bien le centre de l'objet réel ;
     - *Route* → *polyligne* : l'axe de la ligne est l'axe de la route.
   - Les angles presques droits sont transformés en angles droits.
4. Contraintes de saisie : condition de saisie par l'IGN d'un objet : par exemple, une cabane est ignorée si sa taille est trop petite ;
5. Contraintes de forme

- Contrainte majeure : Invariance des relations topologiques : ne pas rompre une adhérence ou une inclusion ;
- Partage d'arêtes plutôt que partage de points ;
- Préserver le positionnement relatif : Etre à gauche de, être parallèle à, etc ;
- L'objet important est le graphe : Il peut être simplifié (exemple du croisement de routes - rond point à 5 sorties, on peut n'en garder que 3 mais dans le bon ordre) ;
- Altération du graphe routier : préserver la circulation dans le graphe.

3.3 Modèles

Modèle d'échange Le modèle d'échange à l'IGN est l'achat simple qui s'oriente vers des mises à jour(lots d'évolution). En effet, les fonds cartographiques de l'IGN sont des oeuvres originales protégées par la législation sur la propriété intellectuelle. Voir www.ign.fr.

Modèle de tatouage Les producteurs et les utilisateurs des bases de données géographiques pratiquent souvent des traitements (filtrage et lissage par exemple) sur les données vectorielles dans le but de corriger des erreurs induites lors de la saisie. Ces traitements sont classiques par exemple à l'IGN. En conséquence, le tatouage doit non seulement résister aux attaques malveillantes, mais aussi aux traitements naturels pratiqués sur les données.

3.4 Traitements et attaques classiques sur les bases de données géographiques

3.4.1 Lissage et filtrage

Lissage Le lissage est une opération qui a pour but l'amélioration de l'affichage, en nuançant les points adjacents, donnant l'impression que la forme se fond dans son contexte. Ce traitement est souvent effectué par les clients de l'IGN (même par l'IGN lui-même). Par exemple diminuer le rayon de courbure ou diminuer la distance d'un point avec les points voisins.

3.4.2 Algorithme de lissage de Gauss[13]

Dans l'algorithme de lissage gaussien des lignes, chaque point est une moyenne de sa position originale et les points qui sont dans son voisinage. Les poids des points du voisinage du point en question sont modélisés avec une gaussienne de la distance de ce point. le nombre de points du voisinage pris en compte dépend du paramètre de la fonction gaussienne σ. Ainsi, avec le lissage, les petits détails des lignes sont supprimés.

1 Les segments de la polyligne sont décomposés en un nombre entier de micro-segments en rajoutant des somments à la polyligne avec un certain pas donné en comme paramètre à la fonction gaussienne.

2 Ré-évaluation de la valeur de σ si le nombre de points est insuffisant.

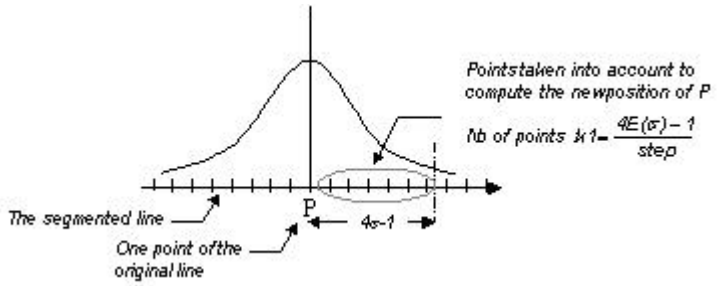

FIG. 4 – Ré-évaluation de σ pour prendre le nombre suffisant de points

3 Si k1 est plus grand que le nombre de points dans la ligne décomposée, alors recalculer la valeur de σ de telle sorte que k1 redevienne égal au nombre de points de la ligne décomposée.

4 Extension des deux extrêmités de la ligne avec symétrie en rajoutant k1 points dans chaque extrêmité, dans le but de pouvoir calculer les nouvelles positions des (k1 - 1) derniers points de chaque extrêmité.

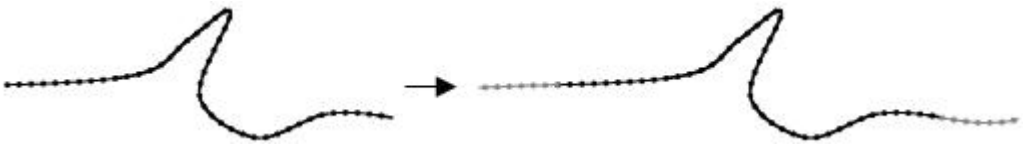

FIG. 5 – Extension de la ligne décomposée

5 Calcul des poids qui vont être utilisés pour le calcul des nouvelles positions : Les poids des points pris en compte sont une fonction gaussienne de leur distance curviligne s du point qui va être lissé.

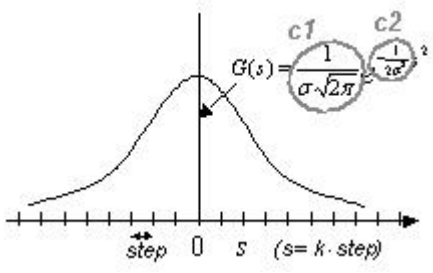

FIG. 6 – Cacul des poids des points voisins au point à lisser

6 Calcul de la nouvelle ligne : Pour chaque point de la ligne, ses coordonnées lissées sont :

$$X_{lissee}(i) = \sum_{k=-k1}^{k=k1} Weight(k).X(i+k) = c1 + \sum_{k=-k1}^{k=k1} c1.e^{c2.(k.step)^2}.[X(i-k)+X(i+k)]$$

$$Y_{lissee}(i) = \sum_{k=-k1}^{k=k1} Weight(k).Y(i+k) = c1 + \sum_{k=-k1}^{k=k1} c1.e^{c2.(k.step)^2}.[Y(i-k)+Y(i+k)]$$

Avec $Weight(k) = c1.e^{c2.(k.step)^2}$

7 Extraire de la ligne décomposée lissée les points correspondant aux points originaux de la ligne originale non décomposée.

Exemple Un exemple de lissage gaussien est le suivant :

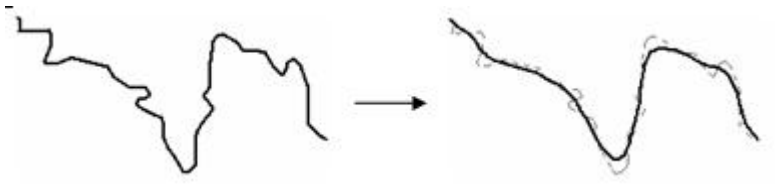

FIG. 7 – Exemple de lissage gaussien

Filtrage Le filtrage est une opération de simplification, qui consiste à supprimer des sommets de polygones dans les zones de forte densité de points.

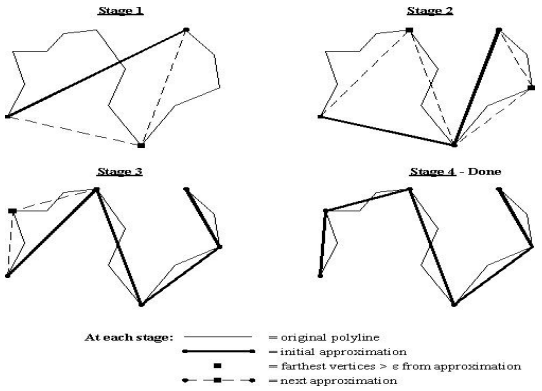

FIG. 8 – Simplification de Douglas Peucker

3.4.3 Algorithme de Douglas Peucker pour la simplification[14]

Algorithme de Douglas-Peucker Pour les simplifications polylignes C'est un algorithme récursif, utilisé pour la simplification des polylignes. Son fonctionnement est comme suit :

1. Considérer les deux extrêmités de la polyligne ainsi que le segment les reliant ;
2. S'il y a des sommets d'une distance supérieure à une distance minimale ϵ au segment, alors rajouter le sommet le plus loin du segment à l'ensemble des sommets du nouveau polygone simplifié ;
3. Itérer l'algorithme sur les nouveaux segments obtenus jusqu'à ce qu'aucun sommet ne vérifie la condition 2.

La figure 8 illustre le fonctionnement de cet algorithme :

4 Solutions proposées

Dans ce qui suit, nous proposons une solution de tatouage des bases de données géorgaphiques avec contraintes. Dans cette étude, notre objectif était de respecter deux types de contraintes, à savoir la précision, et la préservation de surface des objets.

4.1 Principe

Les données polygonales sont idéales pour le tatouage, car il s'agit des données numériques où l'on peut toujours insérer des bits de tatouage (des bits d'information). Notre solution consiste à parcourir séquentiellement tous les polygones de la base de données géographiques, et pour chaque polygone, parcourir séquentiellement les points le constituant et tester à chaque fois avec une fonction aléatoire paramétrée à base d'une clé secrète (Les paramètres étant les bits

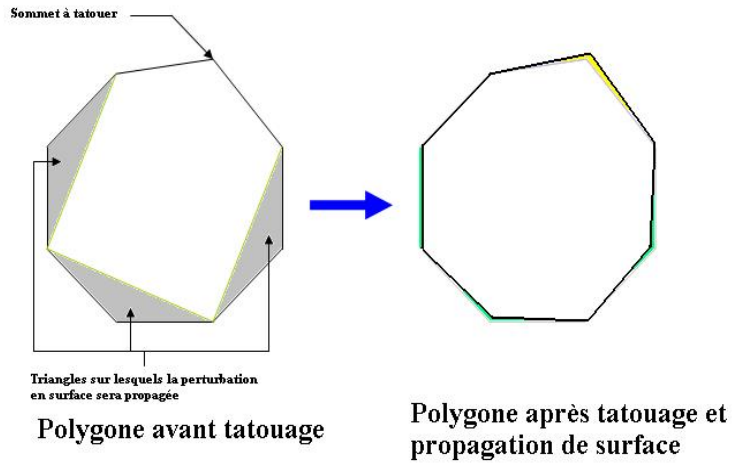

FIG. 9 – Tatouage des polygones avec préservation de la surface

de poids forts des coordonnées du point et une clé) si le point va être tatoué ou non. Si oui, insérer un bit d'information déterminé par la précédente fonction dans la partie des bits de poids le plus faible des coordonnées du point. Ensuite récupérer la pertubation en surface due à la pertubation des coordonnées du point sur le reste du polygone. Si on arrive à récupérer cette surface, on sauvegarde le fait que ce polygone a été tatoué pour la détection après, et on passe au points suivants, sinon on annule toutes les opérations faites sur ce polygone et on passe au polygone suivant.

La figure 9 montre clairement le fonctionnement de notre algorithme de tatouage :

4.1.1 Formalisation des contraintes

Soient : $\mathbb{P}$: espace des points de la base de données. ϕ la fonction de tatouage. $\mathbb{PL}$: espace des polygones de la base de données.

– Précision : soit la fonction de précision définie comme suit :

$$\eta : \mathbb{P} \rightarrow \mathbb{R}$$

tel que :

$$\forall P_i \in \mathbb{P}, \eta(P_i) < \epsilon, et, \eta(\phi(P_i)) < \epsilon$$

– Préservation de surface : soit la fonction de surface définie comme suit :

$$\delta : \mathbb{PL} \rightarrow \mathbb{R}$$

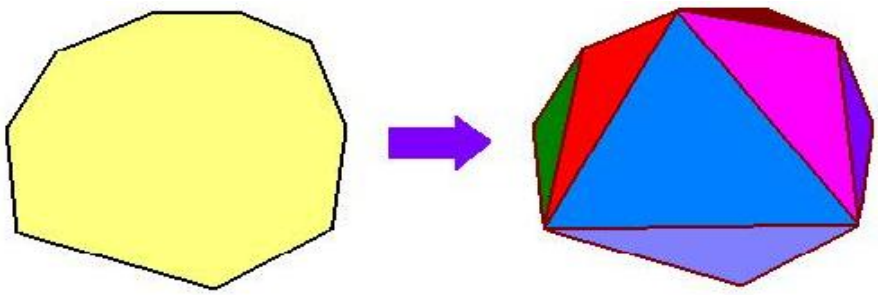

FIG. 10 – Triangulation d'un polygone

tel que :

$$\forall P_i \in \mathbb{PL}, \delta(P_i) = \delta(\phi(P_i))$$

4.1.2 Algorithmes

Tatouer une carte revient à faire des perturbations sur les sommets des polygones qui composent cette dernière. Cependant, si on souhaite effectuer un tatouage avec contraintes, on aura besoin de savoir calculer la surface d'un polygone, le périmètre d'un polygone, et d'autres calculs géométriques sur et entre les polygones. Dans ce qui suit, on suppose des polygones simples, c'est-à-dire dont les segments ne s'intersectent pas(cette hypothèse est standard).

Surface d'un polygone : Les polygones 2D peuvent être décomposés en triangles pour le calcul de la surface. On calcule la surface de chaque triangle et on somme toutes les surfaces des triangles pour avoir la surface du polygone comme illustré sur la figure 10.

Périmètre d'un polygone Le périmètre d'un polygone est la somme de toutes les longueurs de segments qui le composent. Soient p_1 de coordonnées (x_1, y_1) et p_2 de coordonnées (x_2, y_2) deux extrémités d'un segment d'un polygone. La longueur $\overline{p_1 p_2}$ entre ces deux points est donnée par :

$$d(p_1, p_2) = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

Distance entre deux polygones Pour le tatouage avec préservation de la topologie, on a souvent besoin d'estimer une métrique indicatrice, c'est la distance entre deux polygones.

Distance de Hausdorff[15] L'une des mesure les plus utilisées dans le calcul de la distance entre deux polygones simples est la mesure d'Hausdorff. Elle

consiste à calculer la distance minimum entre un ensemble de points d'un polygone et le point le plus proche d'un autre ensemble de points d'un autre polygone. Formellement :

$$h(A,B) = \max_{a \in A}\{\min_{b \in B}\{d(a,b)\}\}$$

Avec A et B deux ensembles de points, et d(a,b) la distance entre les points a et b. La complexité en temps de calcul de cette distance est de l'ordre de O(m + n), où m est le nombre de sommets du premier polygone, et n le nombre de sommets du deuxième polygone.

Maintenant nous présentons notre algorithme de tatouage et celui de détection que nous avons conçus pour les bases de données géographiques.

Algorithme de tatouage :

```
1  Soit Ω = P_1, .., P_N l'ensemble des polygones de la carte à tatouer;
2  msb : Most significant bit;
3  (ω : nombre moyen de points dans tous les polygones);
4  LISTE : liste des polygones tatoués;
5  for j = 1..N do
6  |  Soit P_j = (p_0(x_0, y_0), p_1(x_1, y_1), .., p_n(x_n, y_n)) je j ième polygone;
7  |  for i = 1..n do
8  |  |  R = Randomize(cle||cleSecrete||msb(x_i)||msb(y_i));
9  |  |  if R(0, .., ω) = 0 then
10 |  |  |  (d_max = log_2(√2/2 * d));
11 |  |  |  Index_x = R(0, .., d_max);
12 |  |  |  Index_y = R(0, .., d_max);
13 |  |  |  Mark = R(0, .., 1);
14 |  |  |  x_i[Index_x]=Mark;
15 |  |  |  y_i[Index_y]=Mark;
16 |  |  |  PropagerSurface();
17 |  |  |  if Propagation reussie then
18 |  |  |  |  Sauvegarger j dans LISTE;
19 |  |  |  end
20 |  |  |  else
21 |  |  |  |  Annuler toutes les transformations sur ce polygone;
22 |  |  |  end
23 |  |  end
24 |  end
25 end
```

Algorithm 1: Algorithme de tatouage

A la ligne 8, la fonction R est initialisée par un générateur pseudo-aléatoire cryptographique paramétré. Les paramètres de cette fonction aléatoire sont :

une clé privée secrète qui n'est connue que du propriétaire de la carte, et les bits les plus significatifs (à définir) des coordonnées du point tatoué. Ainsi, un pirate doit détenir la clé secrète et altérer les bits les plus significatifs des coordonnées du point tatoué pour que son attaque ne soit pas détectée (un changement du bit le plus significatif change considérablement la position du point dans la carte et la rend inutilisable). Ensuite, à la ligne 9 on teste si on va tatouer le point ou non, et seul le propriétaire de la carte sait quels sont les points tatoués. Si cette condition est vérifiée, on va insérer un bit dans la partie des bits des poids les plus faibles de l'abscisse du point et un autre dans son ordonnée (lignes 11, 12, 13, 14, 15), cette insertion doit respecter une distance d à ne pas dépasser (précision). A partir de cette distance on saura quel est l'ensemble de bits de poids faibles dans lequel on peut insérer le bit d'information . A la ligne 16 on lance la fonction de propagation de surface. Si la propagation réussit on sauvegarde dans une liste le fait que le polygone a été tatoué (lignes 17 et 18), sinon on annule le tatouage du polygone en cours (ligne 21).

La propagation de surface : C'est la fonction de propagation et préservation de surface qui consiste à modifier les positions des points suivants le point tatoué de telle sorte à préserver la surface du polygone.

1. Soit P un polygone, et p le point à tatouer, et $p^{'}$ le point obtenu après le tatouage.
2. Soit p_1 et p_2 les points avant et après p.
3. Soit δs =*SurfaceTriangle(p_1pp_2)* - *SurfaceTriangle($p_1p^{'}p_2$)*.
4. **if** $\delta s < 0$ **then**
   Perte de surface
   end
   else
   Gain de surface
   end
5. Soit p_3 le point suivant p_2, et p_4 le point suivant p_3.
6. Soit $p_3^{'}$ la projection du point P_3 sur la droite *(p_2p_4)*.
7. Trouver un point p_5 sur la droite *($p_3p_3^{'}$)* tel que δs = *SurfaceTriangle($p_2p_3p_4$)* - *SurfaceTriangle($p_2p_5p_4$)*.
8. **if** $p_3p_5 < d$ **then**
   Fin Propagation
   end
   else
   Repositionner p_5 tel que $p_3p_5 = d$
   end
9. Soit $\delta s = \delta s$−*(SurfaceTriangle($p_2p_3p_4$) - SurfaceTriangle($p_2p_5p_4$))*

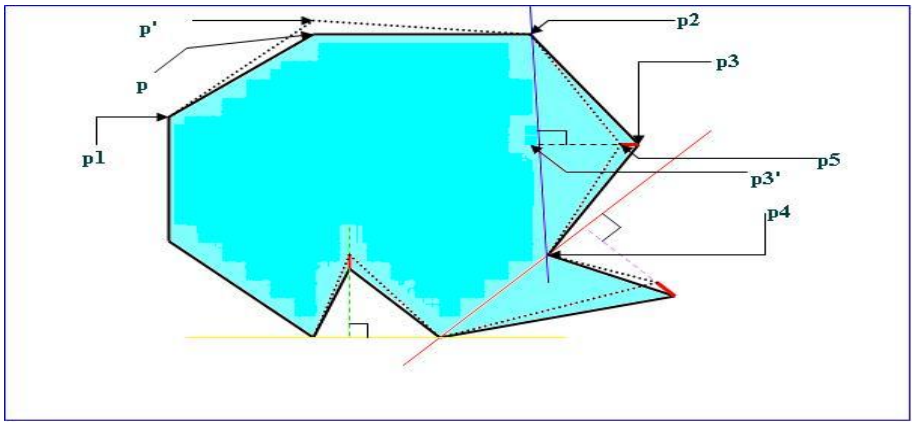

FIG. 11 – Propagation de la surface

10. **if** exist(suivant(p_4)) **then**
    Aller à 5 avec p_3 = *suivant(p_4)*
    end
    else
    Echec de la propagation
    end

La figure 11 illustre mieux le fonctionnement de l'algorithme de propagation de la surface.

Algorithme de détection :

```
1  Soit Ω l'espace de dimension N de tous les polygones de la carte à
   tatouer;
2  Integer : compteur = 0, total = 0;
3  for j = 1..N do
4  |  Soit P_j = (p_0(x_0, y_0), p_1(x_1, y_1), .., p_n(x_n, y_n));
5  |  for i = 1..n do
6  |  |  R = Randomize(cle||cleSecrete||msb(x_i)||msb(y_i));
7  |  |  if R(0, .., ω) = 0 ET P_j ∈LISTE then
8  |  |  |  (Point tatoué);
9  |  |  |  total = total + 1;
10 |  |  |  Index_x = R(0, .., d_max);
11 |  |  |  Index_y = R(0, .., d_max);
12 |  |  |  Mark = R(0, .., 1);
13 |  |  |  if Index_x = Mark ET Index_y = Mark then
14 |  |  |  |  compteur = compteur +1 ;
15 |  |  |  end
16 |  |  end
17 |  end
18 end
19 Retourner(compteur/total) ;(Pourcentage de points tatoués détectés);
```

Algorithm 2: Algorithme de détection

L'algorithme de détection consiste à faire l'inverse de l'algorithme d'insertion, à savoir vérifier si une marque a été dissimulée dans les coordonnées des points des polygones(ligne 7). Notre algorithme de détection est semi-aveugle car on vérifie si le polygone détecté est dans la liste des polygones tatoués(ligne 7). On définit deux compteurs(*compteur*, et *total*) pour le comptage du pourcentage des points détectés comme tatoués sur l'ensemble des points qui devraient être tatoués respectivement(lignes 9 à 14). En fonction de ce pourcentage (ligne 19), on peut décider si la carte est piratée ou non. Si c'est un grand pourcentage, la base de données peut être considérée comme tatouée, sinon on ne peut rien dire.

Afin que l'algorithme de tatouage soit également résistant aux traitements effectués par les utilisateurs des bases de données géographiques, ainsi que pour la préservation des relations topologiques proprement dite, nous proposons de faire quelques estimations statistiques avant la validation du tatouage. Ces estimations porteront sur l'effet du tatouage sur les relations topologiques entre les objets. Il s'agit aussi de trouver les paramètres optimaux de l'algorithme de tatouage qui le rendent le plus résistant possible aux traitements de lissage, filtrage, et d'autres traitements effectués par les producteurs et utilisateurs des données géographiques. Par exemple, pour un filtrage ou lissage faible, moyen, ou fort, trouver les paramètres optimaux de l'algorithme de tatouage qui minimisent le nombre de points tatoués et supprimés ou décalés par les traitements

accomplis. Aussi, pour une précision maximum *Precision*, définie par le propriétaire de la base de données géographique, trouver une autre précision *Precision'*<*Precision*, qui minimise l'effet du tatouage sur la mesure d'Hausdorff, et en conséquence sur les relations topologiques.

Discussion La solution que nous avons proposée aborde un nouveau type de données : les données géographiques. Notre solution peut s'appliquer sur toute base de données géographique, étant donné que ces dernières contiennent toujours des données numériques (polygones). Notre algorithme de tatouage respecte des contraintes de précision et de surface sur les données, et pour la robustesse, nous avons opté pour le principe de Kerckhoff, qui dit que pour une sécurité plus fiable, la méthode doit être publique, et seule la clé reste secrète au propriétaire. Néanmoins notre algorithme de détection n'est pas totalement aveugle, pour une détection très fiable, nous avons besoin de la liste des polygones tatoués, c'est le prix à payer pour un tatouage avec contraintes. Comme optimisation de notre algorithme de tatouage, nous avons proposé de faire des tests et d'estimer statistiquement les paramètres optimaux pour un tatouage optimal qui résiste aux traitements classiques sur les bases de données géographiques.

Faiblesses La faiblesse de notre algorithme est la complexité en temps et en espace de calcul de l'algorithme de tatouage et de détection en fonction N(Nombre de polygones dans la base de données) et ω(paramètre du générateur aléatoire).

4.2 Implémentation

Pour le codage des algorithmes que nous avons proposés, nous avons dû programmer dans deux environnements, celui que nous avons développé au CNAM, et la plate forme de l'IGN. Bien sûr nous n'avons pas eu de difficultés pour exécuter nos programmes sur les deux environnements, c'est l'un des avantages de programmation en java. Le nombre de lignes du code développé est d'environs *2000* lignes.

4.2.1 BD PostgreSQL[16]

Le système de gestion de bases de données utilisé pour le stackage des données est le système objet-relationnel connu sous le nom de Postgres (abréviation de « Postgres95 ») qui est dérivé du logiciel écrit à Berkeley. C'est un logiciel à unitilisation libre.

4.2.2 OpenMap[17]

L'interface d'accès à l'information que nous avons utilisée dans notre prototype repose sur l'outil Openmap. Ecrit en JAVA, il permet la visualisation de

données géographiques sur un espace sélectionné. Composée d'une fenêtre principale dans laquelle sont ajoutées différentes couches d'information, l'applette java Openmap intègre les fonctionnalités de zoom, de déplacement, de centrage et d'historique. L'utilisateur peut sélectionner et régler l'affichage de chaque couche à sa convenance avec une fenêtre de contrôle appelée palette. Il peut choisir la profondeur d'affichage, les couleurs utilisées pour les dessins ou les textes, les polices de caractères, etc. Chaque couche étant pilotée par un composant Java (« bean ») indépendant, elle peut même utiliser des caractéristiques très différentes, comme chercher dynamiquement ses données sur un serveur, offrir une interaction fine avec l'utilisateur en lui permettant par exemple d'accéder aux données brutes ayant servi à la cartographie ou en changeant leur mode de représentation graphique.

4.2.3 Intégration dans les outils IGN

Lors de mon séjour à l'IGN, j'ai travaillé sur la plate forme GeOxygene[18], et les données stockées sur une base de données Oracle. J'ai intégré les programmes (classes java) des algorithmes que nous avons développés au laboratoire Cedric dans leur système, et j'ai utilisé l'éditeur Eclipse. Pour la visualisation des données, j'ai utilisé le logiciel JUMP qui est d'une grande convivialité et qui offre plusieurs fonctionnalités d'affichage pour les données géographiques. Quant aux données, au départ j'ai manipulé des données de l'administratif extraites de la ville de Pau, par la suite j'ai manipulé des données d'une zone plus large(Pau et autour de Pau), pour avoir toutes les formes géomètriques possibles(zones rurales, bâtiments, administratif, etc.).

4.3 Tests et évaluation

Les algorithmes que nous avons conçus ont été programmés en java JDK version 1.4, j'ai également utilisé au CNAM l'éditeur Java d'utilisation libre Netbeans. Les exécutions ont été réalisées sur une machine Pintium 3, d'une vitesse du processeur de 500MHZ, système d'exploitation LINUX Debian avec accès aux données réelles IGN sur le serveur, stockées dans une base de données Postgres version 1.8. Les données utilisées sont celles mises sur le site de l'IGN, il s'agit de deux zones : La BD_TOPO d'Orléans (20814 polygones), et celle de Pamiers(13490 polygones) que nous avons extraites et stockées sur la base Postgres. Le système que j'ai développé au CNAM permet au même temps le tatouage, la simulation des attaques, et la détection. A l'IGN, j'ai intégré les mêmes programmes développés au CNAM dans leur système GeOxygene basé également sur Java.

La figure 12 montre le nombre de points que contient chaque polygone de la base de données.

On voit que les polygones ayant entre cinq et dix points sont majoritaires. Celà est dû au fait que la majorité des polygones représentent des objets de petites

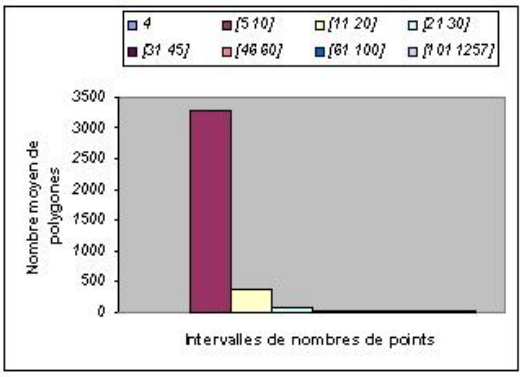

FIG. 12 – Nombre moyen de points par polygone

tailles tels que les bâtiments. Ceci augmente les chances d'échec du tatouage lorsqu'on rajoute des contraintes : la taille des polygones est proportionnelle au nombre de polygones tatoués. Pour les tests, on a pris ω=9 : le nombre de points moyen par polygone.

La figure 13 montre la croissance du nombre de polygones tatoués en fonction du nombre de polygones dans la base de données.

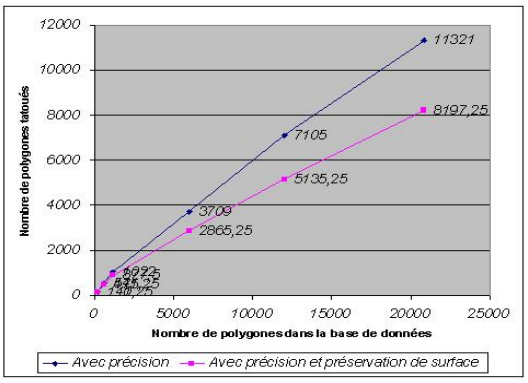

FIG. 13 – Nombre de polygones tatoués en fonction du nombre de polygones dans la base de données

On constate que la croissance est linéaire. En conséquence, la distribution des polygones tatoués sur toute la base est uniforme, celà prouve que notre algorithme est résistant aux attaques qui consistent à extraire des parties de la base correspondant à des parties précises sur la carte. On remarque aussi que le nombre de polygones tatoués avec précision et préservation de la surface

est inférieur au nombre de polygones tatoués avec précision seulement. Des deux figures précédentes on peut conclure ce qui suit : le nombre de contraintes est inversement proportionnel au nombre de polygones tatoués. La précision est proportionnelle à la discrétion du tatouage, par contre elle est inversement proportionelle au nombre de polygones tatoués avec contraintes.

La figure 14 montre la même chose que la précédente, sauf qu'elle montre implicitement l'influence des paramètres du générateur aléatoire utilisé dans nos algorithmes.

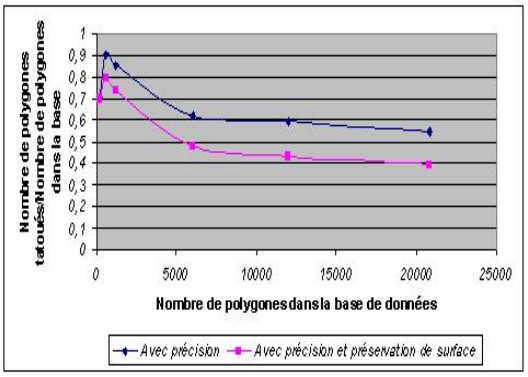

FIG. 14 – Taux de polygones tatoués en fonction du nombre de polygones dans la base de données

En effet, les premiers n-uplets de la base de données ont été sélectionnés et plusieurs exécutions ont été lancées, de telle sorte à avoir un maximum de polygones tatoués en changeant les paramètres de la fonction aléatoire. On peut donc bien avoir un taux plus élevé de polygones tatoués si on choisit bien les paramètres du générateur aléatoire.

La figure 15 montre l'évolution du temps d'exécution de l'algorithme de tatouage en fonction du nombre de polygones dans la base de données.

On constate d'abord que le temps d'exécution de l'algorithme de tatouage est relativement long, et bien sûr il est linéaire (environs 22 polygones tatoués par seconde).

Maintenant, on souhaite tester la robustesse de nos algorithmes aux attaques, en plus de sa résistance à l'extraction d'une partie d'une carte tatouée. Dans un premier temps, on a testé la détection en l'absence d'attaques, avec d'abord la clé secrète utilisée lors du tatouage, et ensuite avec différentes clé autres que celle utilisée lors du tatouage. On a lancé plusieurs exécutions et on a obtenu le graphe sur la figure 16.

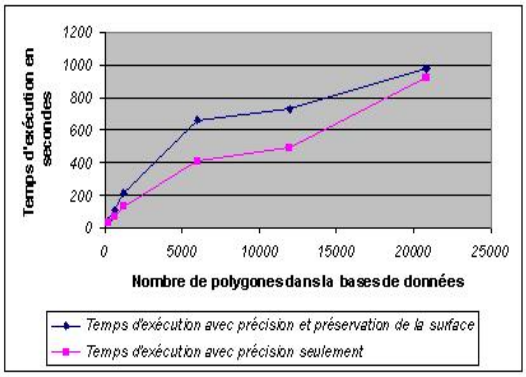

FIG. 15 – Temps d'exécution de l'algorithme de tatouage en fonction du nombre de polygones dans la base de données

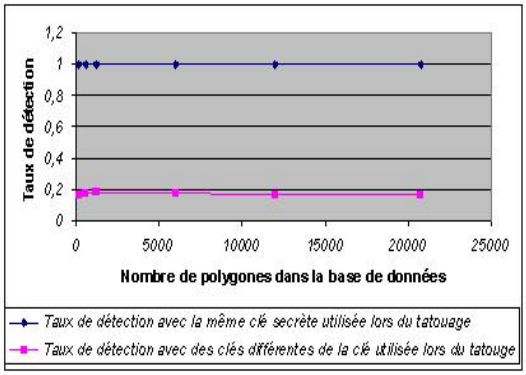

FIG. 16 – Taux de détection en l'absence d'attaques

On remarque qu'en l'absence d'attaques, on a une détection de presque 100% avec la clé secrète utilisée lors du tatouage, et avec les autres clés, la détection est très négligeable (moins de 20%). D'où l'importance de la clé secrète.

Maintenant, on a simulé une attaque sur la base tatouée en rajoutant un bruit aléatoire à tous les points de tous les polygones de toute la base, dans le but de détruire le tatouage, c'est l'attaque la plus sévère qui puisse se produire dans notre qui peut détruire le tatouage. On a lancé plusieurs exécutions avec à chaque fois un bruit aléatoire plus ou moins grand. On a obtenu la figure 17.

Dans la majorité des cas, lors de l'ajout d'un bruit aléatoire très petit, qui n'altère pas les bits de poids fort des coordonnées des points, on a obtenu une détection en moyenne de plus de 65%, et lorsqu'on a simulé des attaques qui altèrent les bits de poids forts des coordonnées, on a eu un faible taux de

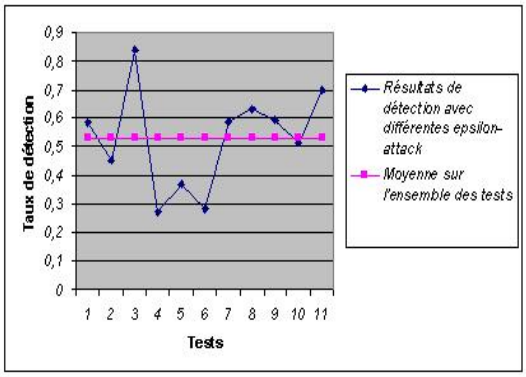

FIG. 17 – Taux de détection après rajout d'un bruit aléatoire sur tous les points de la carte

détection, en revanche la carte a été fortement altérée, et devient inutilisable. En moyenne de toutes les attaques, le taux de détection dépasse les 53%. Celà montre la résistance de notre algorithme à ce genre d'attaques.

Tests effectués à l'IGN Les tests effectués à l'IGN concernent essentiellement la résistance aux traitements de filtrage et de lissage. Les tests, après application du filtrage de Douglas Peucker et le lissage de Gauss, ont montré qu'il est toujours possible de trouver des paramètres de l'algorithme de tatouage qui nous permet de reconnaître notre tatouage avec une forte probabilté. Nous avons également évalué l'effet du tatouage sur la distance d'Hausdorff, et on a obtenu le graphe sur la figure 18. On voit que le tatouage n'a pas un grand

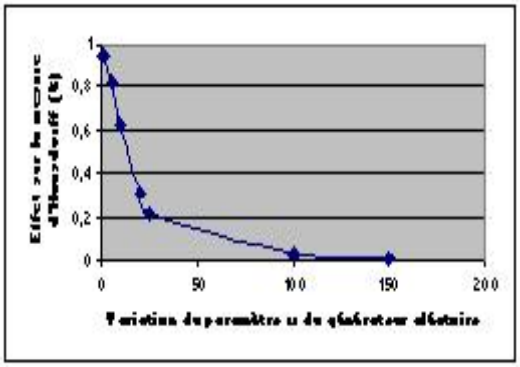

FIG. 18 – Effet du tatouage sur la mesure d'Hausdorff

effet sur la mesure d'Hausdorff, même avec un paramètre de tatouage $\omega = 1$ qui permet de tatouer plusieurs points par polygone. La mesure d'Hausdorff est

une mesure indicatrice de l'effet du tatouage sur les relations topologiques entre les objets. En conséquence, on est très optimiste quant à l'extension de notre algorithme de tatouage à un algorithme qui préserve également les relations topologiques entre les objets de la base de données.

5 Conclusion et perspectives

Durant ce stage de Master, nous avons étudié les différentes techniques et algorithmes utilisés pour le tatouage des données numériques, nous avons également étudié les bases de données géographiques, leur structure, ainsi que les contraintes que doit respecter le tatouage. Ensuite, nous avons proposé une solution de tatouage qui consiste à bruiter légèrement des sommets des polygones constituant les objets de la base de données géographique. Les algorithmes que nous avons proposés se basent sur une clé secrète que seul le propriétaire de la base de données connaît, et respectent les contraintes les plus importantes, à savoir la précision, et la surface des objets. Pour évaluer nos algorithmes, nous avons développé un système de tatouage, et nous avons fait des tests sur des données réelles de l'IGN. Les résultats obtenus ont montré la dicrétion de notre tatouage(Les déformations ne sont pas visibles) et la robustesse de nos algorithmes contre les attaques qui peuvent viser la Base de Données. L'intégration de nos algorithmes dans le système GeOxygene de l'IGN (Simulation d'attaques avec le filtrage de Douglas Peucker, et le lissage de Gauss) ont approuvé les résultats obtenus au CNAM. Notre solution peut facilement s'étendre à un tatouage avec préservation de relations topologiques entre les objets, les tests sur la mesure d'Hausdorff montre cela. Néanmoins un problème de complexité surgit de ce fait, étant donné qu'il faudra vérifier pour chaque objet ses relations topologiques avec tous les autres objets de toutes la base de données. Comme solution à ce problème, et comme perspective à notre travail, nous proposons une solution d'optimisation du temps de calcul dans ce cas qui consiste diviser l'espace de tous les polygones de la Base de Données Géographique en sous-ensembles, où chaque sous-ensemble représente une composante connexe, c'est à dire les objets dans la même composante connexe ont des relations topologiques entre eux, mais aucune avec les autres objets des autres composantes connexes. Ensuite appliquer l'algorithme de tatouage avec préservation de la topologie en parallèle sur chacune de ces composantes connexes.

6 Remerciements et dédicaces

Je tiens à remercier d'abord Mr David Gross-Amblard, maître de conférences au la boratoire Cedric du Cnam, de m'avoir donné la chance de faire ce stage, mais aussi pour sa disponibilité, ses encouragements, ses conseils, et son suivi permanents et presque quotidiens durant toute la durée de mon stage. Je remercie aussi mon collègue Julien Lafaye, doctorant au laboratoire Cedric du Cnam, pour son aide précieuse, surtout sur le plan technique. Je remercie Mr Bernd

Amann, professeur à l'Université Pierre et Marie CURIE, pour m'avoir aidé à trouver le stage. Je remercie Eric Grosso et Anne Ruas du laboratoire Cogit de m'avoir suivi lors de ma période de stage à l'IGN. Sans oublier de remercier mon père,mon cousin Achour, Safa, ma mère, mes frères, et mes amis en Algérie pour leur soutien et leurs encouragements depuis mon arrivée en France, ainsi que mon ami Makiou Hamid pour son aide, et tous ceux qui, de près ou de loin, m'ont soutenu cette année.

Je dédie ce mémoire de Master à Safa, à toute ma petite et grande famille en Algérie, à tous les membres de l'équipe de recherche Vertigo et notamment ceux avec qui j'ai été au laboratoire qui étaient tous très agréables : Michel , David, Dan, Valérie, Julien, Noha, Eric, Cedric, Imen, et Marie. A tous les amis là où ils se trouvent(Paris, Alger, Bougie, et ailleurs).

Références

[1] http ://cedric.cnam.fr/ dgram/Watermill

[2] www.ign.fr

[3] http ://cedric.cnam.fr/vertigo/tadorne

[4] http ://www.mrnfp.gouv.qc.ca/territoire/portrait/portrait-donnees-mille.jsp

[5] Rakesh Agrawal, Jerry Kienan.,Watermaking Relational Data Bases, *Procedings of the 28th VLDB Conference,* (Hong Kong, China :2002).

[6] R. Sion, M. Atallah, S. Prabhakar.,Watermarking numerics sets, *In Proceedings in of IWDW,* (Magdeburg, 2002).

[7] Gerrit Schultz, Michael Voigt.,I High Capacity Watermarking System for Digital Maps, *MM&Sec'04,* (Magdeburg, Germany : September 20-21 2004).

[8] Ryutarou Ohbuchi, Shuh Endoh.,Watermaking 2D Vector Maps in The Mesh Spectral Domain, *International Conference on Shape Modelling and Applications,* (Seoul, Korea : pp.216225, May. 2003).

[9] http ://etudiant.univ-mlv.fr/g̃rozier/Docs/IndexationT&I/Recherche/Kd-Tree

[10] lien vers doc de Karni de l'article ohbuchi

[11] Ryutarou Ohbuchi, Akio Mukaiyama, Shigeo Takahashi : A Frequency-Domain Approach to Watermarking 3D Shapes, *Comput. Graph. Forum,* (2002)

[12] livre de michel

[13] M. Bader, X. Barillot, S. Mustière, M. Barrault, C. Duchêne., Line gaussian smoothing, *20/01/99*

[14] http ://geometryalgorithms.com/Archive/algorithm_0205/

[15] http ://www-cgrl.cs.mcgill.ca/g̃odfried/teaching/cg-projects/98/normand/main.html

[16] postgresql livre au labo cnam

[17] http ://openmap.bbn.com

[18] http ://oxygene-project.sourceforge.net

[19] livre de cedric

[20] article de david

[21] Rakesh Agrawal, Peter J. HAAS, Jerry Kienan.,Watermaking Relational Data, *Framework, algorithms and analysis,* (2002).