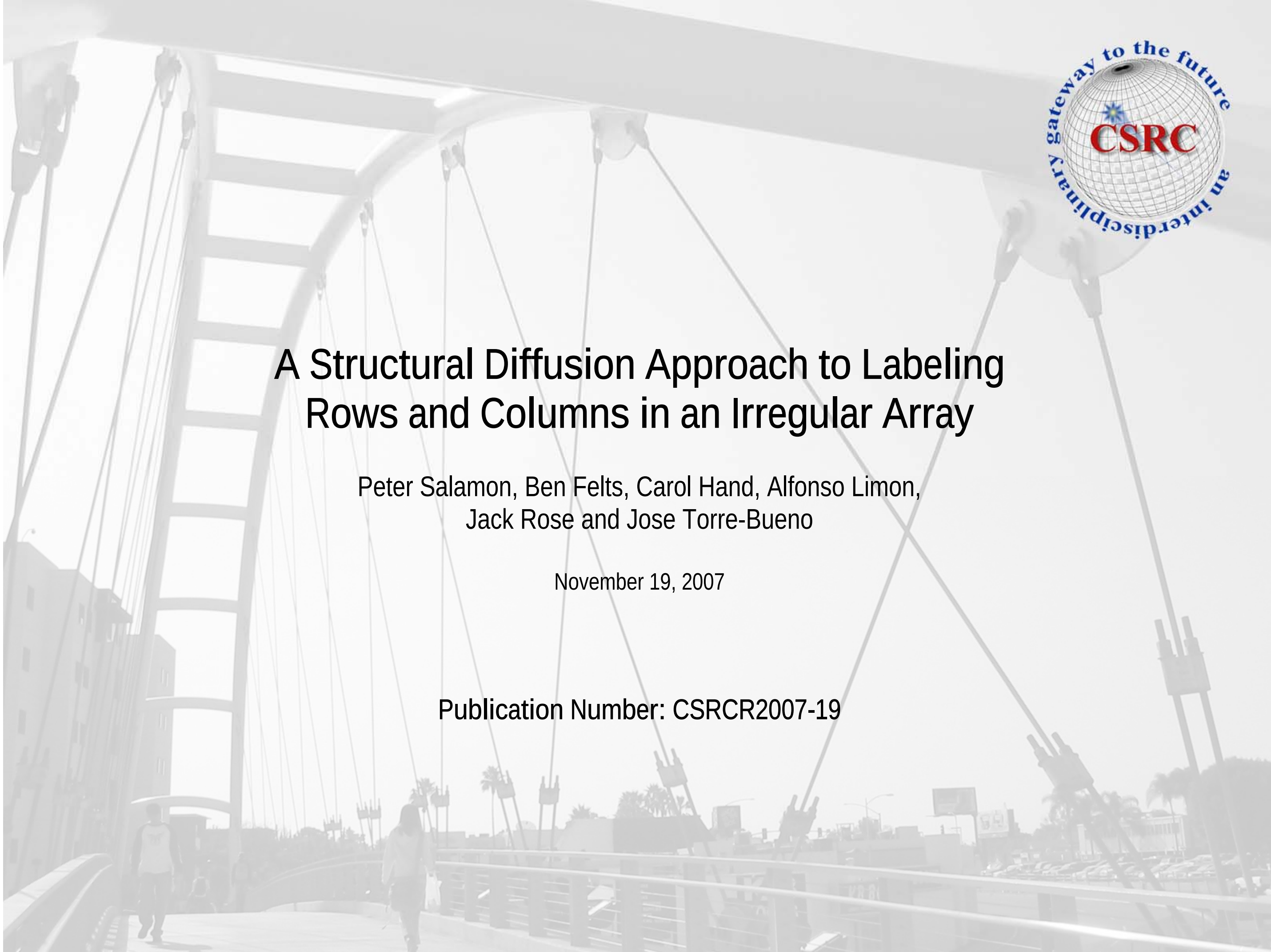


A Structural Diffusion Approach to Labeling Rows and Columns in an Irregular Array

Peter Salamon, Ben Felts, Carol Hand, Alfonso Limon,
Jack Rose and Jose Torre-Bueno

November 19, 2007

Publication Number: CSRCR2007-19

Computational Science &
Engineering Faculty and Students
Research Articles

Database Powered by the
Computational Science Research Center
Computing Group & Visualization Lab

COMPUTATIONAL SCIENCE
& ENGINEERING

SAN DIEGO STATE
UNIVERSITY

Computational Science Research Center
College of Sciences
5500 Campanile Drive
San Diego, CA 92182-1245
(619) 594-3430

A Structural Diffusion Approach to Labeling Rows and Columns in an Irregular Array

Peter Salamon, Ben Felts, Carol Hand, Alfonso Limon, Jack Rose and Jose Torre-Bueno

Department of Mathematical and Computer Sciences
San Diego State University
San Diego, CA 92182-7720

ABSTRACT

A robust algorithm is presented for labeling rows and columns in an irregular array. The algorithm is based on hierarchical pattern matching to a local lattice which is used as a template. Starting from the best local match, the pattern is expanded hierarchically to encompass the entire array. An application to labeling digitized images of an array of tissue sections mounted on a microscope slide is discussed.

Introduction

In a combinatorial approach to tissue response, an experimental treatment on diseased tissues can be tested simultaneously by placing an array of several hundred tissue samples on a microscope slide. To produce a series of nearly identical slides, cylindrical tissue samples are embedded in a block of paraffin. Successive slices of paraffin are then mounted on the slides and the series of slides is subjected to a battery of treatments. The sheer volume of data generated by this technique necessitates automatic processing, which is performed on digitized images of the stained slides.

Before further processing can proceed, the regions of the image corresponding to each tissue sample needs to be identified. The digitized image is analyzed to locate connected regions and their centroids. The algorithm described in this paper works from the list of centroids; the algorithm either associates to each centroid a row and a column in the array or identifies the centroid as a suspect point which should be excluded from the regular array.

Several image irregularities hinder machine determination of the row and column coordinates. Even when a guide is employed, the regularity with which the cylinders of tissue can be placed into the paraffin is less than perfect. The tissue sections are subject to deformation during the transfer to the slide, and some samples fail to adhere. Entire rows of samples are intentionally omitted to separate different groups of samples. The digitization process introduces further noise to challenge the row-column identification. The noise level is sufficiently high that a simple round to the nearest integer in

$$n_{cols} * X / X_{max} \tag{1}$$

does not reveal the centroid's column coordinate, where X is the x coordinate of the centroid in question, X_{max} is the greatest x coordinate of all the centroids, and n_{cols} is the number of columns on the slide.

The Problem

Figures 1 and 2 below show two different instances of the problem. The instances consist of arrays of (x,y) pairs representing the positions of the centroids of the tissue samples on the slides. In each figure, we show two close-ups of troublesome regions. Figure 1a seems to be fairly regular to the eye with well-defined rows and columns, but zooming into the data set (figures 1b and 1c) illustrate problem areas. Figure 1b illustrates problems due to missing points and deviation of points from nice straight columns. Figure 1c illustrates two different problems: two dots lie in what seems to be a missing row (third row from the top) and the correspondence between the bottom rows in the left half and the right half of the image is not clear. There are various reasons why this might happen, ranging from human error to shearing of the paraffin during handling, but this type of shift makes it impossible to know with certainty what is the correct labeling in these areas.

The problem of sparse data is also evident in various regions of figure 2a. Figure 2b shows a close-up of the top left hand corner of figure 2a. It is difficult to know the row and column numbers of the points in the top left corner, especially when using an algorithm based on local structure. Finally, figure 2c shows a close-up of the bottom left corner. The problems encountered in labeling the datasets in figures 1 and 2 are detailed in the *Results* section below.

The Solution

We have developed an algorithm for labeling rows and columns in irregular arrays of dots which robustly handles the irregularities described above. The basic philosophy behind the algorithm is to proceed via a sequence of deductions concerning the lattice structure. At each stage of the algorithm, we work from information assumed correct to a high degree of certainty. The deductions we make are of the form: if a certain centroid is in row n, then the centroid just above it must be in row n+1. We begin from a region that conforms well to our template of a local-lattice structure and spread our labeled region out from this initial seed. In so doing, the algorithm essentially works through a process of structural diffusion.

Structural diffusion is a model that has been used in the study of liquid structure [Baer et al. 1995]. The basic idea behind the model is that the local structure in the liquid strongly resembles the regular lattice structure of a solid but that the longer-range structure breaks down and undergoes a distortion in moving from point to point. The process is modeled as a diffusion process in the space of local structures and the parameters of such structures are fit to neutron scattering experiments to characterize the global structure in the liquid. In what follows we make use of this analogy in two ways. First, we associate to each location a local structure, which is a best match to the local configuration -- a point and its .nearest neighbors. Second, our algorithm moves systematically outward from a starting position mimicking a diffusion process. The analogy stops here; the diffusion described below is a diffusion of identification.

a. Locate Starting Point

As our first step, the algorithm searches for a good starting region defined as a region of centroids, which is the best match to our template of a regular local lattice. The local lattice that the algorithm uses as its template consists of a three-by-three grid of points representing a center and its eight nearest neighbors. For each point in the input list, the algorithm selects a best local lattice and calculates a measure of the goodness of fit. The points are then sorted by goodness of fit to determine the starting point.

The measure we used for the quality of fit at a given lattice site, (x_c, y_c), was computed as follows. The lattice points relative to the center (x_c, y_c) are given by

$$(2) \qquad (x_{grid}, y_{grid}) = (x_c, y_c) + K \cdot v_a + L \cdot v_b$$

where *K* and *L* are integers and v_a and v_b are the two lattice vectors that describe the local lattice and that remain to be selected as part of the fit. The eight closest centroids *N8* $=\{(x_k, y_k), k=1,\ldots,8\}$, are each associated with a node on the model grid by minimizing

$$(3) \qquad (x_k - x_{grid})^2 + (y_k - y_{grid})^2$$

over the values of *K* and *L* for any given values of v_a and v_b, where (x_{grid}, y_{grid}) are as defined in Eq. (2). The grid parameters v_a and v_b are then adjusted to minimize the sum of the squared distances from the actual positions of the eight points closest to (x_c, y_c) and their nearest lattice sites. Specifically,

$$(4) \qquad \text{Fit}(x_c, y_c) = \underset{v_a, v_b}{Min}\left(\sum_{k=1}^{8}\left((x_k - x_{grid(k)})^2 + (y_k - y_{grid(k)})^2\right)\right)$$

The sum of these eight squared distances is the measure of the goodness of a region's local lattice match.

Given reasonably close starting values of v_a and v_b, a local optimization can be carried out analytically. The choice of *K* and *L* for a given center (x_c, y_c), neighbor (x_k, y_k), and lattice vectors v_a and v_b is achieved by expressing the displacement (x_k-x_c, y_k-y_c) as a linear combination of v_a and v_b, and rounding the coefficients to the nearest integers. Using these integers K_k and L_k for the ($x_{grid(k)}$, $y_{grid(k)}$) positions, the resulting objective function is quadratic in the four unknowns $v_a = (v_{ax},v_{ay})$ and v_b, $= (v_{bx},v_{by})$. Setting the four partial derivatives of the objective function in Eq. (4) equal to zero, gives four linear equations which may be solved for the optimal values of v_a and v_b,. The solution determines the horizontal and vertical vectors that generate the best model grid for the region. The starting values of v_a and v_b are

$$v_a = (X_{max}/n_{cols}, 0) \quad \text{and} \quad v_b = (0, Y_{max}/n_{rows}).$$

Once a region with a good fit value is found, its v_a and v_b are used as initial values for the next optimization.

The previous paragraphs describe how each centroid is assigned an associated pair of lattice vectors v_a and v_b and a Fit value. The starting point for the diffusion is the centroid with the best fit value. It is assigned relative row and column coordinates (I,J)=(0,0) and its lattice vectors are denoted by v_a* and v_b*.

b. Diffuse

Before proceeding with the diffusion outward from this center, we scan the list of centroids for suspicious points; these points will not be visited during the diffusion. Such suspicious points are identified by either (a) having a Fit value in Eq. 4 above some tolerance ε_1, or (b) having a match between the local lattice vectors at the point and the local lattice vectors at the starting center point (which had the best Fit) above some tolerance ε_2. Labeling these points as suspicious is in keeping with our mandate of assigning coordinates only when there is high confidence in the assignment. The selection of the two tolerances ε_1 and ε_2 is discussed below in the results section. Once these points have been labeled as suspicious, the diffusion proceeds.

At each step of the diffusion, the algorithm selects a centroid (x_k,y_k) to be used as the next center of a new local lattice and then assigns row and column coordinates to all the points lying sufficiently near to lattice sites of a 5x5 lattice centered at (x_k,y_k). The new center must fulfill four criteria:

(a) it has not been labeled *suspicious* as described in the previous paragraph
(b) it has not been labeled *ambiguous* as described below
(c) it has not been previously used as a center in the algorithm
(d) it has been assigned relative row and column coordinates (I,J).

Among the points that fulfill these four conditions, we select the next center as the one which lies closest to the starting center (I,J)=(0,0). Figures (3a,b,c) illustrate the diffusion pattern the algorithm generates as it looks for neighbors to label. Note that condition (d) implies that initially only the starting point can be chosen as a center. We denote the coordinates of the next center by (x_k,y_k) and its assigned relative row and column coordinates by (I_k,J_k). The algorithm proceeds to generate the 25 points in the local 5x5 lattice centered at this point; for each K and L value between –2 and 2, we find the centroid (x,y) closest to

$$(x_{grid}, y_{grid}) = (x_k, y_k) + Kv_a^k + Lv_b^k$$

and tentatively assign it the coordinates

$$(I_{new}, J_{new}) = (I_k, J_k) + (K, L).$$

These tentatively assigned coordinates are actually assigned to this centroid only if no previous coordinates have been assigned to this point and the distance

$$\sqrt{\left(x - x_{grid}\right)^2 + \left(y - y_{grid}\right)^2}$$

is less than a specified tolerance ε_3. If coordinates have previously been assigned and they do not match the newly calculated coordinates, then the centroid is marked as *ambiguous* and thus will not be used as a center.

The diffusion process continues until all eligible dots have been used as the center dot. The algorithm then translates the relative row and column coordinates (I,J) to absolute row and column coordinates

$$(M, N) = (I, J) - (I_{\min}, J_{\min}) + (1,1)$$

where I_{min} and J_{min} represent the smallest assigned I and J values. Besides the row and column assignments, the output of the algorithm includes lists of suspicious, ambiguous, and unlabeled centroids.

c Parameter Settings

The algorithm described above requires specified values for three tolerances ε_i, i=1,2,3. These tolerances specify limiting values of certain quantities calculated from the (x,y) coordinates of the centroids and, as such, are highly dependent on the scale used in digitizing the image. To eliminate this dependence, the values of these three ε's are calculated from three user specified in the grid tolerances δ_i, i=1,2,3 which are measured in units of a lattice spacing. This section describes the correspondence between these ε's and δ's and specifies precisely how the ε's are used.

To rescale from units of a lattice spacing to the units used in the digitized grid, it is convenient to define a parameter L, which represents the mean lattice spacing[1]

$$L = \frac{1}{2}\left(\left\| v_a^* \right\| + \left\| v_b^* \right\|\right)$$

where v_a^* and v_b^* are the lattice vectors of the best lattice as defined at the end of the *Locate Starting Point* section.

(a) Epsilon 1

$$\varepsilon_1 = N_{nn}\left(L\delta\right)^2$$

where N_{nn} is the number of nearest neighbors (which our algorithm currently takes as eight). Since ε_1 is used in the test

$$\varepsilon_1 < \mathrm{Fit}(x_c, y_c)$$

and $\mathrm{Fit}(x_c,y_c)$ is a sum of N_{nn} squared deviations, δ_1 represents the average deviation between a grid point and a neighbor point in units of L.

(b) Epsilon 2

The second user specified tolerance, δ_2 is used to label suspicious points on the grounds that their lattice vectors deviate too much from the ideal. In the algorithm, this is measured by

[1] It is possible to separate the scales of Δx and Δy but since our application did not require it, we did not do so.

$$\varepsilon_2 > \left| \frac{\| v_i \|}{\| v_i^* \|} - 1 \right| \; ; i = \mathrm{a,b}.$$

Since ε_2 specifies a fractional difference, we take

$$\delta_2 = \varepsilon_2.$$

This makes δ_2 a fractional threshold comparing the size of the locally optimized lattice to the best lattice. This test condition serves to eliminate harmonics[2]. If the optimized lattice is too small, we say that the lattice vectors correspond to higher harmonics; similarly, if the lattice is too large we call it lower harmonics.

(c) Epsilon 3

The last user specified tolerance ε_3 is used in

$$\varepsilon_3 > \sqrt{\left(x - x_{grid}\right)^2 + \left(y - y_{grid}\right)^2}$$

to decide whether the neighbor (x,y) of our local center should be labeled based on its proximity to the nearest grid point emanating from the local center. Defining δ_3 by

$\delta_3 = \varepsilon_3 / L$

makes δ_3 scale independent.

(d) Experimental values for δ's

Table 1 below shows the range of parameter values that produced satisfactory performance using our labeling algorithm. The table also summarizes the meaning and use of each of the parameters.

Tolerances	Liberal	Conservative	Meaning	Use
δ_1	0.20	0.15	Mean deviation of neighboring points from the local lattice	Decide whether to use centroid as center
δ_2	0.30	0.10	Fractional deviation of local lattice vectors from best lattice vectors	Decide whether to use centroid as center
δ_3	0.25	0.15	Deviation of one neighbor to nearby grid point	Decide whether to assign label to neighbor

[2] If a set of points are well described by the lattice vectors v_a, v_b then they are also well described by $v_a/2$ and $v_b/2$. We call these locally optimal solutions harmonics.

Results

The algorithm described above performed satisfactorily on all datasets considered. The data shown in figures 1 and 2 were the most problematic. The troublesome regions of each figure are in the bottom left corner, shown in the close-ups in figures 1c and 2c. The centroids in these regions received labels that depended strongly on the parameters used and therefore served to define the range of reliable values shown in Table 1.

To discuss the problems encountered, it is useful to introduce three complementary ways to visualize the results of the algorithm. The first is just the finished version of the lattices generated during the diffusion phase. Figure 4 shows this view of the labeled dataset from figure 1. The left labeling was generated using conservative values of the tuning parameters δ while the right half used liberal values. Below each of the subplots showing all of the centroids is a labeled close-up of the bottom left corner corresponding to figure 1c. For legibility only the labels suspicious (susp), ambiguous (ambg) and unlabeled (unlb) were included in the figure. Note that conservative values for the parameters label fewer points and result in one unlabeled centroid while liberal values tend to overlabel and result in six ambiguously labeled points.

Note that the lattice visualization shows each edge (the line segment connecting two adjacent points in a row or column) as represented in six different local lattices. These lattices are well aligned if the six line segments appear as a single line segment. Small misalignments cause the line segment to appear thicker, and large misalignments produce shadows or completely separate traces. As a result, this way of displaying the data explains any difficulties encountered by the algorithm. The ambiguous labels are due to the very slanted local lattice clearly visible in the liberal labeling. This lattice does not show up in the conservative labeling since its center was labeled suspicious and therefore not used to nearby label points according to its local lattice.

The second visualization of the labeling is shown in figure 5. Again, the figure shows the result of the conservative and the liberal labelings for the dataset of figure 1. This view shows the grid generated by connecting each centroid to the adjacent centroids in its assigned row and column. This representation displays the global structure resulting from the assignments. The rows and columns can be easily discerned, and deviations from a regular grid shape stand out. Irregularities in the grid pattern identify centroids with questionable assignments. For example, the centroid left unlabeled by the conservative parameter values is seen to lie at a “fault line” where the grid to its left is shifted by about half a grid spacing relative to the grid to its right. In this situation, it is better for the algorithm to focus the attention of the user to this region than to come up with purported labels. Both extremes of parameter values fulfill this goal of pointing out problem areas.

The third visualization for this same dataset is shown in figure 6. This graphical representation displays the lines that are the least squares fit to the centroids in each row and in each column. These lines summarize the global structure that has been assembled by the algorithm. Any nonlinear trends or deviations within a row or column are clearly visible within this representation and show the variation of local structures relative to the global structure. The lone pair of centroids in the 5th row from the bottom are very easy to spot in this view. In retrospect it is also easy to spot these two points in the grid representation of figure 5 and represent another reason to call in a human operator. The "fault line" of figure 5 is also evident here in figure 6.

The least squares lines in this representation can be used to extend the global structure into sparse regions, and enable the assignment of row and column indices to centroids that were left unlabeled by the local structure algorithm. The grid can also be extended across empty regions to connect isolated groups of centroids.

The closeness of a centroid to the nearest grid intersection could have been used as a fourth measure of the confidence in its row and column assignment. The row and column assignments of centroids that are further from an intersection of the grid may be questionable. Using this measure would enable the algorithm to perform further consistency checks thereby resolving some ambiguities and identifying others.

All three visual representations make it easy to spot the missing rows and columns. The grid and the lattice representations make clearly visible any misalignment between two intentionally separated regions. Note that figure 4 shows how a region boundary projects the local structure of its region into the missing row or column. A misalignment between the regions will appear as a series of misalignments between the local lattices centered on opposite sides of the missing row or column. Thus, the local structure for the boundary centroids spans the missing row or column and resolves the offset between the regions. Recall that in order to span the gap across missing rows, the diffusion used 5 by 5 lattices to attempt labeling. Our visualization shows only local 3 by 3 lattices to minimize clutter in the image.

The sensitivity to parameter values is further illustrated by considering the labelings for the dataset from figure 2 shown in figures 7 (local lattice view), 8 (grid view) and 9 (lines view). Both the conservative and the liberal values for the parameters focus the operator attention on the problem area in the bottom left corner. Again we see that conservative settings leave many points unlabeled, while liberal settings label many points ambiguous. The best results are obtained by taking a conservative approach to allowing centers by keeping the conservative setting δ_1=0.15 while allowing liberal labeling of neighbors from approved centers δ_3=0.25. The third labeling in figures 7-9 show the results of the mixed setting δ_1=0.15, δ_2=0.30, and δ_3=0.25. This labeling shows that the culprit is the point in the second column which lies a little too far above the fourth row from the bottom.

Conclusions

We presented a robust algorithm for machine labeling of rows and columns in slightly irregular arrays of points. Arrays of samples are used in many areas, for instance 96 well plates, slides for combinatorial chemistry and multi well carriers for synthesizers. In all these cases current design relies on the wells or dots being in predictable positions so they can be processed or read by robotic equipment. If for any reason the positioning is not as expected, the only possible response is to shut down. Augmenting a vision system with this algorithm would allow robotic systems of this type to recover from minor positioning errors.

Beyond its possible uses in similar chemical contexts, the algorithm described in the manuscript is an interesting example of extracting *global* information from *local* structure. To accomplish this task, the algorithm hierarchically extends the structure from an initial seed where the match to a local template is optimal. It thus adheres to the classic paradigm for extracting global information described for example in [Duda and Hart 1973] or [Ballard and Brown 1982]. The present effort represents a novel example of this technique based on structural diffusion.

Several graphical representations of the output data were presented. These different representations show complementary aspects of the local and global structure in the data and enable fast visual analysis of any problems that the algorithm needs to bring to the attention of the operator.

References

1. Baer, S., Gutman, L., Silbert, S., J. Non-Cryst. Solids 192-193, v. 106 (1995).

2. Ballard, D.H., Brown, C.M., Computer vision, (Prentice Hall, Englewood Cliffs, 1982).

3. Duda, R.O., Hart, P.E., Pattern Classification and Scene Analysis, (J. Wiley, New York, 1973).

FIGURE CAPIONS:

Figure 1
An example of the arrays of points to be classified showing details of difficult areas.

Figure 2
A second example of the arrays of points to be classified, again showing details of difficult areas.

Figure 3
The algorithm proceeds by finding the most regular local lattice and diffusing out from that seed using the local lattice as a guide. The three portions correspond to progressively later times during the algorithm.

Figure 4
The dataset from figure 1 at the completion of labeling using both conservative and liberal values of the parameters. Note that conservative values for the parameters result in fewer labeled points and result in one unlabeled centroid while liberal values tend to overlabel and result in six ambiguously labeled points.

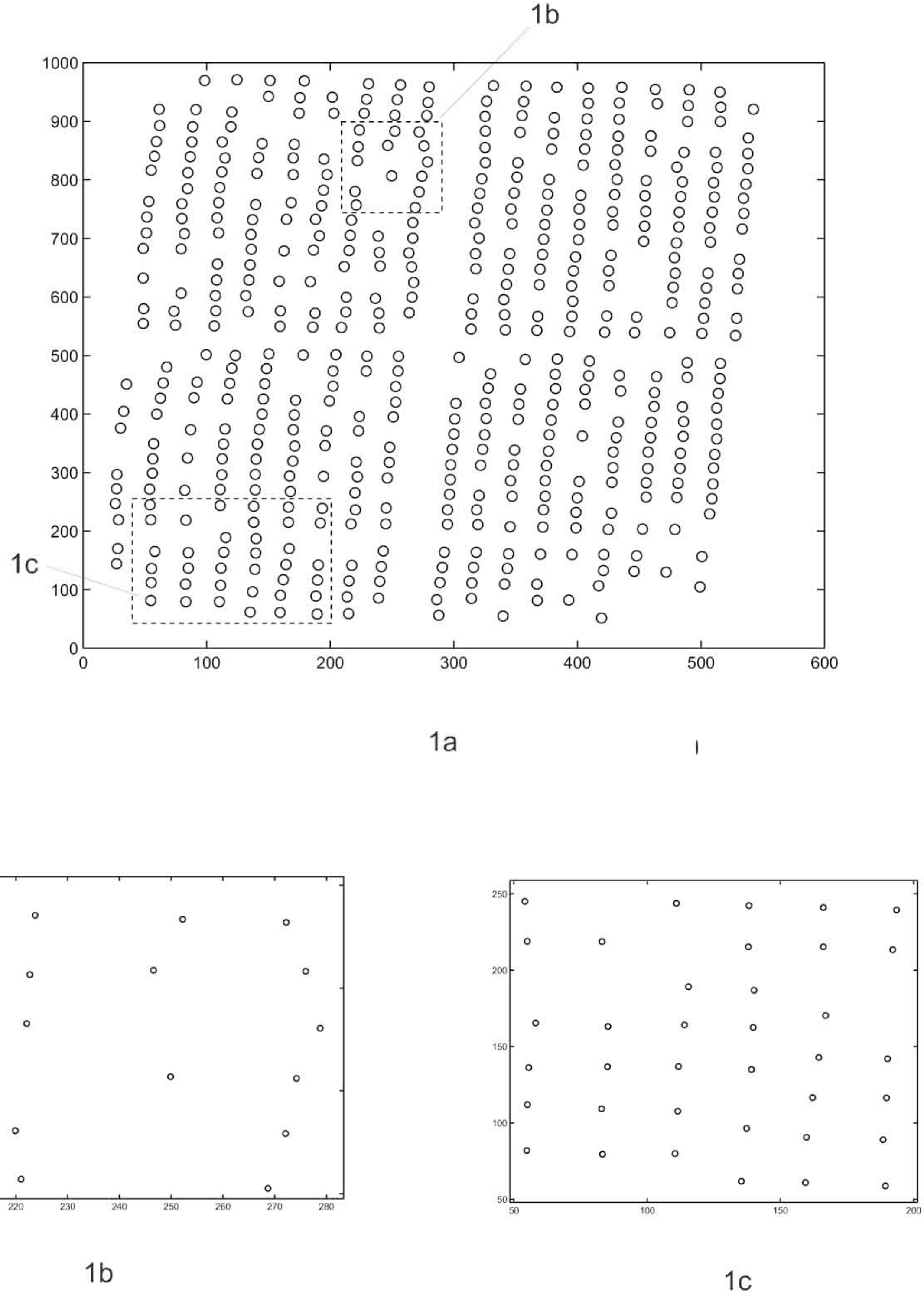
Figure 5
This view of the labeling from figure 4 replaces the local lattice lines by line segments connecting the labeled points. This representation displays the global structure resulting from the assignments. The rows and columns can be easily discerned, and deviations from a regular grid shape stand out. For example, the centroid left unlabeled by the conservative parameter values is seen to lie at a "fault line" where the grid is shifted by about half a grid spacing.

Figure 6
The third visualization for the labeled dataset of figures 1, 4 and 5. This graphical representation displays the lines that are the least squares fit to the centroids in each row and in each column.

Figure 7
The dataset from figure 2 is shown with all the local lattices drawn at the completion of labeling. The results are shown using conservative, liberal, and mixed values of the parameters. Again, conservative values for the parameters label fewer points while liberal values tend to overlabel.

Figure 8
The results for the example in figure 2 are shown in the connected grid points view.

Figure 9
The results for the example in figure 2 are shown overlayed by the least squares lines for each row and column.

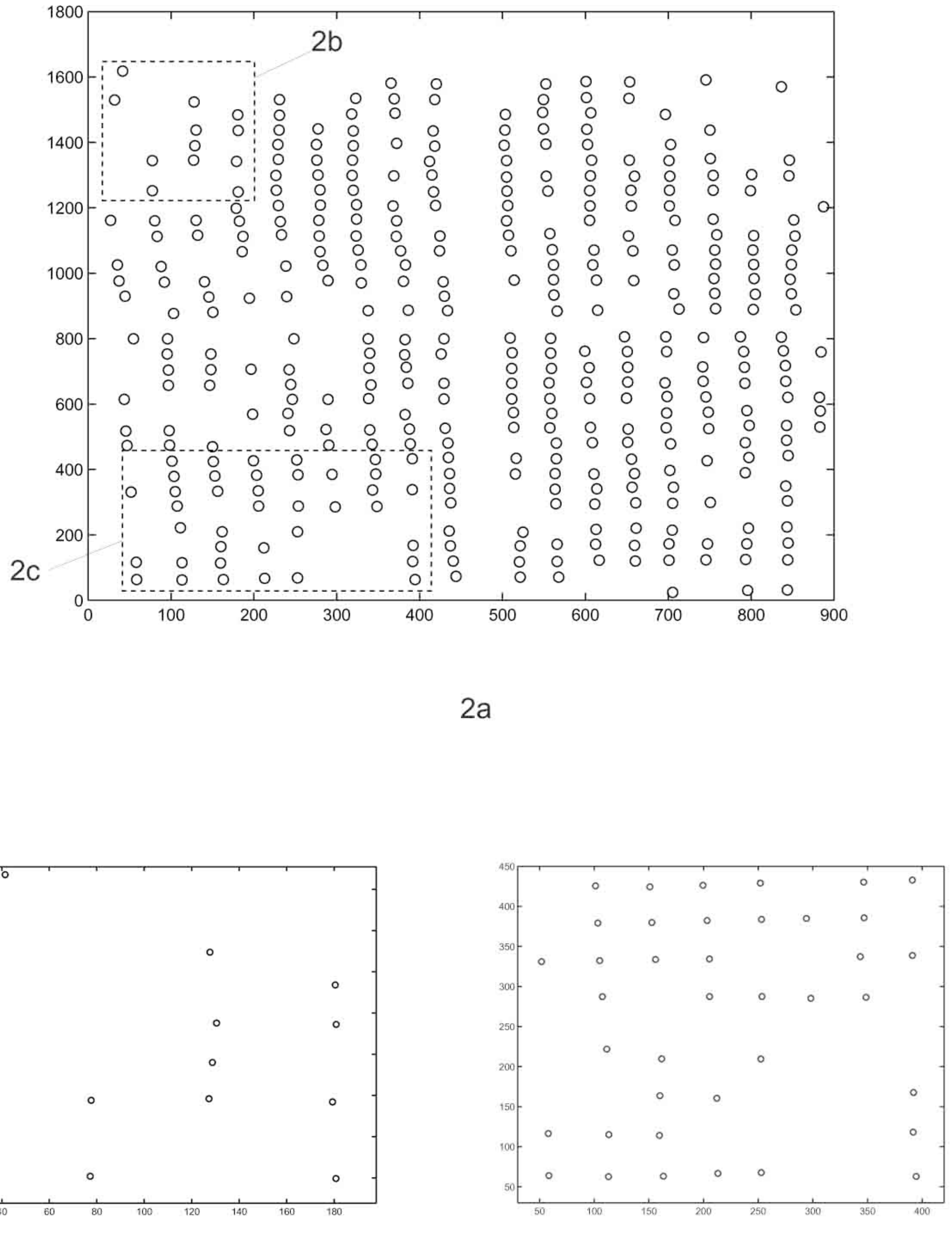


1a

1b

1c

Figure 1



2a

2b

2c

Figure 2

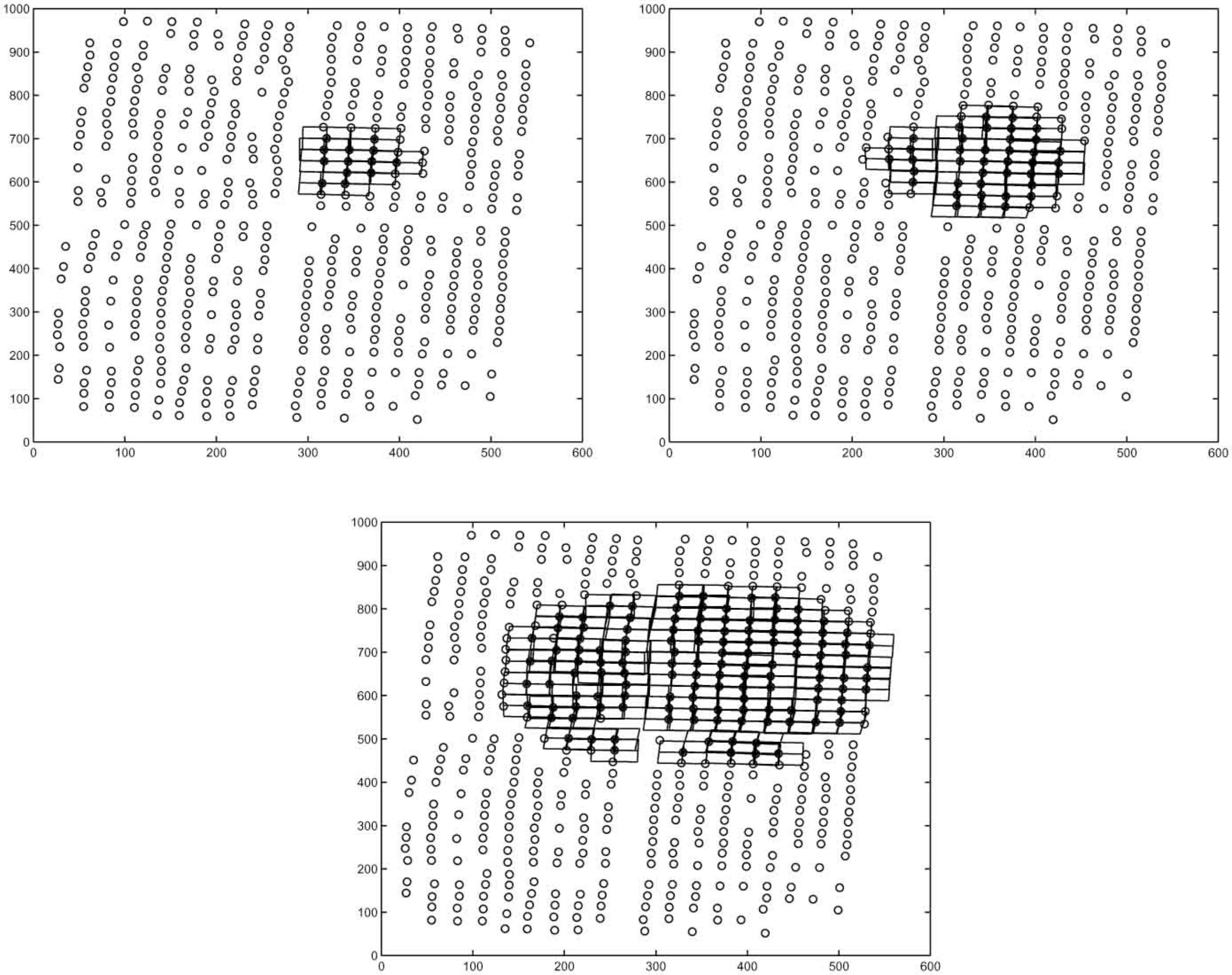
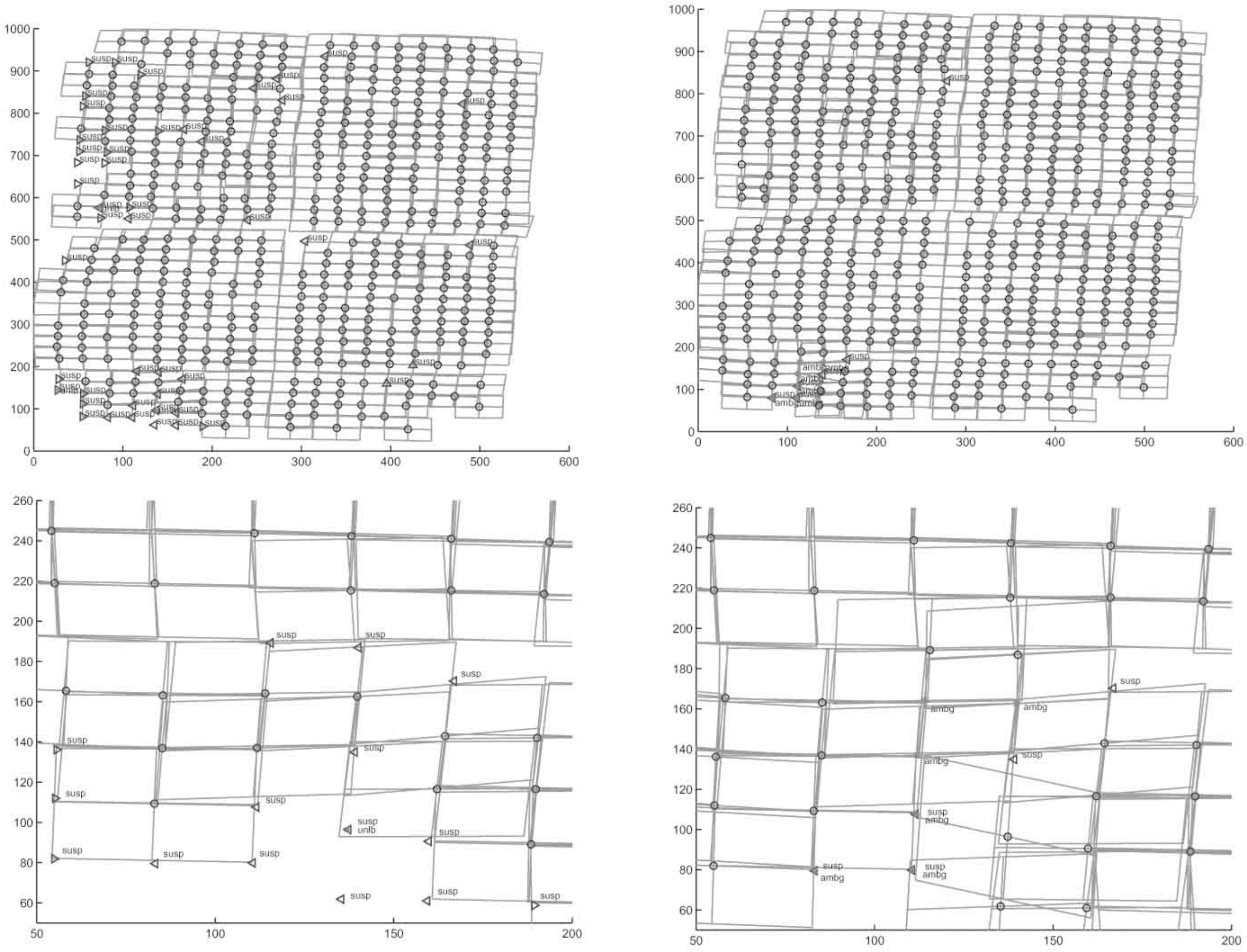

1000
900
800
700
600
500
400
300
200
100
0
0
100
200
300
400
500
600
1000
900
800
700
600
500
400
300
200
100
0
0
100
200
300
400
500
600
1000
900
800
700
600
500
400
300
200
100
0
0
100
200
300
400
500
600

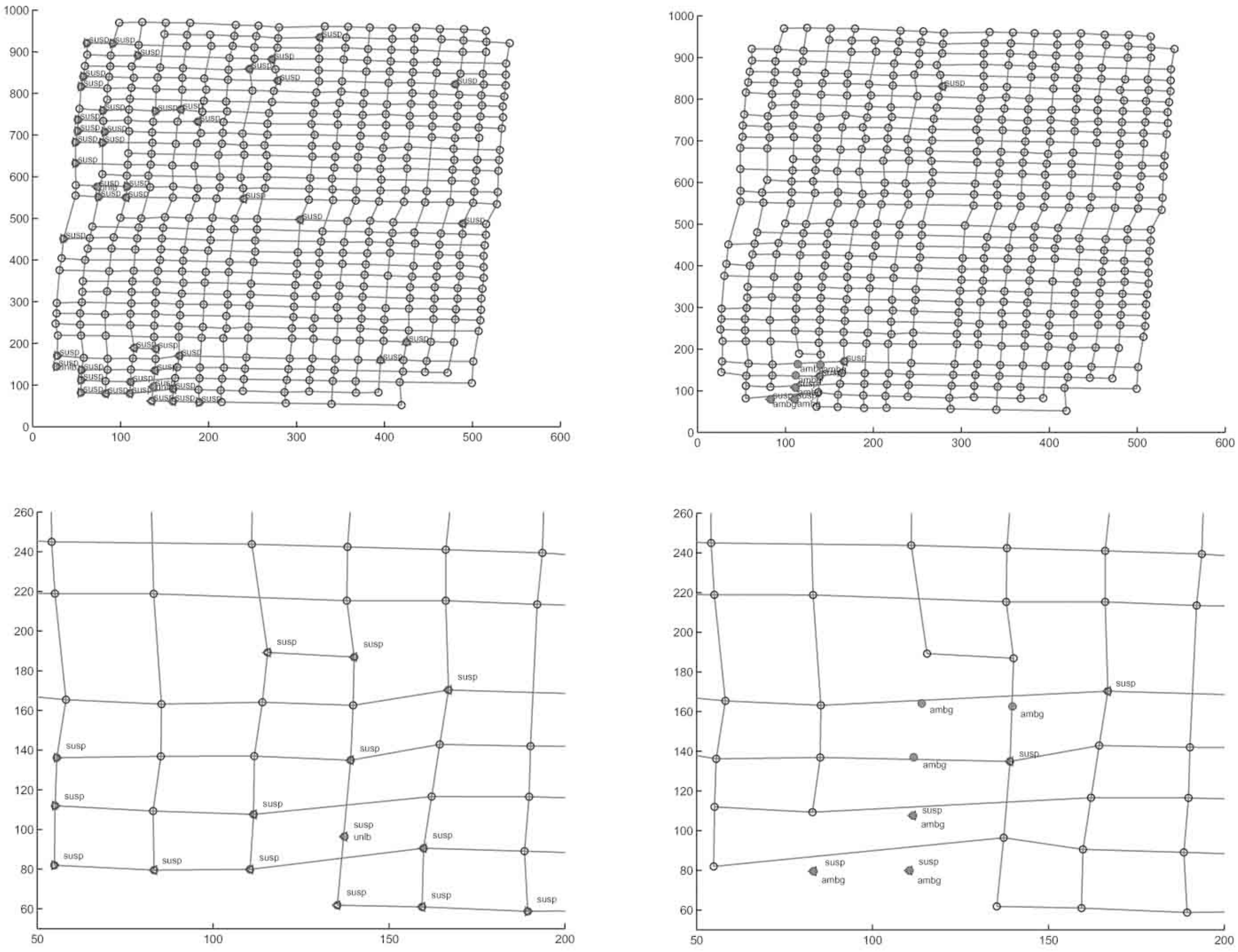
Figure 3



Conservative

Liberal

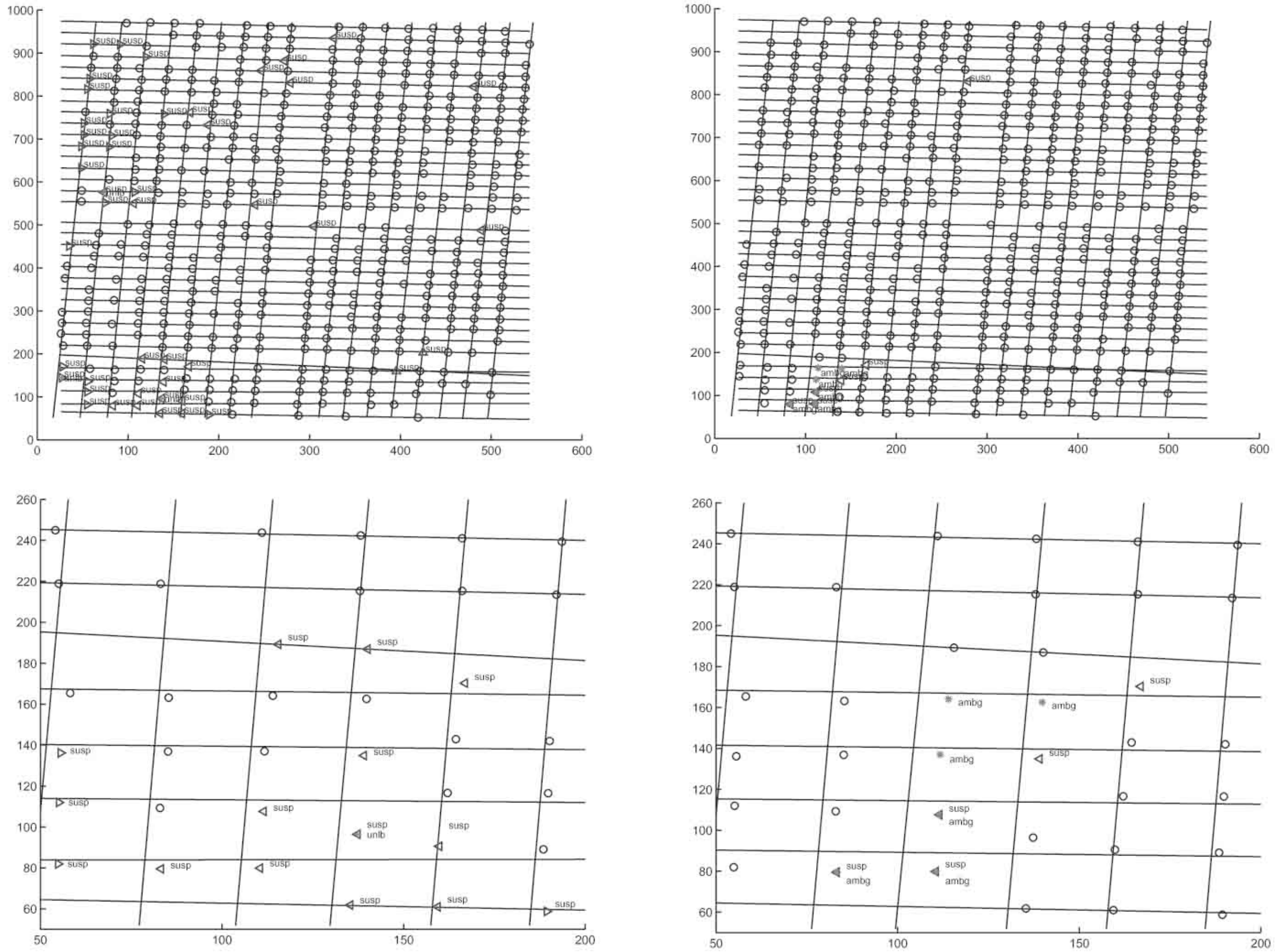
Figure 4



Conservative

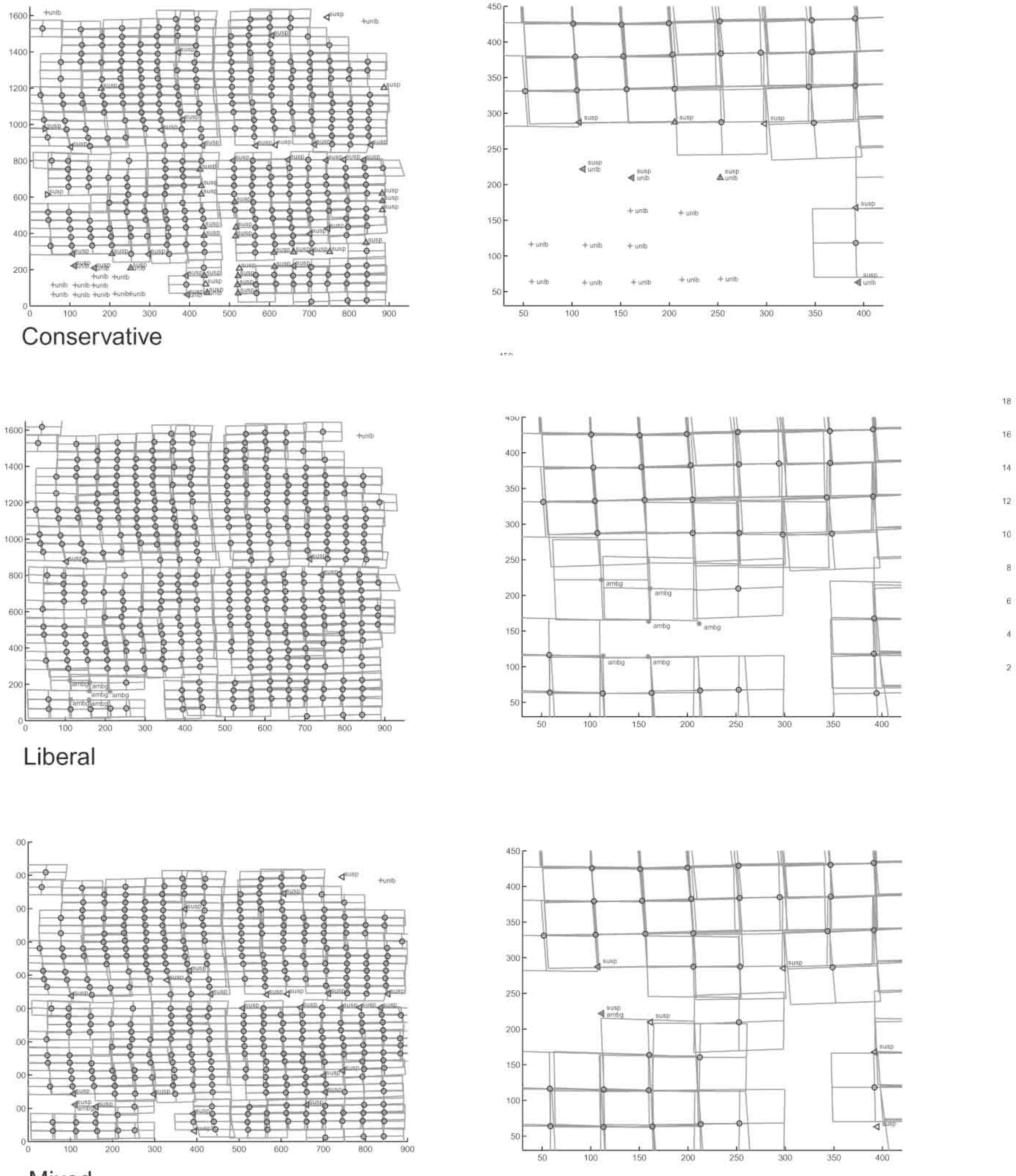
Liberal

Figure 5



1000
900
800
700
600
500
400
300
200
100
0
0
100
200
300
400
500
600
susp
susp
amb
ambg
260
240
220
200
180
160
140
120
100
80
60
50
100
150
200
susp
unlb
ambg

Figure 6

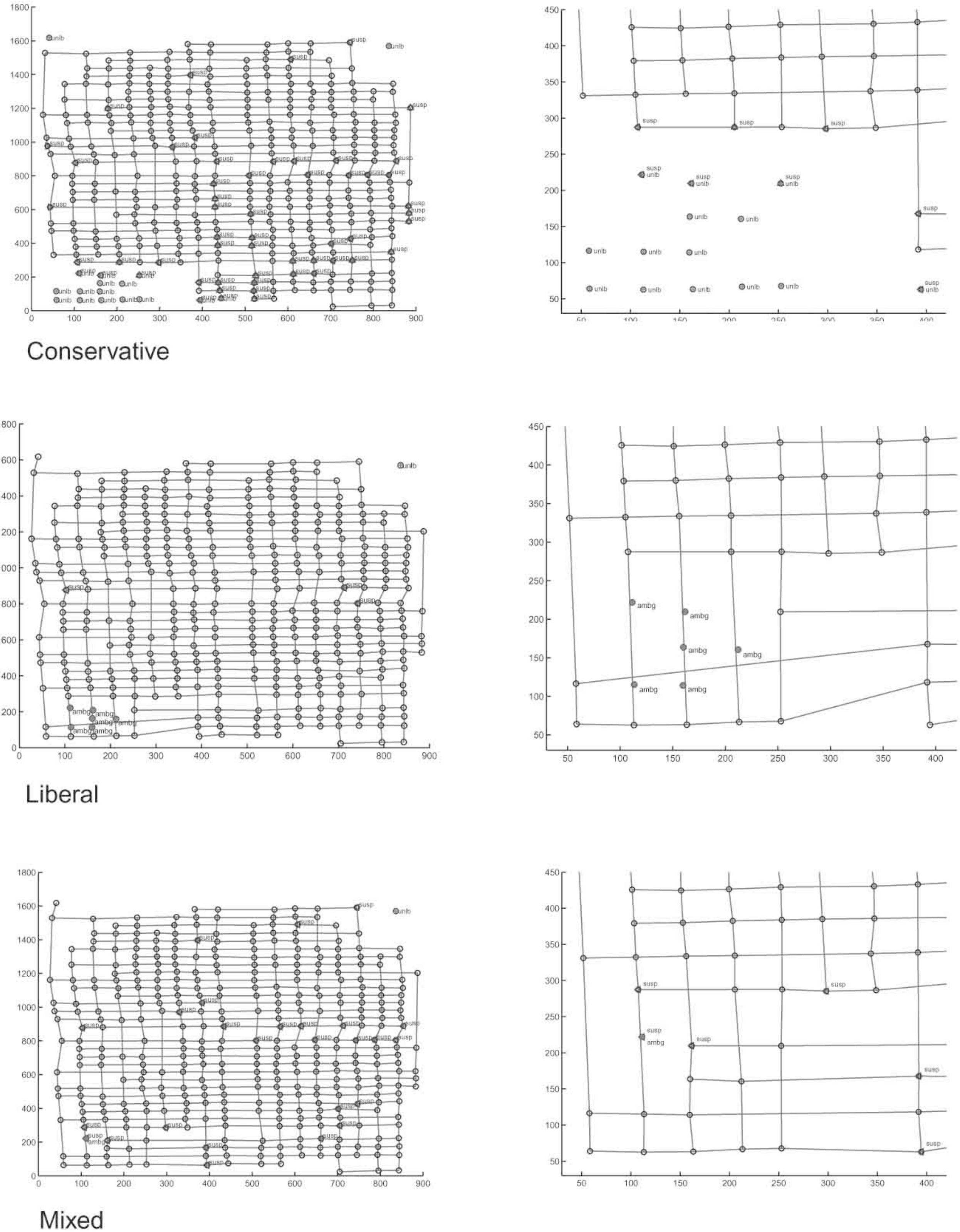


Conservative

Liberal

Mixed

Figure 7

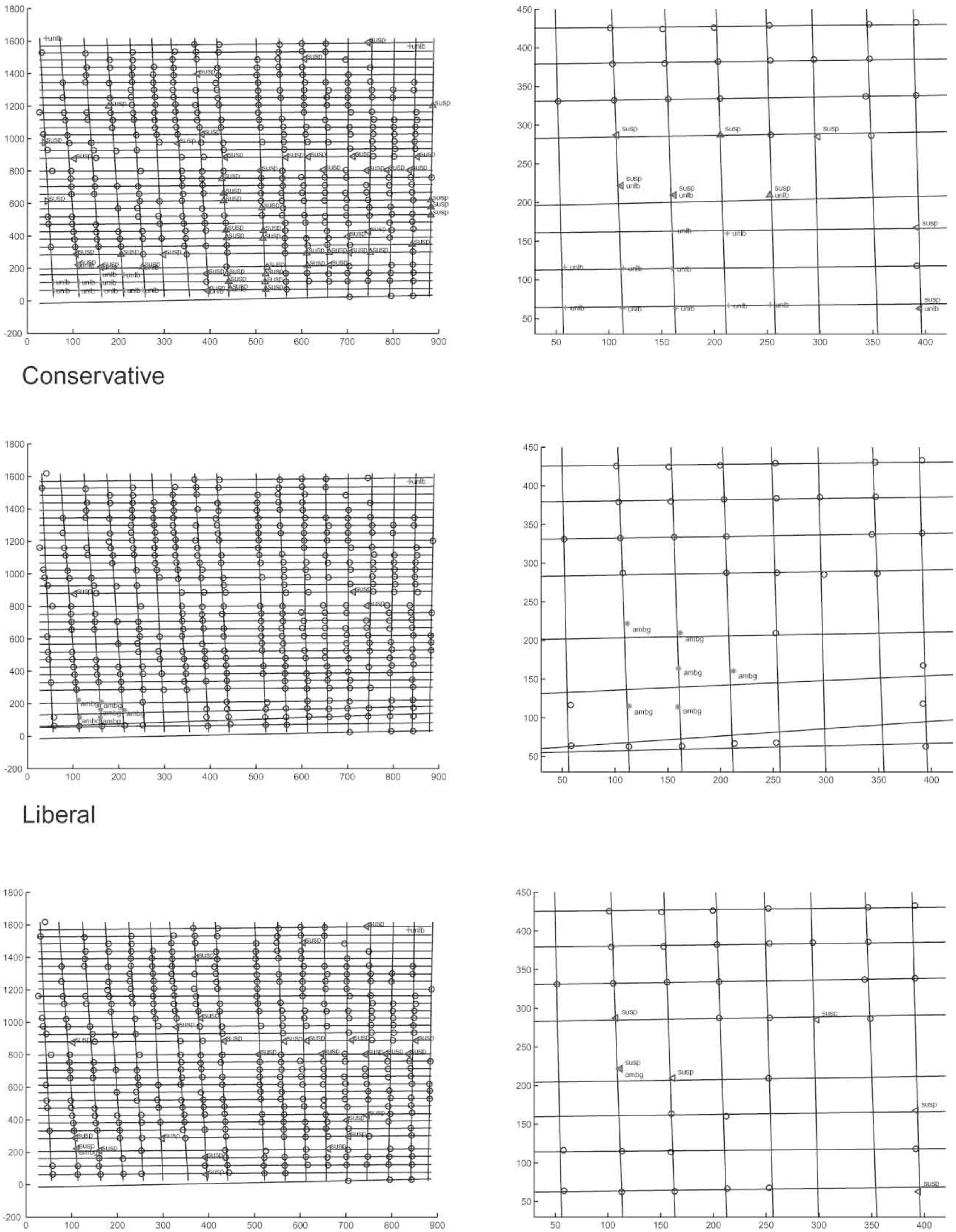


Conservative

Liberal

Mixed

Figure 8



Conservative

Liberal

Mixed

Figure 9