Evaluating conflation methods using uncertainty modeling

Peter Doucette[*,1], John Dolloff[*], Roberto Canavosio-Zuzelski[*],
Michael Lenihan and Dennis Motsko

National Geospatial-Intelligence Agency, 7500 GEOINT Dr., Springfield, VA 22150
[*]Contractor

ABSTRACT

The classic problem of computer-assisted conflation involves the matching of individual *features* (e.g., point, polyline, or polygon vectors) as stored in a geographic information system (GIS), between two different sets (*layers*) of features. The classical goal of conflation is the transfer of feature metadata (*attributes*) from one layer to another. The age of free public and open source geospatial feature data has significantly increased the opportunity to conflate such data to create enhanced products. There are currently several spatial conflation tools in the marketplace with varying degrees of automation. An ability to evaluate conflation tool performance quantitatively is of operational value, although manual truthing of matched features is laborious and costly. In this paper, we present a novel methodology that uses spatial uncertainty modeling to simulate realistic feature layers to streamline evaluation of feature matching performance for conflation methods. Performance results are compiled for DCGIS street centerline features.

Keywords: conflation, GIS, vector data, uncertainty modeling, perturbation, feature matching, performance evaluation

1. INTRODUCTION

The basic definition of conflation, which is a form of fusion, is to integrate data sets such that the outcome is an enhanced version of the inputs individually, i.e. 'the whole is greater than the sum of its parts'. The end goals of conflation are the same as fusion, i.e., to facilitate methods of discovery, clarification, reinforcement, augmentation, or update. Automating parts of the conflation process is a desired goal, as is the ability to evaluate automation performance from conflation methods in the marketplace. In this paper, we present a novel methodology by which to dramatically streamline the process for quantitative performance evaluation of conflation methods.

1.1. Background

The problem of conflating spatial data dates back to circa 1980, and automation principles were proposed by Saalfeld[1]. The classic problem of computer-assisted conflation involved the matching of individual GIS *features* (e.g., point, line, or polygon vectors) of two disparate feature *layers* so that attribute information could be transferred from one layer to the other. The cause of spatial disparity (position and topology) between layers is often attributable to the fidelity, scale, technique of the collection method, and source data used, which contribute random and systematic error components. Methods of modeling spatial uncertainty for raster and object data types are discussed in the literature[2-6]. Some of these effects for object (i.e., feature vector) data are apparent in Fig. 1, which shows two coincident, but independently collected GIS layers of street centerline data. Modeling these effects is the thrust of this paper.

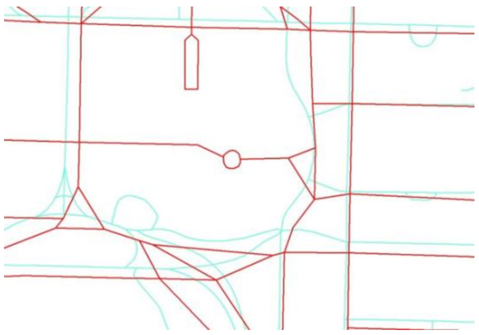
Fig. 1. Public data of street centerline features; TIGER (red) from US Census Bureau, and DCGIS (cyan) from the District of Columbia.

[1]peter.j.doucette.ctr@nga.mil
[Approved for public release, NGA case 13-213]

By the 2000s, the domain of spatial data conflation had broadened considerably to include cross-type feature matching, including feature-to-image alignment[7]. The age of open source and commodity geospatial feature data (e.g., OpenStreetMap, Wikimapia, and Google Maps) has significantly increased the opportunity to leverage these emerging data layers via conflation with standardized layers. There are currently several spatial conflation tools in the marketplace with varying automation capabilities[8].

Although conflation scenarios can vary widely, basic steps of an automated conflation process include 1) determine matching (correspondence) features between input data sets; 2) use the matched features to solve for the parameters of a pre-determined transfer function; 3) transfer and/or combine information (physically or via relational links) between input layers; and 4) edit the errors as needed.

A conflation process is closely related to that of image registration. In the case of the latter a relatively small set of correspondence features (usually points) is used as the basis for solving transfer function parameters, which are then used to transfer coordinates for 'new' points. However, in a conflation process the correspondence features themselves (points, lines or polygons) may constitute the bulk (or all) of the features for which information transfer subsequently occurs, and possibly new features. From one perspective, it could be argued that image registration is a special case of conflation, although colloquial usage of these terms implies application to image versus vector data, respectively[9].

By convention, conflation inputs are typically divided into *reference* and *source* layers. The former is generally considered more spatially accurate (and hence often fixed), relative to the latter, but mixed variations may occur. In the classic conflation scenario, the assertion is that the reference layer is spatially preferred, but where source layer features contain unique attributes (metadata) relative to their reference counterparts. In this case, the goal of conflation is to first match source to reference features, then transfer attribute information from source to reference features. In practice, one-to-one correspondences seldom occur for all features. One-to-none, one-to-many, and many-to-many feature correspondences (and vice-versa) need to be determined, and attribution transfer handled accordingly.

1.2. Problem Statement and Approach

In order to evaluate the performance of automated conflation methods quantitatively, feature matching *truth* must first be established between reference and source input layers. When performed manually, feature match truthing is a tedious, error prone, and costly task. If a source layer could be derived (i.e., simulated) from the reference layer, feature match truth would be known by definition. Ideally, a simulated source layer would be a *realistic* representation for a source-reference input pairing for a conflation scenario of interest.

In this paper we propose a novel technique called *Perty*—short for "perturbator", by which spatial uncertainty modeling is used to simulate a realistic source layer from an existing reference layer. To model realistic discrepancies, reference feature coordinate geometry and topology need to be perturbed appropriately. To simulate coordinate geometry discrepancies, vector coordinates are randomly perturbed using a strictly positive definite correlation function (SPDFC) that is based upon inter-point distances to account for spatial autocorrelation. Feature topology discrepancies simulate one-to-one, one-to-none, and one-to-many feature matching scenarios. Since any number of randomly simulated source feature sets can be derived from the reference set, a virtually unlimited number of source-reference input pairings with known feature match truth can be generated for match performance evaluation of conflation tools.

In this study, only feature matching performance is evaluated. While feature matching does not represent a measure of spatial alignment, or operational time savings per se, it is the primary criterion upon which subsequent conflation processing steps are based. That is, operational time saving is affected according to some functional relationship with match accuracy. Although this relationship may be impractical to model precisely from match accuracy alone, the simple assumption that one will measurably affect the other is sufficient for the purposes of this study.

It should be noted that differences between source and reference inputs often manifest spatial or topological correlation according to procedures used to create vector data. For example, errors created from base images for feature extraction (e.g., warping and mosaicking effects), scale differences at which features are extracted, the skill level of the extractor, the quality of a hardcopy map scanning process, the quality of GPS-derived coordinates, etc., may present manifestations of spatial correlation in different features within and between layers. Consider the manual extraction of road features in which intersections are likely to be digitized with more care versus interior (shape) points due to the perception of a relative greater importance of intersection topology. To the extent that such phenomena can be understood, *Perty* provides a powerful mechanism by which to model them for the purposes of simulating realistic feature layers for Monte Carlo type analyses.

In this paper, the *Perty* technique is demonstrated concurrently across four different vector-to-vector conflation tools. Only *polyline road* features are considered in order to constrain the scope of the study, although the technique is easily extendable to points or polygon features. The approach of this study was to simulate random and spatially correlated source feature sets as a point of departure to investigate generic feature matching capabilities of conflation tools. Attempts to model the actual correlation that exists between any two real vector data sets was beyond the scope of this study. However, this knowledge would be of considerable value for future study. Recognizing that a given conflation tool may be intended for application toward particular scenarios, e.g., mosaicking effects, such a tool may be predisposed to underperform relative to other tools under the generic modeling scenario proposed herein. Therefore, comparative analysis among different tools tested in study is intended for demonstrational purposes of the perturbation model, rather than critiquing the different tools based upon their performance.

2. SIMULATING A SOURCE LAYER FROM A REFERENCE LAYER

We propose a novel technique called *Perty*—short for "perturbator", in which an existing (reference) layer of features is copied and then perturbed to simulate a realistic source layer. Since the source is derived from the reference, feature match truth between reference and source is completely known. The reference-source pair can then be input to a conflation tool to measure its feature matching accuracy. The process can be iterated for *N* simulations over multiple conflation tools to facilitate numerical analyses as desired. The *Perty* technique is divided into perturbative processes for point coordinates (section 2.1), and topology, i.e., the connective relationships between features (section 2.2).

2.1. A Method to Perturb 2D Spatial Coordinates for Polylines

A polyline feature is defined as a sequence of connected lines (as from a digitizing process), for which the start and end points of the assembly are referred to as *nodes*. Any feature point that is not a node is a *vertex*, a.k.a. shape point. The baseline perturbation algorithm presents a method to simulate 2D positional errors to an existing set of 2D point locations. In this context, an *error* is defined as a specific type of uncertainty that can be measured relative to a known location[3], i.e., *truth*. The errors are modeled statistically as a sum of individual random errors (per point) with systematic errors corresponding to a homogeneous random field that are correlated as a function of distance(s) between the points. The model assumes that the random field is stationary in first and second order statistics, and isotropic with a decaying correlation function[10]. Also, *x*-errors are assumed uncorrelated with *y*-errors.

2.1.1. Baseline Algorithm

Point coordinates are the base input to the base algorithm. For polyline features, the nodes and vertices that compose them are input. When spatially perturbing nodes of intersection, node topology must be determined to ensure consistent perturbation among lines that share nodes. That is, any shared node should be perturbed only once. 2D point coordinates are assumed in the description that follows, although the method is extendable to 3D coordinates.

The base algorithm generates *nxn* covariance matrices for random and systematic error, where *n* is the number of input points. There are practical computational limits on the size of *n*. When *n* >> 1000, which is expected for non-trivial GIS polyline layers, a practical alternative is to perturb a representative point grid for the input space, and interpolate for the actual feature coordinates. Perturbation grids with bilinear interpolation were implemented as a standard part of the baseline algorithm.

We define total error as the sum of systematic and random error. Random error implies that it is uncorrelated, and systematic error is more general than bias, although can be considered bias-like over "short" distances. The use of a spatial SPDCF in our approach to simulate systematic errors is more robust and realistic versus simulating errors as a simple sum of a bias and random errors. The following 10 steps provide a detailed description of the baseline algorithm.

1) Assume *n* unique 2D point locations in any arbitrary order.
2) Assume *n*x*n* random error covariance matrices for *x* and *y* equal to,

$$R_x = \begin{bmatrix} \sigma_{rx_1}^2 & 0 & 0 & 0 \\ . & \sigma_{rx_2}^2 & 0 & 0 \\ . & . & \ddots & 0 \\ . & . & . & \sigma_{rx_n}^2 \end{bmatrix}, \quad R_y = \begin{bmatrix} \sigma_{ry_1}^2 & 0 & 0 & 0 \\ . & \sigma_{ry_2}^2 & 0 & 0 \\ . & . & \ddots & 0 \\ . & . & . & \sigma_{ry_n}^2 \end{bmatrix} \tag{1}$$

3) Assume *n*x*n* systematic error covariance matrices for *n* points in *x* and *y* equal to,

$$S_x = \begin{bmatrix} \sigma_{sx_1}^2 & \rho_{x_1x_2}\sigma_{sx_1}\sigma_{sx_2} & \cdots & \rho_{x_1x_n}\sigma_{sx_1}\sigma_{sx_n} \\ . & \sigma_{sx_2}^2 & \cdots & \rho_{x_2x_n}\sigma_{sx_2}\sigma_{sx_n} \\ . & . & \ddots & \vdots \\ . & . & . & \sigma_{sx_n}^2 \end{bmatrix}, \quad S_y = \begin{bmatrix} \sigma_{sy_1}^2 & \rho_{y_1y_2}\sigma_{sy_1}\sigma_{sy_2} & \cdots & \rho_{y_1y_n}\sigma_{sy_1}\sigma_{sy_n} \\ . & \sigma_{sy_2}^2 & \cdots & \rho_{y_2y_n}\sigma_{sy_2}\sigma_{sy_n} \\ . & . & \ddots & \vdots \\ . & . & . & \sigma_{sy_n}^2 \end{bmatrix} \tag{2}$$

4) Assume a strictly positive definite correlation function (SPDCF) between points *i* and *j* that is common to all point components. A SPDCF ensures the generation of a valid covariance matrix with positive eigenvalues[10-11]. For example, an exponential SPDCF is shown in eq. 3,

$$\rho_{ij} = e^{-(d_{ij}/D)} \tag{3}$$

where d_{ij} is the horizontal distance between any points *i* and *j*, and *D* is a horizontal distance constant.

5) Calculate covariance matrices R_x, R_y, S_x, S_y in eqs 1-2, by assigning desired perturbation values for σ_{rx}, σ_{ry}, σ_{sx}, σ_{sy}. For simplicity, assume a constant value for each sigma, e.g., if $\sigma_{rx} = 10$, then $\sigma_{rx_i} = 10$ for *i* = 1, 2, …*n*. Calculate ρ_{ij} for each point pair using a desired value for *D* in eq. 3, and then plug into matrices in eq. 2.

6) Defined *n*x1 (*x*,*y*)-error vectors corresponding to the *n* points:

$$\epsilon_x = [\epsilon_{x_1} \quad \epsilon_{x_2} \quad \cdots \quad \epsilon_{x_n}]^T \quad \text{and} \quad \epsilon_y = [\epsilon_{y_1} \quad \epsilon_{y_2} \quad \cdots \quad \epsilon_{y_n}]^T. \tag{4}$$

7) Define corresponding *nxn* (symmetric) error covariance matrices P_x and P_y as,

$$\begin{aligned} P_x &= E\{\epsilon_x {\epsilon_x}^T\} = R_x + S_x \\ P_y &= E\{\epsilon_y {\epsilon_y}^T\} = R_y + S_y. \end{aligned} \tag{5}$$

8) Generate a realization of the corresponding *n* (*x*, *y*)-errors as,

$$\epsilon_x = P_x^{1/2}\boldsymbol{a}, \quad \text{and} \quad \epsilon_y = P_y^{1/2}\boldsymbol{b} \tag{6}$$

where each instance of $\boldsymbol{a}$ and $\boldsymbol{b}$ is a *n*x1 vector realization of *n* independent distributed random variables, e.g., distributed as *N*(0,1), *U*(-1,1), or as desired. $P^{1/2}$ is the *nxn* principal matrix square root of P.

9) Join the *x*-errors and *y*-errors for the *n* points in a 2*n*x1 vector as,

$$\epsilon_X = \left[\epsilon_{x_1} \quad \epsilon_{y_1} \quad \epsilon_{x_2} \quad \epsilon_{y_2} \quad \cdots \quad \epsilon_{x_n} \quad \epsilon_{y_n}\right]^T \tag{7}$$

10) To generate the source point coordinates, add the errors to the reference point coordinates for $i = 1, 2, 3, \dots, n$, as,

$$x_{(source)_i} = x_{(reference)_i} + \epsilon_{x_i} \quad \text{and} \quad y_{(source)_i} = y_{(reference)_i} + \epsilon_{y_i} \tag{8}$$

2.1.2. Demonstration with Polyline Features

Implementation of an appropriate correlation function in step 4 above is central to modeling realistic spatial dependence (autocorrelation) when perturbing polyline nodes and vertices. That is, nodes and vertices that are closer together should be perturbed with greater correlation (i.e., similar direction and magnitude) than those further apart. Also, topological integrity must be maintained following perturbation[3], i.e., intersections among lines that did not exist prior to perturbation must not result.

For all testing cases in this study, the SPDCF in eq. 9, referred to as the CSM (community sensor model) function[9,11], was used. This function allows for flexible parametric adjustment of the resulting shape.

$$\rho(\Delta d) \; = \; A\left[\alpha + \frac{(1+\beta)}{\beta + e^{\Delta d/D}}\right], \text{ where } 0 < A \leq 1; 0 \leq \alpha < 1; 0 < D; 0 \leq \beta \leq 10. \tag{9}$$

As a simplification, assumptions of fixing A = 1 and α = 0 are made, leaving,

$$\rho(\Delta d) \; = \; \frac{(1+\beta)}{\beta + e^{\Delta d/D}} \tag{10}$$

where Δd is the horizontal distance between two given points, and β and D are the two remaining CSM parameters. Modification of β or D generates a family curves. An example is shown in Fig. 2 [left] for $\beta = 9$ and D = 100m (blue curve). The abscissa represents horizontal distances Δd (e.g., meters) between points, and the ordinate is the correlation $\rho(\Delta d)$ for a given distance, e.g., ρ(100m) = 0.85, ρ(200m) = 0.61, …, ρ(800m) ≅ 0. The result of modifying D by ±D/2 is shown by the dotted curves flanking the blue curve. Also, as $\beta \rightarrow 0$ the curve approaches a classic exponential shape, and as $\beta \rightarrow 10$, becomes more Gaussian-like in appearance. Fig. 2 [right] shows isotropic surfaces in x and y for β = 9 [top] and β = 1 [bottom] as a function of correlation. It is possible to generate anisotropic surfaces with this technique using separable correlation functions[11] in x and y, though not performed in this study.

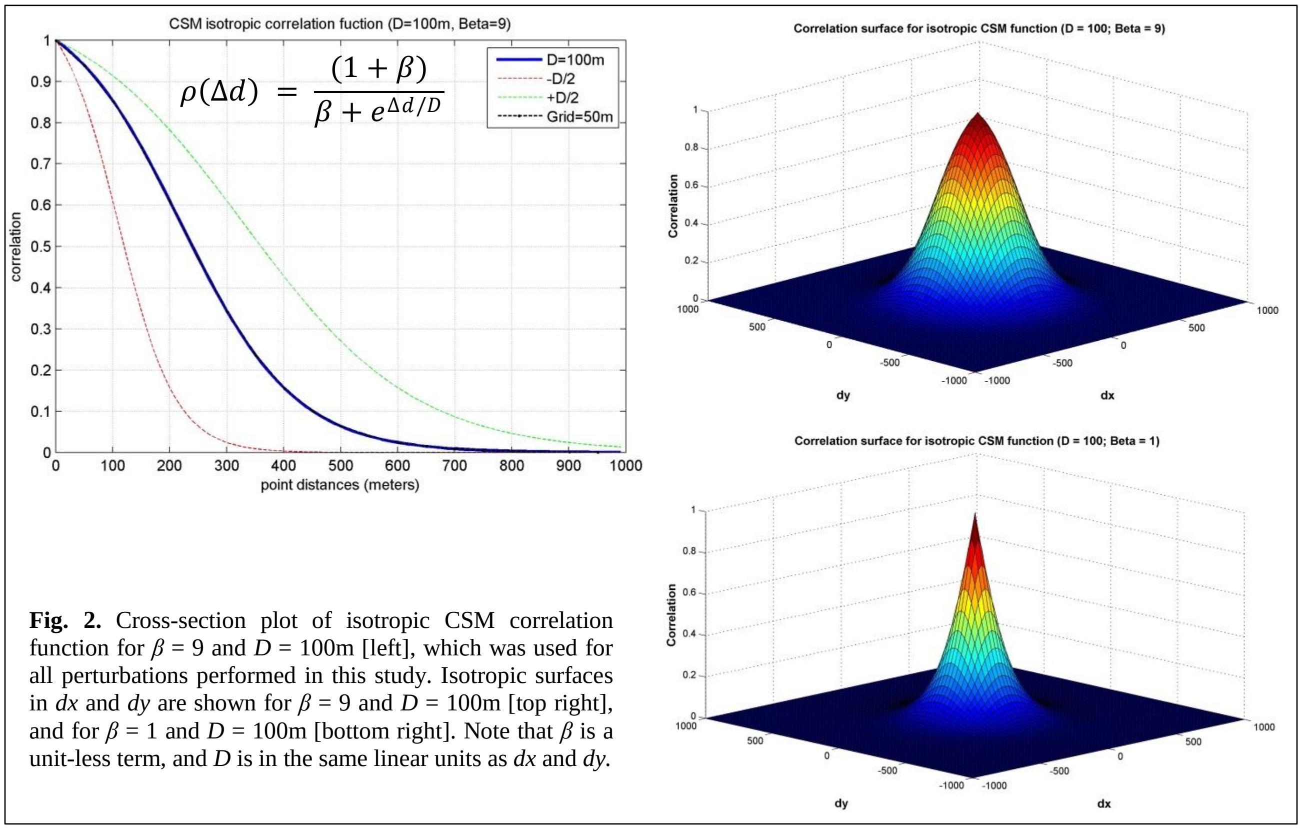

Fig. 2. Cross-section plot of isotropic CSM correlation function for β = 9 and D = 100m [left], which was used for all perturbations performed in this study. Isotropic surfaces in *dx* and *dy* are shown for β = 9 and D = 100m [top right], and for β = 1 and D = 100m [bottom right]. Note that β is a unit-less term, and D is in the same linear units as *dx* and *dy*.

A demonstration of polyline coordinate perturbation is shown in Fig. 3 for DCGIS street centerline features at two different scale views. Using the baseline algorithm from section 2.1.1, the CSM function of eq. 10 is used in step 4 for D = 100m and β = 9, which yields correlations of ρ = 0.85 at distances of 100m, ρ = 0.61 at distances of 200m, …, and $\rho \cong 0$ at distances of 800m. For simplicity, perturbation values in step 5 were set as $\sigma_{rx} = \sigma_{ry}$= 0m (random error), and $\sigma_{sx} = \sigma_{sy}$ = 50m (systematic error), using a uniform distribution U(-1 1) in step 8. The left panes show the perturbation field with a quiver plot. Black arrows show how the point grid (100m spacing) is perturbed, and red arrows show the interpolated perturbations for the individual vertices and nodes that compose the street features. The right panes show the corresponding coordinate perturbations to the actual street features (reference=blue, and perturbed=red).

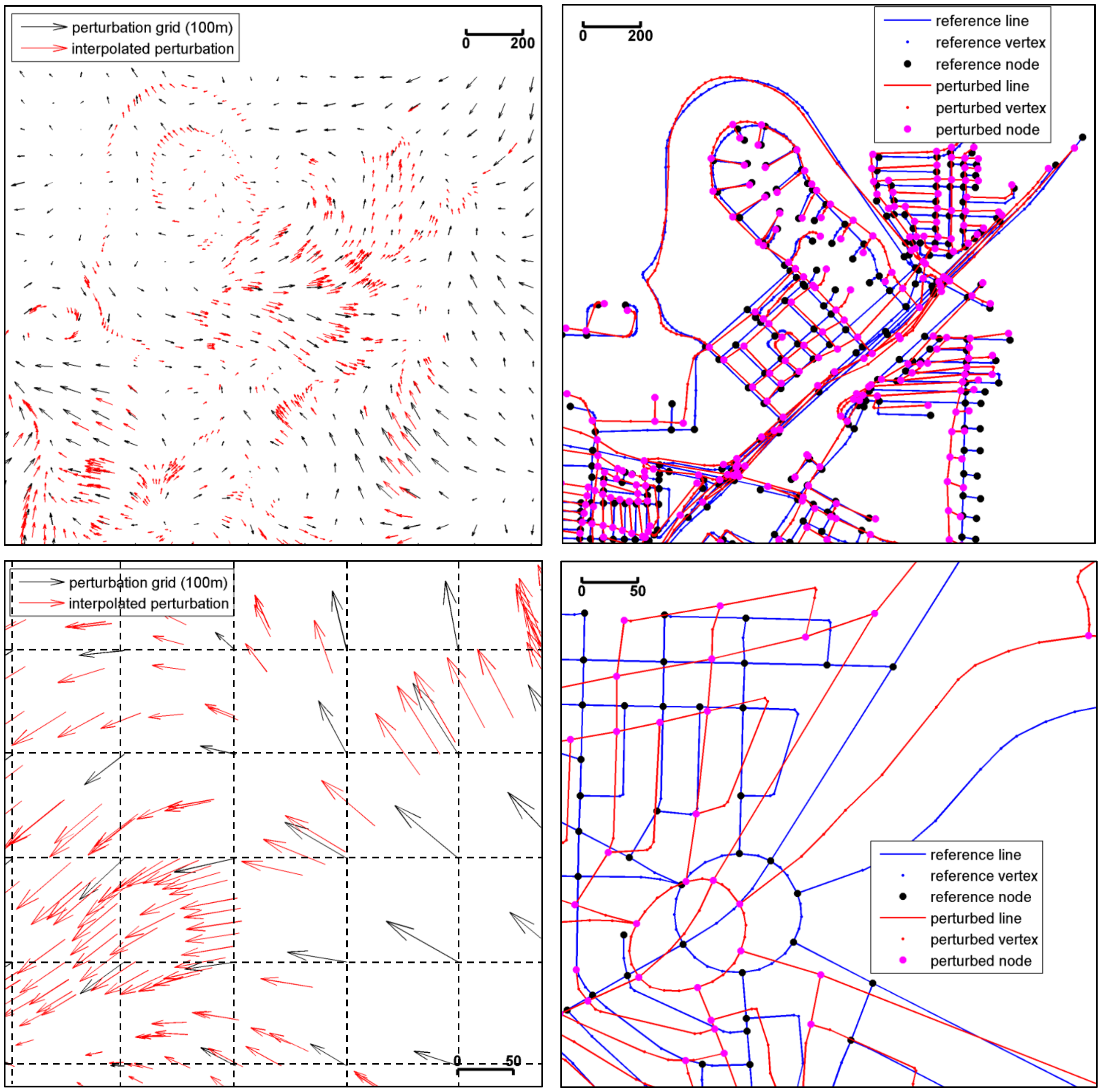

Fig. 3. Demonstration of coordinate perturbation for street centerline features (from DCGIS public data dcgis.dc.gov). The left panes show the perturbation field with a quiver plot at two different scale views [top and bottom]. The right panes show the corresponding coordinate perturbations to the actual street features, where blue are the reference, and red the perturbed. (Note: scale bars in meters.)

There are several techniques to further generalize this approach beyond the demonstration above that include[9-11], 1) different random fields for the vertical versus horizontal coordinates, 2) other correlation functional forms for coordinate perturbation, as well as feature attributes and topology, 3) separable correlation functions versus isotropic, and 4) non-stationary random fields. Altogether, these options offer considerable flexibility to generate perturbations that could be used to model a wide range of scenarios between reference and source feature layers.

2.2. Topology Perturbations for Polylines

As with coordinate perturbations, the goal of perturbing topology is to simulate *realistic* uncertainty between reference and source layers. In the absence of uncertainty in topology, every feature has a unique matching reference feature. Since the *one-to-one* scenario for *all* features is not realistic in practice, topology should also be perturbed.

Topology perturbations in this study are performed to source features only. Reference feature geometry is unaltered for simplicity of the study. Topology simulations considered include: 1) the *one-to-none* scenario, i.e., features exist in the reference layer, but not in source; 2) the *one-to-many* scenario, i.e., certain features in the reference layer can have correspondence with to two or more combined features in the source layer; and 3) *generalization*, i.e., source features are represented by a reduced number of vertices using the Douglas-Pueker simplification method[12]. Since generalization is used in this study to remove vertices only, it does not perturb node-based topology. However, it does alter the shape of source features, as well as reducing the number (n) of 2D point coordinates to perturb (from section 2.1).

2.2.1. The One-to-None Scenario

To simulate one-to-none scenarios, a desired percentage of features are randomly removed (pruned) from the source layer. Pruning can be performed according to desired correlation criteria, e.g., specific attributes, or specific topology, i.e., only those features containing m-connected nodes, where m could be any combination of 1 or more connections. For example, 30% of the street features in Fig. 4 [left pane] have been randomly pruned (dotted red) with no other criteria.

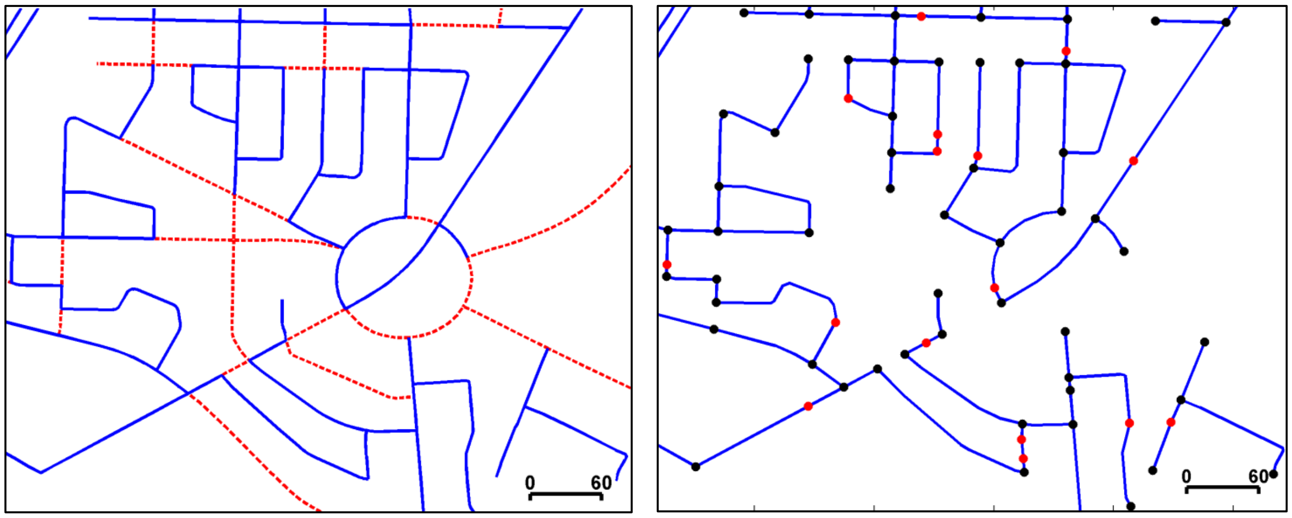

Fig. 4. [left pane] Random feature pruning (dotted red lines) to simulate one-to-none scenarios; [right pane] random recursive feature splitting (by red nodes) to simulate one-to-many scenarios. (DCGIS public street centerline data from dcgis.dc.gov)

2.2.2. The One-to-Many Scenario

To simulate one-to-many scenarios, a single feature is randomly selected and split into multiple features by random insertion of new node(s). The feature pruned results from Fig. 4 [left pane] are subsequently split (by red nodes) in Fig. 4 [right pane]. Feature splitting is performed in one of two ways: 1) if vertices exist, one is randomly selected to convert into a node; and 2) if no vertices exist, a coordinate for a new node on the line between existing (black) nodes is randomly selected. In either case, a minimum node spacing parameter is specified as desired. Random feature splitting is performed recursively as often as desired such that any given feature could be split multiple times.

As with pruning, feature splitting can be performed according to desired correlation criteria, e.g., specific attributes and/or topologies. For simplicity, child features created from splitting inherit the attributes from their parent features. A more sophisticated simulation could perturb attributes of the child features relative to their parent, but was beyond the scope of this study.

2.3. Measurement of Feature Match Accuracy

A feature layer represents a *set* of features, which are further composed of heterogeneous data, e.g., coordinate geometry and attributes. The process of matching features as *entities* (i.e., each with a unique database identifier) can be defined as performing an intersection between feature sets A and B as,

$$A \cap B = C \tag{11}$$

where,

$$A = \{feature\_a_1, feature\_ a_2,..., feature\_ a_m\}$$
$$B = \{feature\_b_1, feature\_ b_2,..., feature\_ b_n\},$$
$$C = \{entity\_c_1, entity\_ c_2,..., entity\_ c_p\}$$

and C represents the feature ID set of those matched. Although the features of A and B may manifest differently in their respective database containers, an intersected (i.e., matched) feature is defined as a unique entity to all intersected features. For example, $\{feature\_a_1\}$ may represent polygon geometry for a building, and $\{feature\_b_1\}$ may be composed of a single point coordinate. But if they represent the same entity (i.e., 'building'), then an entity exists as,

$$\{feature\_a_1\} \cap \{feature\_b_1\} = entity\_c_1 \tag{12}$$

where,

$$\{feature\_a_1, feature\_b_1\} \in entity\_c_1.$$

Eqs. (11-12) apply in cases of one-to-one, one-to-many, or many-to-many matching scenarios. In one-to-one, or one-to-none scenarios, a feature is equivalent to an entity; but this is not the case for a one-to-many scenario. For example, consider the following one-to-many scenario,

$$A = \{feature\_a_1, feature\_ a_2, feature\_ a_3\} \tag{13}$$
$$B = \{feature\_b_1\} \tag{14}$$

where set A represents three rooms of a building, and B represents the entire building. Then,

$$A \cap B = C = \{entity\_c_1\} \tag{15}$$

where,

$$\{feature\_a_1, feature\_ a_2, feature\_ a_3, feature\_ b_1\} \in entity\_c_1.$$

The method to measure match accuracy is to first establish the reference (truth) match set, C_{ref}, for the intersection of any two given sets A and B as defined in eq. (16). An estimated match set, C_{est}, (e.g., as output from an automated tool), represents the estimated intersection, denoted with parentheses $(\cap)$, between sets A and B as defined in eq. (17).

$$A \cap B = C_{ref} \tag{16}$$

$$A \ (\cap) \ B = C_{est} \tag{17}$$

The matching accuracy is then determined from eq. (18), where the number of intersected elements between C_{ref} and C_{est} is counted (denoted by set cardinality $| \ |$), and divided by the number of elements in C_{ref}. This ratio between 0 and 1 provides the entity match accuracy, M, where 1 means 100% accurate.

$$M = | C_{est} \cap C_{ref} | \ / \ | C_{ref} | \tag{18}$$

3. EXPERIMENTS

Perty was used to simulate realistic reference-source layer pairs to evaluate the automated feature ID matching performance only of four conflation tools in the marketplace. Automated feature matching accuracy is measured as a function of coordinate and topology perturbation. All testing was performed with DCGIS street centerline public data.

3.1. Feature Match Accuracy Performance Curves

Feature match accuracy M is measured as a function of coordinate perturbation in x and y. If the nx1 vector realizations $\boldsymbol{a}$ and $\boldsymbol{b}$ in eq. 6 are normally distributed as $N(0,1)$, then M is plotted versus σ_{sx} and σ_{sy}, assuming $\sigma_{sx} = \sigma_{sy}$ for simplicity. But if $\boldsymbol{a}$ and $\boldsymbol{b}$ are uniformly distributed as $U(-1\ 1)$, then M is plotted versus $max(|\epsilon_X|)$ from eq. 7. All tests were performed using $U(-1\ 1)$ to provide hard bounds for simplicity. The perturbation range used was 5m to 50m in x and y, at 5m increments. Thus, at 50m the maximum possible horizontal perturbation is $H_{max} = [(50)^2 + (50^2)]^{1/2} = 70.7$m.

The *search distance*, R_{search}, is a fundamental parameter specification common to feature matching algorithms used in the conflation tools tested. It generally specifies the horizontal radius within which each conflation tool attempts to find matching features. To eliminate the possibility for out-of-range errors for all conflation tools tested, R_{search} was set to 75m to ensure $H_{max} < R_{search}$.

3.1.1. Varying Coordinate Maximum versus Fixed Topology Configurations

Fig. 5 shows 6 feature matching performance curves for 4 conflation tools tested. The upper 4 graphs plot the match accuracy versus increasing the coordinate perturbation from a uniform distribution. In these cases, topology perturbations were held fixed in quantity, but allowed to randomly vary in which features are removed or split for each simulation. The tool legend indicates the mean CPU computation time required per simulation per tool. Data points are plotted for each tool at 5m increments, which represent the average match accuracy ${}_iM_{ave}$ for the number of simulations S_i performed for the i-th perturbation increment, for $i = 1, 2, \ldots, 10$. ${}_iM_{ave}$ is computed to smooth out the effects from outlier simulation results. The total number of simulations S_T per tool is then given by S_i x 10, e.g., S_T = 3x10 = 30 simulations per tool for the upper left graph in Fig. 5. Each simulation represents execution of one automated conflation tool. Therefore, S_T = 30 over 4 tools means 120 independent conflation processes.

The secondary legend in each graph provides the following information:

- Input shapefile name and (total number of reference features)
- Search Distance: R_{search} used for all tools.
- (PruneFrac): the fraction (0 to 0.99) of features removed in the perturbed data to simulate one-to-none scenarios
- (FeatSplits): the number of features split (divining one into two) in the perturbed data to simulate one-to-many scenarios (splits can be recursive)
- CSM correlation function parameters: *D* and *Beta* (to adjust shape), and interpolation grid spacing *G*.

Graph 1 [upper left] shows results for one-to-one only matching, which generally represents the simplest scenario since topology is identical between reference and source layers. Although this situation is unlikely with real data, it is used to establish a performance baseline. Graph 2 [upper right] shows results for a random 50% removal of features to simulate one-to-none scenarios, indicating a significant matching performance drop. Graph 3 [middle left] shows results for 150 split features to simulate one-to-many scenarios, indicating a similar drop. Graph 4 [middle right] shows a combination of one-to-none (25%) and one-to-many (150) scenarios, and the most significant performance drop in general (note Tool_3 was unable to perform in this mode).

3.1.2. Varying Topology versus Fixed Coordinate Maximum Configurations

Graph 5 [lower left] in Fig. 5 shows performance results as a function of increasing the number of features removed from 0 to 90% from the source layer, while holding the coordinate perturbation maximum fixed to $max(|\epsilon_X|)$ = 25m. Match accuracy bottoms out for 3 of 4 tools between 30-50%, but increases with higher rates as there is decreasing opportunity for mis-matching. Graph 6 [lower right] shows that increasing the number of split features from 0 to 900 has relatively little effect on tools 1 and 4 for a fixed $max(|\epsilon_X|)$ = 25m. (Tool_3 had stability issues with FeatSplits > 200).

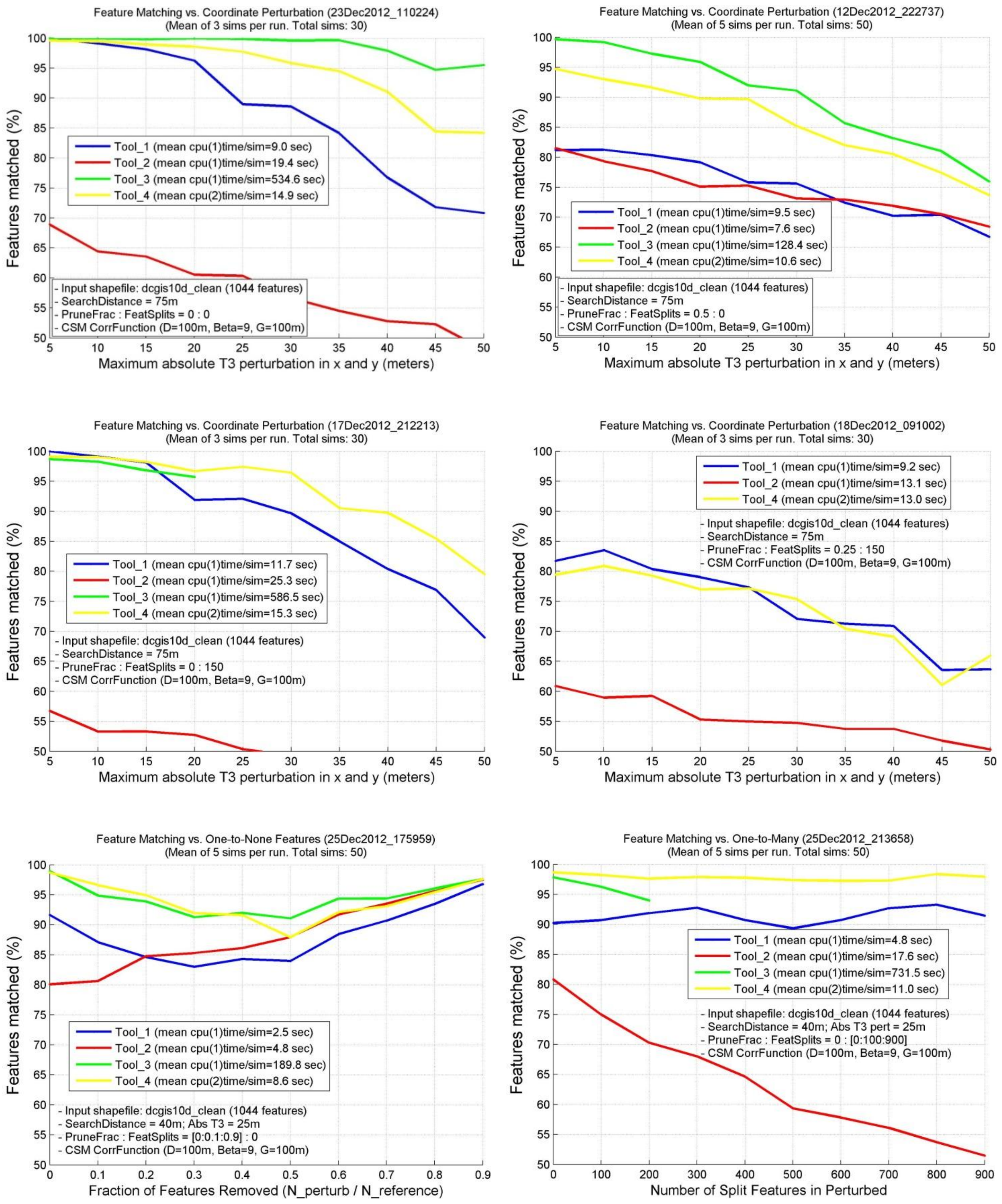

Fig. 5. Automated feature matching performance curves for 4 conflation tools per graph. Upper 4 graphs show matching as a function of increasing coordinate perturbation with fixed topology configurations. Lower 2 graphs show matching as a function of increasing topology perturbation with fixed coordinate perturbation.

3.2. Heat Map Generation for Identifying Problem Features for Matching

Another utility of *Perty* is the generation of a simulation heat map, in which feature matching accumulations are spatially visualized. Fig. 6 demonstrates this concept for each of the four conflation tools tested, and which accompany the results from Fig.5, graph 1. Features that were matched in all 30 simulations are colored black, and others color coded according to the match error rate shown in the color to the right. Heat maps may provide an effective method by which to spatially identify and predict problem features when performing conflation with real data sests. Interestingly, complementary matching performance is at times apparent among the different tools tested.

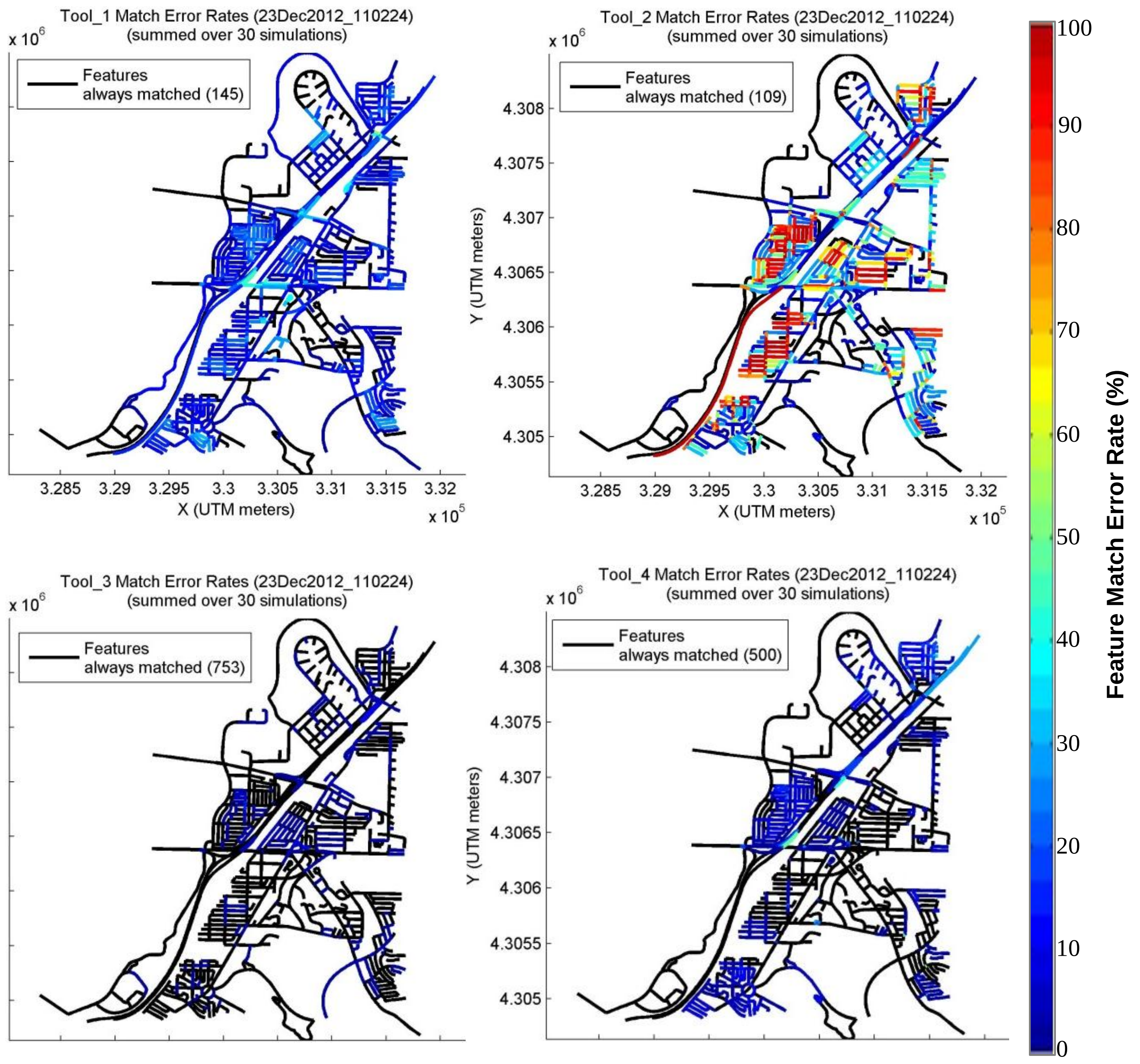
Tool_1 Match Error Rates (23Dec2012_110224)
(summed over 30 simulations)
Features always matched (145)
Tool_2 Match Error Rates (23Dec2012_110224)
(summed over 30 simulations)
Features always matched (109)
Tool_3 Match Error Rates (23Dec2012_110224)
(summed over 30 simulations)
Features always matched (753)
Tool_4 Match Error Rates (23Dec2012_110224)
(summed over 30 simulations)
Features always matched (500)
x 10^6
4.308
4.3075
4.307
4.3065
4.306
4.3055
4.305
Y (UTM meters)
3.285
3.29
3.295
3.3
3.305
3.31
3.315
3.32
X (UTM meters)
x 10^5
Feature Match Error Rate (%)
100
90
80
70
60
50
40
30
20
10
0

Fig. 6. Perty heat maps for conflation tools 1 through 4 (labeled in each sub-graph) that show the accumulated match errors spatially for the corresponding plotted curves in Fig. 5 graph 1 (upper left). Street centerline features colored black are correctly matched in all 30 simulations. Others are color coded according to accumulated match errors. The spatial coordinate system is Universal Transverse Mercator (UTM) in units of meters. Heat maps can provide an effective method by which to spatially identify and predict problem features for matching when performing conflation with real data. (DCGIS public street centerline data)

Fig. 7 shows *Perty* response curves and corresponding heat maps for conflation tools 1, 2, and 4. Results are shown for 30 simulations with one-to-one matching, using ~10X more street centerline features than before. The response curves are very similar to those in Fig. 5 graph 1 (~1k feature set), which demonstrates strong stationarity of the perturbation model, although D = 200m in this case to keep the nxn covariance matrices wieldy. An interesting outcome among the heat maps reveals remarkable consistency with which the higher match errors tend to cluster spatially across all 3 tools.

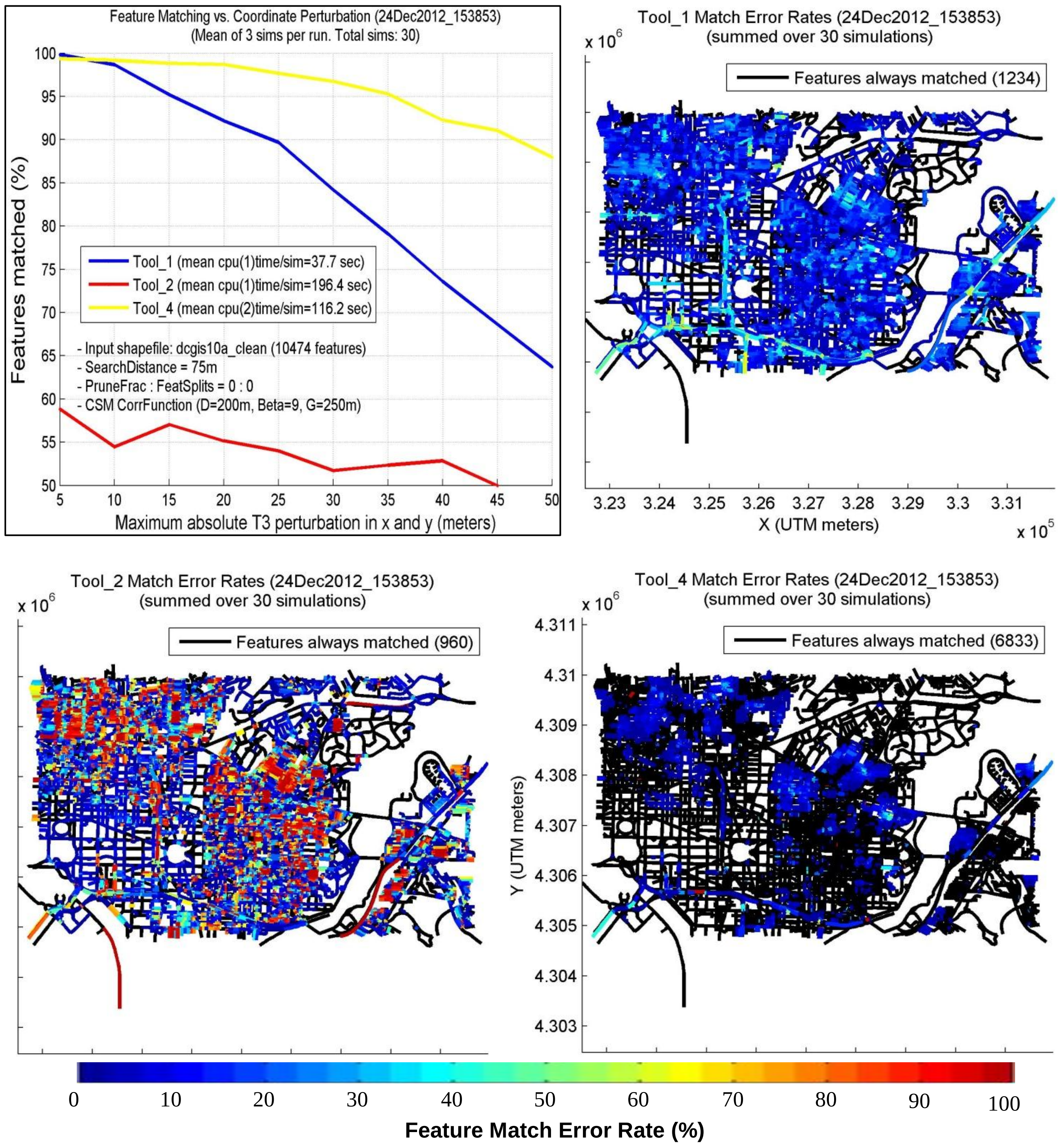

Fig. 7. *Perty* response curves and corresponding heat maps for conflation tools 1, 2, and 4. The results shown are for 30 simulations of 10,474 features. Note strong cross-spatial relationships of error clusters. (DCGIS public street centerline data)

4. CONCLUSIONS

The age of free public and open source geospatial feature data has significantly increased the opportunity to conflate such data to create enhanced products. There are currently several spatial conflation tools in the marketplace with varying degrees of automation. An ability to evaluate conflation tool performance quantitatively is of operational value, although manual truthing of matched features is laborious and costly. In this paper, we presented a novel methodology called *Perty* that uses spatial uncertainty modeling for coordinate and topology perturbations to simulate realistic feature layers from which to conflate. Since the source layer is derived from a reference layer, feature match truth between reference and source is completely known. The reference-source pair can then be input to a conflation tool to measure its feature matching accuracy. Although automated feature matching by itself does not represent a complete measure of overall conflation per se, it can serve as a useful performance predictor. This process can be iterated for simulations over multiple conflation tools to facilitate powerful numerical analyses as demonstrated for evaluating matching performance.

Perty was demonstrated at measuring the automated feature matching accuracy of four vector-to-vector conflation tools in the marketplace. Urban 2D polyline street centerline features from DCGIS public data were considered to constrain the scope of the study. Results for test set sizes of 1,044 and 10,474 features were demonstrated. Although not demonstrated in this paper, *Perty* could be easily extended to points or polygon features, including 3D coordinates. The approach was to perturb coordinates and topology as stationary processes for first and second order statistics to investigate generic feature matching capabilities of conflation tools. Attempts to model correlation that exists between any two real vector data sets was beyond the scope of this study, but would be of value for future investigation.

The utility of a simulation heat map was demonstrated with *Perty*, in which accumulated feature matching errors could be spatially visualized. Heat maps may provide an effective method by which to spatially identify and predict problem features for matching when performing conflation with real data sets.

Vector encoded uncertainty information at a local spatial level could significantly enhance conflation algorithm performance matching[6]. Conflation algorithm developers should also consider incorporating methods to leverage this kind of information which may become more prevalent in the future.

Use of feature attributes (i.e., textual descriptions) to further facilitate automated matching is a commonly provided option in conflation tools. Although not investigated in this study, a perturbation methodology for attributes could be incorporated into the *Perty* concept.

ACKNOWLEDGEMENTS

The authors would like to express their appreciation to the following individuals for their efforts in supporting this research: Mr. Jason Surratt, Mr. Drew Bower, Mr. Thomas Hersey, Mr. Matthew Sorenson, and Mr. Adam Gurson.

REFERENCES

[1] Saalfeld, A., “Conflation: automated map compilation”, International Journal of Geographical Information Systems, 2(3): 217-228 (1988).
[2] Heuvelink, G., [Error propagation in environmental modelling with GIS], Taylor & Francis, London, UK (1998).
[3] Zhang, J., and Goodchild, M., [Uncertainty in geographical information], Taylor & Francis, Inc., New York, NY (2002).
[4] Foody, G., and Atkinson, P., (eds.) [Uncertainty in remote sensing and GIS], John Wiley & Sons Ltd, W. Sussex, England (2002).
[5] Shi, W., [Principles of modeling uncertainties in spatial data and spatial analyses], Taylor & Francis Group/CRC Press, London and New York (2010).
[6] Doucette, P., Motsko, D., Sorenson, M., White, D., “Uncertainty handling in geospatial data”, Proc. of SPIE, vol. 8396 (2012).

[7] Chen, C., Knoblock, C., “Geospatial data conflation”, In: Shekhar, S., Xiong, H. (eds.) Encyclopedia of GIS, SpringerScience+Business Media, LLC (2008).
[8] Rosettex Technology & Ventures Group, and Swiftsure Spatial Systems, Inc., “Conflation services in an R&D fusion environment”, Report Number: TR-001-085-022309-527-PR, 70p (2009).
[9] Doucette, P., Theiss, H., Marshall, J., Dolloff, J., Lenihan, M., Mikhail, E., Johanesen, T., “Chapter 11: Measurement and Automation Practices in Photogrammetry”, in Manual of Photogrammetry, 6th ed. (McGlone, J.C., ed.), ASPRS (2013).
[10] Dolloff, J., Lofy, B., Sussman, A., Taylor, C., “Strictly positive definite correlation functions”, Proc. of SPIE, vol. 6235 (2006).
[11] Dolloff, J., “The full multi-state vector error covariance matrix: Why needed and its practical representation”, Proc. of SPIE, vol. 8747 (2013).
[12] Douglas, D. and Peucker, T., “Algorithms for the reduction of the number of points required to represent a digitized line or its caricature”, The Canadian Cartographer, 10(2), 112–122 (1973).