GeoInformatica (2006) 10: 495–530
DOI: 10.1007/s10707-006-0344-6

Automatically Conflating Road Vector Data with Orthoimagery

Ching-Chien Chen · Craig A. Knoblock · Cyrus Shahabi

Received: 16 December 2004 / Revised: 10 March 2006 /
Accepted: 29 March 2006

Abstract Recent growth of the geospatial information on the web has made it possible to easily access a wide variety of spatial data. The ability to combine various sets of geospatial data into a single composite dataset has been one of central issues of modern geographic information processing. By conflating diverse spatial datasets, one can support a rich set of queries that could have not been answered given any of these sets in isolation. However, automatically conflating geospatial data from different data sources remains a challenging task. This is because geospatial data obtained from various data sources may have different projections, different accuracy levels and different formats (e.g., raster or vector format), thus resulting in various positional inconsistencies. Most of the existing algorithms only deal with vector to vector data conflation or require human intervention to accomplish vector data to imagery conflation. In this paper, we describe a novel geospatial data fusion approach, named AMS-Conflation, which achieves automatic vector to imagery conflation. We describe an efficient technique to automatically generate control point pairs from the orthoimagery and vector data by exploiting the information from the vector data to perform localized image processing on the orthoimagery. We also evaluate a filtering technique to automatically eliminate inaccurate pairs from the generated control points. We show that these conflation techniques can automatically align the roads in orthoimagery, such that 75% of the conflated roads are within 3.6 meters from the real road axes compared to 35% for the original vector data for partial areas of the county of St. Louis, MO.

C.-C. Chen (✉)
Geosemble Technologies, 2041 Rosecrans Ave., Suite 245, El Segundo, CA 90245, USA
e-mail: jchen@geosemble.com

C. A. Knoblock · C. Shahabi
Department of Computer Science & Information Sciences Institute,
University of Southern California, Los Angeles, CA 90089-0781, USA

C. A. Knoblock
e-mail: knoblock@usc.edu

C. Shahabi
e-mail: shahabi@usc.edu

Keywords conflation · fusion · vector data · orthoimagery · template matching · rubber-sheeting

1 Introduction

With the rapid improvement of geospatial data collection techniques and the growth of Internet, large amount of geospatial data are now readily available on the web. The examples of well-known vector datasets are US Census TIGER/Line files[1] and NAVSTREETS from NAVTEQ.[2] The National Map,[3] Google Map,[4] and Microsoft TerraService[5] [3] are good examples of map or satellite imagery repositories. The users of these data products often want these geospatial data and other related data to be displayed in some integrated fashion for knowledge discovery. Instead of simply being able to display all of the related data in a single framework, we need to actually fuse the data to provide additional inferred information that is not contained in any single information source.

In fact, geospatial data fusion has been one of the central issues in GIS[6] [24]. Geospatial data fusion requires that the various datasets be integrated, and then a single composite dataset from the integrated elements be created. Towards geospatial data fusion, a vital step is reducing the spatial inconsistencies among multiple datasets. Figure 1 shows an example of combining a road network (NAVTEQ NAVSTREETS) and an image (geo-referenced USGS color imagery with 0.3 m/pixel resolution). Certain geospatial inconsistencies between the road network and imagery are noticeable (as shown in Figure 1(a)). An integrated view of the imagery with the aligned road network of the area (as Figure 1(b)) can annotate streets in the imagery with detailed attribution information often contained in vector dataset. In addition, recent advances in satellite imaging technology are making it possible to capture imagery with ever increasing precision and resolution (0.3 m/pixel or better). Once the road network is aligned to higher accuracy imagery, its relatively poorer positional accuracy can be improved.

One cannot rely on a manual approach to align diverse geospatial datasets, as the area of interest may be anywhere in the world and manually aligning a large region (e.g., the continental United States) is very time consuming and error-prone. Moreover, performing alignment offline on two geospatial datasets is also not a viable option in online GIS-related applications as both datasets may be obtained by querying different information sources at run-time. However, automatically and accurately aligning geospatial datasets is a difficult task. Essentially, the challenge is that various geospatial datasets may not align due to multiple reasons: they may use different spheroids, projections or coordinate systems; they may have been collected in different ways or with different precisions or resolutions, etc. If the geographic projections of both datasets are known, then both datasets can be converted to the same geographic projections. However, the geographic projection for a wide variety

[1] http://www.census.gov/geo/www/tiger/

[2] http://www.navteq.com/

[3] http://seamless.usgs.gov/

[4] http://maps.google.com/

[5] http://terraserver-usa.com/

[6] http://www.cobblestoneconcepts.com/ucgis2summer2002/researchagendafinal.htm

(a) Imagery with road network without alignment (Black lines: NAVTEQ NAVSTREETS roads)

(b) Imagery with road network with alignment (roads are annotated with vector information)

Figure 1 The vector and imagery integration.

of geospatial data online is not known. Furthermore, converting datasets into the same projection does not address the issue of different inaccuracies between two spatial datasets.

Conflation is often a term used to describe the integration or alignment of different geospatial datasets.[7] The conflation process can be divided into the following subtasks: (1) Feature matching: Find a set of conjugate point pairs, termed control point pairs, in two datasets, (2) Match checking: Detect inaccurate control point pairs from the set of control point pairs for quality control, and (3) Alignment: Use the accurate control points to align the rest of the geospatial objects (such as points or lines) in both datasets by using the triangulation and rubber-sheeting techniques. Please note that finding accurate control point pairs is a very important step in this kind of feature-based conflation process as all the other points in both datasets are aligned based on the control point pairs.

Traditionally, the problems of vector and imagery conflation have been in the domain of image processing and GIS. The focus of the image processing techniques has been on automatic identification of objects in the image in order to resolve vector-image inconsistencies. However, these techniques require significant CPU time to process an image in its entirety and still may result in inaccurate results. Moreover, various GIS systems, such as ESRI ArcView,[8] ESEA MapMerger,[9] and Able R2V[10] provide the functionality to perform different layers of geospatial

[7] In this paper, we use the terms conflation, integration and alignment interchangeably.
[8] http://www.esri.com/
[9] http://www.esea.com/products/
[10] http://www.ablesw.com/r2v/

dataset integration. However, these products do not provide automatic vector and imagery conflation, and manual intervention is needed to consolidate multiple geospatial datasets. The goal of our research is to develop an automatic, efficient and accurate vector to imagery conflation technique to align vector and imagery for GIS-related applications. Once the vector datasets are aligned to higher accuracy imagery, their positional information can be updated. Furthermore, the aligned vector data can annotate spatial objects in the imagery with detailed attributions information often contained in vector datasets.

Particularly, in this paper, we consider the automatic conflation of road network and orthoimagery (i.e., this imagery is altered from original photos so that it has the geometric properties of a map). We propose our approach, a geospatial information integration approach, named *A*utomatic *M*ulti-*S*ource conflation (AMS-conflation), to automatically integrate vector data and imagery. AMS-conflation exploits information from each of the sources to be integrated to automatically identify control points for aligning datasets. Furthermore, rather than processing each source of information separately in isolation, AMS-conflation processes the sources and exchanges information obtained from one source to help the processing of the other source and vice-versa. Essentially, there are three general sources of information for automatically identifying control points: (1) inferences on the data source (e.g., analyzing road vector to detect intersections or classifying imagery to identify road regions), (2) metadata/attributes about the data sources (e.g., resolutions/coordinates of imagery and road width information of vector data), (3) other sources of data that can be linked to the source (e.g., the online telephone books that store the addresses of a named point in the imagery). These automatically exploited information are dynamically exchanged and matched across these geospatial datasets to accurately identify corresponding spatial features as control points in AMS-conflation.

Figure 2 shows the overall approach for conflating vector and imagery. AMS-conflation is a multi-step data alignment process that involves identification of matching features as control points, filtering of misidentified control points and local transformation of other spatial objects. In fact, AMS-conflation is based on the preliminary techniques that we proposed in [7]. We enhanced our techniques in several ways: (1) we proposed a histogram-based classifier to more accurately identify road intersections as control points on median resolution (about 1 m/pixel) to high resolution (up to 0.3 m/pixel) color orthoimages,[11] (2) we improved our localized image processing technique by exploiting road vector directions and widths to generate templates to match against the orthoimages, (3) we presented a novel evaluation methodology evaluate our conflation results based on three different metrics, and (4) we used different accuracy level real-world vector datasets and images of different resolutions for evaluation.

The remainder of this paper is organized as follows. Section 2 describes the algorithm to automatically identify control point pairs in two types of geospatial

[11] Our approach could apply to low resolution imagery as well. However, low resolution imagery is not the focus of our research. This is because natural objects in the imagery become vague with the decrease of imagery resolution. Furthermore, high resolution imagery becomes more and more available today and provides clearer ground truth information (hence often poses more challenging automatic spatial object recognition issues). In addition, the misalignments between vector data and low resolution imagery are rather imperceptible.

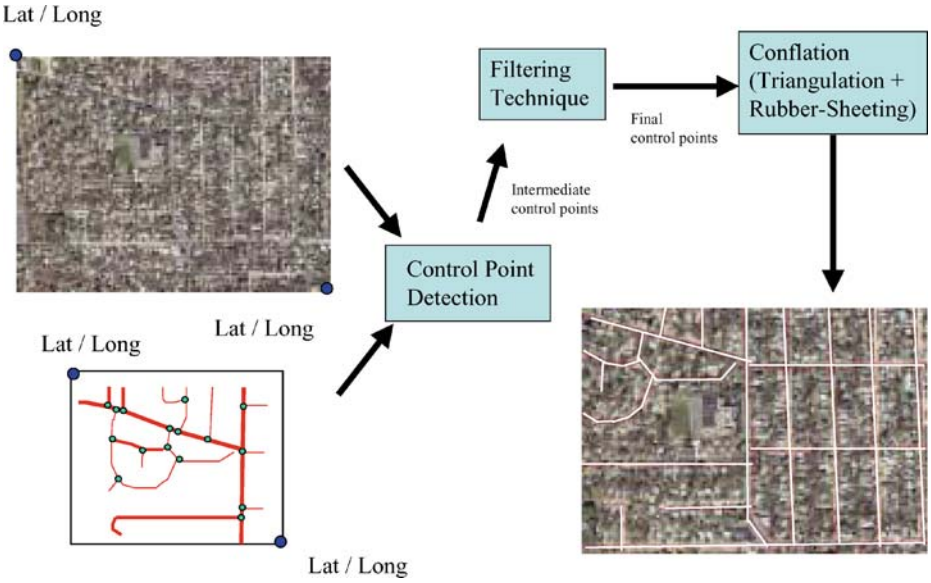


Figure 2 Overall approach to align vector with orthoimagery.

datasets. Section 3 describes the algorithm to filter out inaccurate control point pairs from the automatically generated control point pairs. Section 4 describes an enhanced conflation process to align two geospatial datasets. Section 5 presents our evaluation methodology and provides results of utilizing our approach to align real world data. Section 6 discusses the related work and Section 7 concludes the paper by discussing our future plans.

2 Finding control points

A control point pair consists of a point in one dataset and a corresponding point in the other dataset. Finding accurate control point pairs is a very important step in the conflation process as all the other points in both datasets are aligned based on the control point pairs.

Consider the conflation of road network and orthoimagery. Road intersections are good candidates for being control points, because road intersections are salient points to capture the major layouts of road network and the road shapes around intersections are often well-defined. In addition, various GIS researchers and computer vision researchers have shown that the intersection points on the road networks are good candidates to be identified as an accurate set of control points [7], [10], [13]. In fact, several image processing algorithms to detect roads in the imagery have been utilized to identify intersection points in the imagery. Unfortunately, extracting road segments directly from the imagery is a difficult task due to the complexity that characterizes natural scenes [1]. Thus, extracting roads from the imagery is error-prone and may require manual intervention. Moreover, processing an image of a large area to extract roads requires a lot of processing time.

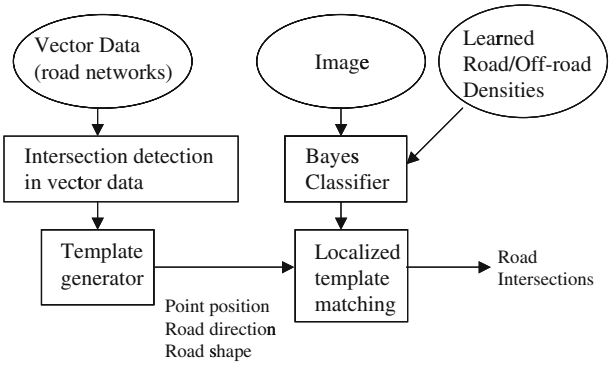


Figure 3 Road intersection identification.

Integrating vector data into the road extraction procedures alleviates these problems. In this section, we describe our technique, called localized template matching (LTM) [6], which takes advantage of the vector data, image metadata as well as the color of imagery to accurately and efficiently find the road intersections in the imagery. Conceptually, the spatial information on the vector data represents the existing knowledge about the approximate location of the roads and intersection points in the imagery, thus improving the accuracy and running time to detect intersection points in the image. The entire process of locating road intersections in imagery using road network data is shown in Figure 3. We discuss the detailed procedure in the following sections.

2.1 Road networks intersection detection

The process of finding the intersection points on the road network from the vector data is divided into two steps. First, the system examines all line segments in the vector data to label the endpoints of each segment as the candidate points. Second, the system examines the connectivity of these candidate points to determine if they are intersection points. In this step, each candidate point is verified to see if there are more than two line segments connected at this point. If so, this point is marked as an intersection point and the directions of the segments that are connected at the intersection point are calculated.

2.2 Imagery road intersection detection

Towards the objective of identifying road intersections on imagery, the vital step is to understand the characteristics of roads on imagery. In low resolution imagery, roads are illustrated as lines, while in high resolution imagery, roads are exposed as elongated homogeneous regions with almost constant width and similar color along a road. In addition, roads contain quite well-defined geometrical properties. For example, the road direction changes tend to be smooth, and the connectivity of roads follows some topological regularities.

Road intersection can be viewed as the intersection of multiple road axes that are located at the overlapping area of these elongated road regions. These elongated road regions form a particular shape around the intersection. Therefore, we can

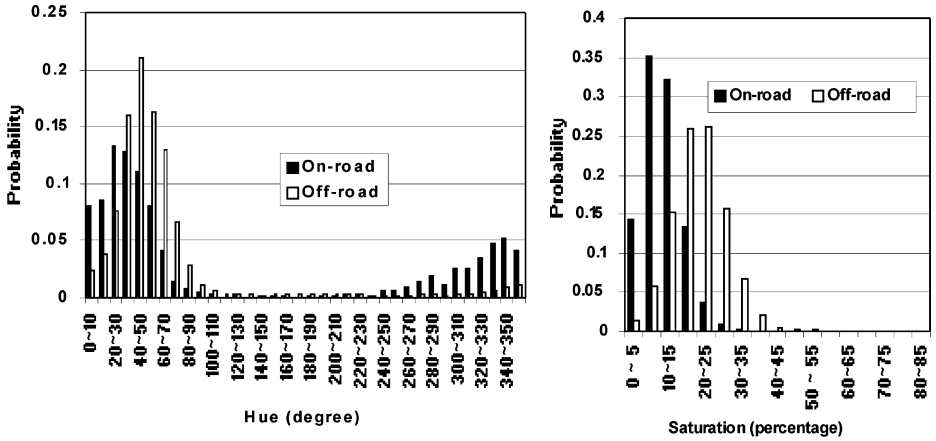
(a) High resolution color orthoimagery (b) Detected edges by Canny edge detector (c) Road-classified image (White pixels: image pixels pre-classified as roads)

Figure 4 An example of edge-detected image and road-classified image.

match this shape against a template derived from road network data (discussed next) to locate the intersection. Based on the characteristics of roads, the formation of this shape is either from detected road-edges or homogeneous regions. In our previous work [7], an edge detector (such as [20]) was utilized to identify linear features from imagery to match against vector data to locate intersections. However, on high resolution imagery (such as up to 0.3 m/pixel, as an example shown in Figure 4(a)), more detailed outlines of spatial objects, such as edges of cars and buildings, introduce noisy edges. Hence, in some cases the system may obtain fragmented edges that include real road edges, building edges, tree edges, etc (Figure 4(b)). This makes grouping-based method (i.e., the method that groups pixels belonging to the same edge as a line or a curve) used for road-edges linking to identify road intersections a difficult task. As the example shown in Figure 4(b), the system must exploit auxiliary constraints (e.g., the road sides of the same roads are often in parallel) to eliminate the impacts from noisy edges. However, we can make use of other useful information about roads, such as the color of roads, to alleviate this problem. Therefore, in contrary to traditional edge-detection approach, we propose a more effective way to identify intersections on imagery. Our approach uses the Bayes classifier, a histogram based classifier [11], [18], to classify images' pixels as on-road or off-road pixels (as in Figure 4(c)). In Section 2.2.1, we discuss the method to pre-classify image pixels, and in Section 2.2.2, we describe the localized template matching algorithm in detail.

2.2.1 Labeling imagery using the Bayes classifier

The histogram-based classification is based on the assumption of consistency of image color on road pixels. That is, road pixels can be dark, or white, or have color spectrum in a specific range, however; for the imagery set whose images were taken around the same time period using similar remote sensing equipments, we expect to find the same representative color on nearby road pixels. We construct the statistical color distribution (called class-conditional density) of on-road/off-road pixels by utilizing a histogram learning technique as follows. We first randomly train the system on a small area of the imagery by interactively specifying on-road regions and off-road regions respectively. From the manually labeled training pixels, the system learns the color distribution (histograms) for on-road and off-road pixels.



(a) Hue density function (b) Saturation density function

Figure 5 Learned density function on HSV color space for On-road/Off-road pixels.

The off-line learning process requires manual labeling to obtain conditional density functions, but it is performed only when a new imagery dataset is introduced to the system. In addition, we can apply the learned results to automatically identify intersections of the area that is much larger than the area we learn from. In order to determine whether new training is needed for a given image or the current learning is sufficient for the classification of the new area, in the future, we plan to develop an automatic approach based on the statistical analysis of the color distribution of the target imagery.

Figure 5 shows the hue probability density and saturation probability density,[12] after conducting the learning procedure on nearly 500 manually labeled rectangles from a set of USGS 0.3 m/pixel imagery (covering St. Louis County in Missouri). There are 250 rectangles for each category (On-road and Off-road) and totally three are 50,000 pixels within these rectangles. It is about 0.05% of our target imagery. It only took about 1 h to perform the labeling process. Reducing the amount of training samples may result in a narrower range of learned roads color. Thus, it may not provide sufficient color information for road classification. Figure 5(a) shows the conditional probabilities *Prob(Hue/On-road)* and *Prob(Hue/Off-road)*, respectively. The X-axis of this figure depicts the hue value grouped every 10 degrees. The Y-axis shows the probability of on-road (and off-road) pixels that are within the hue range represented by the X-axis. For a particular image pixel, we can compute its hue value h. Given the hue value h, if the probability for off-road is higher than on-road, our system would predict that the pixel is an off-road pixel. As shown in Figure 5, these density functions depict the different distribution of on-road and off-road image pixels on hue and saturation dimensions, respectively. Hence, we may use

[12] We eliminated the intensity (i.e., brightness of HSV model) density function. There is no obvious difference between the brightness distribution of on-road and off-road pixels, since these images were taken at the same time (i.e., under similar illumination conditions).

either of them to classify the image pixels as on-road or off-road. In our experiments, we utilized hue density function for classification. In general, we can utilize all color components, hue, saturation and brightness, with a more effective classifier (e.g., the machine learning classifier *Support Vector Machines*) to classify these higher dimensional datasets.

Based on the learned hue density functions, an automated road-labeling is conducted as follows. A particular image pixel whose hue value is h is classified as road if $\frac{\Pr ob(h/On-road)}{\Pr ob(h/Off-road)} \geq \theta$, where θ is a threshold. θ depends on the application-specific costs of classification errors and it can be selected using ROC technique discussed in [18].

2.2.2 Analyzing imagery using road network data (localized template matching)

Using the classified image (an example is shown in Figure 6(b)(c)) as input, the system can now match it with a template determined from the road network data to identify intersections. First, our LTM technique finds the geo-coordinates of all the intersection points on the road network data. Since the system also knows the geo-coordinates of the images (from image metadata), it can obtain the approximate location of intersections on the imagery (as in Figure 6(c)). For each intersection point on the road network data, LTM determines the area in the image where the corresponding intersection point should be located by picking a rectangular area (with width W and height H) in the image centered at the location of the intersection point from the road network data. Meanwhile, as an example shown in Figure 6(a), a template (with width w and height h) around an intersection on road network data is generated by the presence of regions inferred from the road network data using information, such as the road directions and road widths. LTM will then locate regions in the road-labeled image (see Figure 6(c)) that are similar (in shape) to the generated template (as in Figure 6(a)) as follows. Given a road-labeled image I with W×H pixels a template T with w×h pixels, the system then moves the template around the image and compares the template against the overlapped image regions. The adapted similarity measure is a normalized cross correlation defined as:

$$C(x,y) = \frac{\sum_{y'=0}^{h-1}\sum_{x'=0}^{w-1} T(x',y')I(x+x',y+y')}{\sqrt{\sum_{y'=0}^{h-1}\sum_{x'=0}^{w-1} T(x',y')^2 \sum_{y'=0}^{h-1}\sum_{x'=0}^{w-1} I(x+x',y+y')^2}}$$

where $T(x,y)$ equals one, if (x,y) belongs to a road region in the template T, otherwise; $T(x,y)$ equals zero. $I(x,y)$ equals one, if (x,y) is pre-classified as a road pixel, otherwise; $I(x,y)$ equals zero. C(x,y) is the correlation on the pixel (x,y). In our implementation, we set that W equals to H (i.e., a square area).

The highest computed correlation C(x,y) implies the location of the best match between the road-labeled image and the template. Furthermore, C(x,y) determines the degree of similarity between the matched road-labeled image and the template. An intersection will be identified, if C(x,y) is greater than a similarity threshold t ($0 \leq t \leq 1.0$). When setting t to 0.5, the system keeps the detected intersection that has higher similarity value (i.e., C(x,y)) than its dissimilarity value(i.e., 1.0−C(x,y)). Hence, in our experiment, we set the threshold t to 0.5.

(a) Layout (left) of the original road network data around an intersection point and template (right) inferred by using the road network data

(b) Original image

(c) Road-labeled image (White pixels: labeled road pixels; Black lines: existing road network data; Black circles: intersections on vector data, implying approximate location of intersections on imagery)

Figure 6 An example of the localized template matching.

The square area dimension (i.e., the width W) can be determined based on the accuracy and resolution of the two datasets. One option is to utilize the domain knowledge about maximum error or offset between two datasets. If this kind of knowledge is inaccessible, we can conduct experiments for a small set of these datasets to determine the area dimension by using various sizes and selecting the size that has better performance. More precisely, as an example shown in Figure 7, we utilized a high quality road network, NAVTEQ NAVSTREETS, and the proposed LTM technique with various area sizes to identify intersections in a 1.5 km by 1.5 km USGS high resolution color imagery (with 106 real road intersections).

Figure 7 shows the performance of LTM with different area dimension by applying a "buffer method" to calculate recall and precision of identified intersections. Road intersection can be viewed as the intersection of multiple road axes that are located at the overlapping area (called buffer) of elongated road

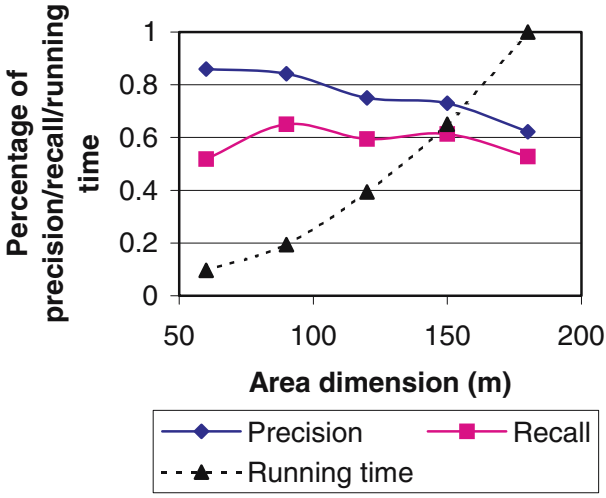


Figure 7 The impact of area dimension (dimension is increased by 30 meters, i.e., 100 pixels on 30 cm/pixel resolution imagery).

regions. Identified road intersections that fall within the buffer are considered as "correctly identified intersections". Using this term, we define:

$$\text{Recall} = \frac{\text{Number of correctly identified intersections}}{\text{Number of intersections in the image}} \quad (1)$$

$$\text{Precision} = \frac{\text{Number of correctly identified intersections}}{\text{Number of identified intersections}} \quad (2)$$

As shown in Figure 7, we increase the area dimension from 60 m to 180 m with the incremental dimension value 30 m. Then, we calculate the precision and recall. We also compute the normalized intersection detection running time (with respect to the running time of using 180 m as area dimension). The results show that the detection time dramatically increases as area dimension increases.

As shown in Figure 7, the precision decreases when the area dimension increases. This is because that larger area may involve more road intersections that have similar shape as the road template (e.g., some urban areas where roads are sometimes constructed in a grid shape). In addition, there could be more misclassified pixels for a larger area, thus detecting some incorrect intersections. Due to the same reason, the recall also slightly decreases as the area dimension increases from 90 m to 180 m. However, we obtained lowest recall when setting dimension size to 60 m. This implies that dimension 60 m is not large enough to capture most of the positional displacements between the vector and imagery. Therefore, based on these experimental results, we can select 90 m as our area dimension to identify intersections on other neighboring areas. This is because setting the area size to 90 m, we achieved 84% precision and 64% recall. Although it is slightly smaller than the precision (86%) obtained using 60 m as area dimension, we have much better recall (64 v.s. 52%).

The histogram-based classifier, as illustrated in the previous section, may generate fragmented results due to noisy objects, such as cars, tree-clusters and building shadings on the roads. Furthermore, some nonroad objects whose color is similar to road pixels might be misclassified as roads. However, LTM can alleviate these problems by avoiding exhaustive search of all the intersection points on the entire image and usually locates the intersection point on the image that is the closest intersection point to the intersection point on the road network data. Moreover, this technique does not require a classifier to label each pixel for the entire region. Only the areas near the intersections on the image need to be pre-classified. In addition, when utilizing localized template matching, it implicitly implies that the topology constraint (such as adjacency) is considered. This is because the template is generated based on the connectivity of several road segments merging at the same intersection point.

Furthermore, note that the objective of histogram-based classifier is not to extract the roads by correctly classifying every single pixel of the image as off-road or on-road. Therefore, it is not essential to apply morphology techniques (e.g., dilation or erosion) to resolve the fragmented classification results. Instead, as long as a majority of on-road pixels are identified so that the intersection-shape on the image is captured, LTM can successfully match it to the corresponding vector template. Even in the worst case, if we miss an entire intersection, still the entire

Figure 8 The intersections (rectangles) on road network data and the corresponding intersections (circles) on imagery.

conflation process may be successful as long as enough number of intersections is identified. Moreover, due to some noisy information in the imagery, applying morphology techniques to the fragmented classification results can produce road shapes that are inconsistent to the original road shapes. This may in turn result in poorer performance of LTM.

In sum, the running time for the LTM technique is dramatically lower than traditional image processing techniques due to performing image processing on localized areas. Furthermore, exploiting the road direction information improves both the accuracy and efficiency of detecting road intersections in the image. Figure 8 shows an image indicating the intersection points on road network data and the corresponding intersection points identified from the imagery.

3 Filtering control points

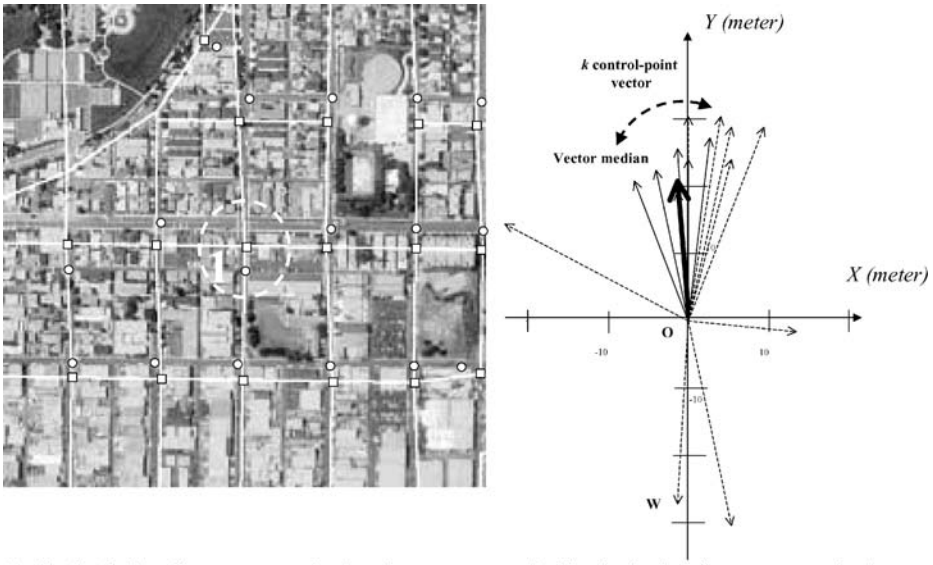
Due to the complexity of natural scene in the imagery, the LTM technique may still misidentify intersections as control points. For example, in Figure 9, the detected control point pairs 1, 2 and 3 are inaccurate control point pairs. It is essential to use a filter to eliminate misidentified intersections and only keep the accurately identified intersection, hence improving the precision with the cost of reducing recall. To address this issue, we can exploit the fact that there is a significant amount of regularity in terms of the relative positions of the controls points across data sets. This is due to the fact that we are not trying to correct individual errors, but rather to determine some local transformations across datasets that allow us to integrate two separate data sources. More precisely, while there is no global transformation (or systematic behavior of the offsets) to align imagery and vector data, in small

Figure 9 The intersections (rectangles) on road network data and the corresponding intersections (circles) on imagery (intersections 1, 2 and 3 are misidentified points).

areas the relationship between the points on the imagery and the points on the vector data can be described by some similar transformations. These transformations can be attributed to different projections, accuracies, or coordinate systems used in the imagery data and the vector data. Due to the above-mentioned nature of the datasets, in a small region the control points on the imagery and the counterparts on vector data should be related by similar transformations. Therefore, the inaccurate control point pairs can be detected by the filters that identify those pairs with significantly different relationship as compared to the other nearby control point pairs. Vector Median Filter (VMF) [2] is an example of such filter. VMF is a mathematical tool for signal processing to attenuate noise, and it is a popular filter to do noise removal in image processing. The VMF perceives the data points as vectors and filters out the data point with vectors significantly different from the median vector (i.e., outliers).

VMF first interprets the coordinate displacement between the points of each control point pair as a 2D vector, termed as control-point vector. Assuming that N control point pairs are generated in a small area by LTM. Hence, there are N control-point vectors denoted as $\{\vec{x}_i / \vec{x}_i = \overrightarrow{P_iQ_i}$ (i=1, 2, 3, ... N), where the tail P_i is an intersection point on the vector dataset, and the head Q_i is the corresponding point on the imagery}.

Since vectors are invariant under translation, it is convenient to consider the tail P_i as located at the origin. Hence, the tail of each control-point vector coincides to the same origin. For example, the control-point vectors for the 17 detected control point pairs of Figure 10(a) are illustrated in Figure 10(b) as the arrows (vectors). Due to the similarities of these control-point vectors in a local area, the directions and magnitudes of them can be represented by the vector median. We modified the



(a). The distribution of seventeen control point pairs (represented as rectangles and circles) in a small area centered at the control point 1

(b). The distribution of seventeen control-point vectors

Figure 10 VMF filter.

vector median filter as follows to identify the control-point vectors that are significantly different, hence obtaining the best matching set of control points.

The vector median of these N vectors $\vec{x}_i$ (i=1, 2, 3, ... N) is defined as the vector $\vec{x}_{vm}$ such that

(1) The sum $\sum_{i=1}^{N} \|\vec{x}_{vm} - \vec{x}_i\|$ is minimized.
Here $\| \|$ stands for L_2 norm (Euclidean distance).
(2) $\vec{x}_{vm} \in \vec{x}_i; \mathrm{i} = 1, 2, 3, \ldots, \mathrm{N}$

Vector median has similar properties as the median operation. Intuitively, the median vector is the vector that has the shortest summed distance (Euclidean distance) to all other vectors.

The inputs for a vector median filter are N vectors $\vec{x}_i(\mathrm{i} = 1, 2, 3, \ldots \mathrm{N})$ and the output of the filter is the vector median $\vec{x}_{vm}$. We revised the output of vector median filter to accommodate not only $\vec{x}_{vm}$, but also k closest vectors to the vector median. We defined the distance D:

$$D = \left\| \vec{x}_k - \vec{x}_{vm} \right\|_2 \text{ where } \vec{\mathrm{x}}_k \text{ is the } k\text{-th closest vector to } \vec{\mathrm{x}}_{vm}.$$

Then, the output of modified vector median filter is

$$\left\{ \vec{x}_i \middle| \text{ where } \left\| \vec{\mathrm{x}}_i - \vec{\mathrm{x}}_{vm} \right\| \leq D \text{ and } i = 1, 2, 3, \ldots N \right\}$$

As a result of the modified Vector Median Filter, k closest vectors to the vector median are selected (because they have similar directions and magnitudes to the vector median) and the other control-point vectors are filtered out. The possible value of k is an integer between 1 and N. Large value of k provides more control-

Figure 11 After applying VMF on Figure 10(a). The circles mark the control points categorized as outliers by VMF.

point vectors, but may not filter out all inaccurate control point pairs. Based on our experiments of tuning different values for k, VMF filter performs well when setting k to $\lceil \frac{N}{2} \rceil$. Hence, the system kept the $k = \lceil \frac{N}{2} \rceil$ closest vectors to the vector median and filtered out the rest of the control point pairs. As a result, some accurate control-point vectors may be lost. However, the missing control point pairs would not greatly affect the conflation results, as some of the selected control point pairs close to the lost accurate control point pairs have the similar directions and displacements.

Figure 10 graphically shows how the Vector Median Filter works. For example, to determine whether the control point pair *1* (Figure 10(a)) is an outlier or not, its corresponding control-point vector would be compared to other control-point vectors nearby. The 17 neighboring control-point vectors within a radius of 300 meters to the control point pair *1* are shown in Figure 10(b) as the arrows. The thickest arrow is the vector median among these control-point vectors. After applying the modified Vector Median Filter, only nine (k=9) closest vectors to the vector median are not categorized as outliers. The control point pair *1* will be filtered out (see Figure 11), because its corresponding control-point vector (represented as $\overrightarrow{OW}$ in Figure 10(b)) is categorized as an outlier. The system repeats the same process to filter out other outliers.

4 Conflating imagery and vector data

After filtering the control point pairs, the system identifies an accurate set of control point pairs. Each point of the control point pair from the vector data and imagery indicates the same position. Transformations are calculated from the control point

pairs. Other points in both datasets are aligned based on these transformations. The Delaunay triangulation [4] and piecewise linear rubber sheeting [26] are utilized to find the appropriate transformations. The Delaunay Triangulation is discussed in Section 4.1, and rubber-sheeting is explained in Section 4.2.

4.1 Space partitioning using Delaunay triangulation

To achieve overall alignment of imagery and vector data, vector data must be adjusted locally to conform to the imagery. The system can align the two datasets based on local adjustments, because small changes in one area usually do not affect the geometry at long distances. To accomplish local adjustments, the domain space is partitioned into small pieces based on accurately identified control point pairs. Then, the system applies local adjustments to each individual piece. Triangulation is an effective strategy to partition the domain space into triangles to define local adjustments.

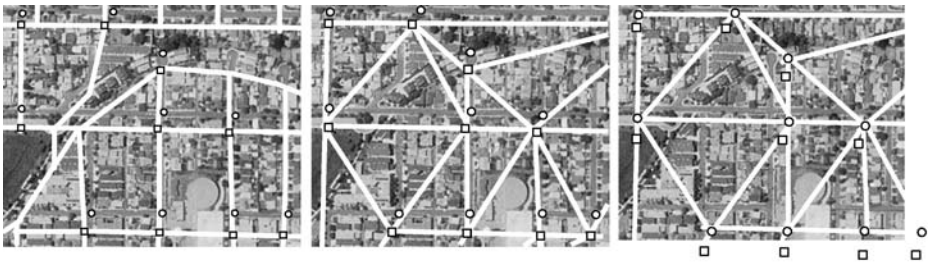
One particular type of triangulation, Delaunay triangulation, is especially suited for conflation systems [23]. A Delaunay triangulation is a triangulation of the point set with the property (called "INCIRCLE" property) that no point falls in the interior of the circumcircle of any triangle (the circle passing through the three triangle vertices). The Delaunay triangulation maximizes the minimum angle of all the angles in the triangulation, thus avoiding triangles with extremely small angles [4]. More precisely, an incremental Delaunary triangulation algorithm to partition the space (based on the point set P) works as follows: (1) initialize triangulation TR with a big bounding triangle that contains all points P, (2) randomly choose a point p' from P, (3) find the triangle T_{in} in TR which contains p', (4) subdivide T_{in} into smaller triangles that have p' as a vertex, (5) fix up neighboring regions by performing diagonal swaps on edges based on the INCIRCLE property test, (6) repeat steps 2 to 5 until all points have been added to TR.

Our system performs the Delaunay triangulation with the set of control points on the vector data, and makes a set of equivalent triangles with corresponding control points on the imagery. Figure 12 shows an example of a resulting Delaunay triangulation on some detected control points.

The time complexity of performing Delaunay triangulation is O(n*log n) in worst case, where n is the number of control points. The details of the triangulation algorithms are discussed in [4], [23].

4.2 Piecewise linear rubber-sheeting

Imagine stretching a vector dataset as if it were made of rubber. Our system deforms algorithmically, forcing registration of control points over the vector data with their corresponding points on the imagery. This technique is called "Piecewise linear rubber sheeting" [26]. There are two steps to rubber sheeting. First, the transformation coefficients (i.e., the affine transformation that is composed of translation, rotation and scaling) to map each Delaunay triangle on vector data onto its corresponding triangles on the imagery are calculated. Second, the system applies the same transformation coefficients to transform the endpoints of each vector line segment within each Delaunay triangle to the imagery. Consider the example shown in Figure 13(a). White lines represent the road network. The rectangles are the



(a) The original road network and detected control point pairs (one is marked by rectangle and the corresponding point is represented as circle)

(b) Delaunay triangulation based on detected control points on road vector data

(c) Corresponding Delaunay triangulation based on detected control points on image

Figure 12 An example of Delaunay triangulation.

control points on the road vector data and the circles are the corresponding control points on the imagery. The two triangles shown are Delaunay triangles formed by three corresponding control point pairs and one endpoint A of the original road segments is located within the triangle on the road vector data. The rubber sheeting technique transforms endpoint A to the point B on the imagery (B becomes a road endpoint on the image). The conflated road network is constructed by connecting these transformed endpoints (see Figure 13(b)).

Piecewise linear rubber sheeting based on triangles with extremely small angles (i.e., long and thin triangles) results in distorted conflation lines. Since the Delanuay triangulation avoids triangles with extremely small angles, it reduces the problem. The detailed piecewise linear rubber-sheeting algorithms can be found in [23], [26].

5 Performance evaluation

In this section, we evaluated AMS-conflation by conducting several experiments on various real world data. Section 5.1 describes the test datasets in details. The purpose of the experiment is to evaluate the utility of our algorithms in integrating real world data. We are interested in measuring the improvement in the accuracy of the integration of road vector and imagery using AMS-conflation. To that end, we performed experiments to validate the hypothesis: using AMS-conflation, we can automatically improve the alignments of different accuracy level road vectors with orthoimagery.

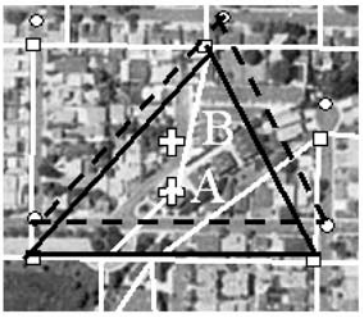
Section 5.1 describes the experimental setup and the test datasets. Section 5.2 presents our evaluation methodology to measure the performance. Section 5.3 discusses the experimental results.

5.1 Experimental setup

We used the following two different datasets for our experiments:

(1) Orthoimagery

The imagery used in the experiments is the geo-referenced USGS color orthoimagery (0.3 m/pixel resolution) and geo-referenced USGS gray-scale



(a) The endpoint A on the original road segments (white lines) will be transformed to the point B on the imagery, after applying rubber-sheeting

(b) Partial conflated roads after applying rubber-sheeting to Figure 12

Figure 13 An example of rubber-sheeting.

DOQ imagery with lower resolution (1 m/pixel). In particular, we tested the color imagery that covers some areas of county of St. Louis, MO, and the gray-scale imagery that covers some areas of the city of El Segundo, CA. This imagery is available online and queryable from Microsoft TerraService web service [3].

(2) Vector data (road networks)

Three road networks from different data providers were used as the vector data:

- TIGER/Lines from U.S. Census: The TIGER system was developed by the U.S. Bureau of the Census. The Bureau provides the public with the TIGER/Line files, which are extractions of selected geographic and cartographic information from the TIGER database. In particular, we focus on the road networks queried from TIGER/Line files (called TIGER/Lines henceforth).
- NAVSTREETS from NAVTEQ: NAVSTREETS is a commercial product and is high quality vector data with highly accurate geometry. It is regularly updated by NAVTEQ using some base maps acquired from a variety of sources including local governments, commercial mapping agencies and other public agencies. Many online street map services, such as Google Map, utilize NAVSTREETS as the base dataset for road route planning.
- Road network data (called MO-DOT henceforth) from department of transportation, St. Louis, MO [24]. It is also a high quality road vector dataset.

In general, all these road network data listed above have rich attribution; however, TIGER/Lines has both poor positional accuracy and road geometry. With TIGER/Lines and MO-DOT, the number of lanes can be inferred from the attribute "*CFCC (Census Feature Class Codes)*" associated with each road segment, while the number of lanes can be obtained from the attribute "*LANE_CAT*" in NAVSTREETS. Furthermore, the locations of road intersections and the road directions around each intersection are calculated by analyzing these road networks using the algo-

Figure 14 Original road vector (white lines) superimposed with imagery.

rithms described in Section 2.1. In general, NAVSTREETS and MO-DOT are high quality road vectors, but with various accuracy levels. As some sample images shown in Figure 14, there are some spatial inconsistencies between the USGS high resolution color imagery and the three different vectors.

Our automatic conflation system was developed in C#. The algorithm allows the user to specify the two datasets to conflate (e.g., "imagery and TIGER/Lines", "imagery and NAVSTREETS" or "imagery and MO-DOT"). The output of our conflation system was a set of conflated roads for the three different types of vector datasets. The experiment platform is Pentium III 1.2 GHz processor with 512 MB memory on Windows XP (with .NET framework installed). In order to evaluate our approach on various real world data, we applied AMS-conflation to diverse areas as summarized in Table 1. In addition, we manually plotted the real road axes (called reference roads) as the ground truth with which we compare our conflated roads.

5.2 Evaluation methodology

We compared the reference roads with conflated roads by developing an evaluation schema to measure (1) *The percentage of the reference roads in imagery for which we generated conflated roads*, (2) *The percentage of correctly conflated roads with respect to the total conflated roads*, (3) *The percentage of the total length of the conflated roads that is within a specific distance of the reference roads*.

In the computer vision literature on automatically extracting roads from imagery, there are many methodologies proposed to evaluate the extracted roads against real roads [14], [27]. Due to the natural similarity between the problem of evaluating extracted roads and our problem of evaluating conflated roads, we can utilize these existing algorithms to evaluate our conflation results. In particular, we adapted the "road-buffer method" proposed in [27]. The road-buffer method is utilized to measure the accuracy of automatically extracted roads with respect to real roads. We revised this method to measure the accuracy of conflated roads with respect to real roads.

According to the algorithm proposed in [27], to compare two road networks (in our case, they are reference road network and conflated road network), the first step is to split both networks into short pieces of equal length. Then, a constant predefined road-width is constructed around the reference road network. Every portion of the conflated road network within the given distance (i.e., the buffer width) from the reference road network is considered as matched. Furthermore, the

Table 1 Tested datasets used in the experiments.

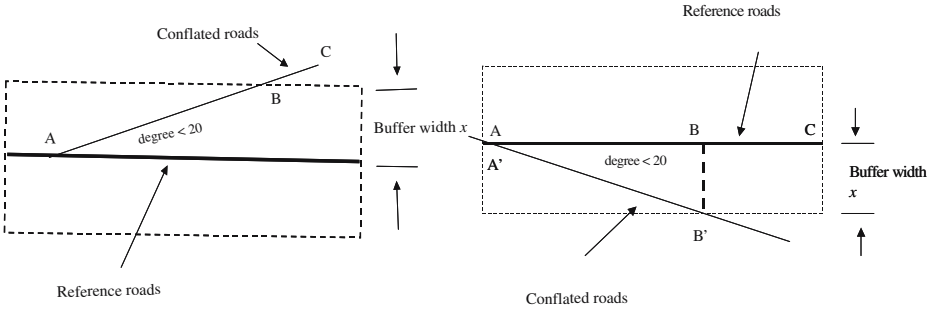
	Test area 1	Test area 2	Test area 3	Test area 4
Area covered*	Latitude: 38.5808 to 38.5947 Longitude: −90.4049 to −90.388 Width: 1.5 km Height: 1.5 km	Latitude: 38.5703 to 38.5842 Longitude: −90.543 to −90.526 Width: 1.5 km Height: 1.5 km	Latitude: 38.5962 to 38.6101 Longitude: −90.490 to −90.473 Width: 1.5 km Height: 1.5 km	Latitude: 33.914 to 33.932 Longitude: −118.4209 to −118.399 Width: 2 km Height: 2 km
Total road length of TIGER/ Lines (m)	23,534.52	21,443.96	7,966.62	46,580.64
Total road length of NAVSTREETS (m)	24,360.00	21,921.29	9,876.02	N/A**
Total road length of MO-DOT (m)	24,759.3	21,796.92	9,431.68	N/A**
Total road length of reference roads (m)	25,471.63	21,999.00	9,252.01***	46,660.2
Area features	0.3 m/pixel color imagery. Suburban area (covering some urban area) with high road density (11.3 km/km^2) and high house density	0.3 m/pixel color imagery. Suburban area with high road density (9.7 km/km^2) and high house density. Perceptually, the majority of road color in this area is different from the road color in test area 1	0.3 m/pixel color imagery. Rural area with medium road density (4 km/km^2). 12% of the roads are highways	1 m/pixel gray-scale imagery. Urban area with high road density (12.83 km/km^2) and high house density

* Test area 1, 2, and 3 cover partial areas of the county of St. Louis, MO. Test area 4 covers a partial area of the city of El Segundo, CA.

** These road vector datasets were inaccessible at the time the experiments were performed.

*** These reference roads are shorter than vector data because some straight reference roads are depicted as curves in vector datasets.

direction difference between matched road axis and reference road axis must be less than a pre-defined threshold d (d was set to 20 degree in [27]). The drawback of this procedure is that the performance is highly affected by the predefined constant buffer width. Instead of using the constant buffer width for each road segment, we used the real road widths in the imagery as the buffer. Hence, the roads with different widths have different buffer widths. The pieces of the conflated roads within the buffer to the reference roads with consistent direction are considered as matched.



(a) Segment AB: matched conflated road (b) Segment AB: matched reference road

Figure 15 Buffer method for evaluating completeness and correctness.

Figure 15(a) shows the examples of matched conflated roads with respect to reference roads. Segments AB is calculated as *matched conflated roads*, while BC is not. Figure 15(b) shows the example of matched reference roads with respect to conflated roads. Segments AB is categorized as *matched reference road*, since the conflated road segment A'B' can be used to "complete" (or "match") the reference road segment AB. Segments BC is unmatched reference road.

Using this term, two measurements, completeness and correctness, are defined as follows [27].

$$\text{Completeness} = \frac{\text{Length of matched reference roads}}{\text{Total length of reference roads}}$$

$$\text{Correctness} = \frac{\text{Length of matched conflated roads}}{\text{Total length of conflated roads}}$$

Basically, the completeness is *the percentage of the reference roads in imagery for which we generated conflated roads*. On the other hand, correctness is *the percentage of correctly conflated roads with respect to the total conflated roads*. However, the other measurement, RMS (root-mean-square error), described in [27] does not meet our requirements to compute *how far the conflated road network is from the reference road network*, since it only measures *how far the matched conflated road network is from the reference road network*. Instead of computing a number (e.g., average distance) to illustrate how far from each other the two networks are, we would like to measure the percentage of the total length of the (conflated) roads that is within a specified distance x to the reference roads (e.g., 95% of the conflated roads are within 5 meters of the reference roads). The method proposed in [12] is a technique to assess this positional accuracy. As an example shown in Figure 16, we consider a buffer with width x around the reference road network, then compute the proportion p of the conflated roads that lies within the buffer [12]. Using this technique, only the distance difference between two roads is considered. The errors due to the difference of directions between roads are captured by completeness and correctness.

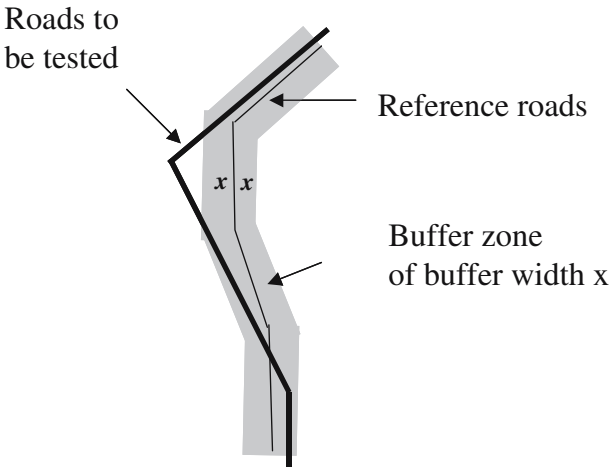


Figure 16 Positional accuracy evaluation.

We conducted our experiments as follows for test area 1, 2 and 3:

Step 1: Learn the histogram (as shown in Figure 5) from nearly 500 manually labeled rectangles[13] from some color orthoimages covering partial areas of the County of St. Louis, MO.

Step 2: Apply AMS-conflation algorithm to conflate each area (image) with TIGER/Lines, NAVSTREETS and MO-DOT respectively.

In particular, for test area 1, 2 and 3, we conducted experiments to measure the accuracy of original road vectors and conflated road vectors generated by AMS-conflation. Then, we compare the evaluation results for conflated road vectors with the results for original road vectors. Additionally, we also measured the quality of detected control points (before and after applying filtering techniques) by the precision and recall metrics defined in Section 2.2.2 (Eq. 1 and Eq. 2). For the fourth test area (covering partial area of city of El Segundo, CA), basically, we repeated the same process as above, but we learned road color information from the lower resolution black–white images that covers some areas of El Segundo and we only conflated TIGER/Lines. We discuss the detailed experimental results in the following section.

5.3 Experimental results and interpretations

Since finding accurate control point pairs is an important step in the conflation process, Section 5.3.1 describes the performance of LTM and VMF to detect control points. Section 5.3.2 describes the overall performance of AMS-conflation.

[13] There are 50,000 pixels covered by these rectangles and it took about 1 h to perform the labeling process.

Table 2 Results of identified intersections.

		Test area 1	Test area 2	Test area 3	Test area 4
(a) Precision/recall of identified intersections for TIGER/Lines					
Original road network	Precision	7.1%	8.7%	4.8%	3.8%
	Recall	6.7%	7.7%	4.5%	3.7%
Without filtering	Precision	72.3%	82.1%	57.9%	78.9%
	Recall	68.1%	24.2%	52.4%	52.1%
Using VMF filtering	Precision	87.1%	83.1%	69.2%	94.8%
	Recall	45.4%	21.2%	42.9%	34.0%
(b) Precision/recall of identified intersections for NAVSTREETS					
Original road network	Precision	15.6%	23.3%	19.1%	
	Recall	15.2%	23.1%	18.2%	
Without filtering	Precision	87.7%	82.1%	64.7%	
	Recall	76.2%	40.7%	52.4%	
Using VMF filtering	Precision	97.1%	92.6%	83.3%	
	Recall	54.1%	27.5%	47.6%	
(c) Precision/recall of identified intersections for MO-DOT					
Original road network	Precision	57.1%	32.2%	28.6%	
	Recall	55.2%	31.9%	27.2%	
Without filtering	Precision	83.8%	82.1%	83.3%	
	Recall	73.9%	50.5%	71.4%	
Using VMF filtering	Precision	98.1%	97.1%	90%	
	Recall	42.9%	37.3%	43%	

5.3.1 Experimental results of precision and recall of identified intersections using LTM

Finding accurate control point pairs is a very important step in the conflation process as all the other points in both datasets are aligned based on the control point pairs. Hence, we evaluated the LTM performance by calculating precision and recall of detected control points (before and after applying filters). The results are listed in Table 2. We also included the precision/recall of original road network in Table 2 to demonstrate that our LTM improved both precision/recall of original vector data.

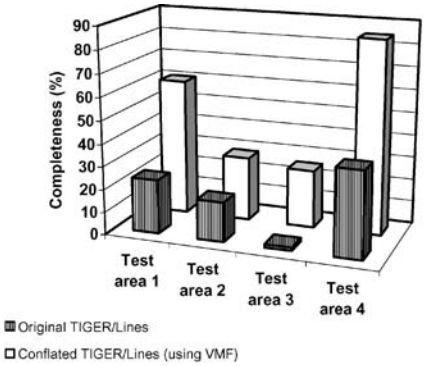
As shown in Table 2, our LTM performed differently for various real world scenarios. This is because these vectors have different qualities and the orthoimagery has various levels of complexity. Hence, we obtained high precision (up to 98%) control points in some areas (such as test area 1), while medium precision (about 70%) in other areas (such as the alignment of TIGER/Lines and imagery in test area 3). In general, we improve the precision after applying filtering techniques. The filtering techniques improve the precision at the cost of reducing the recall. However, for the conflation process higher precision is more important than higher recall, since we are not trying to correct individual errors, but rather to determine the local transformations to align vector and imagery.

5.3.2 Experimental results of completeness and correctness of conflated roads

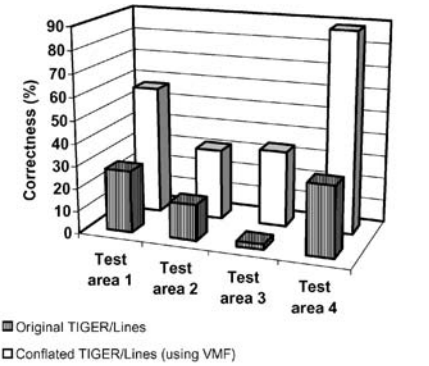
The experimental results of completeness and correctness for each vector dataset are listed in Figure 17(a)–(f). Intuitively, the completeness corresponds to the users'

demands: *how much is missing in the original road network or in the conflated road network with respect to reference road network*. The correctness, on the other hand, is related to *the probability of the original or conflated road segments to be considered as reference road segments*. We showed the completeness (and correctness) for each utilized vector dataset respectively. In addition, the completeness (and correctness) values are grouped by tested areas as the X-axis of Figure 17. The Y-axis in Figure 17(a)(c)(e) shows the completeness for original road vector and conflated road vectors using VMF-filtered intersections as control points. Furthermore, in Figure 17(b)(d)(f), the Y-axis depicts the correctness. For example, as shown in Figure 17(a) and Figure 17(b), when utilizing VMF-filtered intersection points to generate conflated TIGER/Lines for test area 1, we improved the completeness from its original value of 19 to 59.15%, and correctness from its original value of 25 to 55%. Another example, as the results for test area 4 shown in Figure 17(a)(b), we improved the completeness and correctness 2.5 times better than the original TIGER/Lines. In addition, there are some immediate observations from this figure:

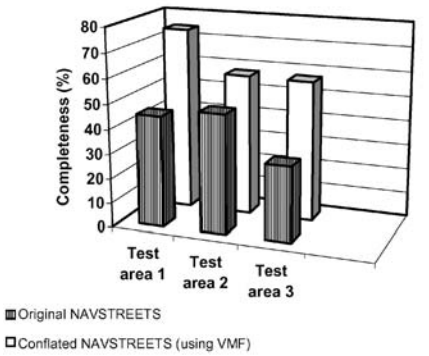
- The data quality of MO-DOT is superior to NAVSTREETS and much better than TIGER/Lines, which is consistent with our prior knowledge about these three different datasets. Moreover, from the diverse completeness and correctness in each vector dataset for different test areas, we concluded that each vector dataset itself has various accuracies. This is also consistent with the vector data quality statements quoted by the data providers.
- Consider TIGER/Lines as the vector data source. The shapes (and geometry) of the original TIGER/Lines are sometimes inconsistent with the corresponding roads in the imagery, because large portions of curve-shaped roads were simplified as straight lines. Hence, as shown in Table 2 and Figure 17, original TIGER/Lines has low completeness/correctness and low precision/recall for the intersections. For a particular road segment, if the shape of the original vector data is inconsistent with roads in the imagery (as the example of TIGER/Lines), our system may not align them well, although the majority of intersections might be aligned. This is mainly because our matching is at the point level, not at the edge level. As the example of TIGER/Lines in test area 1 (see Table 2(a)), we improved the node (intersection) alignment (as the precision improved from original 7 to 87.1%), while we achieved completeness and correctness to 55%. However, recently, not only is the imagery quality enhanced, the quality of vector data is also significantly improved. Consider the conflation of high quality imagery and high quality vector dataset, such as NAVSTREETS. The road shapes of NAVSTREETS are very similar to the road shapes in the imagery. Hence, the major issue is that there are some local inconsistencies between them. AMS-conflation can capture these local transformations (based on intersection to intersection matching information) and maintain the road shapes.
- On average, good improvements were achieved for TIGER/Lines (as shown in Figure 17(a)(b)). For NAVSTREETS, we perform 1.3 to 1.9 times better than the original data (as in Figure 17(c)(d)), while we only gain marginal improvements for MO-DOT data on test area 1 and 3 (see Figure 17(e)(f)). This is due to high completeness (92.54%) and correctness (93.38%) of the original MO-DOT data in test area 1. In addition, some roads are not aligned well around highways in test area 3. The road widths of highways vary and are difficult to predict. The problem could probably be addressed by integrating other information sources



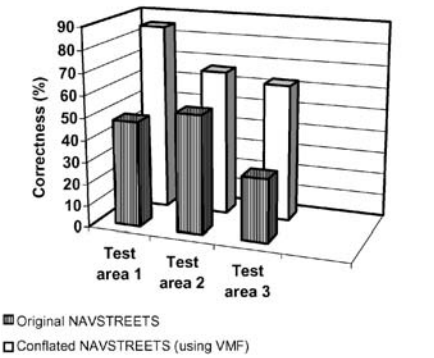
(a) Completeness assessment for TIGER/Lines



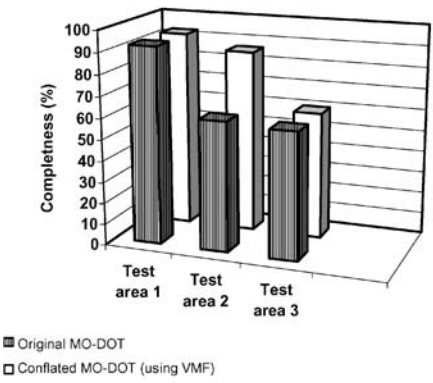
(b) Correctness assessment for TIGER/Lines



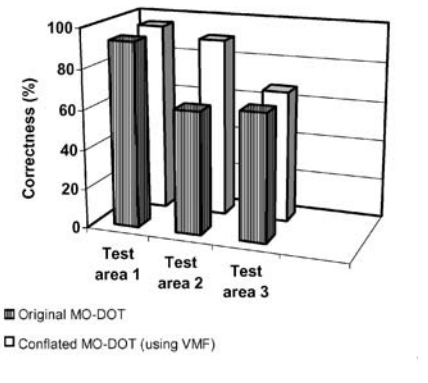
(c) Completeness assessment for NAVSTREETS



(d) Correctness assessment for NAVSTREETS

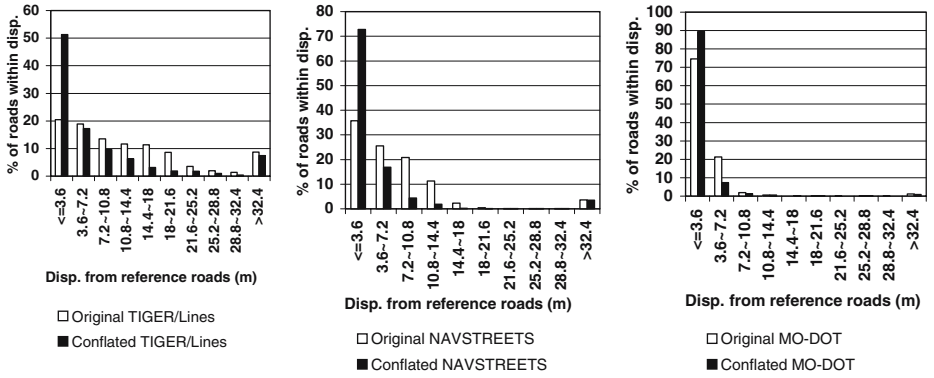


(e) Completeness assessment for MO-DOT

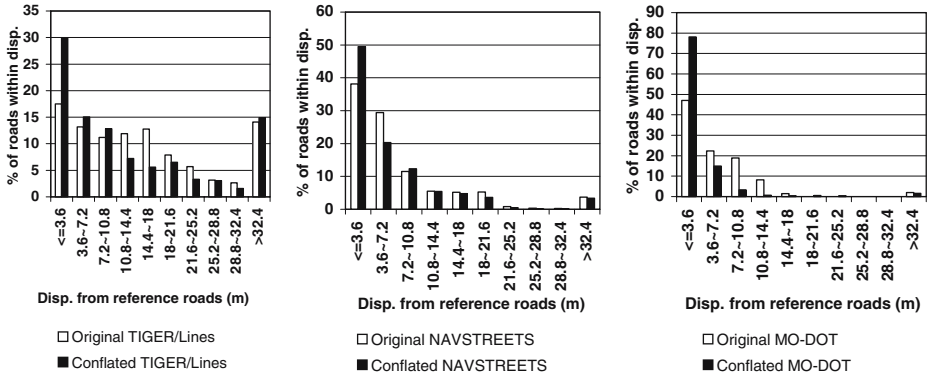


(f) Correctness assessment for MO-DOT

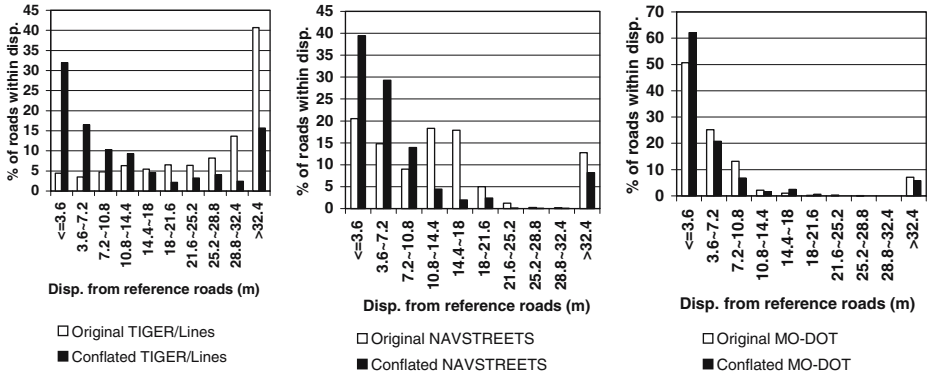
Figure 17 Evaluation results for completeness and correctness.



(a) Positional accuracy assessment for test area 1



(b) Positional accuracy assessment for test area 2



(c) Positional accuracy assessment for test area 3

Figure 18 The percentage of original/conflated roads lying within the buffer versus the buffer width (all conflated roads were generated based on VMF-filtered control points).

with those details. In fact, after visual examination, we found many misaligned road segments are close to the margins of road buffers (i.e., roadsides). When relaxing the "buffer-widths", we can obtain higher completeness and correctness. This kind of assessment is illustrated by our "positional accuracy" evaluation described next.

5.3.3 Experimental results of positional accuracy of conflated roads

The experimental results of "positional accuracy" categorized by road vectors for each test area are illustrated in Figure 18(a)–(c). Intuitively, the "positional accuracy" corresponds to the users' demand: *how far is the conflated road network from the centerlines of the real (reference) road network*. We evaluated these displacements between two networks by gradually increasing the buffer-widths constructed around reference road network. The buffer-width was increased by 3.6 meters (i.e., the U.S. standard lane width). As shown in the X-axis of Figure 18, the displacement values are grouped every 3.6 meters. The Y-axis shows the percentage of conflated roads lying within the displacement range represented by the X-axis. For example, as shown in Figure 18(a), when utilizing VMF-filtered intersection points to generate conflated NAVSTREETS for the first test area, about 75% of the conflated roads are within 3.6 m from the reference roads, and only 35% of the original NAVSTREETS are within 3.6 m displacement. Although we did not achieve significant improvements on completeness/correctness for MO-DOT data (as stated earlier), we achieve better positional accuracy: On average, 91% of the conflated MO-DOT roads are within 7.2 m of the reference roads compared to 80.3% of the original MO-DOT.

Even higher improvements were achieved for TIGER/Lines and NAVSTREETS. On average, 76% of conflated NAVSTREETS are within 7.2 meters displacement versus 54.6% of original NAVSTREETS and 53.96% of conflated TIGER/Lines are within 7.2 meters versus 25.93% for the original TIGER/Lines. In particular, comparing to NAVSTREETS and MO-DOT data, again, TIGER/Lines have poor positional accuracy and poor geometry. For such kind of severely distorted original TIGER/Lines segments, our approach is limited in aligning imagery curves and vector lines, although the detected intersections are matched (as shown in Table 2). Hence, only about 47% of conflated TIGER/Lines in test area 2 and 3 are within 7.2 meters from reference roads, while 70 to 85% of conflated NAVSTREETS and MO-DOT are within 7.2 meters displacement. However, comparing to the original TIGER/Lines, our approach significantly improved the positional accuracy.

Particularly, in the first test area, there are about 25% streets in grid shape and all three road vectors provide accurate road shapes over these street grids. Therefore, we have better performance in test area 1. The issue of quality of the vector data can be addressed by starting with higher quality data such as NAVSTREETS or MO-DOT data. However, there are small portions of our conflated roads not aligning well to the imagery. This is mainly because the color of these misaligned roads are very different from what we learned or the roads are close to the conflation area margins where long and thin Delaunay triangles were constructed. These issues could be alleviated by doing more training based on all

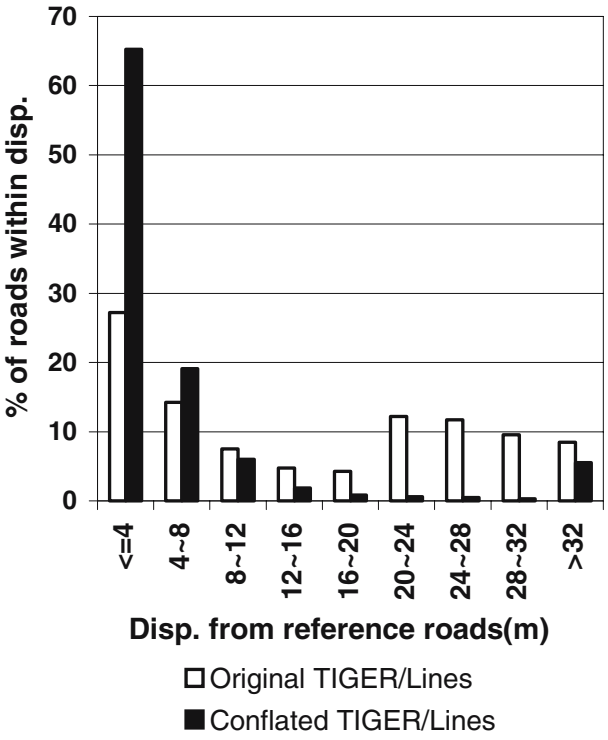


Figure 19 Positional accuracy assessment for test area 4.

available color components to recognize a wider range of road colors and applying the conflation on larger areas.

In the fourth test area, AMS-conflation achieved high improvement for TIGER/Lines. 85% of conflated TIGER/Lines are within 8 meters displacement versus 43% of original TIGER/Lines (see Figure 19). This demonstrates the utility of our approach to conflate vector data with black–white and lower resolution imagery.

5.3.4 Experimental results using filtered control points vs. using unfiltered control points

Finally, we also compared the performance of running the conflation algorithm with filtered control points and unfiltered control points. We conducted experiments to measure positional accuracy for original MO-DOT and conflated MO-DOT in the tested area 2. As the results shown in Figure 20, conflated roads generated by filtered control points outperforms those generated by unfiltered control points. This is because the conflation process does not require a large number of control point pairs to perform an accurate alignment. In fact, a set of sufficient amounts of control points with higher accuracy would serve better for the conflation process, which is what our filtering technique performs. Therefore, we only consider our conflation performance (as illustrated in Figure 17 and Figure 18) by utilizing the filtered control points.

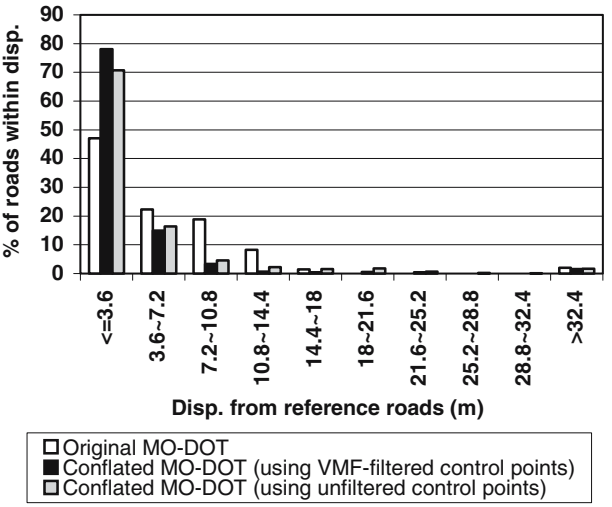


Figure 20 Comparison of filtered vs. unfiltered conflation results for test area 2.

5.3.5 Execution time and summary

Since the running time of AMS-conflation was mainly dominated by the LTM routine to detect road intersections, we used the running time of LTM as the overall execution time (the query time for retrieving online images or vector dataset was ignored). On average, the execution time for locating an intersection in a localized area was about 3 to 5 s (it depends on the radius setting for LTM). For example, the total running time for generating conflated roads for a small area of 1 km by 1 km with 20 intersections is about 1 min. When users manipulate a GIS system, they often navigate from a small area to another small area. Therefore, it is possible to apply our approach to better align imagery and vector data on the fly for GIS data navigation.

Figure 21 shows some sample images, after applying conflation to each of the three road vectors, respectively. In sum, AMS-conflation automatically and efficiently improved the alignments of various vector datasets with orthoimagery. This validates our experimental hypothesis.

6 Related work

Conflation was first proposed and implemented in 1988 by Saalfeld [22], and the initial focus of conflation was to eliminate the spatial inconsistency between two overlapping vector maps in order to improve the spatial accuracy of vector maps. From then, various conflation techniques have been proposed and many GIS systems have been implemented to achieve the alignments of geospatial datasets with different resolutions or different types.

Geospatial datasets conflation is a complex process that may utilize work from a broad range of disciplines that include GIS, cartography, computational geometry,

Figure 21 Vector-Imagery conflation (white lines: original road network; black lines: after applying conflation).

graph theory, image processing, pattern recognition, and statistical theory. In general, based on the types of geospatial datasets dealt with, the conflation technologies can be categorized into following three groups:

- Vector to vector data conflation: For example, the integration of two road networks of different accuracy levels.
- Vector to raster data conflation: For example, the integration of road network and imagery or road network and raster maps.
- Raster to raster data conflation: For example, the integration of two images with different resolutions or the integration of raster maps and imagery.

Many *vector to vector* conflation techniques have been proposed [8], [23], [25], [28] and various GIS systems (such as ESEA MapMerger) have been implemented to achieve the alignments of vector datasets with different accuracies. However, there are few research activities on *vector to imagery* data conflation. Due to recent advances in remote sensing technology to capture high resolution imagery, *vector to imagery* conflation has become one of the central issues in GIS. Utilizing vector to imagery conflation, this accurate imagery can be used to update vector datasets. Moreover, the abundant information often contained in diverse vector data can be utilized to annotate objects, such as roads and buildings, in the imagery.

In this section, we review related work on *vector to imagery* data integration. In particular, Section 6.1 reviews the techniques that detect some counterpart elements on the datasets and apply traditional conflation algorithm (i.e., establishing the correspondence between the matched entities and transforming other objects accordingly). Section 6.2 describes the approaches that utilize active counter models [19] to align each vector road segment to corresponding road edge in the imagery.

6.1 Aligning vector data and imagery using some (automatically) identified features

Hild and Fritsch [16] process vector data to extract vector polygons and performed image segmentation on imagery to find image polygons. Then, a polygon matching algorithm is applied on both images and vector to find a set of 2D conjugate points. In order to obtain a successful matching between an image and vector data, the datasets must contain polygonal features like forest, villages, grassland or lakes. This approach will fail when polygonal features cannot be found, such as in high resolution urban areas.

Filin and Doytsher [9] propose a linear conflation algorithm to align vector and imagery. First, all edges (such as road edges and building edges) are detected from the imagery (without using the existing vector data as prior knowledge) and converted to vector format. Then, their approach matches the detected edges with vector data to identify real road edges. Based on the influence regions formed by the matched edges, their system then transforms other vector data where there is no corresponding edge detected in the imagery. However, their approach suffers from the difficulties of extracting and converting features directly from imagery as vectors. There exists many algorithms for extracting roads utilizing the characteristics of roads and context of imagery as prior knowledge [17], [21], while none of them give good results in all circumstance [1], [15] and most of them are heavily CPU intensive.

Flavie et al. [10] try to find all the junction points of all detected lines in the imagery , then match the junction points of the road vector with the image junctions. Then, the vector lines are moved according to the matched junctions (i.e., no space partition method is used to build the influence regions of matched junctions). Finally, their system applies the active contour models technique [19] (discussed next) to refine the matched road segments. However, their method sufferers from the high computation cost of finding all possible junctions of detected lines on images.

Our AMS-conflation significantly differs from the work mentioned above in terms of our approach to locate matched entities. To the best of knowledge, AMS-conflation is the first vector to imagery conflation approach that automatically exploits auxiliary structured data (such as image color, image metadata, road directions, road intersections and road coordinates provided by vector data) to

improve the feature recognition techniques on imagery. In addition, AMS-conflation performs a template matching around each road intersection (i.e., a localized area, instead of the entire image). This will improve both the accuracy and efficiency.

6.2 Aligning vector data and imagery using Snakes-related techniques

In the computer vision literature on automatically aligning vector lines with imagery edges, the active contour models (i.e., Snakes [19]) is one of the most prevalent methods to "attach" vector dataset (e.g., road segments) to the corresponding features (e.g., road edges) in the imagery (often with the objective to detect changes of roads or detect real road edges to update pre-existing vector data). Snakes is a parametric curve and it is often modeled as a spline linked by multiple control points. The active contour models evolve their shape by moving their control points towards the image features and maintaining their smoothness at the same time. The evolution is based on the principle of energy (including internal and external energy) minimization. The "internal energy" enforces geometric constraints, such length and smoothness of the Snakes, while the "external energy" pushes the Snakes towards images features. By minimizing internal and external energy simultaneously, image information and geometric properties are fused to accomplish the evolution of the Snakes. The Snakes method requires some seed points as control points to start the evolution and these seed points should be close to the real roads. One option is utilizing pre-existing vector data as the (initial) approximate outline of the roads. However, the Snakes method is not appropriate for aligning roads in highly textured areas such as urban areas, due to the following weaknesses: (1) The Snakes might attach to noisy pixels and this prevents the Snakes from converging on the desired edges, (2) It is a greedy algorithm and demands a lot of calculations when trying all possible (and local) solutions and picking the best one, (3) If the placement of the Snakes is not well initialized, the Snakes will diverge, (4) Relaxing the internal energy tends to destroy the shape of the Snakes.

Furthermore, each vector road segment needs to perform the Snakes evolution to accomplish the alignment, while the conflation techniques (described in previous section) only detect some corresponding features (e.g., points, lines or polygons) and transform other features accordingly. Particularly, comparing our AMS-conflation to the Snakes method, our matching mechanism is not based on entire road segments but on partial road segments around the intersections. For a particular road segment, if the shape of the original vector data is inconsistent with roads in the imagery, AMS-conflation may not align them well (although the intersections might be aligned). Considering the Snakes techniques, this type of poorly aligned original vector will also harm the evolution of Snakes. In a worst case, it may cause the Snakes to diverge. However, recently, not only the imagery quality is enhanced, the quality of vector data is also significantly improved. Consider the conflation of high quality imagery and high quality vector dataset, such as NAVSTREETS. Most road shapes of NAVSTREETS are consistent to the road shapes in the imagery. Therefore, using some local transformations can significantly reduce the positional inconsistencies (and maintain the road shapes) between them, while the Snakes method may wiggle the road shapes. Hence, we save computation time by only detecting some (salient) feature points and transforming other points (and lines) utilizing Delaunay triangulation and rubber-sheeting. Moreover, AMS-conflation avoids these high quality original road segments to attach to noisy edges. In fact, our conflated roads are very close to the road axes on

the imagery (based on our experiments) and they can be utilized as good seed points for any more robust Snakes-related algorithms[14] in the future.

7 Conclusion and future work

The main contribution of this paper is the design and implementation of a novel data fusion approach to automatically integrate vector data and imagery of median to high resolutions. AMS-conflation is the first technique that exploits different sources of information and metadata from each of the sources to be integrated to automatically generate control points for conflation. Moreover, we propose an effective filtering algorithm to automatically eliminate inaccurate pairs from the generated control points. Experimental results on the county of St. Louis, MO, and city of El Segundo, CA, demonstrate that our approach can automatically and accurately align and annotate orthoimages with vector data.

We are further enhancing our approach by improving both LTM technique and the VMF filtering technique. More precisely, we are improving road classifications by utilizing a machine learning classifiers, called *Support Vector Machines (SVM*) to categorize images pixels based on all available image color information/channels (e.g., RGB or HSV). This better classification consequence results in better intersection detection by LTM. As stated in Section 3, there is often no systematic behavior of the offsets (i.e., no global transformation) between the geospatial datasets. The revised classifier can help to identify more accurate control point pairs. Hence, we can produce more accurate local transformations to reduce the positional inconsistencies between different datasets. Meanwhile, we are enhancing our VMF filtering technique by investigating the differences of angular distances (i.e., both vector direction and magnitude) between control-point vectors. This is because angular distance provides the essential information to dynamically determine different thresholds (i.e., the parameter k of VMF) for diverse regions. More precisely, for the majority of the scenarios, the similar vectors tend to form clusters around the median vectors. From this observation, we can modify the filtering technique to accommodate more vectors that are close to the median vector. The revised filter can retain more accurate control points that are detected by improved classifier and LTM (i.e., the enhanced filter can increase the recall rate of identified control points without losing precision).

Our approach for conflating road networks with imagery can be generalized to a wide variety of geospatial data sources. The basic idea is to use whatever information is available about the different geospatial products to automatically determine a set of control point pairs. Thus, we can apply this approach to conflating images with map, vector, and point data. In particular, consider applying our approach to conflate vector data and geo-referenced maps. The same types of image processing on vectors and can performed on maps. In fact, for many maps, finding the roads on the maps is an easier problem. Once a set of vector data is aligned to two different sets of images or maps, then the same set of control points can be utilized to conflate image with image, map with map, or map with image [5]. Several important

[14] Many variants of the active contour models are developed to improve the efficiency and accuracy to make it appropriate for different scenarios. This proposed improvement of the active contour models is a different research topic that is beyond the scope of this paper.

application domains that can benefit from such integration are the crisis management applications, city traffic planning and military intelligence applications.

Acknowledgment We would like to thank Snehal Thakkar for his valuable discussions on this project.

This research has been funded in part by NSF grants EEC-9529152 (IMSC ERC), IIS-0238560 (PECASE), and IIS-0324955 (ITR), in part by the Air Force Office of Scientific Research under grant numbers F49620-01-1-0053 and FA9550-04-1-0105, in part by a gift from the Microsoft Corporation, and in part by the US Geological Survey (USGS) under order number 05CRSA0551. The U.S. Government is authorized to reproduce and distribute reports for Governmental purposes notwithstanding any copyright annotation thereon. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of any of the above organizations or any person connected with them.

References

1. M.-F. Aculair-Fortier, D. Ziou, C. Armenakis, and S. Wang. "Survey of work on road extraction in aerial and satellite images, Universite de Sherbrooke," *Technical Report*, 2000.
2. J. Astola, P. Haavisto, and Y. Neuvo. "Vector median filter," in *Proceedings of IEEE*, April, 1990.
3. T. Barclay, J. Gray, and D. Stuz. "Microsoft terraserver: A spatial data warehouse," in *Proceedings of the 19th ACM SIGMOD International Conference on Management of Data*, 2000.
4. M.D. Berg, M.V. Kreveld, M. Overmars, and O. Schwarzkopf. "Computational geometry: Algorithms and applications," Springer: Berlin Heidelberg New York, Vol., 1997.
5. C.-C. Chen, C.A. Knoblock, C. Shahabi, Y.-Y. Chiang, and S. Thakkar. "Automatically and accurately conflating orthoimagery and street maps," in *Proceedings of the 12th ACM International Symposium on Advances in Geographic Information Systems (ACM-GIS'04)*, pp. 47–56, Washington, DC, ACM, November 12–13, 2004.
6. C.-C. Chen, C. Shahabi, and C.A. Knoblock. "Utilizing road network data for automatic identification of road intersections from high resolution color orthoimagery," in *Proceedings of The Second Workshop on Spatio-Temporal Database Management(STDBM'04) colocated with VLDB*, Toronto, Canada, August 30, 2004.
7. C.-C. Chen, S. Thakkar, C.A. Knoblok, and C. Shahabi. "Automatically annotating and integrating spatial datasets," in *Proceedings of the International Symposium on Spatial and Temporal Databases*, Santorini Island, Greece, 2003.
8. M. Cobb, M.J. Chung, V. Miller, H.I. Foley, F.E. Petry, and K.B. Shaw. "A rule-based approach for the conflation of attributed vector data," *GeoInformatica*, Vol. 2(1):7–35, 1998.
9. S. Filin and Y. Doytsher. "A linear conflation approach for the integration of photogrammetric information and GIS Data," *International Archives of Photogrammetry and Remote Sensing*, Vol. 33:282–288, 2000.
10. M. Flavie, A. Fortier, D. Ziou, C. Armenakis, and S. Wang. "Automated updating of road information from aerial images," in *Proceedings of American Society Photogrammetry and Remote Sensing Conference*, 2000.
11. D. Forsyth and J. Ponce. Computer Vision: A Modern Approach. Prentice-Hall, 2001.
12. M.F. Goodchild and G.J. Hunter. "A simple positional accuracy measure for linear features," *International Journal of Geographic Information Sciences*, Vol. 11(3):299–306, 1997.
13. A. Habib, R. Uebbing, and A. Asmamaw. "Automatic extraction of road intersections from raster maps," *Technical Report*, The Center for Mapping, The Ohio State University, 1999.
14. W.A. Harvey. "Performance evaluation for road extraction," *The Bulletin de la Societe Francaise de Photogrammetrie et Teledetection*, Vol. 153(1999-1):79–87, 1999.
15. C. Heipke, H. Mayer, and C. Wiedemann. "Evaluation of automatic road extraction," *IAPRS, International Society for Photogrammetry and Remote Sensing*, Vol. 32(3-2(W3)), 1997.
16. H. Hild and D. Fritsch. "Integration of vector data and satellite imagery for geocoding," *International Archives of Photogrammetry and Remote Sensing*, Vol. 32, 1998.
17. S. Hinz, A. Baumgartner, C. Steger, H. Mayer, W. Eckstein, H. Ebner, and B. Radig. "Road extraction in rural and urban areas," *Semantic Modeling for the Acquisition of Topographic Information from Images and Maps*, 1999.

18. M. Jones and J. Rehg. "Statistical color models with application to skin detection," in *Proceedings of IEEE Conf. on Computer Vision and Pattern Recognition*, 1999.
19. M. Kass, A. Witkin, and D. Terzopoulos. "Snakes: Active contour models," *International Journal of Computer Vision*, Vol. 1, 1988.
20. R. Nevatia and K.R. Babu. "Linear feature extraction and description," *Computer Graphics and Image Processing*, Vol. 13:257–269, 1980.
21. K. Price. "Road grid extraction and verification," *International Archives of Photogrammetry and Remote Sensing*, Vol. 32(Part 3-2W5):101–106, 1999.
22. A. Saalfeld. "Conflation: Automated map compilation," *International Journal of Geographic Information Sciences*, Vol. 2(3):217–228, 1988.
23. A. Saalfeld. Conflation: Automated map compilation, Ph.D. Dissertation, Computer Vision Laboratory, Center for Automation Research, University of Maryland, 1993.
24. E.L. Usery, M.P. Finn, and M. Starbuck. "Data integration of layers and features for the national map," in *Proceedings of American Congress on Surveying and Mapping*, Phoenix, Arizona, 2003.
25. V. Walter and D. Fritsch. "Matching spatial data sets: A statistical approach," *International Journal of Geographic Information Sciences*, Vol. 5, 1999.
26. M.S. White and P. Griffin. "Piecewise linear rubber-sheet map transformation," *The American Cartographer*, Vol. 12(2):123–131, 1985.
27. C. Wiedemann, C. Heipke, and H. Mayer. "Empirical evaluation of automatically extracted road axes," in *Proceedings of 9th Australasian Remote Sensing and Photogrammetry Conference*, Sydney, 1998.
28. S. Yuan and C. Tao. "Development of conflation components," in *Proceedings of Geoinformatics*, 1999.

Ching-Chien Chen is the Director of Research and Development at Geosemble Technologies. He received his Ph.D. degree in Computer Science from the University of Southern California for a dissertation that presented novel approaches to automatically align road vector data, street maps and orthoimagery. His research interests are on the fusion of geographical data, such as imagery, vector data and raster maps with open source data. His current research activities include the automatic conflation of geospatial data, automatic processing of raster maps and design of GML-enabled and GIS-related web services. Dr. Chen has a number of publications on the topic of automatic conflation of geospatial data sources.

Craig Knoblock is a Senior Project Leader at the Information Sciences Institute and a Research Professor in Computer Science at the University of Southern California (USC). He is also the Chief Scientist for Geosemble Technologies, which is a USC spinoff company that is commercializing work on geospatial integration. He received his Ph.D. in Computer Science from Carnegie Mellon.

His current research interests include information integration, automated planning, machine learning, and constraint reasoning and the application of these techniques to geospatial data integration. He is a Fellow of the American Association of Artificial Intelligence.

Cyrus Shahabi is currently an Associate Professor and the Director of the Information Laboratory (InfoLAB) at the Computer Science Department and also a Research Area Director at the NSF's Integrated Media Systems Center (IMSC) at the University of Southern California. He is also the Chief Technology Officer of Geosemble Technologies. He received his B.S. degree in Computer Engineering from Sharif University of Technology in 1989 and his M.S. and Ph.D. degree in Computer Science from the University of Southern California in 1993 and 1996, respectively. He has two books and more than hundred articles, book chapters, and conference papers in the areas of databases, GIS and multimedia. Dr. Shahabi's current research interests include Geospatial and Multidimensional Data Analysis, Peer-to-Peer Systems and Streaming Architectures. He is currently an associate editor of the IEEE Transactions on Parallel and Distributed Systems (TPDS) and on the editorial board of ACM Computers in Entertainment magazine. He is also in the steering committee of IEEE NetDB and ACM GIS. He serves on many conference program committees such as ACM SIGKDD 2006, IEEE ICDE 2006, ACM CIKM 2005, SSTD 2005 and ACM SIGMOD 2004. Dr. Shahabi is the recipient of the 2003 Presidential Early Career Awards for Scientists and Engineers (PECASE).