

L'extension "Topologie" pour OpenJUMP

v 0.8.2 (2016-06-12) by Michaël Michaud

1.1 History

0.8.2 (2016-06-12) : corrige 2 bugs dans « Ajuster les frontières des polygones »

- l'angle entre segments était mal calculé
- dans certains cas, les polygones troués n'étaient pas traités

0.8.1 (2014-05-27) : corrige un bug dans "**Projeter des points sur des lignes**" concernant l'option permettant de projeter un point plusieurs fois.

0.8.0 (2014-05-26) : ré-écriture complète du code de "**Projeter des points sur des lignes**" qui gérait mal l'insertion de plusieurs points sur un même segment.

0.7.0 (2013-01-27) : **ajout du plugin "Ajuster les frontières des polygones"** après un gros travail de débogage et d'optimisation

0.6.1 (2013-01-09) : ajout des traductions es/it/fi (de G. Aruta et J. Rahkonen)

0.6.0 (2012-09-15) : ré-écriture complète du plugin "Projeter des points sur des lignes" qui permet maintenant de faire des multi-projections.

0.5.1 (2012-08-30) : correction d'un problème dans l'interface

0.5.0 (2012-06-25) : **ajout du plugin "Projeter des points sur des lignes"**

0.4.0 (2012-05-20) : NetworkTopologyCleaning : correction d'un bug dans le calcul de la valence d'un noeud du réseau et ajout d'une option permettant de conditionner une opération de snap par l'égalité des valeurs d'attribut des deux objets.

0.3.4 (2012-05-17) : Nettoyage topologique d'un réseau : certains noeuds étaient insérés dans le mauvais segment

désactivation de "Ajuster les frontières des polygones" (jugé pas assez robuste pour être intégré dans OpenJUMP 1.5.2 PLUS edition)

0.3.3 (2011-11-22) : **ajout de "Ajuster les frontières des polygones"**

0.3.2 (2011-11-08) : correction d'une erreur dans la traduction de l'interface

0.3 (2011-04-22) : création d'une seule extension regroupant les plugins d'Assurance Qualité et de Topologie

0.2 (2011-04-11) : intégration du nouveau MultiInputDialog venant dans OpenJUMP 1.4.1

0.1 (2010-04-22) : version initiale de l'extension

1.2 Concepts généraux

L'extension "Topologie" est un ensemble de plugins fournis dans un fichier jar à placer dans le répertoire d'accueil des plugins d'OpenJUMP (répertoire lib/ext par défaut).

Tous les plugins de l'extension Topologie sont relatifs à la détection et/ou à la correction d'erreurs topologiques.

Important : cette extension travaille sur les géométries et utilise intensivement les fonctionnalités de JTS. A contrario, l'extension graph utilise plus les algorithmes de la bibliothèque JGraphT et travaille des des graphes construits à partir des relations d'adjacence entre objets.

Cette extension n'a pas d'autres dépendances que JTS et OpenJUMP.

TODO : créer un plugin exposant les différentes fonctionnalités du package noding de JTS.

1.2.1 Les plugins de l'extension topologie

Tous les plugins de l'extension topologie sont installés dans le menu Extensions d'OpenJUMP, dans un sous-menu appelé... Topologie.

La plupart des options ont une bulle d'aide donnant de plus amples information sur la signification de l'option.

2 Supprimer les micro-segments...

Ce plugin supprime les micro-segments des lignes et des polygones.

Ce processus est souvent utile avant d'effectuer un nettoyage topologique.

En fait, la plupart des opérations de nettoyage de la topologie utilisent un paramètre de tolérance pour décider si deux objets (ou nœuds ou segments) doivent être confondus, et la présence de micro-segments plus petits que cette tolérance à l'intérieur même d'une géométrie vient souvent perturber le processus.

Ce que fait le plugin et ce qu'il ne fait pas :

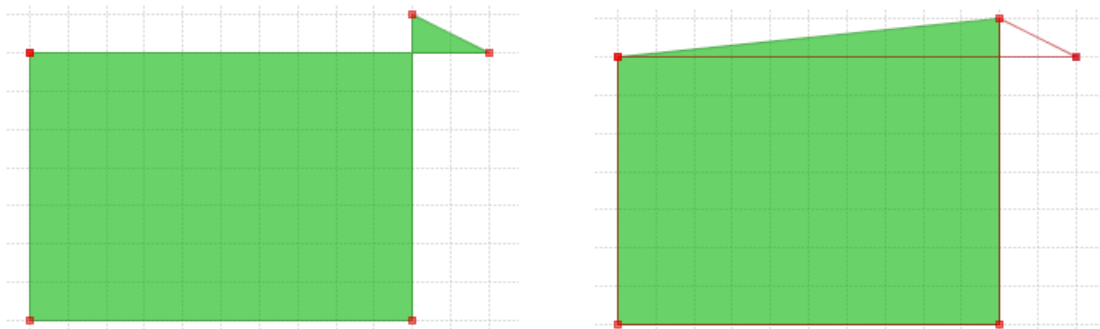
- "Supprimer les micro-segments" choisit soigneusement le point à supprimer de la géométrie de manière à la déformer le moins possible : les micro-segments sont composés de deux points. Les points situés sur la frontière d'un objet ne sont jamais supprimés, et si les deux points sont strictement intérieurs à la ligne, le plugin supprime le point où l'angle intérieur est le plus proche de l'angle plat (180°).



- "Supprimer les micro-segments" ne traite que les lignes simples et les polygones. Les multilignes, multipolygones et autres multigéométries sont actuellement ignorées.
- "Supprimer les micro-segments" ne vérifie pas si le point qu'il supprime est partagé par une autre géométrie. Ce traitement, purement géométrique, peut donc détériorer la topologie. Heureusement, les erreurs topologiques ainsi générées devraient théoriquement pouvoir être réparées par les autres outils de correction topologique.

Cas d'utilisation particulier :

"Supprimer les micro-segments" permet de corriger les polygones rendus invalides par la présence de petites auto-intersections :



Noter que le problème est résolu par la suppression d'un point, ce qui ne marche que pour les auto-intersections composées d'un seul micro-segment.

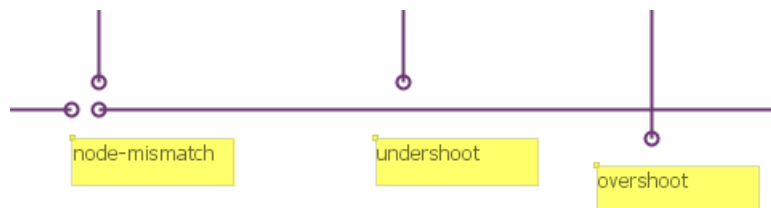
Actuellement, il manque à OpenJUMP un outil convivial permettant de corriger des auto-

intersections plus complexes. Un processus compliqué consisterait à extraire les frontières de la couche polygonale, calculer les noeuds, polygoniser, re-cr  er les trous, r  cup  rer les attributs par appariement...

3 Nettoyage topologique d'un réseau...

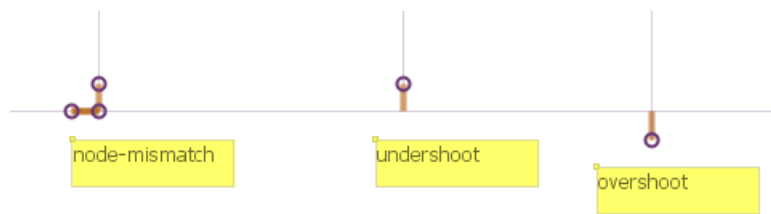
Ce plugin peut traiter un jeu de données représentant un réseau, et détecter et/ou réparer quelques erreurs topologiques répandues comme :

- node mismatches : nœud du réseau proches les uns des autres mais non superposés
- undershoot : une extrémité de ligne arrive proche d'une autre ligne sans être réellement accrochée dessus



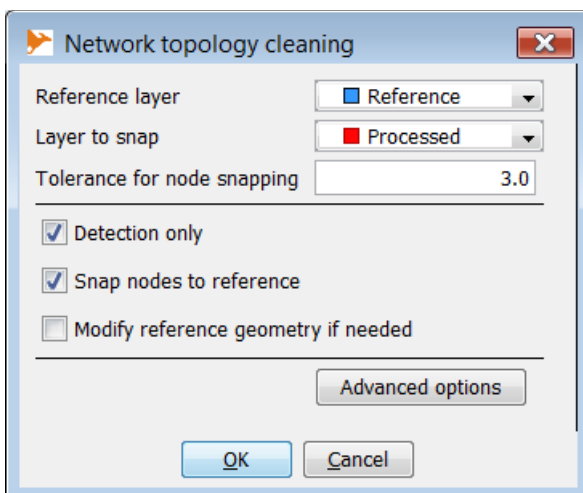
- overshoot : une ligne en croise une autre et se termine juste après l'intersection

Définition formelle : le plugin "nettoyage topologique d'un réseau" recherche tous les nœuds d'un réseau (extrémités des lignes le composant) qui sont suffisamment proche d'une autre ligne ("proche" étant paramétrable), mais non accrochés à cette ligne (pas de point partagé).


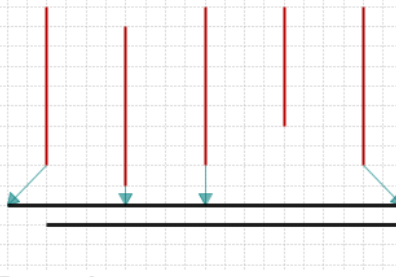
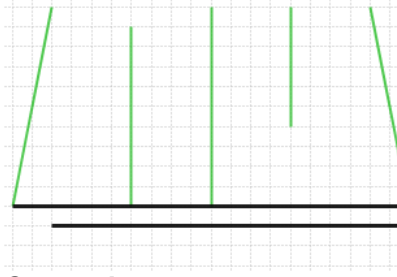


Les anomalies peuvent être simplement détectées ou bien alors corrigées.

3.1 Le paramètre principal du nettoyage topologique



Le principal paramètre est une distance entre le nœud d'extrémité d'un objet et un autre objet, distance au-dessous de laquelle le plugin considère que le nœud aurait probablement du être accroché sur cet autre objet.

		
<p>Dans cet exemple, la tolérance est de 3 mètres.</p> <p>Les nœuds du bas des objets rouges 1, 2, 3 et 5 sont sous le seuil de tolérance par rapport à l'objet de référence (en noir).</p>	<p>Detection</p> <p>Le nœud du 1er objet est à moins de 3 m du nœud gauche de l'objet de référence.</p> <p>Les objets 2 et 3 sont proches de l'objet de référence mais à plus de 3 m de ses points de construction.</p> <p>L'objet 4 est hors tolérance ;</p> <p>Le 5ème objet est à moins de 3 m d'un point intermédiaire de l'objet de référence.</p>	<p>Correction</p> <p>Le premier et le dernier objets sont accrochés au point le plus proche.</p> <p>Le 2ème et le 3ème objet sont ramenés sur le lieu le plus proche de l'objet de référence (un point devra alors être inséré dans la géométrie de ce dernier).</p>

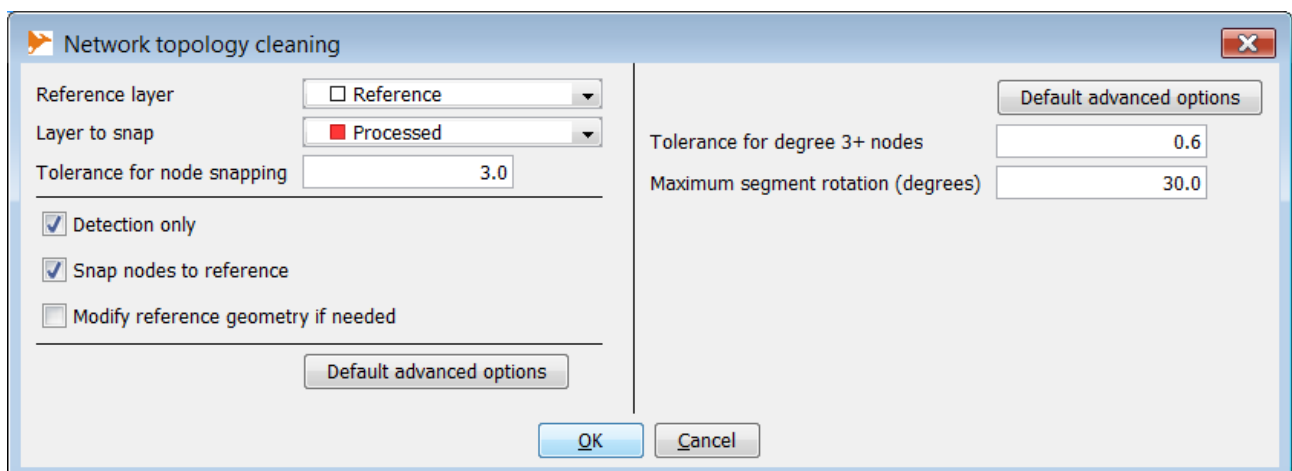
3.2 Options avancées : les nœuds de degré 3

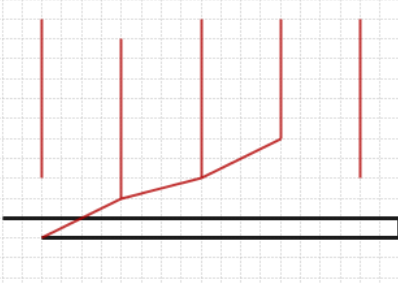
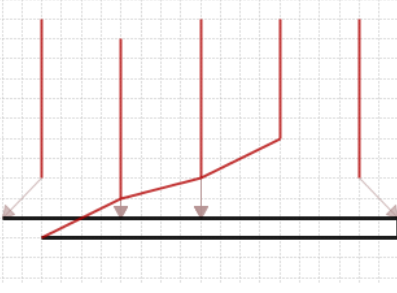
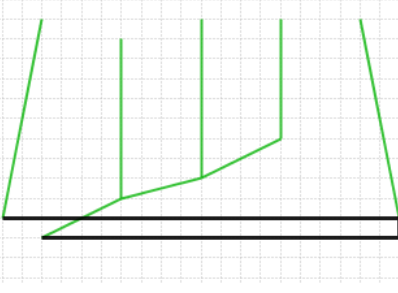
Une option permet de paramétrer la tolérance d'accrochage pour les noeuds de degré 3 ou plus de manière indépendante de la tolérance globale (qui ne vaut plus alors que pour les nœuds de degré 1 ou 2).

Il peut en effet être utile de diminuer la tolérance pour ces noeuds, car s'ils ne sont pas "accrochés" à la couche de référence, la probabilité pour que cela soit volontaire est plus grande que pour des noeuds d'ordre 1 ou 2.

Par défaut la tolérance d'accroche pour les noeuds de degré 3 ou plus est de 1/5 fois la tolérance définie pour les noeuds pendants.

Changer la tolérance pour ces nœuds particulier est assez facile :






		
Le nœud du bas des deuxième et troisième objet verticaux sont de degré 3 (cela signifie que non seulement ces nœuds sont "accrochés" entre eux, mais aussi qu'ils se rejoignent à leurs extrémités)	La détection des mauvais accrochages détecte des "undershoots" parce que les nœuds sont situés au dessous de la distance de tolérance globale (3m) par rapport aux objets de référence...	...mais seuls les nœuds de degré 1 situés sous cette tolérance sont accrochés. Les nœuds de degré 3 auraient toutefois été accrochés s'ils avaient été situés au-dessous de la distance par défaut pour les nœuds de degré 3 ($3/5 = 0,6$ m)

3.3 Options avancées : rotation maximum

Il existe un second paramètre qui correspond à la rotation maximum acceptable pour le segment qui doit être déformé pour être accroché à l'objet de référence.

Ce paramètre n'est utilisé que pour les nœuds pendants (nœuds de degré 1).

L'utilisation d'un seuil de rotation pour les nœuds d'ordre 2 ou plus pourrait en effet conduire à rompre la continuité du réseau initial. Prenons l'exemple de 2 lignes consécutives dont la première pourrait être accrochée à la référence sans dépasser le seuil de rotation et l'autre non. L'accrochage de la première ligne à la référence romprait ainsi la continuité avec la deuxième ligne qui elle, ne pourrait être accrochée sans dépasser la limite fixée par notre paramètre de rotation.

		
Ici, nous avons deux candidats à l'accrochage (tous deux situés à moins de 3 m de la référence).	Les deux pourraient être accrochés à la référence, mais ce faisant, l'orientation du premier segment serait considérablement modifiée (au-delà du seuil de rotation fixé), tandis que l'accrochage du deuxième ne modifiera que très	Finalement, avec un paramètre de rotation de 15°, la première ligne n'est pas affectée par le traitement tandis que la deuxième est accrochée à l'objet de référence.

	peu son orientation.	
--	----------------------	--

Lorsque les deux points délimitant le plus proche segment de référence sont tous deux dans la distance de tolérance, le point d'accroche choisi est celui qui minimise la rotation du segment et non le plus proche (à moins que l'un des points de référence ne soit un point d'extrémité, auquel cas il est toujours privilégié).

L'extrémité inférieure de la ligne rouge est située à moins de 3 m de l'objet de référence et les deux points du segment le plus proche sont tous les deux situés à moins de 3 m de cette extrémité.	Le point d'accroche choisi est alors le point provoquant la moindre rotation et non le point le plus proche	

3.4 Cas particuliers

Cas particulier où le traitement ne donne pas forcément le résultat attendu

Du fait du critère de rotation évoqué plus haut, dans le cas présenté ci-dessus où les 2 points de référence sont situés à moins de 4 m des objets à accrocher (tolérance principale = 4 m)...	... le processus qui cherche à minimiser la rotation des segments va générer deux lignes qui s'intersectent.	Un moyen d'éviter ce type de problème est de corriger la topologie des objets verts entre eux (avec le même traitement topologique) avant d'essayer de les accrocher à la couche de référence.

3.5 Options de sortie

--	--	--

Détéction	Correction	Correction de la référence
Création d'une couche contenant les vecteurs de translation à appliquer aux nœuds de la couche à traiter pour les accrocher sur la couche de référence. Cette phase ne tient compte que du critère principal de distance et contient par conséquent des vecteurs de translation qui ne seront pas utilisés dans la correction (lorsque le degré du nœud ou le critère de rotation l'interdit).	La couche à traiter est recopiée, puis, les nœuds à déplacer pour les accrocher aux objets de référence sont déplacés.	Pour être corrigée, la couche de référence doit être modifiable. Attention, la modification de la couche de référence n'est pas annulable. La correction de la couche de référence consiste à ajouter les points intermédiaires manquant chaque fois que nœud de la couche à traiter est venue s'accrocher au milieu d'un segment de référence.

3.6 Description de la couche contenant les vecteurs de translation

L'attribut "ACCROCHAGE" indique si le nœud a finalement été accroché, à quoi il a été accroché et le cas échéant, pourquoi il n'a pas été accroché :

Accrochage au sommet : le nœud a été accroché à un sommet de la couche de référence

Accrochage au point : le nœud a été accroché sur un point intermédiaire de la couche de référence

Accrochage au segment : le nœud est venu s'accrocher au milieu d'un segment de la couche de référence

Non accroché : le nœud n'a finalement pas été accroché

Non accroché ($D > xx$) : le nœud n'a finalement pas été accroché parce que la distance au sommet est supérieure à xx (tolérance fixée pour les nœuds d'ordre 3+, si inférieure à la tolérance de base)

Non accroché ($A > xx$) : le nœud n'a finalement pas été accroché parce que cela induisait une rotation du segment supérieure à xx (tolérance fixée dans les options avancées)

L'attribut "ROTATION" indique, pour les nœuds qui ont été accrochés à un objet de référence, la rotation qui a dû leur être appliquée (en degrés).

4 Plugin projeter des points sur des lignes

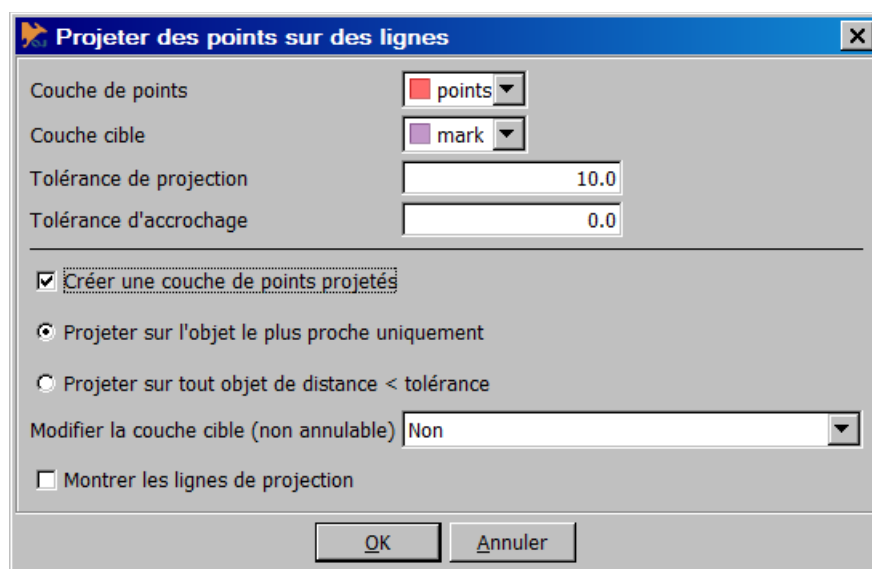
Le plugin projette les points d'une couche sur les lignes d'une autre couche. Optionnellement, les points projetés peuvent être insérés dans les lignes sur lesquels ils sont projetés, ou servir à les découper.

4.1 Entrées

La Couche de points doit contenir au moins un point.

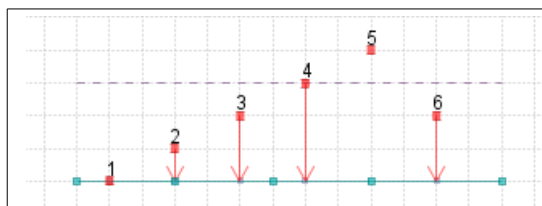
La couche de lignes doit contenir au moins une ligne de type LineString. Les MultiLineStrings ne seront pas utilisées. Si votre jeu de données contient des lignes multiples que vous souhaitez utiliser, décomposer les en lignes simples avant de mettre en œuvre ce traitement.

4.2 Les paramètres du Plugin

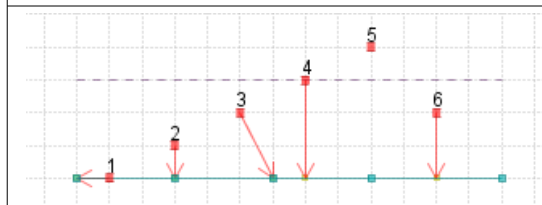


Tolérance de projection : c'est le paramètre principal. Si la distance minimale d'un point à la ligne la plus proche est supérieure à cette tolérance, le point ne sera pas projeté. Par défaut, le point est projeté orthogonalement.

Tolérance d'accrochage : si la distance minimum entre la projection orthogonale et le point de construction le plus proche est inférieur à cette valeur, le point projeté vient s'accrocher sur le point de construction.



Avec une **Tolérance de projection = 3** et une **Tolérance d'accrochage = 0**, les points sont projetés orthogonalement sur la ligne la plus proche, sauf si la distance à la ligne est strictement supérieure à la tolérance définie (point 5).



Regardons maintenant les effets du paramètre **Tolérance d'accrochage avec une valeur de 1** :

- 1 - la projection glisse sur le vertex voisin (≤ 1)
- 2 - la projection orthogonale coïncide avec un vertex
- 3 - la projection orthogonale est déviée jusqu'au vertex le plus proche (≤ 1)
- 4 - la projection orthogonale ne peut être décalée sur le vertex le plus proche sans que la distance totale ne dépasse la Tolérance de projection (3)
- 5 - la distance est $>$ à la distance de projection
- 6 - la projection orthogonale est trop loin du plus proche vertex

Créer une couche de points projetés : crée une nouvelle couche contenant la projection des points sur la ligne la plus proche. Le nom de la couche produite est "coucheSource - projeté". Elle a le même schéma que la couche de points plus un attribut contenant la distance minimale entre le point source et l'objet cible.

Projeter sur tout objet de distance < tolérance : l'outil peut projeter un même point sur plusieurs objets cibles si ceux-ci respectent les critères de distance.

Modifier la couche de ligne (non annulable) : par sécurité, ce composant n'est actif que si la couche de ligne est modifiable. Il possède trois options :

- aucune opération : ne modifie pas les lignes
- insérer les points projetés : ajoute les points projetés dans la géométrie des lignes
- découpe les lignes (MultiLineStrings) : découpe les lignes au niveau des points de projection. Le nombre d'objets total n'augmente pas, mais les lignes simples sont fragmentés en MultiLineStrings et peuvent être décomposées en plusieurs objets simples par la suite.

L'insertion de points et la découpe de lignes sont effectués sur la géométrie d'origine et ne sont pas annulables. Ce choix a été effectué pour des questions de performance et de consommation mémoire. Merci de prendre vos précautions en faisant une copie de votre couche auparavant si vous n'êtes pas sûr que le résultat de l'opération vous conviendra.

Montrer les lignes de projection : crée une couche contenant les segments reliant chaque point à sa projection. Cette couche reprend le schéma de la couche de points afin de préserver tous les attributs, avec en plus l'attribut distance au plus court (qui peut être inférieur à la longueur du segment si la tolérance de snap n'est pas nulle).

5 Ajuster les frontières des polygones (topology-0.7.0+)

5.1 Mise en oeuvre

Ce plugin nettoie une couverture polygonale qui comprend des défauts tels que micro-recouvrements ou micro-interstices.

Il est basé sur le code de la Java Conflation Suite, principalement écrite par Martin Davis (architecte de JTS) et publiée par la société Vividsolutions. Les améliorations par rapport au plugin d'origine sont :

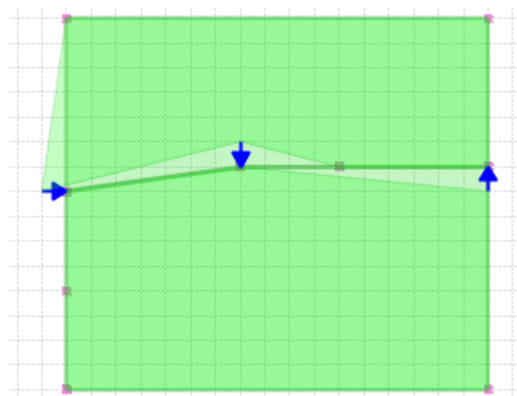
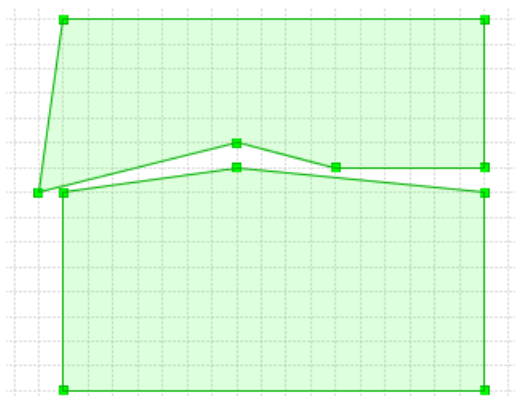
- améliorations des performances
- support complet des polygones à trous
- débogage

Comment marche ce plugin ? Le plugin essaye d'apparier chaque segment de chaque polygone avec les segments des polygones voisins. Les segments sont appariés lorsque :

- ils ne sont pas topologiquement égaux ($AB \neq CD$ et $AB \neq DC$)
- la distance minimum entre les segments est inférieure à une tolérance définie par l'utilisateur,
- l'angle entre segments (modulo π) est inférieur à la tolérance angulaire
- la projection orthogonale de chaque segment sur son homologue est non vide

Si des segments appariés sont détectés sur un contour, ces contours sont fusionnés :

- en snapant les points des segments appariés des deux contours lorsqu'ils sont suffisamment proche l'un de l'autre
- en ajoutant les points nécessaires du contour A dans le contour B et vice-versa.



5.2 Options

Les options suivantes sont disponibles dans le plugin "Ajuster les frontières des polygones" :

Décomposer les multi-polygones : le plugin ne traite pas les multi-polygones. C'est pourquoi cette option est cochée par défaut. Elle reste toutefois optionnelle, car il peut être préférable de pré-traiter la couche hors du plugin et de décocher cette option au moment du nettoyage, afin de ne pas cumuler les difficultés.

Normaliser les polygones : le plugin ne traite pas les polygones mal orientés. C'est pourquoi cette option est cochée par défaut. Elle reste toutefois optionnelle, car il peut être préférable de pré-traiter la couche hors du plugin et de décocher cette option au moment du nettoyage, afin de ne pas cumuler les difficultés.. Note : `buffer(0)` produit des géométries normalisées.

Tolérance : si la distance entre les segments dépasse cette tolérance, les polygones ne sont pas ajustés.

Tolérance angulaire : si l'angle entre les deux segments dépasse cette tolérance (modulo π), les polygones ne sont pas ajustés.

Utiliser le cadre : si cette option est cochée et qu'un cadre est présent dans le projet, seules les parties situées à l'intérieur du cadre seront traitées.

5.3 Comment tirer le meilleur parti du plugin

Le plugin d'ajustement des frontières est un plugin très puissant, mais le nettoyage d'une couche de polygones est une opération délicate qui n'a malheureusement pas une solution unique. Il peut être utile de s'y prendre en plusieurs fois pour obtenir le meilleur résultat possible :

- appliquer l'opération une première fois avec un paramètre de tolérance très inférieur à la tolérance maximum que vous êtes prêt à accepter permet de traiter dans un premier temps les erreurs indiscutables et faciles à corriger avant de s'aventurer vers les cas plus difficiles. Lorsque plusieurs points/segments sont situés dans le même voisinage, cela aidera l'algorithme à traiter les problèmes dans le bon ordre.

- l'algorithme traite les objets deux par deux. Lorsque plus de deux objets sont impliqués par un ajustement de frontière, le traitement peut laisser quelques incohérences. Dans ce cas, un deuxième passage peut aider à résoudre les problèmes résiduels.

5.4 Fonctionnement interne du plugin

Le plugin d'ajustement de polygones est complexe. Voici quelques explications qui peuvent aider à sa maintenance.

5.4.1 Principales structures de données mises en oeuvre

De la plus grande à la plus élémentaire, les principales structures de données utilisées sont :

Coverage : l'objet couverture polygonale, unique, contient une table de correspondance entre les objets source et des objets `CoverageFeature` (voir ci-dessous). Il inclut également une table de tous les Vertex impliqués (`VertexMap`) et un tableau des coordonnées à réajuster.

CoverageFeature : contient une référence vers l'objet source, les contours extérieur et intérieurs de l'objet modélisés par des objets "Shell" et des indicateurs sur le déroulement de l'appariement (`isProcessed` et `isAdjusted`).

- `computeAdjustmentSingle` calcul les ajustements avec les contours limitrophes et
- `getAdjustedGeometry` retourne la nouvelle géométrie ré-ajustée.

Shell : c'est la structure la plus importante de l'algorithme. Elle contient un tableau de coordonnées contenant les coordonnées d'origine, un tableau contenant les coordonnées ajustées, et un tableau des segments dans lesquels les nouveaux points vont être insérés. Les principales méthodes sont :

- *match*(Shell, SegmentMatcher, SegmentIndex) retourne un booléen et
- *computeAdjusted*() calcul les coordonnées définitives

FeatureSegment : une extension de l'objet LineSegment de JTS qui surcharge la méthode equals (et hashCode) pour les besoins de leur indexation (voir aussi *Segment*).

Segment : un GeometryComponent qui représente un segment de la géométrie source dont les extrémités peuvent être réajustées et dans lequel on va pouvoir insérer des points de la géométrie limitrophe.

Vertex : un point dont la position peut être ajustée, mais dont le déplacement doit entraîner le déplacement de tous les points initiaux de coordonnées identique.

VertexMap : une table mettant en relation les coordonnées d'origines avec des noeuds qui peuvent se déplacer pendant les opérations d'ajustement. C'est cette structure qui garantit que les déplacements de certains points ne perdent pas la cohérence topologique pré-existence.

5.4.2 Processus

La classe déroulant l'ensemble du processus est CoverageCleaner. Les principales étapes sont :

1 - Récupérer les segments appariés et les objets concernés

Recherche des FeatureSegments appariés (et intersectant le cadre si cette option est choisie). Cette recherche utilise *InternalMatchedSegmentFinder* pour

- filtrer les segments uniques (élimination des segments déjà en partage)
- création d'un index spatial de ces segments uniques
- récupération des segments uniques s'appariant avec un autre segment non identique

Identification des objets incluant des segments appariés.

2 - Création d'un jeu de segments appariés (Set) et des coordonnées les définissant.

Crée un *SegmentIndex* (appelé *matchedSegmentIndex* dans le code) contenant tous les FeatureSegment appariés mais non identiques.

Crée un Set contenant toutes leurs coordonnées (*matchedSegmentCoordSet* dans le code)

3 - SetAdjustableCoordinates (attribut *matchedSegmentCoordSet* de l'objet Coverage)

Associe le jeu de coordonnées appartenant à un segment apparié à l'objet global Coverage.

4 - Récupérer tous les objets impliqués (grâce à *matchedSegmentCoordSet*)

Récupère tous les objets ayant un point commun avec *matchedSegmentCoordSet*. Remarquer que des objets sans aucun segment apparié peuvent être impliqués par l'ajustement des frontières si un de leur point est commun avec celui d'un segment apparié.

5 - Crée un nouveau jeu de données qui accueillera les géométries ajustées

6 - Ajuster tous les objets

6.1 - Ajustement des objets avec les objets limitrophes selon le processus suivant :

- Pour chaque objet candidat, chercher les voisins
 - Pour chaque voisin, utiliser *CoverageFeature#computeAdjustmentSingle* pour
 - appairer le contour externe avec le contour externe de l'objet voisin
 - appairer chaque contour interne avec le contour externe de l'objet voisin

Pour chaque paire de contours, comparer les segments deux à deux.

Eliminer les paires de segments quand l'un d'eux (ou les deux) n'appartiennent pas au jeu des segments appariés,

- tester si les segments s'apparient avec *SegmentMatcher#isMatch*
- s'ils sont appariés, les ajouter l'un à l'autre avec *Segment#addMatchedSegment*

6.2 - Then update features (*Coverage#computeAdjustedFeatureUpdates*)