H: SCENE OBJECT POLYGONS

Like *Frequently Asked Questions*, a question is posed, e.g., *H1. What is a Scene-Object Polygon (SOP)?* Then, an answer is given.[1]

This tutorial deals with OBJECT.sml, its uses, and its options.

This script produces three output products:

1. A georeferenced Edge-Probability (EP) raster,
2. A set of georeferenced vector Scene-Object Polygons (SOPs), and
3. A text report.

These outputs are initially based on a single input raster, Rin. However, an EP raster from one run may be carried over for modification by OBJECT.sml during another run with another Rin raster. This feature lets you build SOPs based on two or more Rin rasters. In most cases, you will be working with just one Rin raster.

The author suggests that you use a Rin raster that has SRFI units. For example, Rin could be a SRFI raster for a particular spectral band, such as, SRFINA. Better yet, a Rin raster could be a Tasseled Cap (TC) raster from TASCAP.sml, e.g., TC Greenness. In any case, OBJECT.sml will accept a non-SRFI type raster as the Rin raster.

In any case, the TNTmips Spatial Data Editor can be used to modify the SOPs that are produced by OBJECT.sml. This follow-on activity also serves as an important Quality Assurance / Quality Control (QAQC) step.

In addition, after you run OBJECT.sml, you can attach sets of raster attributes to the final SOPs by using an easy-to-use process available from the TNTmips main menu. Then, Raster attributes *and* standard polygon attributes of each SOP element can be used to make a final Thematic Map.

These kinds of value-added modification and follow-on tasks would be done by using interactive TNTmips processes and tools that are available through the TNTmips main menu.

This overall combination of automatic processing and manual editing will produce a better map than one based strictly on pixel-level processing and spectral / temporal classification.

[1] Jack F. Paris, Ph.D., 2407 Maplewood Cir. E., Longmont, Colorado 80503 USA, *jparis37@msn.com* , 303-775-1195

In Brief …

This tutorial discusses key model concepts related to **OBJECT.sml**. If you are interested in a particular topic below, please go directly to it.

KEY MODEL-CONCEPT ITEMS

Quick Guide to Using OBJECT.sml …

If you are already familiar with SML functions and syntax … and you just want to Run OBJECT.sml, this Quick Guide will help you.

BEFORE you run OBJECT.sml …

- You may want to use SRFI.sml to produce the SRFI raster that OBJECT.sml could use as the input raster, Rin. This preliminary step is not an absolute requirement. OBJECT.sml scans the Rin raster to determine the range of tentative Edge Probability (EP) values. Then, the script applies a scaling factor (epFac) as it makes final EP values. Thus, the final EP values will have been standardized to a fixed upper limit of 10,000. Based on the final EP raster, OBJECT.sml then produces a set of initial SOPs. The boundary lines of these SOPs are then examined for possible deletion based on the relative Rin medians of neighboring SOPs. When the designated lines are deleted, the final SOPs are written to a TNTmips Project File called SOP.rvc. The final EP raster is also written to a TNTmips Project File called EP.rvc.
- Alternatively, you may use TASCAP.sml to produce a set of Tasseled Cap (TC) rasters that are related to a full set of SRFI rasters. TC rasters have "SRFI" units; therefore, they are well-suited for use as the Rin raster for the OBJECT.sml script. OBJECT.sml will work with other vegetation-index rasters that do not have SRFI units.

DISCUSSION:

Using a TC raster as the Rin raster has distinct advantages. Suppose you are trying to make a map of vegetated scene objects, e.g., trees, forests, agricultural management zones, and/or agricultural fields. In this case, a TC Greenness raster is your best choice for the Rin raster. The script lets you set limits on the Rin raster values (from floorRin to ceilRin). If you set the floorRin value for a TC Greenness raster to a level greater than zero, the script will make SOPs *only* for vegetated areas. If you also set the ceilRin value for a TC Greenness raster to a level that is below the usual maximum, then the script will not create internal boundaries within densely vegetated areas.

Another advantage of using TC Greenness as the Rin raster is that boundaries visible only in a non-TC-Greenness raster, e.g., TC Brightness, will not be produced by the script.

AFTER you start the script, it will ask you to provide values for specific control parameters via a series of **Popup Windows**. These parameters are discussed on the next page. If you are making another run, you will also be asked to identify the location of the EP raster from the previous run.

- **CONSOLE-WINDOW ADJUSTMENT:** This pause lets you use your mouse to adjust the size and placement of the Console Window. You need to be able to view its contents as the script runs and "prints" text data to it. Starting with TNTmips Version 7.1, you need to make this adjustment only the first time you use SML. When you have completed your adjustments, click OK.
- **SELECT A RASTER OBJECT FOR Rin:** Navigate to the location of the raster that you want to use as the Rin input raster. See the previous discussion. When you have made this selection, the related Raster Name will appear in the Console Window with its values range and with its cell sizes.
- **EP RASTER OPTION:** If this is the first time you are using OBJECT.sml, enter **1**. If you want to modify the EP raster from a previous run by using information from a different Rin raster, enter **2**. Then click OK.
- **INPUT-RASTER FLOOR-VALUE, floorRin:** The default floorRin value is the minimum value in the Rin raster. You may enter a different floorRin value. See the previous discussion. As with other parameters, you can rerun OBJECT.sml with a new set of parameters as you optimize the results.
- **INPUT-RASTER CEILING-VALUE, ceilRin:** The default ceilRin value is the maximum allowed value in the Rin raster. You may enter a different ceilRin value. See the previous discussion. As with other parameters, you can rerun OBJECT.sml as you optimize the results.
- **MINIMUM ALLOWED EDGE-PROBABILITY VALUE, minEP:** Weak SOP boundaries are associated with very low EP values. To delete more weak SOP boundaries, enter a minEP value that is higher than **1**, e.g., 100. The allowed range for minEP is from 1 to 5,000. To retain all of the SOP boundaries, whether weak or strong, enter a value of **1** for minEP.
- **EXTRAPOLATION-LINE LENGTH, eLen:** OBJECT.sml uses a Sobel 3 x 3 edge-enhancement filter algorithm to calculate EP values. This 3 x 3 filter matrix has 9 pixels. For each of the 8 outside pixels in this matrix, OBJECT.sml estimates a replacement Rin value by fitting a straight line to the Rin values in neighboring pixels that extend *outward* from the center of the filter. The length of this extrapolation line (in raster cells) is called the Extrapolation-Line Length (eLen). With these extensions, the shape the pixels involved in this enhanced Sobel calculation, looks like a "Star." Thus, the author calls this the Sobel Star filter. The purpose for fitting a straight line though extended Rin pixel values is to smooth the Rin values before making the Sobel calculation. If you keep the eLen value at the default value of **1**, then OBJECT.sml will perform no smoothing operations. If you enter an eLen value of **3** or more (up to **9**), then the script will smooth the Rin values prior to calculating the related EP value. Selecting a large value for eLen, e.g., **9**, will produce *more smoothing*; <u>however</u>, *the resulting SOPs may also be adversely affected*. As with other parameters, you can rerun OBJECT.sml as you optimize the results.

- **CUMULATIVE-PERCENTILE FOR LINE DELETION, cPTLinD:** OBJECT.sml makes an initial set of SOPs based on the EP raster's values. Each neighboring pair of SOPs is separated by a single edge line. But, the related SOP pair *may* be more alike than different. Using an *adjusted normalized difference* metric, based on the medians of the Rin values for each SOP in each pair, the script ranks the bounding edges into a cumulative distribution from weakest to strongest. The requested control parameter, cPTLinD, is the cumulative percentile threshold. If the actual cumulative percentile is less than cPTLinD, then the related edge line will be deleted. This causes the related pair of SOPs to merge into one SOP. The default for cPTLinD is 0 %; this value causes OBJECT.sml to retain all of the initial SOP edges and polygons. As with other parameters, you can run OBJECT.sml again with a new set of parameters as you optimize the results.
- **VECTOR-LINE THINNING FACTOR, fThin:** Without thinning, all SOP edge lines will faithfully follow each edge of the related pixels. This causes a "staircase" type of edge line that has many vertices. Line thinning reduces the number of vertices and reduces the "staircase" appearance of the SOP edge lines. Too much thinning, however, can distort the SOPs so that they align poorly with the input image. The default fThin value of **0.00** results in **no thinning**. In fact, if fThin is less than 0.8, little thinning will be done. A value of **1.00** for fThin is a good choice if you want moderate thinning. As with other parameters, you can rerun OBJECT.sml to achieve optimal results.
- **MINIMUM ISLAND-POLYGON AREA, aIPMin:** Some candidate SOPs wind up being island polygons. You may want to delete SOPs that have an area less than a specified minimum area (aIPMin), in units of sq. m. As with other parameters, you can run OBJECT.sml again with a new set of parameters to optimize the results.
- **SELECT RASTER OBJECT FOR "EP":** This selection is necessary only when you are modifying an existing EP raster from a previous run.
- **OFFSET FOR SOP MEDIAN VALUES, medOff:** In the Line-Deletion process, the median values for each of related pair of SOPs are compared. The adjusted normalized difference formula for this comparison is:

  delPar = 1000 (Med1* – Med2*) / (Med1* + Med2*)

  where:

  Med1 = median of the Rin values for one SOP of the pair;
  Med2 = median of the Rin values for the other SOP;
  Med1* = Med1 + medOff2; Med2* = Med2 + medOff2; and
  medOff2 = medOff – floorRin + 10.
- **NOTE 1:** Increasing medOff causes the SOP pairs that have relatively low average Rin values to be merged into a set of single SOPs. As with other parameters, you can rerun OBJECT.sml with a new set of control parameters to optimize the results.
- **NOTE 2:** If cPTLinD is equal to 0, the script will not delete any edge lines. Therefore, a value for medOff is not needed.

When you are optimizing the 8 control parameters, you should keep track of these changing values as you can modify them from run step to run step. To help you do this, a control-parameters tracking log sheet is provided below:

Rin Raster Information:

Step	floorRin	ceilRin	minEP	eLen	cPTLinD	fThin	alPMin	medOff
default			1	1	0	0	1	0
2								
3								
4								
5								
6								
7								
8								
9								
10								
11								
12								
13								
14								
15								
16								
17								
18								
19								
20								
21								
22								
23								
24								

NOTES and COMMENTS (During Each Run):

Now let's consider the various aspects of OBJECT.sml and it uses.

*H1. What is a **Scene-Object Polygon (SOP)**?*

A raster image consists of millions of picture elements (pixels) that are arranged in a regular, rectangular pattern of raster lines and columns. In a displayed image, however, you will likely recognize many items of interest. These are called Scene Objects (SOs). This kind of recognition is the essence of manual photointerpretation.

OBJECT.sml is a script-controlled, *automatic* process that creates georeferenced Scene Object Polygons (SOPs) in a vector object (called SOP) in a TNTmips Project File (called SOP.rvc) from a selected input raster, Rin.

The figure below shows a typical, high-resolution color infrared (CIR) image. In it, you can identify a number of SOs. Each scene object involves many adjacent *or* neighboring pixels in a recognizable spatial pattern.

Figure H1A. QuickBird CIR Image (Collected Near Lancaster, CA)

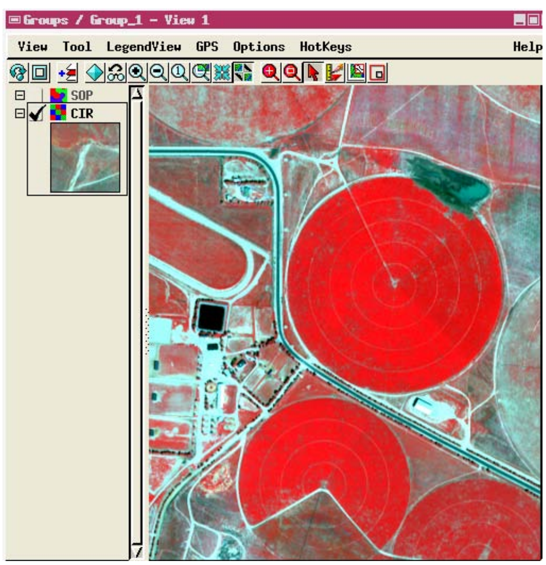
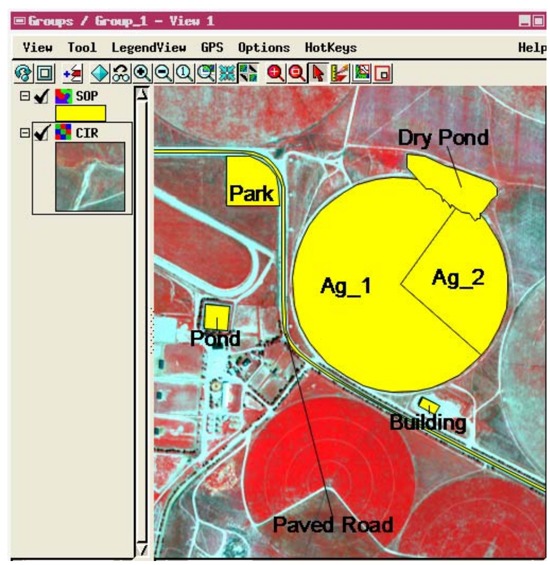


Figure H1B. Related SOPs (Manually-Drawn Yellow-Area SOPs)



You could use the TNTmips Spatial Data Editor to *manually* draw a georeferenced vector line around the entire extents of each recognized (and important) SO. As a result, a set of GIS vector polygons may be produced with one polygon per scene object – an SOP for each SO (see *Figure H1B*).

Some of these SOPs have regular boundaries, e.g., the Pond and the Building. The rest have irregular boundaries, e.g., the Park, the Dry Pond, Ag_1, Ag_2, and the Paved Road. The Paved Road SOP is a long boundary that follows both sides of the narrow road; this kind of object could have been represented by a vector line, instead of by a vector polygon.

Some of the SOP boundaries are easy to recognize in the reference CIR image; however, others are much less distinct, (e.g., the distinction between Ag_1 and Ag_2). The Park SOP has many internal boundaries, which were ignored by the analyst when he defined the geographic extents of this object. An automatic process that defines SOPs will likely produce unwanted SOP edges inside of this kind of SOP.

For some of the SOPs, the properties of the pixels inside the object are spectrally homogeneous, e.g., the consistently dark pixels inside of the Paved Road and of the Pond. For other SOPs, e.g., the Park, the spectral properties of the pixels inside of the object are quite variable.

These examples show how difficult it is to develop a *suitable* automatic SML script that will find *all* of the SOPs of interest in a typical image. *Nevertheless*, it is useful to have a script like OBJECT.sml for scenes that lend themselves to this kind of automation. The results may be, and should be, edited manually to achieve the best set of final SOPs.

H2. What is the ***Usefulness of SOPs****?*

Many algorithms exist to process imagery *at the pixel level* using one or more spectral-band images as inputs. In a sense, each image pixel is a small, uniform-sized SOP that has a rectangular or square boundary. A pixel may be handled as a data point or as a SOP that has related raster values. Many *pixel-oriented* algorithms exist in TNTmips, e.g., the supervised and *so-called* unsupervised classification processes found under Process / Raster / Interpret / Auto Classify… .

However, if you first segment an imaged scene into a number of SOPs, then the properties of each SOP can be calculated (using Process / Vector / Attributes / Raster Properties…) and attached as statistical information in a related, named database table. This enhances the standard vector attributes attached to each SOP during the vector-validation process.

One or more records in other non-image-based database tables can then be assembled so that all of the attached information can be used to classify the SOP as likely being a member of some desired mapping-element class. This requires making logical combinations of these information units. For example, the SOPs called the Pond, the Dry Pond, and the Paved Road that were discussed in *H1* all share similar spectral properties – they are relatively dark in all of the spectral bands; however, they differ greatly in area and in shape (e.g., the ratio of area to boundary length).

OBJECT.sml is a part of an overall Object-Oriented Classification approach … a popular subject in current GIS literature. Software packages that do this kind of processing are usually very expensive ($10,000 or much higher). OBJECT.sml lets you do this kind of processing with TNTmips, which costs much less. To use OBJECT.sml, you need only to specify a few control parameters … TNTmips SML does the rest! If you don't like the results, you can change one or more control parameters and rerun the script. Plus, you can intervene with manual editing operations while perhaps accepting most of the SOPs that the scripts have produced.

H3. What are Some ***Problems with SOPs****?*

In many cases, the precise definition of which exact pixels should be included in a given SOP is more art than simple scientific deduction. A common complaint is that an automatic script like OBJECT.sml makes too many SOPs. Strangely enough, another complaint is some of the resulting SOPs are too inclusive (too few SOPs)! Often, many SOP boundaries are fuzzy.

Another problem may be that a perceived interior boundary of a SOP may be only temporary. For example, a distinct wet area within an agricultural field may be quite visible in an image on one date, but is not present a few days later in another drier date.

The remaining sections in this tutorial focus primarily on the specific technical aspects of OBJECT.sml within the context of a specific project.

H4. What is an ***Edge-Probability*** *(**EP**) Value?*

The statements in OBJECT.sml start with now familiar SML items that (1) set the Warning level to 3, (2) set a Watershed-process parameter (computeW$) to "Ridge," (3) define most of the variable names and types, (4) specify the content of the writeTitle function, (5) allow the Console-Window to be adjusted, (6) define the checkHisto function, and (7) let the user assign a raster object to Rin raster. See previous *FAQs by Jack™* for details about these opening processes. Only the Watershed parameter is new; this will be discussed later in this tutorial.

Next, the script asks you to make a choice about the Edge-Probability (EP) raster, in essence: Are you CREATing a NEW EP raster or are you CONTINUing with an EXISTING EP raster?

To understand the role of the EP raster, you need to know why the author decided to use the TNTmips Watershed function as part of the processes that produce the vector polygons called SOPs.

Watershed functions operate on a special kind of raster, namely, a digital elevation model (DEM). Relatively high-elevation pixels in a DEM are likely to become locations for vertices along a watershed Ridge line. A watershed is defined as an area that would collect rainwater to an internal low point if any rain were to fall on the related impervious terrain. If a watershed were allowed to fill up with collected water, the rising water will eventually fill the watershed depressions and then flow over the lowest point on the Ridge polygon into an adjacent watershed.

In order for a Watershed function to produce SOPs based on an input image, Rin, a DEM-like raster, called the Edge-Probability (EP) raster, must first be made. The values of EP should be "high" where a given pixel is likely to be near or on a SOP boundary. And, the values of EP should be "low" where a given pixel is not near or on a SOP boundary. If a Ridge line ends (at a dangling node) without connecting to another set of Ridge lines (on both ends), then the related watershed will not be divided into smaller areas. These kinds of Ridges become dangling vector lines that are deleted.

OBJECT.sml produces a 16-bit unsigned-integer EP raster, which is saved as an output-file raster object by the same name. The numeric values in an EP raster represent the *relative* probability that a given pixel is, in fact, on a point on the boundary of a scene object.

*H5. What are the **Characteristics of an SOP Edge**?*

A SOP edge should occur where the perceived properties of the Rin raster change abruptly as you move – *spatially* – across the image.

Consider an ideal Rin raster and its associated set of idealized SOPs as shown in Figure H5A [north (Northing) is at the top of this image].

Figure H5A. An Ideal Rin Raster.

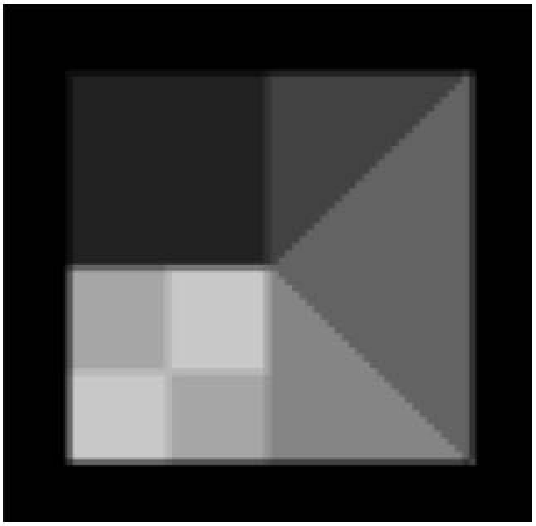
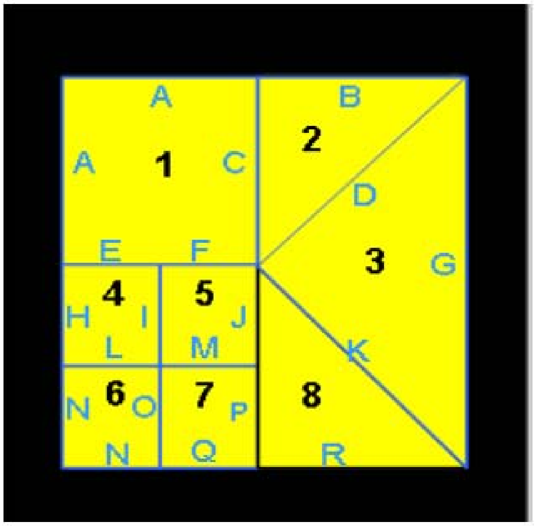


Figure H5B. Manually-Drawn SOPs with Labeled Edges.



In this ideal Rin image, you can easily recognize **8** Scene Objects (SOs). In Figure H5B, these **8** SOs were manually identified by numbers from **1** to **8**. These SOPs are defined by **18** edges as labeled from A to R. The values for Rin inside of each SOP are as follows: SOP **1**: 500; SOP **2**: 1000; SOP **3**:

1500; SOP **4**: 2500; SOP **5**: 3000; SOP **6**: 3000; SOP **7**: 2500, and SOP **8**: 2000. In the black outer area, SOPs is equal to 5.

In this ideal example, most of the SOP edges are straight lines that are oriented either north-to-south (Edges C, G, H, I, J, O, and P) or east-to-west (Edges B, E, F, L, M, Q, and R). Two edges are diagonal straight lines (Edges D and K). Also, two other SOP edges change direction at a vertex point (Edges A and N). Except for Edges A and N, all other edges are defined by two vertices (one at each end of the line). But, Edges A and N involve three vertices. Some vertices serve more than one edge; they are called vector nodes. There is a 9th area that is the black area outside of the domain of the other 8 SOPs. However, this 9th area is not included in the set of SOPs due to its extent being outside the border of the raster.

By design, there are **8** relatively weak edges, as follows:

Edge C (between SOP **1** and **2**), Edge D (between SOP **2** and **3**),
Edge K (between SOP **3** and **8**), Edge P (between SOP **8** and **7**),
Edge O (between SOP **7** and **6**), Edge M (between SOP **7** and **5**),
Edge L (between SOP **6** and **4**), and Edge I (between SOP **5** and **4**).

The other **10** edges are either outside edges (A, B, G, R, Q, N, and H) or stronger edges (J, E, and F, in order from strong to strongest). Since the Rin values in the black area around the SOPs are low (Rin = 5), Edges N, H, Q, R, G, and B are stronger than Edges A, C, D, K, P, O, M, L, and I.

Edge-strength assessments in OBJECT.sml are based on how quickly Rin *changes* across a potential edge pixel from one SOP to its neighboring SOP. Spatial rates of change are called spatial gradients. Spatial gradients are mathematical vector quantities. That is, each spatial gradient has both a direction and a magnitude. EP is based on the magnitude of the gradient.

Consider Edge J. The change in Rin values across Edge J in the east-to-west direction is from 2000 to 3000 – a change of +1000 over a distance of two pixel widths. But, on the same edge, the change in the north-to-south direction is 0 (from 2500 to 2500). But, for Edge D, the direction of change is from northwest to southeast. And, for Edge F, the direction of change is from north to south.

In OBJECT.sml, the probability that a given pixel is located on an edge between two relatively homogeneous SOs (in terms of Rin values) is based on the estimated magnitude of the spatial gradient at that pixel.

One way to evaluate spatial gradient properties is to use a 3 x 3 matrix of Rin values and the Sobel algorithm (Gonzalez, R. and R. Woods, 1992).

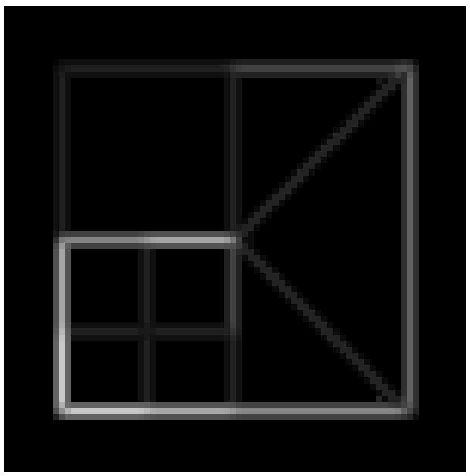
This approach involves calculating the spatial gradient components – one for the east-west direction (i.e., the x-direction) *and* the other for the north-south direction (i.e., the y-direction). Having these (signed) magnitudes (gradx and grady), the magnitude of the gradient (here called gradmag) is:

gradmag = square root of (gradx * gradx + grady * grady)

EP is proportional to the gradmag. The precise Sobel formulas for gradx and grady are defined in the SML script.

An image of the EP raster, related to the ideal Rin raster n Figure H5A, is shown in the figure below.

Figure H5C. The EP Raster as Related to an Ideal Rin Raster.



The strongest SOP edges are the ones where the EP image is brightest.

Note that the strengths of the weakest edges are approximately equal in terms of EP magnitudes. This is usually not the case for a real image. So, this ideal Rin raster is more challenging for the line-deletion part of the OBJECT.sml script than would be a real Rin raster.

*H6. What are the **floorRin** and **ceilRin** Control Parameters?*

In many cases, the Rin raster has special properties that allow the user to focus on certain kinds of SOPs. For example, if a TC Greenness raster is selected for Rin, then the user can take advantage of known characteristics of this kind of calibrated raster that relate to a particular biophysical property of the scene, i.e., vegetation density.

In a typical scene, TC Greenness ranges from large negative values (e.g., near -6000) to large positive values (e.g., near +7000). A value of **0** for TC Greenness is associated with *average* bare soils. Positive values of TC Greenness are associated with vegetated pixels. So, if a user wants to produce SOPs only for moderately vegetated scene objects, the value for floorRin should be set to a number above 0 … say to 500. This causes the script to focus on pixels that have Rin values greater than 500. Likewise, the user may want SOPs that do not have internal boundaries for very dense vegetated areas. In this case, a value for ceilRin would be set high, e.g., to 3000. With these settings (floorRin = 500 and ceilRin = 3000), the resulting SOPs would focus on vegetation having moderate levels of density. The same role of these two control parameters might be appropriate for other kinds of Rin rasters, e.g., TC Yellowness.

*H7. What is the **minEP** Control Parameter?*

The Sobel Star algorithm produces a range of EP values (after being rescaled) that range from 1 to 10,000. Pixels that have low EP values are likely to be associated with very weak edges … ones that the user probably does not want to be included in the final SOP vector. These low-EP edges are weak in an absolute sense. Therefore, if EP values are limited to some specified minimum value (minEP), then the related weak edges will not be created by the script. How low should minEP be? That is a value that can be determined only by iterative runs with the OBJECT.sml script. The default value of 1 is very conservative. Higher values (up to 5,000 … but more likely in the hundreds or low thousands) can be elected for minEP to eliminate weaker SOP edges.

*H8. What is the **eLen** Control Parameter?*

If Rin is a noise-free image, then meaningful edges between homogeneous scene objects are easily defined by the EP values (and the related Watershed process). However, a *real* Rin image has variations that are caused by:

- Noise introduced by the imaging system and other random scene-inducted variations, and
- Gradual trends across a scene object from one side to the other side.

The classic approach for dealing with image noise is to pass the Rin image through a Gaussian filter that reduces the noise while retaining spatial changes in brightness along scene-object edges. This can be done using

TNTmips menu-selected tools. But, this kind of filter does not deal with gradual trends across an image or across a scene object. The Sobel filter produces estimates of the rate of change of Rin values per unit distance (change per meter … based on cell spacing). It is based on the first spatial derivatives in the column direction and in the line direction.

OBJECT.sml employs a different solution to deal with the image-noise and gradual-trend problems. Both are handled by using a linear regression function to estimate the values of Rin for each of the 8 outside cells in a Sobel 3 x 3 matrix.

The Sobel 3 x 3 filter matrix looks as follows:

```
r1  r2  r3

r4  cc  r6

r7  r8  r9
```

The Sobel formulas for directional gradient components do not involve the center cell (cc). Rather, they are based on the weighted differences between cells on the left and the right (for the x-direction gradient, gradx) and between cells on the top and the bottom (for the y-direction gradient, grady).

A better (smoother) estimate of each of the Sobel matrix elements (r1, r2, r3, r4, r6, r7, r8, and r9) can be made by using a number of neighboring Rin values for cells that extend outward (in 8 different directions) from the center of the Sobel 3 x 3 matrix. The diagram below shows the pattern of these extended cells relative to the Sobel matrix.

```
SOBEL STAR CONFIGURATION:  xx: MARKS RELATIVE POSITIONS
OF THE EXTENDED "STAR" CELLS USED TO EXTRAPOLATE TO
THE SOBEL 3 by 3 FILTER ELEMENTS.
THE SOBEL-STAR CONFIGURATION BELOW IS FOR eLen = 3.

xx          xx          xx

    xx      xx      xx

        r1  r2  r3

xx  xx  r4  cc  r6  xx  xx

        r7  r8  r9

    xx      xx      xx

xx          xx          xx
```

For example, to estimate the value for r3, a linear regression line is established on the basis of r3 *and* the two “xx” cells that are above and to the right of cell r3. This pattern of estimation is done for each of the 8 cells involved in the Sobel gradient formulas. The number of cells involved in the linear regression is equal to the eLen control parameter.

If an image is smooth *and* without significant spatial trends, then it would be wise to set eLen = 1. This is the default value for eLen. A normally noisy image with normal trends would be helped by setting eLen = 3. OBJECT.sml allows you to select eLen from **1** to **9** (only as an odd integer). The reason for requiring an odd integer for eLen is that a FocalMax filter is applied to the EP values after they have been created. This filter fills in false low spots in the EP raster that are caused by over estimations by the linear regression near places where legitimate SOP edges occur.

*H9. What is the **cPTLinD** Control Parameter?*

When the first set of tentative SOPs are created, it is still possible that some adjacent polygons, which are separated by a single edge line, are actually more alike than different. The criterion for two SOP elements being alike or dissimilar is based on the two median values of Rin in each of a pair of SOPs.

An adjusted normalized difference algorithm is used to compare the two medians. This expression allows the user to give more (or less) emphasis to differences based on overall (average) Rin properties. OBJECT.sml scans each pair of SOPs, calculates an adjusted normalized difference value, and ranks the edge polyline according to this metric. As a result, each polyline edge has an associated cumulative distribution value (from 0 to 100%). cPTLinD stands for the cumulative PercenTile for edge Line Deletion. A default value of 0 (%) is suggested as a starting point.

If you do not want the script to delete edges based on the comparative values of the medians for the pair of SOPs related to a given edge line, then set cPTLinD = 0.

*H10. What is the **fThin** Control Parameter?*

Initially, OBJECT.sml creates SOP edges that look like stair steps. That is, a line is created that faithfully follows each edge of each pixel that is adjacent to the created edge. While this kind of line is accurate, it uses very many vertices. The fThin control parameter determines how much thinning will be done by the VectorThinLines function. It has units of cell size. So, fThin = 1.00 means that lines that have a 1-cell sized kinkiness to them will be thinned (to more diagonal “shortcuts” around each related stair step). If fThin is less than 0.8, very little thinning will occur. If fThin is greater than 1.0, the amount of thinning is larger. The default value for fThin is **0.00** – which produces **no thinning**.

*H11. What is the **aIPMin** Control Parameter?*

Due to line deletions, some SOPs become "island polygons," which are polygons that are defined by boundary lines that do not connect to any other edge lines. Small SOPs that are island polygons are likely to be extraneous SOPs. So, a control parameter, aIPMin, is provided. It is the smallest island polygon that will be retained by OBJECT.sml. It has units of square meters. A good estimate for aIPMin is to take the area (sq. m.) of one pixel and multiple this by the number of pixels that you consider to be the smallest mapping unit in the Rin image. The default for aIPMin is 1 sq. m.

*H12. Why is a **Temporary Raster, R**, Created?*

Via the two control parameters, floorRin and ceilRin, the actual Rin values used by OBJECT.sml processes will not be the same as the source Rin values. The temporary raster, R, stores these modified Rin values for subsequent processing.

*H13. Why is the **EP** Raster Created Automatically?*

This new raster is automatically created in the same folder as the source Rin raster. Otherwise, if you are continuing with an EP raster from a previous run, you will be asked to open that existing EP raster for further processing.

*H14. Why is the **SOP** Vector Created Automatically?*

This new vector object is automatically created in the same folder as the source Rin raster. This speeds the execution of the script. If you already have a project file called SOP.rvc, a new SOP object will be added to it with a slightly different name, e.g., SOP1.

*H15. What is the Purpose of the **First Scan** Through the **Rin** Raster?*

The initial values for EP arise from the Sobel Star algorithm. It is impossible to predict what will be the actual range (actual maximum value) for these EP values. So, the Rin raster is processed just to find the maximum EP value. Then, a scaling factor (called epFac) is calculated that will force the final maximum value for EP to be 10,000.

*H16. What is the Process that Involves **Isolated EP Cells**?*

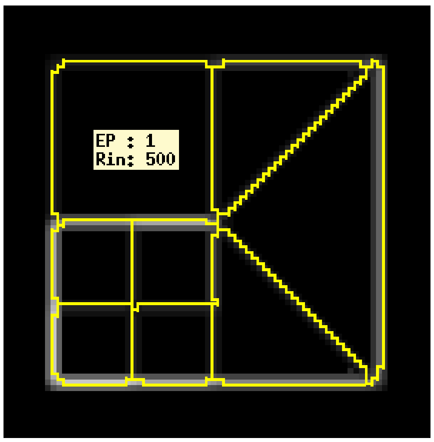
If a focus cell has an EP value that is lower than the ones on either side of it, this low value could cause the related SOP edge to have a break. This would lead eventually to the whole edge line being deleted during the process that deletes dangling lines. Finding isolated EP values and replacing them with higher values helps to prevent the deletion of SOP edge lines due to a single EP cell having a low EP value.

H17. What is an Example of the ***SOPs from an Ideal Rin Raster****?*

In *H5*, an ideal Rin and its associated manually-drawn SOPs were shown (see Figures *H5A* and *H5B*). The associated EP raster was shown in Figure *H5C*. When OBJECT.sml was used, with default control parameters, on this ideal Rin raster, the resulting automatically-produced SOPs are as shown in Figure *H17*. In this figure, the data tip shows the EP value and the Rin value inside of one of the SOPs.

Overall, the SOPs are excellent. But, some SOP edge points are not perfect; however, the defined SOPs agree very well with the extents of the apparent SOPs in Rin.

Figure H17. SOPs Produced by OBJECT.sml from an Ideal Rin Raster.



The one-pixel offsets in these polygons at the SOP corners are caused by how the Watershed function handles places where the direction of flow over the “terrain” from a single EP pixel could be in either direction.

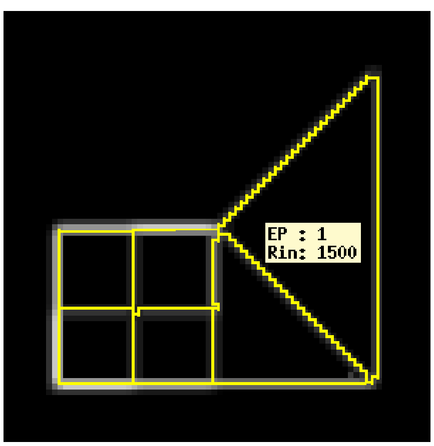
In addition, small changes in the Rin raster at the corners of the SOPs can influence the construction of the SOP vector elements.

At a different scale, these one-pixel offsets will not be noticeable or important to a final thematic map.

H18. For the **Ideal Rin**, *What is the* ***Effect of Changing floorRin****?*

When you increase floorRin to a higher value, e.g., 1001 (keeping all of the other control parameters at their default values), the result is *Figure H18*.

Figure H18. SOPs Produced When floorRin = 1001 (Rest are Default).



As expected, two of the northern SOPs are not created. They contain Rin values that are less than 1001.

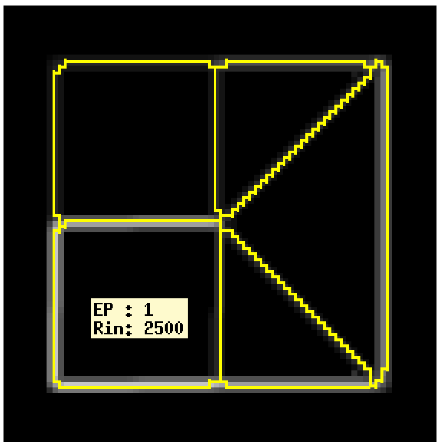
If Rin were a vegetation index raster (e.g., TC Greenness), you could raise floorRin to a value like 1001 in order to exclude low-density vegetation areas from the SOPs.

H19. For the **Ideal Rin**, What is the **Effect of Changing ceilRin**?

Now, consider what happens when you decrease ceilRin from its default value (3000 in this ideal Rin case) to a lower value, e.g., 2499 (which is lower than the Rin values for several of the southern SOPs).

If all of the other control parameters are kept at their default values, the result is what is shown in *Figure H19*.

Figure H19. SOPs Produced When ceilRin = 2499 (Rest are Default).



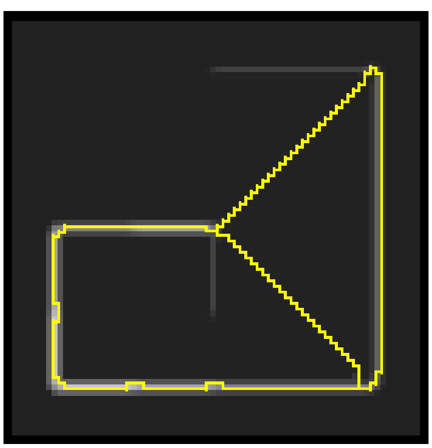
In this case, the four SOPs in the southwestern corner of the Rin image were merged into one SOP. The Rin values for these SOPs are all greater than 2499 (surrounded by lower Rin values).

If this were an SOP having a high vegetation density, then this choice of control parameters leads to the highest density SOPs being merged into one SOP. This option ignores boundaries between high-Rin scene objects.

H20. For the **Ideal Rin**, What is the **Effect of Changing minEP**?

Now, consider what happens when you increase minEP from its default value (1 in this ideal Rin case) to a higher value, e.g., to 1700. If all of the other control parameters have default values, the result is *Figure H20*.

Figure H20. SOPs Produced When minEP = 1700 (Rest are Default).



In this case, many SOPs were merged into two SOPs.

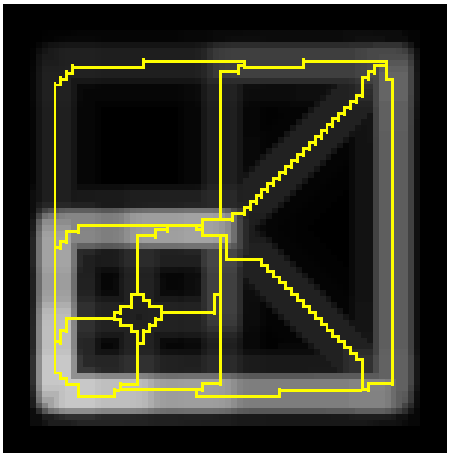
This would be a useful result if you want to reduce the number of SOPs being produced by elimination of weak edges as measured in absolute terms, i.e., in terms of absolute EP values.

*H21. For the **Ideal Rin**, What is the **Effect of Changing eLen**?*

Now, consider what happens when you increase eLen from its default value (1 in this ideal Rin case) to a higher value, e.g., to 5.

If all of the other control parameters are kept at their default values, the result is what is shown in *Figure H21*.

Figure H21. SOPs Produced When eLen = 5 (Rest are Default).



In this case, the results are a bit surprising!

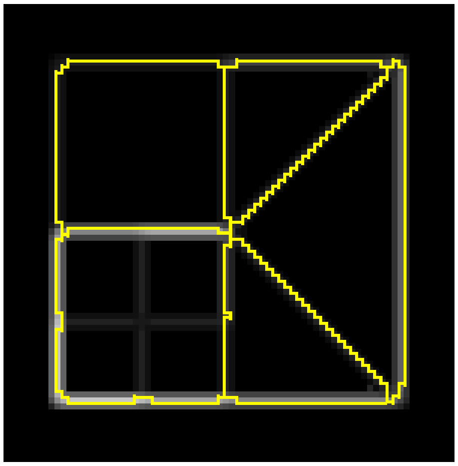
A new SOP element was created at the corner between the four SOPS in the southwest part of the image.

Since the ideal Rin image is very smooth inside of the boundaries, it is not necessary to set eLen equal to a value that is greater than 1. But, for noisy images, it is a good idea to set eLen to **3** or **5**.

*H22. For the **Ideal Rin**, What is the **Effect of Changing cPTLinD**?*

Now, consider what happens when you increase cPTLinD from its default value (0 in this ideal Rin case) to a higher value, e.g., 40. When you do this, the medOff parameter comes into play (see H25).

Figure H22. SOPs Produced When eLen = 5 (Rest are Default).



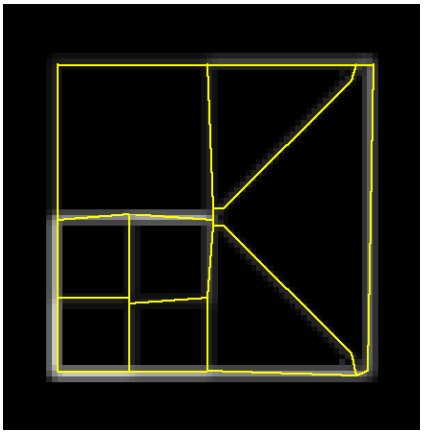
With medOff = 0 (the default value) and cPTLinD = 40, the edges in the area of relatively high Rin values were selected for deletion. These are the edges interior to the previously four SOPs in the southwest part of the image.

If medOff is raised to a much higher value, then the weak edges in the rest of the image, where Rin is low, will have a higher chance of being deleted.

*H23. For the **Ideal Rin**, What is the **Effect of Changing fThin**?*

To decrease the "staircase" appearance of the SOP edges, you can involve the vector-line thinning option in OBJECT.sml. To do this, increase the fThin control parameter from its default value of 0.00 to a higher value, e.g., 1.00. When you do this, the result looks as follows:

Figure H22. SOPs Produced When fThin = 1.00 (Rest are Default).



As expected, this reduces the number of vertices required to represent each SOP edge line. The "staircase" look in this figure is due to display pixels, not to vertices.

While the SOPs are not perfect, it is clear that setting fThin to 1.00 greatly improves them at least as far as minimizing the number of vertices is concerned.

*H24. For the **Ideal Rin**, What is the **Effect of Changing alPMin**?*

There are no island polygons in the SOPs that are being created from the ideal Rin raster. So, the results of changing alPMin will be the same as in *Figure H17*.

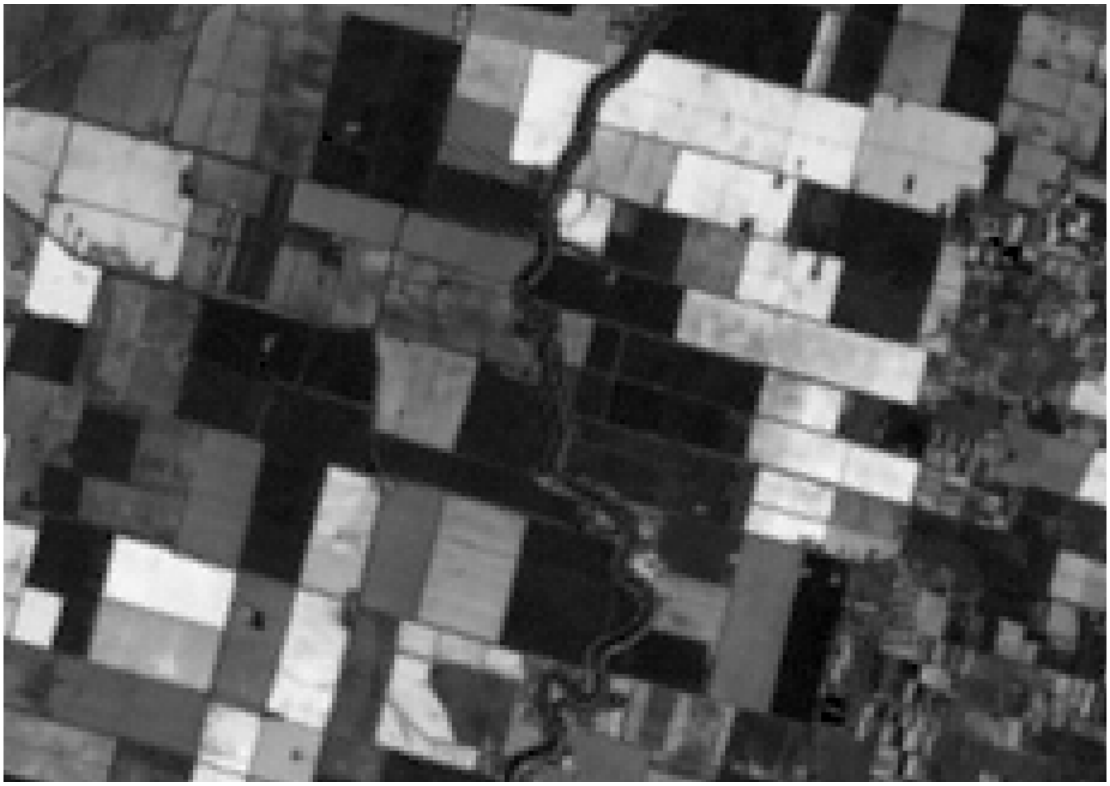
*H25. For the **Ideal Rin**, What is the **Effect of Changing medOff**?*

If cPTLinD is not equal to zero, then some weak edges will be deleted. But, the kinds of edges will depend on the medOff parameter. If you set cPTLinD equal to 40 and also set the medOff parameter equal to 5000 (with all other parameters as default), you will, in the case of this ideal Rin case, get the same results as before (see *Figure H17*). This is due to the fact that many of the weaker edges in the ideal Rin raster have the same strength.

H26. How Does **OBJECT.sml** Work on a Real Image Using **Default Parameters**?

If a TC Greenness raster is used as the Rin from a real image, e.g., a Landsat TM based image (for Stockton, CA) and default parameters are assumed, then the result would be as shown in Figures H26A, H26B, H26C, and H26D.

Figure H26A. Rin Source Image (The TC Greenness Raster).



This is a mostly agricultural scene in a Landsat TM image. It has a few very dense vegetation areas (i.e., the bright values of TC Greenness), some moderately-dense vegetation areas (i.e., the gray areas), and some low or non-vegetated areas including urban landscapes and a river (i.e., the darker areas). Some agricultural fields have significant internal boundaries.

Figure H26B. **Default** EP Image (Based on TC Greenness as Rin).

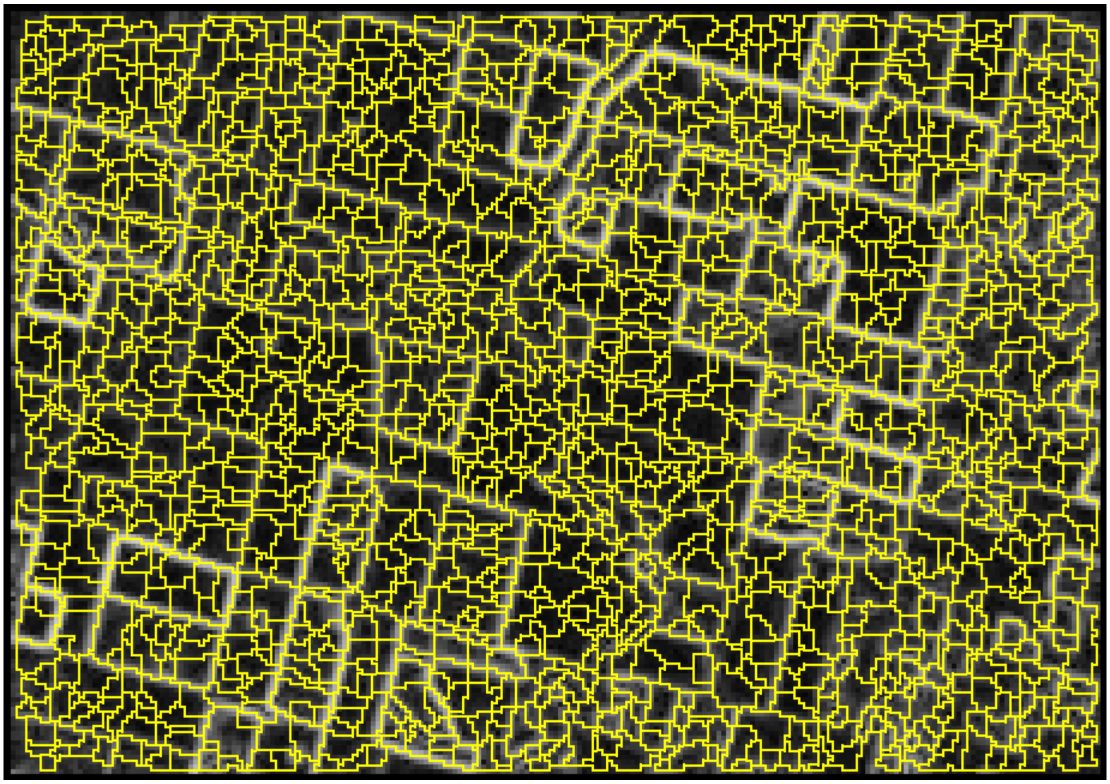
The **default** control-parameter values are as follows:

```
floorRin = minRin
ceilRin  = maxRin
minEP    =      1
eLen     =      1
cPTLinD  =      0
fThin    =      0
aIPMin   =      1
medOff   =      0
```

There are many SOP edges with a wide range of edge strengths. Many of these edges are in non-vegetation areas, especially in the complex urban landscape on the eastern side of the image.

Clearly, some optimization of the OBJECT.sml control parameters is needed.

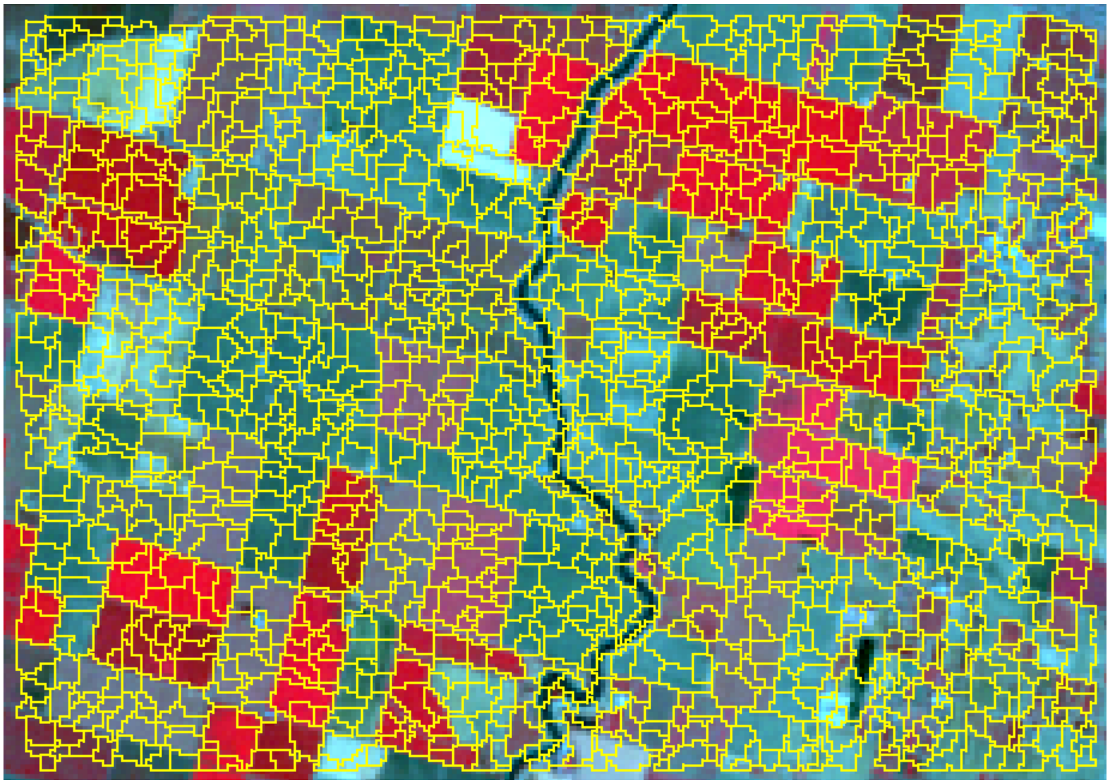
Figure H26C. **Default** SOPs over the EP Image (Based on EP Image).



Again, based on default values for control parameters, the OBJECT.sml script produces many SOPs … for just about every potential EP edge point. While these are accurately positioned, they are probably too numerous for some mapping purposes.

Optimization will improve the SOPs and make them better fit what an analyst would see in the source Rin raster.

Figure H26D. **Default** SOPs over the CIR Image (Landsat ETM Image).



Here the SOPs that resulting from using default control parameters are shown over a color infrared (CIR) image of the scene. These SOPs came from using TC Greenness as Rin.

Though CIR is not used in the SOP creation process, it is a useful reference image for QAQC.

It is clear from this overlay display that the locations of the SOPs are accurate with respect to a wide variety of scene objects including both vegetated scene objects and non-vegetated scene objects.

The absence of SOPs extending to the border of the image is caused by the way that OBJECT.sml handles near-border pixels. A 3 x 3 Sobel matrix operates on a central pixel that is at least 3 pixels away from the border pixels. When eLen = 1, the "Star" extension of the Sobel matrix is not enabled.

H27. How Does OBJECT.sml Work on a Real Image Using ***Optimized Parameters****?*

If you modify the control parameters and iterate the running of the script, you can arrive at a set of optimized control parameters. In this case, the author arrived at the following set of optimized control parameters (after a few steps):

```
floorRin =    500
ceilRin  =   4000
minEP    =    100
eLen     =      3
cPTLinD  =      5
fThin    =      1
aIPMin   =   8000
medOff   =   4000
```

With these optimized control parameters and the same Rin input raster (i.e., TC Greenness), the resulting EP raster and the related SOPs from OBJECT.sml are as shown in the following figures.

Figure H27A. **Optimized** EP Image (Based on TC Greenness as the Rin Raster).

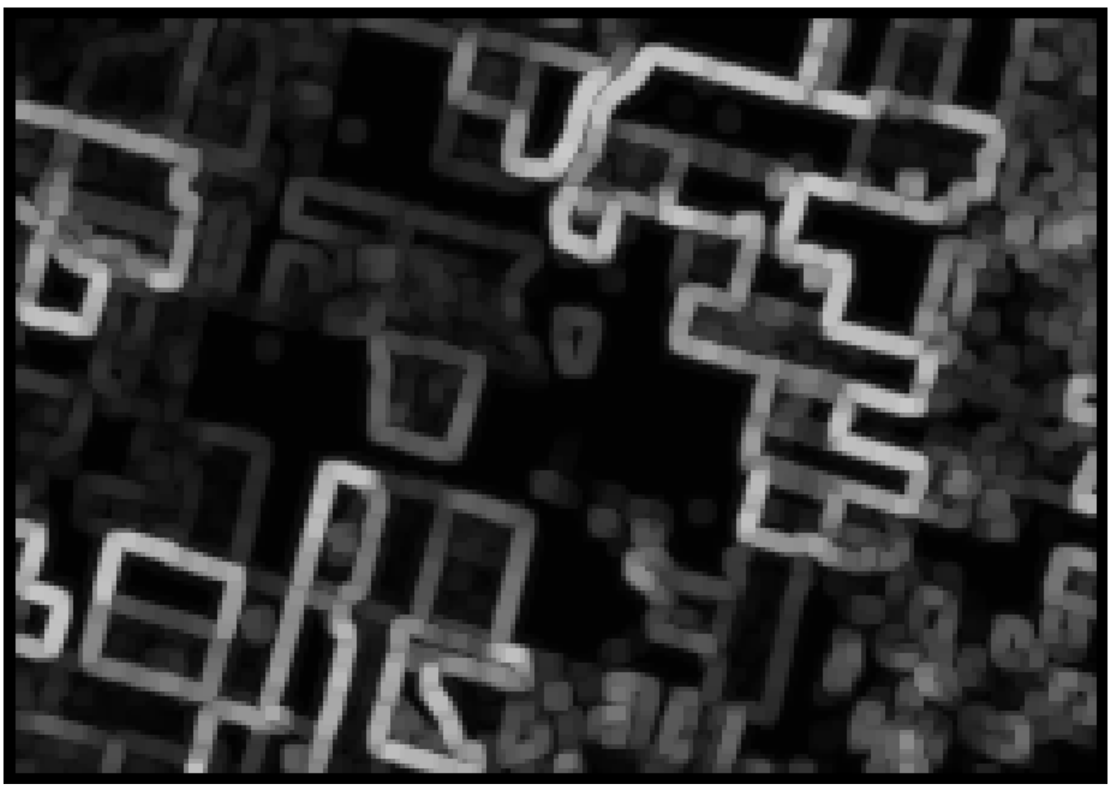
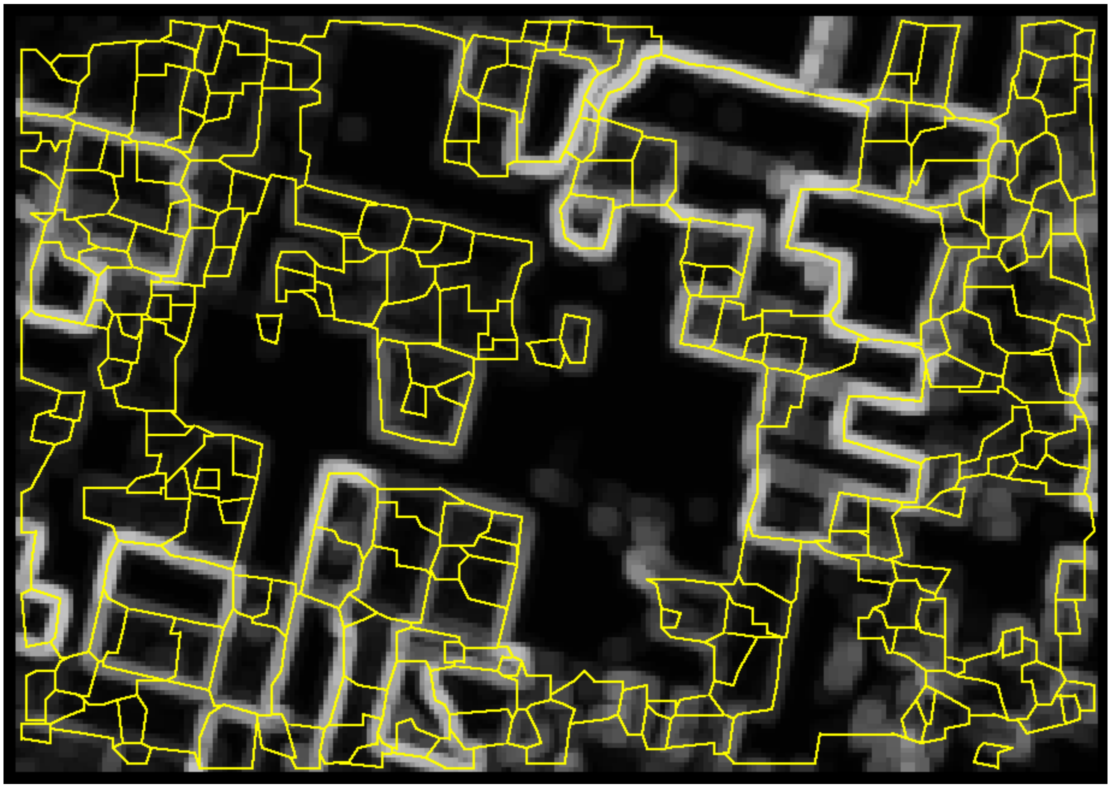
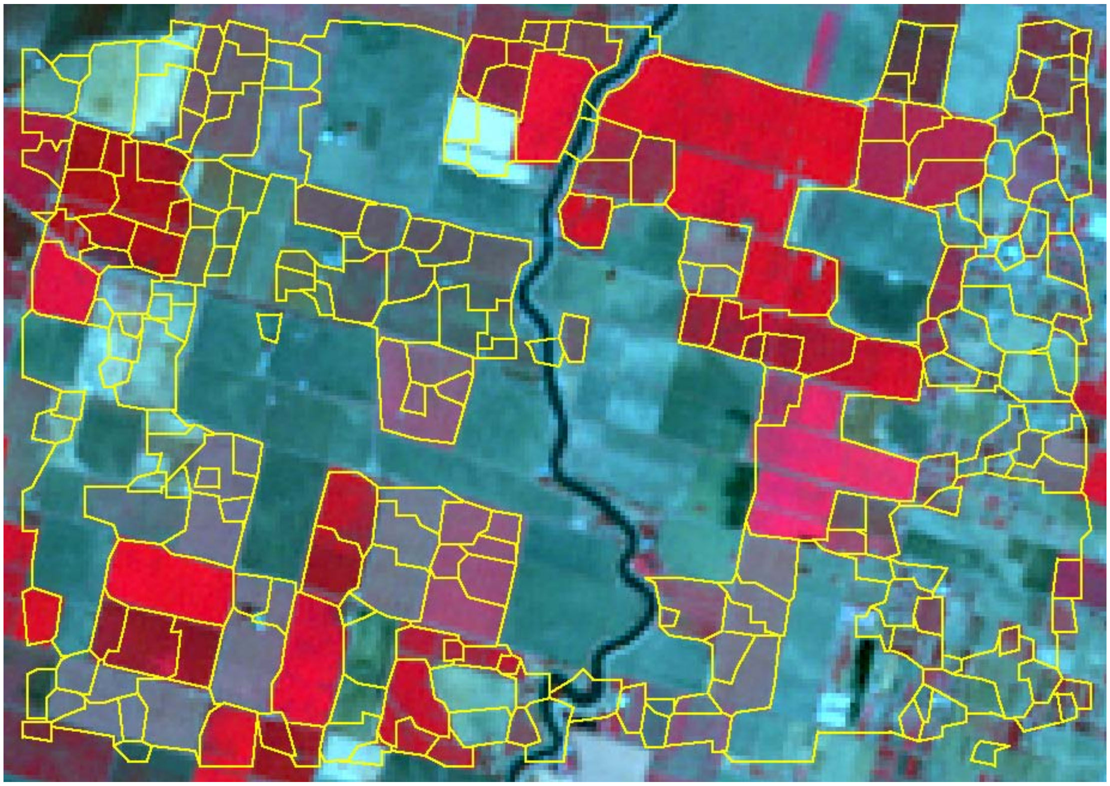


Figure H27B. **Optimized** SOPs over the EP Image (Based on EP Image).



Only 241 SOP polygons have been created and retained. These all are related to moderately-dense vegetation. This will be better seen in the next figure. Note that the fThin parameter (set equal to 1.00) has caused the various edge lines to be represented by relatively simple edge lines.

Figure H27C. **Optimized** SOPs over the CIR Image (Same Landsat ETM Image).



Viewing the SOPs over a CIR image shows that OBJECT.sml performed very well. The only places where interior boundaries appear to be missing are narrow roads that separate ag fields having similar levels of biomass density (similar TC Greenness values). This "neglect" is due to the fact that the EP raster is based on an algorithm that does not use the focus / center pixel in the Sobel 3 x 3 filter matrix.

The author believes that this is a good feature of the algorithm, not something that is being "neglected." But, if the user wants to have these narrow-road lines as SOP boundaries, they are easy to add manually using the TNTmips Spatial Data Editor tools.

*H28. What are Some Ideas about What to Do with the **SOPs**?*

As suggested in the previous sections, you can add value to the automatically created SOP set as follows:

1. Use the TNTmips Spatial Data Editor to delete edge lines and/or to add edge lines to the SOP vector object.
2. You can merge the elements in the SOP vector object with vector elements from another source, e.g., a property-ownership map that has boundaries that may not be visible in a remotely-sensed image.
3. You can transfer raster properties to the SOP elements using processes available from the TNTmips main menu.
4. You can perform logic operations on the vector polygon attributes, both the standard attributes from the vector validation process and the raster-properties attributes from Number 3 above.

These value-added operations are beyond the scope of this tutorial. There are several tutorial documents available on the TNTmips Web site to help you with these.

REFERENCES

Gonzalez, R. and R. Woods**,** 1992: *Digital Image Processing*, Addison Wesley, pp 414 - 428.