SKELETONIZING A DEM INTO A DRAINAGE NETWORK

AMNON MEISELS, SONIA RAIZMAN, and ARNON KARNIELI
Ben Gurion University of the Negev, P.O. Box 653, Beersheva 84105, Israel
e-mail: amo@black.bgu.ac.il

(Received 1 February 1994; revised 24 August 1994)

Abstract—A new method for extracting drainage systems from Digital Elevation Models (DEMs) is presented. The main algorithm of the proposed method performs a skeletonization process of the set of elevations in the DEM and produces a skeleton of flow paths. An enumeration algorithm performs the removal of loops from the initial flow path. A pre-processing for filling depressions is described as is the necessary post-processing for determining the drainage network through depressions. The new method does not suffer from any of the maladies of former methods described in the literature, such as flow cutoffs, loops of flow, and basin flooding. The new method is tested on several real-world DEMs and produced connected, complete, and loopless networks.

Key Words: Digital Elevation Model, Enumeration algorithm, Depressions, Flow paths, Hydrology.

INTRODUCTION

A Digital Elevation Model (DEM) is a numerical representation of a topographic elevation map. It is a fundamental layer in geographic information systems because it assists in modeling, analyzing, and displaying the Earth’s surface. Digital elevation data are a valuable source of information for many earth science disciplines because they can produce other by-products such as contours, slope, aspect, drainage network, drainage basins, visibility, solar isolation, and more.

One of the DEM’s widely used applications is drainage network extraction. During the last two decades several approaches have been developed for extracting the drainage network from a DEM. A few of the early works use a single-step algorithm. This procedure usually is classifying the DEM’s pixels with respect to different hydraulic indicators (such as surface concavity) (e.g. Peuker and Douglas, 1975; Mark, 1983; Haralick, 1983). Results of the single-step algorithms may be poor because they fail to locate drainage features which are not well defined.

More recently, several researchers have gone on to develop two-step procedures in order to overcome the described weaknesses (e.g. Yoeli, 1984; Band, 1986). These procedures add operators for connecting separate channel segments into continuous stream lines. Additionally, there are based on knowledge of watershed characteristics, stream networks, or surface properties. Consequently, the two-step procedures are assumed to be more efficient than single-step algorithms.

There also are several more recent papers which propose approaches to automatic drainage network calculation: Martz and De Jong (1988); Lammers and Band (1990); Qian, Ehrich, and Campbell (1990); Smith, Zhan, and Gao (1990); and Zehana, Desachy, and Zahzah (1991). One feature common to all of these works is the need for mixing at least two types of topographic data or knowledge. One type of data is local and is related to local features such as minima, maxima, and neighboring points which are either lower or higher. Typical data of a local nature is of scales of neighboring pixels of topographic maps, in the range of 10–50 m. The global type of data is composed of concepts such as saddle points, upstream, direction of flow, etc. These global data have larger intrinsic scales (e.g. Qian, Ehrich, and Campbell, 1990; Smith, Zhan, and Gao, 1990). Not all previous work uses exactly these terms to describe their methods, but their different levels of processing (Smith, Zhan, and Gao, 1990) or “phases” of analysis (Martz and De Jong, 1988) are compatible with the given classification.

We propose here a new method for drainage system extraction that is composed of a main algorithm and a few preprocessing and postprocessing procedures. The main algorithm delineates a drainage network from a digital elevation map by a process of (multi-level) skeletonization. This two-stage algorithm is followed by a process of enumeration of the extracted flow-path points. The enumeration process serves mainly to eliminate closed loops in the extracted drainage network. The special topographic features that tend to produce such loops include saddle points, depressions, and small peaks. Our proposed pre-processing and postprocessing procedures eliminate easily all problems arising from the existence of depressions. The complete method produces excellent
results on a set of real DEMs that we present. All of the proposed procedures are completely automatic. The multilevel skeletonization algorithm has only one parameter, K\textsubscript{T}. The sole influence of this parameter is to change the degree of detail of the extracted flow path. Decreasing this parameter will cause additional branches to appear. However, all flow paths extracted with larger values of K\textsubscript{T} form complete parts of flow paths that are produced in more detail by the use of smaller K\textsubscript{T} values.

Many previous works involve human intervention in the extraction process. This may take the form of queries to an expert systems (Qian, Ehrich, and Campbell, 1990), or the user must adjust parameter settings (such as thresholds in Seemuller, 1989). In the DNESYS system (Qian, Ehrich, and Campbell, 1990) the high-level stage uses a process of evidence collection and an uncertainty-reasoning algorithm. In the evidence-collection process there are many thresholds (such as the distance threshold in distance evidence) that provide support for the connection of two segments. These thresholds are determined either from training data or by queries to the experts. In Seemuller (1989) there are about seven parameters on which the resulting drainage network depends. In general, all of these parameters have a significant influence on the results obtained. In our proposed method no parameter tuning or human intervention is needed.

The presence of depressions in DEMs is another problem in automated drainage network extraction noted by several investigators (e.g., Chorowicz and others, 1992). Depressions have been observed to constitute obstacles to flow routing. The depression-filling method usually is used to avoid this problem (e.g., Jenson and dominique, 1988; Gardner, Sasowsky, and Day, 1990; Jenson, 1991; Tarboton, Bras, and Rodrigues-Iturbe, 1991; Tribe, 1991). Another method is smoothing the data before the actual extraction procedure (e.g., Band, 1986; Seemuller, 1989). We use an approach to the problem which is somewhat similar to smoothing the data. Depressions are filled first and then are postprocessed (see the section on "Enumerating the Network").

**EXTRACTING FLOW PATH BY MULTILEVEL SKELETONIZATION**

The main algorithm of the method extracts flow paths by a process of continuous skeletonization of patches of constant elevation. This process, forming the heart of the algorithm, works almost perfectly everywhere, except in special situations which will be treated by an additional enumeration process. First, we will describe the main algorithm and then go on to discuss in detail the enumeration process that eradicates loops from flow paths.

The multilevel skeletonization (MLS) algorithm is a generalization of a recent skeletonization algorithm for binary pictures (Riazanoff, Cervelle, and Chorowicz, 1990) to include the situation of multi-elevation DEMs. Our MLS algorithm starts from pixels of maximal elevation in the map and scans the map elevation-level by elevation-level. The scanned elevation-levels are just those represented by the DTM and are limited by the resolution of the digital map at hand. The scanning process replaces the elevation of processed pixels by a "background" (Bgrnd) level of value zero. This is a special, non-existent, zero elevation value that is used to designate elevations that have been processed by any of the algorithms. As a first step, all local maxima are replaced by the Bgrnd value, so processing starts at the boundary of levels below local maxima. Iterating over all elevations in the map, from Max...Elevation to Min...Elevation, enables the algorithm to test for curvature along lines of equal height by using simple conditions on the number and spread of background pixels in the 3 x 3 neighborhood of each scanned pixel.

To understand the underlying rationale for our MLS algorithm one has to take notice that there are in general two types of criteria that serve to extract pixels that lie on flow paths. One criterion relates to the curvature of contours of constant elevation. High contour curvature signals a flow path. In our skeletonization algorithm this criterion is expressed as a condition for a large enough number of Higher...Elevation pixels in the immediate neighborhood of a pixel belonging to the elevation currently being processed.

The threshold number K\textsubscript{T} can be between 3 and 6; drainage networks of differing detail will be extracted depending on this parameter. The map is processed one elevation at a time and from top to bottom. As a result, flow paths are extracted downhill and neighboring pixels of higher elevation are replaced by the special (Bgrnd) value. A pixel lying at the edge of the map is defined to have "background" neighbors; we will denote this special value by Bgrnd. The condition on a large enough number of neighboring background pixels is in one to one correspondence with the curvature of the neighboring patch of higher elevation.

The contiguous K\textsubscript{T} of each contour pixels is estimated in our MLS algorithm by the maximum number of contingent pixels of Bgrnd value among the pixel's eight neighbors. In the following four examples of 3 x 3 neighborhoods, pixels of Current...Elevation are black and pixels of higher elevations (i.e. currently Bgrnd) are white. The value of the convexity parameter K\textsubscript{T} is given below each example.


\[
\begin{bmatrix}
\bullet & \circ & \circ & \circ & \circ & \circ & \circ & \circ \\
\bullet & \circ & \circ & \circ & \circ & \circ & \circ & \circ \\
\bullet & \circ & \circ & \circ & \circ & \circ & \circ & \circ \\
\end{bmatrix}
\]

\[K\textsubscript{T} = 3\]

\[
\begin{bmatrix}
\bullet & \circ & \circ & \circ & \circ & \circ & \circ & \circ \\
\bullet & \circ & \circ & \circ & \circ & \circ & \circ & \circ \\
\bullet & \circ & \circ & \circ & \circ & \circ & \circ & \circ \\
\bullet & \circ & \circ & \circ & \circ & \circ & \circ & \circ \\
\end{bmatrix}
\]

\[K\textsubscript{T} = 4\]

\[
\begin{bmatrix}
\bullet & \circ & \circ & \circ & \circ & \circ & \circ & \circ \\
\bullet & \circ & \circ & \circ & \circ & \circ & \circ & \circ \\
\bullet & \circ & \circ & \circ & \circ & \circ & \circ & \circ \\
\bullet & \circ & \circ & \circ & \circ & \circ & \circ & \circ \\
\end{bmatrix}
\]

\[K\textsubscript{T} = 5\]

\[
\begin{bmatrix}
\bullet & \circ & \circ & \circ & \circ & \circ & \circ & \circ \\
\bullet & \circ & \circ & \circ & \circ & \circ & \circ & \circ \\
\bullet & \circ & \circ & \circ & \circ & \circ & \circ & \circ \\
\bullet & \circ & \circ & \circ & \circ & \circ & \circ & \circ \\
\bullet & \circ & \circ & \circ & \bullet & \circ & \circ & \circ \\
\end{bmatrix}
\]

\[K\textsubscript{T} = 6\]

The other type of criterion for deciding which pixels form flow paths relates to the connectivity of
flow paths. It is expressed in our algorithm as an OR condition that allows a pixel to be on a flow path even if the number of its background neighbors is less than \( K_T \). The connectivity condition checks for neighboring flow paths by counting the number of times pixels in the neighborhood cross the background value. We take more than two such crossings to imply that the pixel lies in the neighborhood of an existing flow path. Connecting the pixel to this neighboring flow path results in a connected network. Consider a 3 x 3 neighborhood:

\[
\begin{align*}
&b3 & b2 & b1 \\
&b4 & a & b0 \quad \text{(setting b0 = b8)}, \\
&b5 & b6 & b7
\end{align*}
\]

Define the number of 4-connected neighbors to be:

\[
\psi_4 = \sum_{i=0,2,4,6} f_i,
\]

where

\[
f_i = \begin{cases} 
1 & \text{if } h_i \neq \text{Bgrnd} \\
0 & \text{if } h_i = \text{Bgrnd}.
\end{cases}
\]

The number of crossings in an 8-connected neighborhood is:

\[
\psi_8 = \sum_{i=0,2,4,6} h_i
\]

with \( h_i = 1 \)

iff \( (h_i = \text{Bgrnd}) \) AND \( ((h_{i+1} \neq \text{Bgrnd}) \) OR \( (h_{i+2} \neq \text{Bgrnd}) \).

In describing the skeletonization algorithm for DEMs we will use the term “Depression” to describe a connected component of the map that is completely surrounded by pixels of higher elevation. A “Depression” may contain patches that belong to several different elevations.

### THE 8-CONNECTED MULTILEVEL SKELETONIZATION ALGORITHM

The map for our main working example covers a 1 x 2 km area in the Negev, digitized by hand to a high resolution (10 x 10 m pixels). It is presented in two forms in Figure 1. On the LHS of Figure 1 the full DEM is presented (color coded to \( \sim \) 50 gray levels). On the RHS a reduced elevation level resolution map is presented. Only one out of four elevation levels appears on the RHS of Figure 1, making it much clearer for demonstrating the extraction process.

**First stage: flow path pixels of high curvature contours**

1. Initialize \((\text{Last\_Map}(i,j))\) and \((\text{Current\_Map}(i,j))\) to the original DEM values.
2. For all Elevations from Max\_Elevation to Min\_Elevation do
   a. Set pixels of elevation higher than Current\_Elevation to Current\_Elevation.
   b. Repeat
      - For all pixel \((i,j)\) of Current\_Elevation do
        i. if the pixel is currently at the boundary \((\psi_4 < 4)\) and
        ii. if the convexity is higher than the threshold \((K > K_T)\) and
        iii. if the pixel is not connected to a flow path in Last\_Map \((\gamma_i \leq 1)\) and
        iv. if the pixel is not connected to a flow path in Current\_Map \((\gamma_i \leq 1)\) then leave the pixel as flow path in Current\_Map
        this is actually a flow-path pixel
        else remove the pixel from Current\_Map \((\text{Current\_Map}(i,j); = \text{Bgrnd})\)
   - Update Last\_Map and Current\_Map.
   Until no more pixels can be removed.

---

**Figure 1.** Original DEM of small area map: on LHS all elevations; on RHS only one-quarter of elevation levels.
Two images are processed by the algorithm. One stores the result of the previous iteration \( \text{Last\_Map}(i, j) \), the other stores the image currently being processed \( \text{Current\_Map}(i, j) \). There are two stages to the MLS algorithm. The first stage extracts flow-path pixels by using a (local) condition of high curvature of elevation contours. The second stage uses a complementary (local) condition of high curvature of elevation contours. The second stage uses a complementary (local) condition of connectivity and connects all pixels of the flow path.

The setting of pixels to the value \( \text{Current\_Elevation} \), in step 2 (a) of the given algorithm, performs that iterative action of eliminating “leftover” pixels of former (higher) elevations. Such pixels are artifacts of the actual resolution of the DEM in the sense that they are left in their original value after processing the corresponding elevation level because they form a one-pixel bridge between neighboring regions of different elevations. Setting these pixels to the value of the next elevation (i.e. \( \text{Current\_Elevation} \)) has the effect of processing them as usual and not forming artifacts such as flows that start at midpoints.

Figure 2 presents a series of snapshots that follow in detail the process of skeletonization of one elevation level. As the skeletonization process progresses the current elevation level shrinks and flow-path pixels result from the iteration of the convexity condition. The MLS algorithm leaves as flow-path pixels areas of higher curvature contours. Figure 3 presents the complete result of the first stage of the MLS algorithm (i.e. only end-points to flow paths are produced). Flow-path pixels (end-points) were extracted only on the basis of curvature of elevation contours. The second stage of the MLS algorithm uses a different condition in its iterations over all elevation levels and as a result produces a connected network.

Second stage: Connecting flow-path segments

The second stage uses the same data structures, initialization and iterations on all elevations from top to bottom, as the first stage. Processed pixels are either changed to \( \text{Bgrnd} \) or to flow-path value.

In reality only one DEM array is needed and the other can be emulated easily (and cheaply) by one scan line and a few variables holding differences between the two Maps. The only reason for having two simultaneous data structures is to be independent of the changes induced by the direction of the scan of the map.

1. Initialize \( \text{Last\_Map}(i, j) \) and \( \text{Current\_Map}(i, j) \) to the original DEM values.
2. Set all flow-path pixels from the first stage processing to a nonremovable value \( \text{Fl} \).
3. For all Elevations from \( \text{Max\_Elevation} \) to \( \text{Min\_Elevation} \) do
   a. Set pixels of elevation higher than \( \text{Current\_Elevation} \) to \( \text{Current\_Elevation} \).
   b. Repeat
      - For all pixel \( (i, j) \) of \( \text{Current\_Elevation} \) do
        i. if the pixel is currently at the boundary \( \phi_i < 4 \) and
        ii. if the pixel is connected to a flow path in \( \text{Last\_Map} \) \( \gamma_k \leq 1 \) and
        iii. if the pixel is connected to a flow path in \( \text{Current\_Map} \) \( \gamma_k \leq 1 \) then leave the pixel intact in \( \text{Current\_Map} \)
        else remove the pixel from \( \text{Current\_Map} \) \( \text{Current\_Map}(i, j) = \text{Bgrnd} \)
      - Update \( \text{Last\_Map} \) and \( \text{Current\_Map} \).
     Until no more pixels can be removed.

Figure 4 presents the final result of the MLS algorithm. All flow-path segments are connected though some loops of flow exist (in boxes). Many end-points of the flow paths marked by circles in Figure 4 will be removed by the enumeration and traversal process described in the next section. The removal of loops will depend on enumeration. The
Skeletonizing a DEM into a drainage network

removal of artificial end-points is the result of the use, by the traversal algorithm, of the queue of pixels that were extracted by the first stage of the MLS algorithm. In other words, the points extracted by using the curvature condition are reused.

**ENUMERATING THE NETWORK**

The general extraction method that is proposed here is composed of the multilevel skeletonization algorithm and of the path-traversal algorithm. The MLS algorithm operates on the original DEM and produces a flow path. As can be seen in Figure 4, this flow path has loops of flow (in white boxes). In order to eliminate the loops from the drainage network we employ a multilevel flow enumeration algorithm. This algorithm iterates over all elevations of the DEM and enumerates the flow-path pixels from bottom to top. The result of the enumeration process is the enumerated drainage network. Following the enumeration a traversal process follows the network from top to bottom. Traversal takes place in the direction of maximal descent and the result is the complete drainage network.

*Flow-path enumeration algorithm*

The main loop iterates on all elevations from Min_Elevation of the DEM to Max_Elevation. For each of the elevations, flow-path pixels are enumerated from the end-points up. Whenever a higher elevation is encountered, the enumeration at the next elevation starts from the highest numbered pixel of the last elevation. This is a way to climb up a network on all branches of flow in parallel. End-points could be of several types, as is specified by the next procedure (see Fig. 5 for the possible situations).

1. Set mark to 1.
2. Determine flow-path pixels of Min_Elevation which satisfy one of the conditions:
   - touching the frame
   - have three or more neighboring flow-path pixels of the same elevation
   term them *end-points*
3. enumerate all *end-points* to 1.
4. forall Current_Elevation from Min_Elevation + 1 to Max_Elevation do
   - forall unenumerated flow-path points of Current_Elevation do
     - if flow-path point has a flow-path neighbor that is enumerated to some number and mark = n
     - then enumerate it to n + 1 and set mark to n + 1
     - else if flow-path point has a Bgrnd neighbor
     - then enumerate this point to mark.
   Until no more unenumerated points in Current_Elevation

The enumeration process can be seen in Figure 6. Notice the enumeration of pixel 69, for example, which is enumerated by the last step of the given

![Figure 5. All possible situations of end-points.](image-url)
algorithm. That is, the two pixels enumerated 69 are only enumerated after the elevation level they belong to has been completed and the variable mark in the given algorithm had the value 69.

**Traversing the enumerated flow-path**

The enumeration process is followed by a process of path traversal on the result of the MLS algorithm. This path traversal procedure uses a simple queue that includes all pixels that were extracted as flow-path segments during the first stage of the MLS algorithm. To eliminate all flow loops in the flow path we use this queue and the enumerated flow path in the following procedure.

**One-Pass Loop Elimination**

while queue not empty do

(1) Pop Current_Pixel from queue
(2) if Current_Pixel is not in Final_Flow_Path (FFP) then
   (a) Insert Current_Pixel into FFP
   (b) Insert into FFP all enumerated pixels in descending order from Current_Pixel

The final result of the drainage network, with no loops, can be seen on Figure 7. The elimination process as given enables the addition of only such branches that go down in elevation from an endpoint. The paths that are eliminated by this process are artifacts of the "continuity condition" of the skeletonization algorithm. In extreme situations, where two regions of similar elevation are connected by a one-pixel-width path, this condition produces a

Figure 6. Enumeration of a flow-path network.

Figure 7. Resulting drainage network for $K_z = 5$. 

flow path where none should be topographically. The loop elimination procedure removes these paths together with loops. It is clear on Figure 7 that all former loops of flow have been eliminated by the two stage enumeration process.

Finally, a series of simple postprocessing steps is made in order to discard one-pixel-size loops from the extracted drainage network. These loops originate because of resolution effects. The discrete nature of the DEM causes the flow path to be extracted at one-pixel resolution. The nature of such one-pixel loop effects is depicted in Figure 8.

**FILLING DEPRESSIONS ON THE MAP**

In order to eliminate the phenomenon of flow paths starting from local minima areas we replace the values of pixels which are part of depressions in the Processed...Map, by a special elevation value Lake. A window from a DEM, in Figure 9, presents an example of eliminating depressions. When the second stage of the MLS algorithm is performed (using the connecting condition), the filled depressions are processed as the lowest elevation on the map. The value Lake is the last to be processed by the iterative process (skeletonization). Figure 10 presents the differences between flow-path extraction of the MLS in the situation of depression filling and of not filling. Note that when depressions are not filled the extraction of nonexistent flow-path pixels occurs. The reason for this phenomenon is simple. Depressions are local minima and as such give rise to end-points of flow. Because of the continuity condition of the skeletonization algorithm, endpoints are connected to paths. The second stage of our extraction algorithm (the enumeration process) will not help in the situation of depressions because they do not necessarily give rise to loops. So, filling depressions to the surrounding elevation level eliminates the artificial flow branches on the LHS of Figure 10 and results in the flow path of the RHS of the same figure.

The algorithm for eliminating ("filling") depressions in the original DEM is as follows:
(1) Copy the DEM to the Auxiliary_Map.
(2) \textbf{forall} pixels on Auxiliary_Map \textbf{do}
  \hspace{1em} \textbf{if} the pixel's value is Min_Elevation
  \hspace{2em} then
  \hspace{3em} set it to Bgrnd
(3) \textbf{forall} elevations from Min_Elevation + 1 to Max_Elevation \textbf{do}
  \hspace{1em} \textbf{forall} pixels of Current_Elevation on Auxiliary_Map \textbf{do}
    \hspace{2em} \textbf{if} there is at least one neighboring pixel
    \hspace{3em} \textbf{of value Bgrnd}
    \hspace{4em} then
    \hspace{5em} set the pixel elevation to Bgrnd on Auxiliary_Map
    \hspace{6em} until
    \hspace{7em} no pixels of Current_Elevation change elevation value.
  \hspace{1em} \textbf{forall} pixels of Current_Elevation that
  \hspace{2em} remain on Auxiliary_Map \textbf{do}
    \hspace{3em} set the pixel elevation to Current_Elevation + 1 on Auxiliary_Map \textbf{and}
    \hspace{4em} set the pixel elevation to Lakes on Processed_Map

\section*{DISCUSSION}

Our drainage-network extraction is based upon the following hydrological assumptions:
\begin{itemize}
  \item Flow paths start near ridges, in places of high (concave) curvature in contour lines.
  \item Flow paths proceed downward continuously until one of the following is true:
    \begin{itemize}
      \item The path encounters another flow path.
      \item The path reaches the boundary of the map.
      \item The path enters some water body that does not drain out in the present map.
    \end{itemize}
  \item Flow paths follow the direction of steepest slope at each point.
\end{itemize}

In some works a problem with flat-bottom areas arises (Chorowicz and others, 1992). The profile scan algorithm of Chorowicz makes use of a simple threshold (1\% of apparent gradient) to determine whether a point has a flat terrain relationship with its immediate neighbor (Chorowicz and others, 1992). Many typical DEMs, such as the one that we have been using in all of our examples (Fig. 11) are sensitive to this thresholding procedure. However, this is not because these maps have flat areas, but rather because they include wide gaps between neighboring contour lines. In our work such a problem cannot arise because the direction of flow does not depend on the distance between contour lines. It depends only on the high curvature segments of the elevation_level currently being processed (first stage of MLS).

The DEM in Figure 11 was generated by digitizing a small, high-resolution area. It includes only \(\sim 50\) elevation levels and is relatively flat. Especially so are the river beds in Figure 11, which are flat and wide. A different map (Sub-Watershed 11 of the USDA-ARS Walnut Gulch Experimental Watershed, Arizona) has a larger number of elevations and the density of elevations also is higher (Fig. 12). The scale

\begin{figure}[h]
\centering
\includegraphics[width=0.8\textwidth]{figure11.png}
\caption{Drainage network for \(K_r = 4\).}
\end{figure}

\begin{figure}[h]
\centering
\includegraphics[width=0.8\textwidth]{figure12.png}
\caption{Original DEM of Watershed 11 area.}
\end{figure}
Figure 13. Drainage network for Watershed 11 map, $K_t = 4$.

Figure 14. Drainage network for Watershed 11 map, $K_t = 5$.

Figure 15. Drainage network for Watershed 11 map, $K_t = 6$. 
of this map also is longer, about 10 times that of the Negev map. The map in Figure 12 was digitized from real topographic data. The final result of our network extraction method on the Watershed 11 map can be seen in Figures 13–15 for three values of the curvature parameter \( K_T \), either 4, 5, or 6. More detail appears for \( K_T = 4 \) and less for \( K_T = 6 \). The extracted network for \( K_T = 4 \) for the Negev map (in Fig. 11) is complete, with no loops of flow and its details correspond perfectly to a subgraph of the network for \( K_T = 5 \) in Figure 4.

Elevation levels can be separated either narrowly or with wide areas of constant elevation, depending on the topography. As noted, network extraction procedures that rely on locating topographic minima are prone to problems when constant elevation areas are widespread. The extracted flow paths tend to be segmented (see Qian, Ehrlich, and Campbell, 1990, for example). Our proposed procedure is immune to such problems, because it skeletonizes the elevations in the map one at a time and only extracts pixels of flow when the curvature of the surrounding elevation levels is high. Because of the continuity condition our skeletonization procedure is immune to gaps of many pixels between adjacent elevation levels (i.e. relatively shallow parts of the network). Figure 16 presents a close view of a typical area in the processed maps of the present investigation, where gaps between neighboring elevations are many pixels wide. Finally, the code for all of our algorithms can be obtained for testing and running from one of us (A. Karneli via ftp karneli@bguvms.bgu.ac.il).

Acknowledgements—The base map for the Negev watershed was kindly provided by Prof. A. Yair of the Hebrew University, Jerusalem, and the digital data for the Walnut Gulch watershed by the Southwest Watershed Center (USDA-ARS) at Tucson, Arizona.

REFERENCES

Figure 1: The original DEM of the small area map: on the LHS all elevations; on the RHS only a quarter of the elevation levels.
Figure 2: A series of snapshots of the iterations of the first stage (curvature only) MLS algorithm, on one elevation level.
Figure 3: The (very segmented) result of the first stage of the MLS algorithm.
Figure 4: The final result of the MLS algorithm. Loops of flow are boxed in white and artificial end-points are circled.
Figure 5: All possible cases of end-points.
Figure 6: The Enumeration of a flow path network.
Figure 7: The resulting drainage network for $K_T = 5$. 
Figure 8: A one pixel loop, resulting from the discrete nature of MLS.
Figure 9: Depressions of more than one elevation (LHS) were replaced by a special (white) value.
Figure 10: LHS: Results of flow path extraction from a map that includes Depressions. RHS: Same process as in LHS, but, with Depression removal.
Figure 11: The drainage network for $K_T = 4$. 
Figure 12: The original DEM of the Watershed 11 area.
Figure 13: Drainage network for the Watershed 11 map, $K_T = 4$. 
Figure 14: Drainage network for the Watershed 11 map, $K_T = 5$. 
Figure 15: Drainage network for the Watershed 11 map, $K_T = 6$. 
Figure 16: An example area with wide gaps between elevation contours.