

# ANALYSIS OF VECTOR AND RASTER IMAGE INTEGRATION WITH DISPROPORTIONAL SCALING

**Boris Kovalerchuk**, Professor  
**William Sumner**, Researcher  
**Richard Chase**, Graduate Student  
Department of Computer Science,  
Central Washington University,  
Ellensburg, WA 98926-7520  
[borisk@cwu.edu](mailto:borisk@cwu.edu),  
[sumner@cwu.edu](mailto:sumner@cwu.edu),  
[chacer@cwu.edu](mailto:chacer@cwu.edu)

## ABSTRACT

The problems of imagery integration that includes registration, conflation, fusion, and search components require sophisticated methods, which are robust. An algebraic approach is a promising new option for developing such methods. It is based on algebraic structural analysis of features represented as open polylines called linear features. The problem of choosing points when attempting to prepare a linear feature for comparison with other linear features is a significant challenge when image orientation and relative scales are unknown. We have developed a method known as Binary Structural Division (BSD) to deal with this problem. The BSD method is effective in comparing feature structures for cases with significant differences in resolution, rotation, and scale. In other cases, images can also be disproportionally scaled. In this paper, we show that the BSD method can handle limited disproportional scaling for both open and closed polylines. It requires preprocessing by applying a new Reverse Disproportional Scaling (RDS) algorithm proposed in this paper.

## INTRODUCTION

Images to be registered invariably contain different information. At the conclusion of every registration process, there are ambiguities and potential inconsistencies that must be estimated to judge process quality. Robust measures of these differences are necessary to understand the complex variability of image matching and to guide the process. A final step of having an expert review and modify registrations is often essential.

The developed registration process (1) Extracts curvilinear features using standard algorithms; (2) Builds generalized features from extracted features that may contain discontinuities and combine several original extracted features; (3) Calculates algebraic structural characteristics of these features using techniques from abstract algebra; (4) Registers images by matching features based on their intrinsic algebraic structure.

## ALGEBRAIC STRUCTURAL ANALYSIS

Algebraic structural relational invariants (Kovalerchuk et al., 2001-2004) provide a basis for the registration/conflation of images (in raster or vector format) from many sources with various resolutions and reliabilities, giving them common scales and coordinates.

The method of *algebraic invariants*: (1) does not rely on the identification of *control points* (Brown, 1992; Zitová T., Flusser, 2003), (2) does not require *common scales* and (3) does not require that the *orientations* of individual images be known. Thus this method differs from other methods such as presented in (Bartl,

Schneide, 1995; Cohen, Guibas, 1997; Dey et al., 1999; Eppstein, 1999; Hirose et al., 2001; Pinz et al., 1999; Brown, 1992; Zitová T., Flusser, 2003).

The combination of this method with photogrammetric and other sensor information can be useful, especially if images are taken by cameras with very different locations or resolutions.

The algebraic invariant approach assumes that each image has several well-defined “features” that can be represented as polylines (continuous chains of line segments). A feature is a wider concept than is commonly used in image sciences. *Anything with a reasonably well-defined shape will work as a feature.* A closed polygon is also a feature in this sense. The only requirement is that the feature can be fit with a polyline. It is not necessary to know what it is or if there are any correspondences with polylines in other images.

The algebraic technique we use (Mal'cev, 1973) differs from traditional algebraic techniques such as linear algebra. The following definitions make this distinction clear (Kovalerchuk, Schwing, 2001, 2004).

**Definition.** A pair  $\mathbf{a} = \langle A, \Omega \rangle$  is called an **algebraic system** if  $A$  is a set of elements,  $\Omega_a$  is a set of predicates  $\{P\}$  and operators  $\{F\}$  on  $A$  and on its Cartesian products, where

$$P: A \times A \times \dots \times A \rightarrow [0,1] \text{ and } F: A \times A \times \dots \times A \rightarrow A.$$

A set of elements can be more complex and contain several sets (e.g.,  $A$  and  $R$ ). In this case, the system is called *multisort algebraic system*.

A multisort algebraic system  $\mathbf{a} = \langle A, R, \Omega_a \rangle$  is called a **linear feature** if  $R$  is a set of real numbers,  $\Omega_a$  consists of two operators (functions)  $D(a_i)$  and  $L(a_i, a_j)$  and three predicates (linear order relations)  $>_a, \geq_D, \geq_L$ . Operators  $D(a_i)$  and  $L(a_i, a_j)$  satisfy some additional axiomatic properties (Kovalerchuk, Schwing, 2001, 2004) that permit them to represent such polyline characteristics as lengths of intervals and angles.

**Definition.** An algebraic system  $\tilde{\mathbf{a}}$  is called an **abstracted linear feature** of feature  $\mathbf{a}$  if  $\Omega_{\tilde{\mathbf{a}}}$  consists of three predicates (linear order relations)  $>_a, \geq_D, \geq_L$ , with  $\Omega_a = \langle >_a, \geq_D, \geq_L \rangle$  from the linear feature  $\mathbf{a}$ .

**Definition.** Linear features  $\mathbf{a} = \langle A, R, \Omega_a \rangle$  and  $\mathbf{b} = \langle B, R, \Omega_b \rangle$  are *co-reference candidates* if they are homeomorphic and have isomorphic linear subfeatures

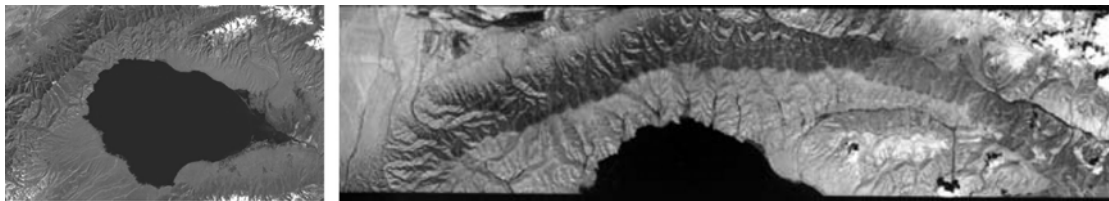
$$\mathbf{e}_a = \langle E_a, R, \Omega_c \rangle \text{ and } \mathbf{e}_b = \langle E_b, R, \Omega_c \rangle,$$

where  $\mathbf{e}_a \subseteq \mathbf{a}$  and  $\mathbf{e}_b \subseteq \mathbf{b}$ .

With these, theorems such as the high speed of the registration process can be proven (Kovalerchuk, Schwing, 2004).

**Theorem:** If the number of elements in linear features  $\mathbf{a}$  and  $\mathbf{b}$  equals  $n$ , then their maximum co-reference subsystem  $\mathbf{e}$  can be found in  $O(n^3)$  matrix comparisons and  $O(n^5)$  binary number comparisons for the worst-case scenario.

The number of points  $n$  needed to identify the feature is much smaller than the number of pixels in the whole image that autocorrelation methods use. The registration and conflation method based on algebraic invariants using polylines may be done in several ways. Consider the two satellite images of the Kyrgyz lake Sonkyl in Figure 1.



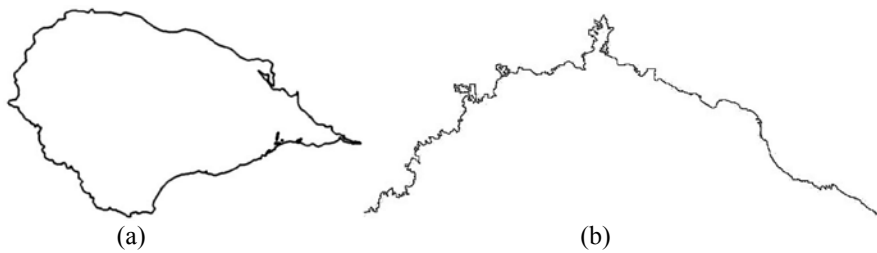
**Figure 1.** Two images of the lake Sonkyl in Kyrgyzstan

Feature extraction programs can be used to construct numerous polylines, with the most obvious one being the shoreline of the lake. The results are shown in Figures 2 and 3. While the overall structure of the extracted shorelines is apparent, the polylines differ in detail for a variety of reasons. Robust ways of

comparing these polylines are necessary to determine image transformations and to assess the quality of the result.



**Figure 2.** Extracted Sonkyl features



**Figure 3.** Lake shores extracted from the photographs of Sonkyl

### **BINARY STRUCTURAL DIVISION (BSD)**

The angles between segments and individual segment lengths are two algebraic characteristics of polylines that are used. For smooth features extracted from images with comparable scales and resolutions, either comparison works well. When there are marked differences in image scale and resolution, the choice of angles or lengths becomes more important. We examine characteristics of extracted polylines and how they may be interpolated and compared. For many cases, including the one illustrated here, comparisons based on segment lengths are shown to be superior.

One common problem with polylines is discontinuity that results from image resolution, differences in image acquisition, and artifacts of feature extraction algorithms. Extracted features can be modified in two ways to give a common resolution complexity to facilitate comparisons.

Consider the case of a curvilinear feature that is segmented as a result of something obscuring it, such as a road shaded by a tree. By connecting segments that are "close enough" and have "small" deviation angles, a composite feature can be formed. The maximum separation distance and the maximum deviation angle parameters permitted are clearly critical to feature creation and to one's confidence in the result.

Another type of feature modification is necessary to simplify curvilinear features with one or more relatively narrow lobes for comparison to one with no narrow lobes. For example, a state road map will typically depict a coastline with very little structure. A high resolution aerial photograph, on the other hand, will show the same coastline with lots of structure, showing that it goes inland for miles along river channels and juts out around spits of land. The higher resolution feature can be simplified by removing these lobes if they are "sufficiently narrow." Critical parameters here are the unit size of feature sampling and maximum jumping distance permitted.

With images prepared with this preprocessing, it is possible to register images that appear at first to have few if any features in common and are of unknown scale and orientation.

Measures of spatial similarity of polylines also need to be developed. Here, the focus is on spatial similarity characteristics while the similarity of non-spatial feature attributes can be matched after a spatial match is confirmed. If two images are matched using only a few reference points, the similarity of other points also needs to be assessed.

The issue of variability of points that form a polyline also needs to be addressed. Different feature extraction algorithms and imagery analysts can assign points differently on the same physical feature. This can affect finding co-reference candidate features. The algebraic technique called **Binary Sequential**

**Division (BSD) method** (Kovalerchuk, Sumner, 2003; Chase, Kovalerchuk, 2003) addresses this problem in a computationally efficient way.

Consider the polyline in Fig. 4 and the successive approximations defined by taking points defined by the mid point of lengths along the polyline as indicated. The BSD method computed by finding a curve **middle point** along the curve, then repeated for each half, halves of halves and so on. In this process a recursive function  $G(n)$  is used to denote the  $n$ -th interpolation of a polyline. Note that above  $n=2^k$ , and  $k$  is called the **BSD level**.

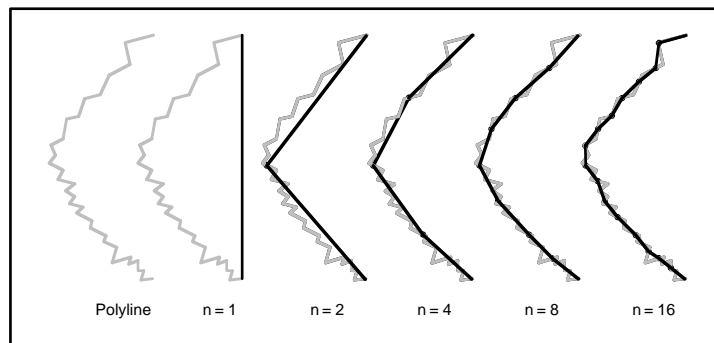
The first four steps of the conflation/registration process are:

Step 1. For raster images *extract* several linear features as sets of points (pixels),  $S$ . For vector images skip this step.

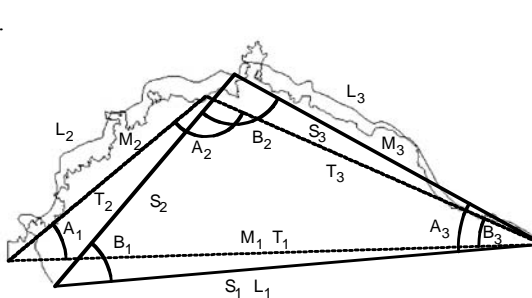
Step 2. *Vectorize* extracted linear features. For vector images skip this step.

Step 3. For both raster and vector images analyze the complexity and connectivity of vectorized linear features. If features are too simple (contain few points and are small relative to the image size) combine several features in a *superfeature*. If features are too complex, simplify features by applying a gap analysis algorithm. In the ideal situation we also should be able to separate feature extraction algorithm artifacts from real features. In the example shown in Fig. 3 (b) the feature extraction algorithm introduced artifacts, by capturing vegetation as a part of the shoreline in several places.

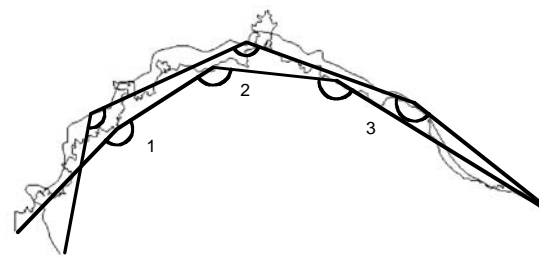
Step 4. *Interpolate* each superfeature as a specially designed polyline using the BSD method.



**Figure 4.** Structural interpolations of a polyline. Our experiments show that 8 sequential divisions with  $256=2^8$  linear segments is typically sufficient for interpolation.



**Figure 5.** Sections of the extracted shorelines with the first level BSD interpolation with  $k=1$  and  $n=2$ .



**Figure 6.** Fragment of BSD level 2 for the two polylines.

Fig. 5 depicts level 1 BSD interpolation with  $k=1$  and  $n=2$  for the vectorized features shown in Fig. 3. The middle points of each feature are as shown, computed along each line. Significant fluctuations have been lost in the lower resolution image. Feature M as interpolated has angles  $A_1$ ,  $A_2$ , and  $A_3$ . Feature L as interpolated has angles  $B_1$ ,  $B_2$ , and  $B_3$ .

## UNKNOWN PROPORTIONAL IMAGE SCALES

Fig. 7 and 8 show different possible matches when scales of two images are unknown. These matches have been found by using the BSD method. The image in Fig. 7(a) shows feature proportions as they were extracted and vectorized from two original images. Our experiments had shown that the BSD method was able to capture the right scale, rotation, and shift. The second image in Fig. 7 shows the correct match. Images in Figure 8 illustrate variability of possible matches when a scale ratio of two images can vary between 1:1 and 1:2.

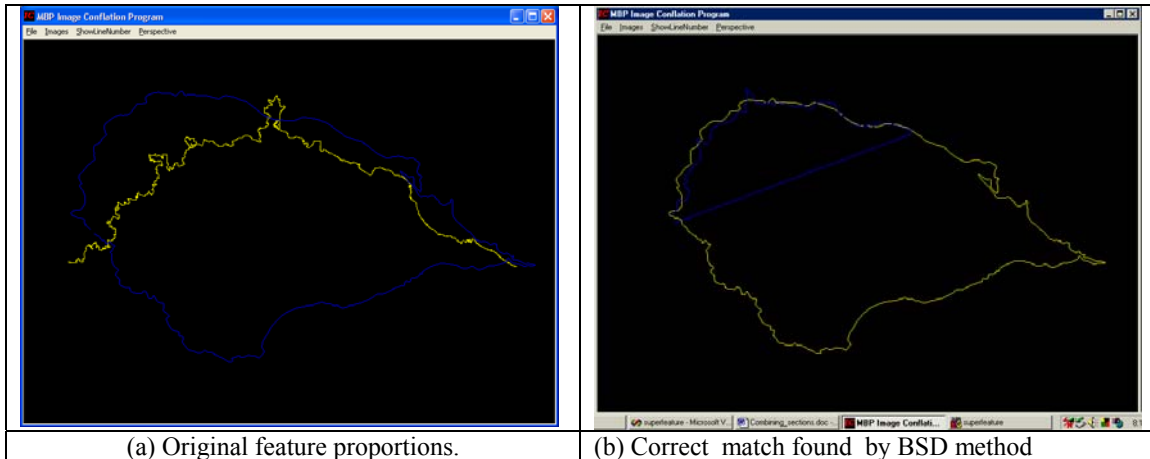


Figure 7. Original feature proportions and correct match.

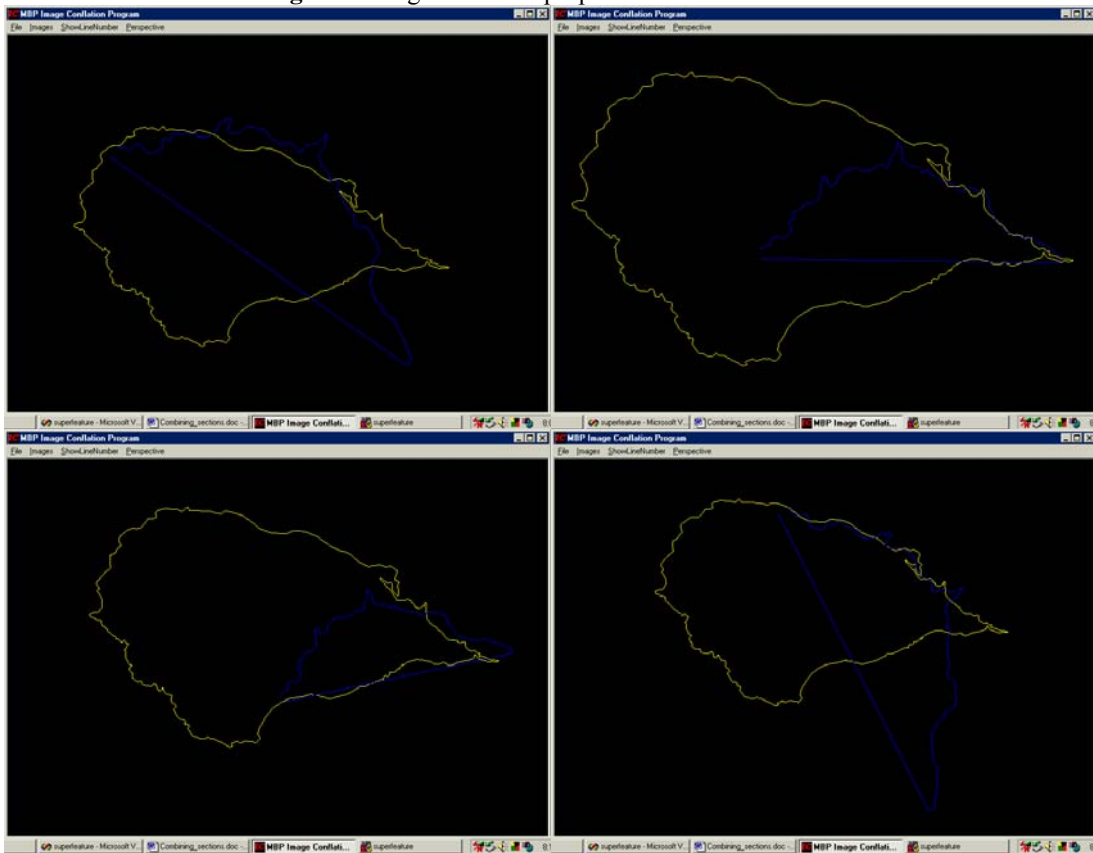


Figure 8. Alternative matches found by using BSD method.

## UNKNOWN DISPROPORTIONAL SCALES

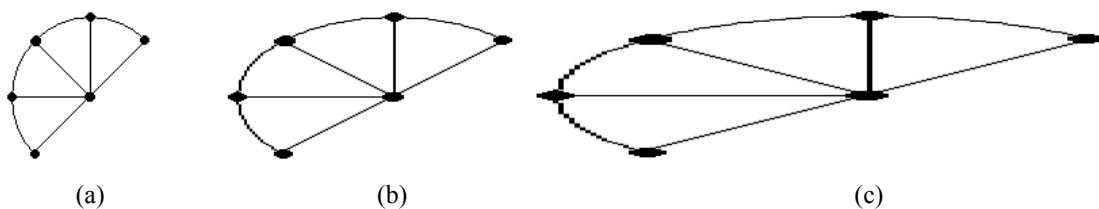
In essence the images in Fig. 7 and 8 are proportionally scaled. The problem of images that are disproportionally scaled presents a more significant challenge. The BSD process of placing middle points to interpolate a polyline is not invariant when images are disproportionally scaled (have a significant distortion in scale uniformity). This means that having the same feature in two images that are disproportionally scaled the middle points of these features can be physically different points. Applying the BSD method to such features may provide incorrect match.

Below we present a method to overcome disproportionality before the majority of points needed for BSD method are placed on the feature. We avoid the point placement until after disproportionality in scaling has been resolved, that is scales of two images made proportional. It is particularly difficult to generate a computer algorithm that can place points on a feature such that after rotation and disproportionally scaling *the same placement algorithm produces the same corresponding points*. This difficulty occurs because in general point placement algorithms produce points that are not invariant to the combination of rotation and disproportionally scaling. Even using the Ramer algorithm (that interpolates the polyline using the furthest points for each its segment) does not place points in the same places because of the nature of real data with noise that are disproportionally scaled. The Ramer algorithm is invariant only for ideal data without noise that is not the case in real imagery. In the Ramer algorithm we would need to use different threshold values for different parts of the feature to generate the same points if the feature has been rotated and disproportionally scaled from the original. Without knowing the transformation metrics it is impossible to calculate what these thresholds should be.

We have developed, however, an **algorithm to detect the rotation and disproportionally scale factors** for a given set of features or subsections of features. After applying the transformations we can then place points in a consistent manner such that we can determine if the candidate sections are indeed matching or not.

Let us examine the nature of disproportionally scaling both mathematically and visually. Given a set of points (coordinate pairs of points) that represent a polyline feature, we will use a half circle to demonstrate the issue.

We find that when disproportionally scaling is applied then the relationships of the initial x coordinates to each other remain the same, and the relationships of the initial y coordinates to each other remain the same. However, the relationships between initial x and y coordinates do not remain the same. While this is obvious it presents interesting implications when comparing to a polyline feature that was of originally different orientation. Consider the deformation of 200% scaling in x (horizontal) direction only (see Fig. 9):



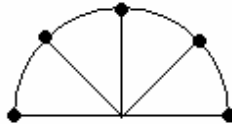
**Figure 9.** Disproportional scaling

It is both clear from observation (see Fig. 8) and mathematically provable that the relationships of x coordinates of the points remain the same in all three shapes. The same is true for y coordinate values taken separately of x coordinate. If the center of the shape is considered the origin then the x coordinates have simply been scaled by a factor of 2 (for the second shape) and by a factor of 4 (for the third shape) with the y coordinates staying the same (scaled by a factor of 1). Let us parameterize the location of each point in the range of the values in the x and y independently. Assume that the lowest x value for the five points on the curve has a parameter  $t=0.0\%$  and the highest value has  $t=100.0\%$ . Similarly the parameter  $u$  is set up for the y coordinate, with  $u=0.0\%$  and  $u=100.00\%$  for the two extremes respectively. In this notation the lowermost point is  $(t,u)=(16.0\%,0.0\%)$  and the rightmost point is  $(t,u)=(100.0\%,75\%)$ . Thus, if

we parameterize the x and y coordinates to the range of the values in the x and y independently we have something like this starting from the low left point:

X coordinates: {16.0%, 0.0%, 16.0%, 58.0%, 100.0%}  
Y coordinates: { 0.0%, 37.5%, 75.0%, 100.0%, 75.0%}

These values are the same for all cases (a)-(c) in Fig. 9. Now consider the same feature but initially it has a different orientation as it is shown in Fig. 10:



**Figure 10.** The feature with a different orientation

The relationships of these points parameterized using the same method as above are very different:

X coordinates: {0.0%, 12.5%, 50.0%, 87.5%, 100.0%}  
Y coordinates: {0.0%, 70.0%, 100.0%, 70.0%, 0.0%}

What must we do to make these representations be the same? The answer is pretty obvious; we must rotate the feature until we find the same relationships. This can be done by iteration either sequentially or in a binary division fashion to find the best rotation to fit the parameters. *When we have found the same relationship set we have found the rotational metric to solve the rotational registration.*

Now, after we have found the orientation we can use the simple geometric properties to find the change in the x and y directions respectively. We use the coordinate values themselves to derive the solution to reverse the effect of the disproportional scaling. To use our previous example of the slanted half circle and the second shape (first mutation), we find the transformation in the x by *dividing the range of the x in the second shape by the range of the x in the first shape to find the scaling coefficient* is 2.0 since the range of the x values in the second shape is twice the range of the first shape. Similarly in Fig. 9 we could find the scale factor for the case (c) is 4.0 to the shape (a) and 2.0 to the shape (b). *The only factor that we had to guess was the rotation that occurred prior to the disproportional scaling.*

We also have the challenge of correctly positioning the points in the features such that we have a one-to-one correspondence in the different shape representations. This can be done using a BSD method following a first level Ramer algorithm implementation. In this method we do not use a threshold epsilon value at all which is a part of the Ramer algorithm. We simply divide the features into a factor of 2 sections using the extreme point as the first division (assuming there is no an ambiguous situation of where to place the point). This representation by itself is in fact enough to establish the rotational metric by which the disproportional scaling can be derived. Using this information, we can disproportionally scale the original data and then place points in the appropriate positions using a point placement technique such as pure BSD. Now we can describe the algorithm more specifically.

## **REVERSE DISPROPORTIONAL SCALING (RDS) ALGORITHM**

**The Reverse Disproportional Scaling (RDS)** algorithm consists of the following steps:

- Step 1. Pick a pair of polyline subsections to be compared. For this we should only compare subsections that are in the scale differential range we have set for our limit such as 1:2 to 2:1. This significantly reduces the number of subsections that we need to test.
- Step 2. If features are not already horizontal, rotate both features to the horizontal.
- Step 3. Find the max deviation point in the same manner that the Ramer algorithm does, but without using epsilon and only iterating the process once.
- Step 4. Parameterize the point set for x and y values as a percentage of each of the ranges independently.

- Step 5. Use the parameterized values of the mid point placements from step 3 to identify the rotational difference between the features when the disproportional scaling was applied. This can be done by iteration, binary search method, or by direct calculation. Then rotate both features to this orientation.
- In practice we are currently using a double loop to find the rotational combination that produces the best match of the parameterized values. Feature A is rotated from 0 to 90 degrees in 0.5 degree increments. For each of these cases feature B is rotated from 0 to 90 degrees in 0.5 degree increments and we compare the parameterized values of the rotated versions. To do this we take the absolute value of the difference in the x parameters and add it to the absolute values of the difference in the y parameters. This combined value gives us the degree of parameter matching. We use the lowest value found to identify the best rotational combination. A comparison value less than 1% means that neither the x parameter nor the y parameter values are more than 1% different and the combined sum of them is less than 1% as well. For the shapes used above in Fig. 9 we find a best rotation for feature A to be 45 degrees and the best rotation for feature B is 26.5 degrees. At this orientation the parameterized values are the closest and the comparison value is at a minimum.
- Step 6. Rotate the features by the best-fit parametric values found in step 5.
- Step 7. Determine the ranges of the x and y values for both. Then use the ratio of the x ranges to find the disproportional scaling factor for the x, and the ratio of the y ranges to find the disproportional scaling factor for the y. Thus disproportional scale factors have been computed.
- Step 8. Apply the disproportional scale factors to the original data to make two new images that are proportionally scaled.

Now we can apply the BSD method than needs proportional scales with confidence to confirm or negate the possibility that these are indeed *matching sections*.

The RDS algorithm has been implemented using C++ and has been found to successfully identify the spatial relationship at which the disproportional scaling took place, and the subsequent scale factors that occurred. Thus, we can transform the distorted version back into the original without having known the relationship initially. For the shapes used above in Fig. 9 we find a best rotation for feature A to be 45 degrees and the best rotation for feature B is 26.5 degrees, and at this combination the computed scale factor is 2.0 in the horizontal and 1.0 in the vertical which has been confirmed to be correct within the error range of the algorithm. The actual rotation value for the second feature is 26.51 degrees, thus an error of 0.01 degrees, which is well within the 0.5 degree resolution being used ( $\pm 0.25$  degrees).

## SUMMARY

The BSD method has been shown to work well in cases where there is no disproportional scaling problem. By guessing that the initial subsections were in fact correctly a matching pair, we can apply the Reverse Disproportional Scaling (RSD) algorithm and compute the scale factors required to compensate for any possible disproportional scaling. From here we can use standard methods or our own BSD method for confirming or negating the correspondence of the two datasets.

The Reverse Disproportional Scaling (RSD) method is rather simple. It does not require an extensive amount of computation and so can be implemented with a minimum of code. We must be careful however that we do not identify a matching section that has a disproportional scale factor (the ratio between the scaling in the x and the scaling in the y) that is different than that found for the majority of matching sections found. It is thus very important to screen the transformation data to identify the dominant cluster. The transformation should be applied to the entire image and a statistical evaluation performed using proximity measures such as were used during the grid evaluation fine-tuning method.

The work has been supported by the grant from the National Geospatial-Intelligence Agency.

## REFERENCES

- Bartl, R., and W. Schneider. Satellite image registration based on the geometrical arrangement of objects. *Proc. SPIE*, v. 2579, p. 32-40, 11/1995.

ASPRS 2005 Annual Conference  
Baltimore, Maryland ♦ March 7-11, 2005

- Brown, L. A Survey of Image Registration Techniques, *ACM Computing Surveys*, vol.24 (4), pp. 325--376, 1992
- Cohen, S. and L. Guibas, Partial Matching of Planar Polylines under Similarity Transformations. *Proceedings of the Eighth Annual ACM-SIAM Symposium on Discrete Algorithms*, 777-786, January 1997.
- Chase R., and Kovalerchuk B., Computationally efficient algorithm for structural feature matching, In: *Proceedings of International Conference on Imaging Science, Systems, and Technology (CISST'2003)*, Las Vegas, p. 253-259, 2003.
- Dare, P. M. and Dowman, I. J. Automatic registration of SAR and SPOT images based on multiple feature extraction and matching. In: *Proceedings of IEEE International Geoscience and Remote Sensing Symposium*, Honolulu, Hawaii, 2000.
- Dey, T., H. Edelsbrunner, and S. Guha. Computational Topology. In *Advances in Discrete and Computational Geometry (Contemporary mathematics 223)*, ed. B. Chazelle, J. E. Goodman, and R. Pollack, American Mathematical Society, 109—143, 1999.  
<http://citeseer.nj.nec.com/dey99computational.html>
- Eppstein D., *Geometry in action: Cartography and Geographic Information Systems*, 1999  
<http://www.ics.uci.edu/~eppstein/gina/carto.html>
- Hirose, M., Furuhashi, H., Kitamura, N., and Araki, K. Method for automatic registration using real-time multiview range images. *Proc. SPIE Vol. 4572*, p. 174-182, 10/2001.
- Kovalerchuk B., and Schwing, J., Mathematical support for combining geospatial data, In: *Proceedings of International Conference on Imaging Science, Systems, and Technology (CISST'2001)*, Las Vegas, 2001, pp. 485-491.
- Kovalerchuk B., and Schwing, J., Algebraic relational approach for geospatial feature correlation, In: *Proceedings of International Conference on Imaging Science, Systems, and Technology (CISST'2002)*, Las Vegas, 2002, pp. 115-121.
- Kovalerchuk B. and Sumner W., Algebraic relational approach to conflating images, In: *Algorithms and technologies for multispectral, hyperspectral and ultraspectral imagery IX. Vol. 5093, International SPIE military and aerospace symposium, AEROSENSE, Orlando, FL, April 21-25, 2003.*
- Kovalerchuk, B., Sumner, W., and Schwing, J., Image Registration and Conflation Based on Structural Characteristics, In: *Proceedings of International Conference on Imaging Science, Systems, and Technology (CISST'2003)*, Las Vegas, 2003, pp. 239-245.
- Kovalerchuk, B., Sumner W., Curtiss, M., Kovalerchuk, M., and Chase, R., Matching Image Feature Structures Using Shoulder Analysis Method, In: *Algorithms and technologies for multispectral, hyperspectral and ultraspectral imagery IX. Vol. 5425, International SPIE military and aerospace symposium, AEROSENSE, Orlando, FL, April 12-15, 2004.*
- Kovalerchuk B., Schwing J. (eds.), *Visual and Spatial Analysis: Advances in Data Mining, Reasoning and Problem Solving (Eds.)*, Springer, 2004.
- Mal'cev A.I. *Algebraic Systems*, Springer-Verlag, New York, 1973
- Pinz, A., Prantl, M., and Ganster, H. A robust affine matching algorithm using an exponentially decreasing distance function. *J.UCS - Journal of Universal Computer Science*, Springer, 1(8), 1995.
- Zitová T., Flusser J., Image registration methods: a survey, *Image and Vision Computing*. 21 (11), 2003, pp. 977-1000