Entropic Quadtrees and Mining Mars Craters

Rosanne Vetro and Dan A. Simovici

Univ. of Massachusetts Boston, Dept. of Comp. Science, 100 Morrissey Blvd. Boston, Massachusetts 02125 USA
{rvetro,dsim}@cs.umb.edu

Abstract. This paper introduces entropic quadtrees, which are structures derived from quadtrees by allowing nodes to split only when nodes point to sufficiently diverse sets of objects. Diversity is evaluated using entropy attached to the histograms of the values of features for sets designated by the nodes.

As an application, we used entropic quadtrees to locate craters on the surface of Mars, represented by circles in digital images.

1 Introduction

In this paper we introduce a variant of quadtrees, a well-known data strucure used in spatial databases. A *quadtree* $\mathcal{T}$ is a tree structure defined on a finite set of nodes that either contains no nodes or is comprised of a root node and 4 quad-subtrees. In a full quadtree, each node is either a leaf or has degree exactly 4. Our variant of quadtrees requires that each node that has descendents is pointing to an area that has a sufficient level of diversity as assessed by the value of an information-theoretical measure.

We provide an algorithm that captures high complexity areas of an image. This algorithm is used for the detection of circular shapes that can possibly correspond to craters.

The algorithm is composed by two methods. The first method uses an information-theoretical approach to create an edge filter that generates a binary image from complex areas which may contain edges. The second method applies a Circle Hough Transform (CHT) with modified threshold to detect the presence of circular shapes in complex areas. The new threshold is imposed to increase the quality of the results given the lack of prior knowledge about the number of craters in an image and the difficulty to estimate a good threshold for the minimum number of votes required in the parameter space to indicate true center points. Efficient methods for crater detection such as [1], [2] and many others referenced by the authors of [3] have been proposed. We provide a distinct approach where no external pre-processing of the original image other than conversion to the JPEG format and resizing is needed. Likewise, no external image filters are used.

In Section 2 we introduce the framework for the rest of the paper. The notion of entropy associated to a partition is presented as well as its usefulness in measuring diversity.

In Section 3 we introduce the proposed algorithm and explain the searching process. In subsection 3.1 we describe the information theoretic method used for mining complex subareas that may contain edges. The CHT method with modified threshold is described in subsection 3.2. Section 4 contains a description of the experiments and major challenges we faced. Finally, Section 5 contains our conclusions and ideas for future work.

P. Perner (Ed.): ICDM 2010, LNAI 6171, pp. 210–221, 2010.

2 Partitions, Entropy, and Trees

Information theory involves the quantification of information and was created with the purpose of finding fundamental limits on compressing, reliably storing, and communicating data. Entropy is a important measure of information in the theory that quantifies the uncertainty associated with probability distributions.

Let S be a finite set. A *partition* on S is a non-empty collection of non-empty subsets of S, $\pi = \{B_1, \ldots, B_n\}$ such that

(i) $B_i \cap B_j = \emptyset$ for $1 \leq i, j \leq n$ and $i \neq j$;
(ii) $\bigcup\{B_i \mid 1 \leq i \leq n\} = S$.

The sets $B_1, \ldots, B_n$ are referred to as the *blocks* of π.

We denote by $\mathsf{Part}(S)$ the set of partitions of S. For $\pi, \sigma \in \mathsf{Part}(S)$ define $\pi \geq \sigma$ if each block B of π is a union of blocks of σ. It is well-known that the relation "$\geq$" is a partial order on $\mathsf{Part}(S)$. The largest partition on S is the single-block partition $\omega_S = \{S\}$, while the smallest partition on S is $\iota_S = \{\{x\} \mid x \in S\}$.

We define now a partial order relation $\geq_k$ on $\mathsf{Part}(S)$ as follows. If $\pi = \{B_1, \ldots, B_n\}$ and $\sigma = \{C_1, \ldots, C_m\}$, then $\pi \geq_k \sigma$ if the following conditions are satisfied:

1. there exists a subcollection of σ that consists of k blocks $\{C_{j_1}, \ldots, C_{j_k}\}$ such that $\bigcup\{C_{j_\ell} \mid 1 \leq \ell \leq k\}$ is a block B_h of π;
2. for $1 \leq i \leq n$ and $i \neq h$, B_i is a block of σ.

For $k = 2$ the relation $\geq_2$ is the direct coverage relation, where the larger partition π is obtained by fusing two blocks of σ.

If $\pi \in \mathsf{Part}(S)$ and $\pi = \{B_1, \ldots, B_n\}$, its entropy is the number

$$\mathcal{H}(\pi) = -\sum_{i=1}^{n} \frac{|B_i|}{|S|} \log_2 \frac{|B_i|}{|S|},$$

which is actually the entropy of the discrete probability distribution

$$\mathbf{p} = \left(\frac{|B_1|}{|S|}, \ldots, \frac{|B_n|}{|S|}\right).$$

Defining the entropy for partitions rather than for probability distributions has the advantage of linking the entropic properties to the partially ordered set of partitions. An important fact is that the entropy is anti-monotonic relative to the partial order defined on partitions. In other words, for $\pi, \sigma \in \mathsf{Part}(S)$, $\pi \leq \sigma$ implies $\mathcal{H}(\pi) \geq \mathcal{H}(\sigma)$. It is easy to verify that $\mathcal{H}(\omega_S) = 0$ and that $\mathcal{H}(\iota_S) = \log_2 |S|$. This shows that the entropy can be used to evaluate the uniformity of the elements of S in the blocks of π since the entropy value increases with the uniformity of the distribution of the elements of S. Note that as the uniformity increases, so does the associated uncertainty.

If C is a non-empty subset of S, and $\pi \in \mathsf{Part}(S)$, the *trace* of π on C is the partition

$$\pi_C = \{B \cap C \mid B \in \pi \text{ and } B \cap C \neq \emptyset\}.$$

The trace of a partition allows us to define the conditional entropy of two partitions. Namely, if $\pi, \sigma \in \mathsf{Part}(S)$ and $\sigma = \{C_1, \ldots, C_m\}$, then the *entropy of* π *conditioned by* σ is the number

$$\mathcal{H}(\pi|\sigma) = \sum_{j=1}^{m} \frac{|C_j|}{|S|} \mathcal{H}(\pi_{C_j}).$$

It can be shown [4,5] that the conditional entropy is an anti-monotonic function of the first argument and a monotonic function of the second. In other words, $\pi_1 \leq \pi_2$ implies $\mathcal{H}(\pi_1|\sigma) \geq \mathcal{H}(\pi_2|\sigma)$ and $\sigma_1 \leq \sigma_2$ implies $\mathcal{H}(\pi|\sigma_1) \leq \mathcal{H}(\pi|\sigma_2)$.

A *measure* on S is a function $m : \mathcal{P}(S) \longrightarrow \mathbb{R}_{\geq 0}$ such that $m(U \cup V) = m(U) + m(V)$ for every disjoint subsets U and V of S. For example, if S is the set of pixels of a gray image S, $m(U)$ can be defined as the number of pixels having a certain degree of grayness contained by the subset U.

Let D be a finite set. A D-feature function on S is a function $f : S \longrightarrow D$. Each feature function $f : S \longrightarrow D$ defines a partition $\ker f$ on S defined by

$$\ker f = \{f^{-1}(d) \mid d \in D, f^{-1}(d) \neq \emptyset\}.$$

We refer to $\ker f$ as the *kernel partition* of f.

For example, if S is the set of pixels of an image, we could define $f(p)$ as the degree of grayness of the pixel $p \in S$. Another example that is relevant in the study of biodiversity is to consider a set S of observation points in a territory, and define $f(p)$ as the number of species of birds sighted in a certain day in p.

If $C \subseteq S$, then the characteristics of the trace partition $(\ker f)_C$ define the concentration of the values that f takes on the set C. If $D = \{d_1, \ldots, d_k\}$, the blocks of the partition $(\ker f)_C$ have the relative sizes

$$\frac{|f^{-1}(d_1) \cap C|}{|C|}, \ldots, \frac{|f^{-1}(d_k) \cap C|}{|C|}$$

and the distribution of these sizes can be conveniently represented using a histogram.

Definition 1. *Let $\Pi = (\pi_1, \pi_2, \ldots, \pi_n)$ be a descending chain of partitions on S such that $\pi_1 = \omega_S$, $f : S \longrightarrow D$ be a feature function, $m : \mathcal{P}(S) \longrightarrow \mathbb{R}_{\geq 0}$ be a measure defined on S and let $\theta, \mu > 0$ be two positive number referred to as the* entropic threshold *and the* measure threshold, *respectively.*

The entropic tree *defined by Π, f, m, θ and μ is a tree $\mathcal{T}(\Pi, f, m, \theta, \mu)$ whose set of nodes consists of blocks of the partitions π_i such that the following conditions are satisfied:*

(i) *the root of the tree is the set S, the unique block of ω_S;*
(ii) *an edge (B, C) exists in the tree only if $B \in \pi_i$, $C \in \pi_{i+1}$, and $C \subseteq B$;*
(iii) *if B is a block of the partition π_i, then $\mathcal{T}(\Pi, f, m, \theta, \mu)$ contains the set of edges $\{(B, C) \mid B \in \pi_i \text{ and } C \in \pi_{i+1}, C \subseteq B\}$ if and only if $\mathcal{H}((\ker f)_B) \geq \theta$ and $m(B) \geq \mu$.*

If $\mathcal{T}(\Pi, f, m, \theta, \mu)$ contains the set of edges $\{(B, C) \mid B \in \pi_i \text{ and } C \in \pi_{i+1}\}$ we say that the node B is *split* in the tree $\mathcal{T}(\Pi, f, m, \theta, \mu)$. Since splitting involves a sufficiently large value of the entropy and a node of sufficiently large measure, longer paths in the tree point towards subsets of S that contain a large diversity of values of the feature function f.

An entropic quadtree is an entropic tree $\mathcal{T}(\Pi, f, m, \theta, \mu)$ such that $\Pi = (\pi_1, \ldots, \pi_n)$ is a descending chain of partitions on S, $\pi_1 \geq_4 \pi_2 \geq_4 \cdots \geq_4 \pi_n$. The entire image area S corresponds to the root of the quadtree.

The expansion of a node B is based on its entropy value and the predetermined threshold used for the splitting condition, as well as the size of the corresponding sub-area. Only nodes with area greater or equal to the defined minimum window size are expanded. The complex areas corresponding to leaves at the highest level on the quadtree are classified according to the possibility of presence of an edge.

3 Algorithm Description

The algorithm proposed constructs a full entropic quadtree related to the image entropy concentration to find high complexity areas that can also contain edges. Later, a slightly modified CHT is used to detect the presence of circles in the complex areas found during the entropy analysis. The algorithm receives as input the 8 bits gray scale version of an image, a minimum window size for analysis, a threshold relevant to the node splitting condition, the minimum and maximum radius values for the searched craters and a threshold for the CHT. Its output lists the detected craters as well as their estimated center points highlighted and superimposed over the original image. A text file with data indicating the center points, radius and Hough Space bin points of each detected crater is also generated.

The construction of the entropic quadtree is based on the measurements of the entropy in image sub-areas, which can also be regarded as tree nodes.

The entire image area corresponds to the root of the quadtree. The expansion of each node is based on its entropy value and the predetermined threshold used for the splitting condition, as well as the size of the corresponding sub-area. Only nodes with area greater or equal to the defined minimum window size are expanded. The complex areas corresponding to leaves at the highest level on the quadtree are classified according to the possibility of presence of an edge.

First, the algorithm determines the average gray intensity of the original image, as well as the low intensity average(average of gray shades below average intensity) and high intensity average(average of gray shades above average intensity). Then, the pixels in each area with minimum size for analysis are mapped to two different sets according to the thresholds corresponding to the average of low intensity shades or the average of high intensity shades of the original image. Crater edges can be found in areas that contain only dark shades of gray or areas containing light shades of gray.

The classification considers the number of pixels in a minimum size window that are above the high intensity average threshold if at least one pixel in the area has gray shade above the average intensity. Otherwise, if all the pixels in the area have low intensity, the classification considers the number of pixels in the minimum size window that have shade below the low intensity average threshold.

Let n be the number of pixels satisfying one of those conditions and h the height of our minimum window. Also, suppose we have a square window. When $h - 2 < n$

$< h^2 - 1$, the entropy value remains considerably high and the area is classified as a leaf that possibly contains an edge. Only those leaves are relevant to our algorithm.

After the high complexity regions that may contain edges are found, the algorithm determines another threshold corresponding to a high intensity shade which is higher than the high intensity average shade, lower than the maximum intensity found in the image and has the highest histogram value among the shades satisfying the couple previous conditions. This last threshold which we will call "near maximum intensity" threshold is used to highlight high intensity pixels corresponding to edges.

Pixels with light shades of gray(higher than average intensity) that form edges usually have intensity greater than the "near maximum intensity" threshold. Finally, the entropy analysis generates a binary image where pixels with shades of gray below the low intensity threshold and pixels with shades of gray above the near maximum intensity threshold are mapped to white. All the other pixels are mapped to black. The resultant binary image corresponds to the output of the entropy analysis and input of the Circle Hough Transform method. As previously mentioned, the original image does not need any pre-processing. The entropy analysis works as an information theoretic edge filter that generates a binary image from complex areas which may contain edges.

Our next step is to apply the CHT to detect circles in the binary image. As described in subsection 3.2, the CHT method maintains an accumulator array to find triplets (a, b, r) that describe circles where (a, b) is the center of a circle with radius r. Each point (a, b) in the image receives a score value referred to as the *number of votes* equal to the number of points (x, y) fall on the perimeter of the circle (a, b, r). This score is stored in a accumulator array. The detected center points have de highest numbers of votes.

Two stopping conditions are commonly used by the CHT algorithm: the maximum number of circles to be found and a threshold for the minimum number of votes related to a point in the parameter space.

In our application, there is no systematic way to reasonably predict both values. Furthermore, it was observed that for any set of radii where the difference between the minimum and maximum radius is relatively small, the chances of a point to represent a real circle center decreases as the number of votes related to the point gets further from the peak value found in the accumulator array. Points with a number of votes relatively far from the peak value usually correspond to near true center points, near center points of poorly delimited circles or points that received votes in the parameter space simply due to noisy pixels that are not part of any circle edge. To alleviate this problem, we created a new threshold for the number of votes corresponding to the maximum distance from the peak value in the accumulator array as our stopping condition for the CHT method. We also restricted each search to small sets of contiguous radii. Details are described in subsection 3.2.

The algorithm is presented in Fig. 1. The function COMPUTE_ENTROPY evaluates the entropy associated with the histogram of the pixels in the node's area.

The recursive method SPLIT introduced in Fig. 2 expands a node if its feature satisfies the splitting condition and if its area is greater or equal to the predefined minimum area size. Thus, each leaf on the quadtree is classified according to the possibility of

Input: 8 bits gray scale version of an image, a minimum area size, entropy threshold, the minimum and maximum radius, CHT threshold
Result: Detected craters as well as their estimated center points highlighted and superimposed over the original image; a text file with data indicating the center points, radius and Hough Space bin points of each detected crater
begin
  $nId \longleftarrow ROOT$;
  $nLevel \longleftarrow 0$;
  $root \longleftarrow newNode(nId, nLevel, image.width, image.height)$;
  $COMPUTE\_ENTROPY(root)$;
  $SPLIT(root)$;
  $entropyImg \longleftarrow PROCESS\_ENTROPY\_IMAGE()$;
  $COMPUTE\_CHT(entropyImg, minRadius, maxRadius, thrCHT)$;
end

Fig. 1. FIND_CRATERS(*image,minArea,thrEntropy,minRadius, maxRadius,thrCHT*)

presence of an edge and only those which may contain an edge are considered at the next method PROCESS_ENTROPY_IMAGE.

This method generates a binary image representing the entropy analysis to find complex areas that may contain edges. Pixels with shades of gray below the low intensity threshold and pixels with shades of gray above the near maximum intensity threshold are highlighting in white. All remaining pixels are mapped to black.

COMPUTE_CHT detects circles in the binary image with radii between the minimum and maximum values given as arguments. It also highlights the detected craters as well as their estimated center points over the original image and generates a text file with data related to the craters found such as radius, center points and number of points in the bins associated with each center point.

3.1 Information-Theoretical Method

Our method evaluates the entropy of the local histograms of image sub-areas to find high complexity regions. The partition blocks of a node, used for the entropy analysis, consist of pixels with the same shade of gray.

Fig. 3 presents the information-theoretic method proposed. It computes the entropy associated with the histogram of the pixels in a node's area. This histogram is created by the method INSERT_GRAYSHADE. The result generated by COMPUTE_ENTROPY is successively used by the recursive method SPLIT shown in Fig. 2. Only the nodes corresponding to sub-areas of the image where the entropy is above the predefined entropy threshold and have area greater or equal to the pre-defined minimum area size are expanded. We observed that leaves at the highest level in the resultant quadtree may naturally have different associated entropy values. CLASSIFY_LEAF classifies a minimum area node as containing or not an edge. As previously mentioned, only leaves that may contain edges are relevant for our algorithm.

Input: A node n from a quadtree
Result: Expands the node creating four children, if node satisfies the necessary requirements
begin
  if $(n.feature > method\_lower\_bound)$ *and* $(n.area > minArea)$ **then**
    $nLevel \longleftarrow n.level + 1$;
    $nId \longleftarrow n.id + A$;
    $topLeft \longleftarrow newNode(nId, nLevel, n.rect.x, n.rect.y, ;$
    $n.rect.width/2, n.rect.height/2)$;
    $COMPUTE\_ENTROPY(topLeft)$;
    $nId \longleftarrow n.id + B$;
    $topRight \longleftarrow newNode(nId, nLevel, n.rect.x +$
    $n.rect.width/2,\ n.rect.y, n.rect.width/2, n.rect.height/2)$;
    $COMPUTE\_ENTROPY(topRight)$;
    $nId \longleftarrow n.id + C$;
    $bottonLeft \longleftarrow newNode(nId, nLevel, n.rect.x, n.rect.y+ ;$
    $n.rect.height/2, n.rect.width/2, n.rect.height/2)$;
    $COMPUTE\_ENTROPY(bottonLeft)$;
    $nId \longleftarrow n.id + D$;
    $bottonRight \longleftarrow newNode(nId, nLevel, n.rect.x + n.rect.width/2, ;$
    $n.rect.y + n.rect.height/2, n.rect.width/2, n.rect.height/2)$;
    $COMPUTE\_ENTROPY(bottonRight)$;
    $RELEASE(n)$;
    $SPLIT(topLeft)$;
    $SPLIT(topRight)$;
    $SPLIT(bottonLeft)$;
    $SPLIT(bottonRight)$;
  else
    $SAVE\_INFO\_NODE(n)$;
    $DRAW(n)$;
    $RELEASE(n)$;
end

Fig. 2. SPLIT(n)

3.2 Circular Hough Transform Method

The Hough Transform is a standard method for shape recognition in digital images. It was first applied to the recognition of straight lines [6,7] and later extended to circles [8,9], ellipses [10], and arbitrary shaped objects [11]. The Circular Hough Transform (CHT) can be used to determine the parameters of a circle when a number of points that fall on the perimeter are known. A circle with radius r and center (a, b) can be described with the parametric equations:

$$x = a + r\cos\varphi \text{ and } y = b + r\sin\varphi.$$

The locus of (x, y) points in the Hough or parameter space falls on a circle of radius r centered at (a, b). The true center point will be common to all parameter circles, and

Input: A node *n* from a quadtree
Result: The node entropy related to the histogram of the pixels in the area.
begin
  $entropy \longleftarrow 0$;
  foreach $pixel \in n.area$ **do**
    INSERT_GRAYSHADE($HISTOGRAM$,$pixel.shade$);
  foreach $shade \in HISTOGRAM$ **do**
    $p \longleftarrow number\_of\_pixels\_with\_shade$;
    $s \longleftarrow total\_number\_of\_pixels\_in\_the\_node$;
    $g \longleftarrow (p \div s)$;
    $entropy -= (g) \times (\lg_2(g))$;
  if $(n.area == minArea)$ **then**
    $relevantLeaf \longleftarrow CLASSIFY\_LEAF(HISTOGRAM)$;
    if $(!relevantLeaf)$ **then**
      return 0;
  return $entropy$;
end

Fig. 3. COMPUTE_ENTROPY(n)

can be found with an accumulator array that stores the number of votes for each point in the parameter space. Multiple circles with the same radius can be found with the same technique.

The main disadvantage of the transform is the fact that the parameter space corresponds to a 3-dimensional space, which makes the computational complexity and storage requirements $O(n^3)$. If the circles in an image are of known radius r, the search can be reduced to a 2-dimensional space.

The method used in our algorithm searches for all circles with radius between two values given as arguments. It differs from other version of CHT methods because of its stopping condition. For reasons previously mentioned, our method does not use the maximum number of circles or the minimum threshold for the number of votes in order to end the search. Instead, it uses a threshold corresponding to the maximum allowed difference between the peak value in the accumulator array of votes and any other number of votes related to a point in the parameter space.

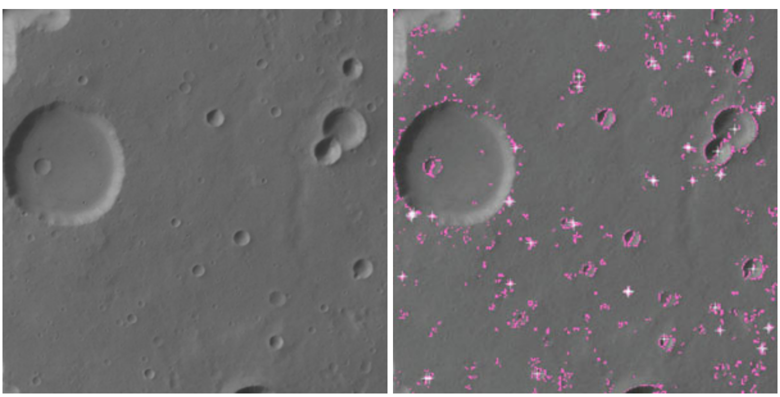
Let $A_{[W][H][R]}$ denote the accumulator array of votes where W is the image width, H is the image height and R depends on the size of the radius set with minimum element $rmin$ and maximum element $rmax$, and on the value for the chosen radius increment i. Let t denote the introduced threshold and v be the greatest value stored in the accumulator array A corresponding to a point (w, h, r) where $0 \leq w \leq W - 1$, $0 \leq h \leq H - 1$ and $0 \leq r <= (rmax - rmin) \div i$. Then an arbitrary point (w', h', r') where $0 \leq w' \leq W - 1$, $0 \leq h' \leq H - 1$ and $0 \leq r' <= (rmax - rmin) \div i$ having v' votes in A is detected as a circle center iff $v - v' \leq t$. We observed that our threshold works well for small groups of contiguous radii. Since the size of the group is small, all the radii are close in value and points corresponding to the center of a circle with one of those radii also have a relatively close number of votes in the parameter space.

Input: image generated by the entropy analysis,minimum and maximum radius, new threshold for stop condition
Result: Detected circles, their estimated center points and radius
begin
    HOUGH_TRANSFORM(entropyImg,minRadius,maxRadius);
    PROCESS_HS();
    GET_CENTER_POINTS(thrCHT);
    DRAW_CIRCLES();
    PRINT_CIRCLES_DATA();
end

Fig. 4. COMPUTE_CHT$(entropyImg, minRadius, maxRadius, thrCHT)$

Therefore, the difference between the number of votes corresponding to centers of true circles that are reasonably well delimited cannot be large when the search is performed for a small group of contiguous radii.

Fig. 4 presents the CHT proposed. HOUGH_TRANSFORM computes the Hough Transform of the binary image generated during the entropy analysis and PROCESS_HS generates an image corresponding to the Hough Space. GET_CENTER_POINTS finds circles and their center points by checking the accumulator array containing votes for each pixel in the image. DRAW_CIRCLES() highlights the detected craters as well as their estimated center points over the original image. PRINT_CIRCLES_DATA generates a text file with data related to the craters found such as radius, center points and number of points in the bins associated with each center point.



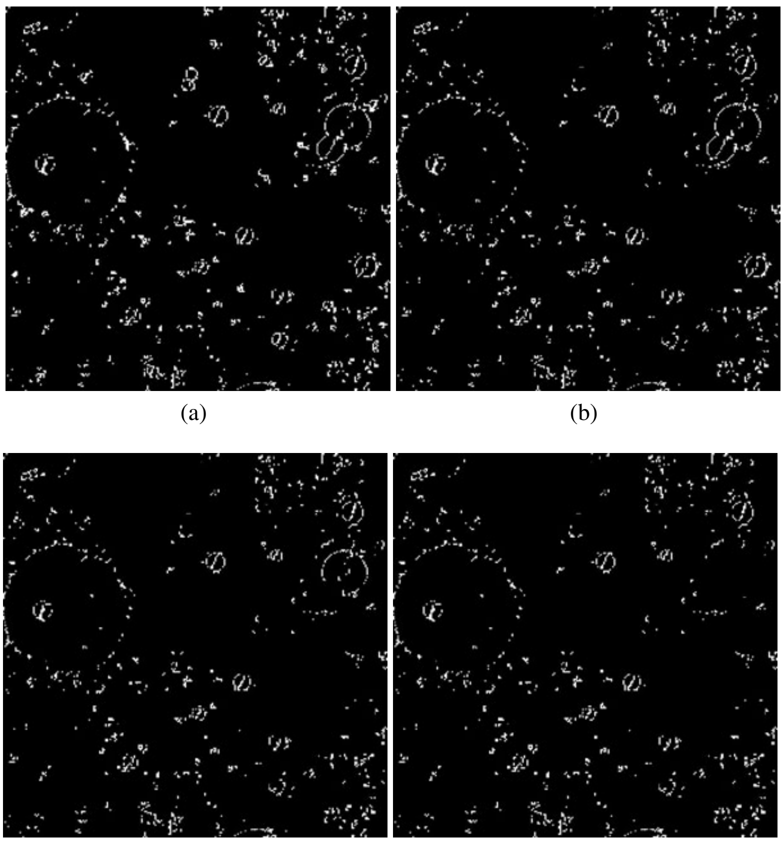
(a) (b)

Fig. 5. Sample image from Mars surface (768x768, 24 bits per pixel) (a) Original. (b) Final image generated by the algorithm. Detected craters and their estimated centers are highlighted in pink and white on the original image.

4 Experimental Results

Experiments were performed over the decompressed 768x768, 8 bits gray scale version of the JPEG digital image corresponding to a picture of Mars surface presented in Fig. 5(a). This image was obtained from the original 24 bits/pixel PGM digital image labeled 3_24 used as training site by the authors of [2]. 3_24 corresponds to one section of a footprint image(h0905_0000) from the High Resolution Stereo Camera (HRSC) instrument of the MarsExpress orbiter. This footprint is about 8248 x 65448 pixels in size and was split into 264(6 x 44) sections of 1700 x 1700 pixels each. Image 3_24 corresponds to one of those sections.

The use of gray scale images allowed the methods to be applied over a reduced color space. We used a 3×3 minimum area for the entropy analysis and an high entropy



(a) (b)

(c) (d)

Fig. 6. Intermediate results of crater detection. (a) Binary image generated by the entropy analysis. (b) Binary image generated after detection of craters with radius between 5 and 20. (c) Binary image generated after detection of craters with radius between 21 and 36. (d) Binary image generated after detection of craters with radius between 37 and 52. For (b), (c) and (d), the pixels corresponding to the circles detected by the CHT are mapped to black.

threshold equal to 3 due to the heavy presence of texture in the original image. Images corresponding to natural scenes, objects and faces with a textured background or images with a high level of noise contain a large amount of information. It is natural that those images contain more areas with high entropy than images with less textured background. Fig. 6(a) shows the binary image generated during the entropy analysis.

We chose a threshold equal to 30 for the CHT method and divided the search into runs containing 15 contiguous values of the radius. We focused on searching for craters with radii varying from 5 to 52 pixels. It was also observed that the choice regarding the search for only 15 radii at a time, combined with a threshold equals to 30 provided reasonably good results. Since the values of the radius in each run are close, the difference between the number of votes for the points corresponding to centers of well delimited circles is usually not greater than 30. Our algorithm was able to detect 50 craters with radii varying between 5 and 20 pixels, 2 craters with radii varying from 20 to 36 pixels and one crater with radius equal to 45 pixels.

Fig. 5(b)shows the final image generated by the algorithm. The detected craters and estimated center points are highlighted over the original image. Notice that for some craters, the center points are slightly shifted to the left or right of the true center point because the characteristic shadow inside the crater is also detected as an edge by the entropy analysis. As presented in Figs. 6(b), 6(c) and 6(d), the algorithm cleans the areas of the entropy analysis image corresponding to the craters found(by mapping their pixels to black) after each CHT run. This cleaning process helps to decrease the amount of noise and therefore undesirable circles overlapping for subsequent runs.

The heavy presence of texture in the image can highly impact the quality of the intermediate image generated by the entropy analysis, which works also as a edge detector tool based on entropy. As the level of texture or noise increases, so does the entropy of regions in the picture. As consequence, the distinction between hight entropy nodes that may contain edges becomes harder. On the other hand, the quality of the results generated by the CHT method highly depend on the quality of the entropy analysis image taken as input. Specially, as the detected edges get more and more similar to the real crater edges, it becomes easier for the CHT method to accurately recognize those circles corresponding to craters. We noticed that the image generated by the entropy analysis does not show all the possible true edges corresponding to crater borders. In order to avoid the capturing of heavy noise, we use a high entropy threshold. By using such high threshold, the algorithm cannot capture true crater borders in areas where the variance among the pixels is not high. As a consequence, those craters cannot be detected by the CHT method. Therefore, improving the detection of edges for heavily noisy or textured images during the entropy analysis can directly impact the quality of the final results. Results also show that the algorithm may detect a larger number of false positives craters as the radius increases. Remains of smaller circles that were not completely cleaned from the binary image due to the imperfection of circle edges, may contribute for undesirable circle overlapping in the Hough Space.

5 Conclusion

An algorithm to detect circles that can possibly correspond to craters in images was introduced. The algorithm performs an information-theoretic analysis of the histogram of sub regions of the image in order to find complex areas that may contain edges. A modified CHT detects circles in those complex areas and provides information about center points and radius of the circles found. A threshold corresponding to the maximum distance from the peak value in the accumulator array is used as stopping condition for the method.

There is no external pre-processing of the original image 3_24 other than conversion to JPEG format and resizing. No external edge filter is used to process the original image prior to the CHT method.The entropy analysis works as an edge filter that generates the binary image given as input to the CHT method. The heavy presence of noise and texture may compromise the quality of the complex areas found during the entropy analysis and impact the quality of the final results. Therefore, by improving the robustness of the entropy analysis against heavy noise and texture, more craters will accurately be detected.

We intend to extend the application of information-theoretical techniques to other structures associated with spatial data sets such as grid-files, (k, d)-trees, and R-trees. Another area of great potential is the application of entropic quadtrees to the identification of terrain areas that contain a high level of biodiversity.

Acknowledgements

The authors thank Dr. Wei Ding for providing the original image 3_24 used in this work as well as the ground truth data corresponding to the manually marked craters in 3_24.

References

1. Bue, B.D., Stepinski, T.F.: Machine detection of martian impact craters from digital topography data. IEEE Transactions on Geoscience and Remote Sensing 45(1), 265–274 (2007)
2. Urbach, E.R., Stepinski, T.F.: Automatic detection of sub-kilometer craters in high resolution planetary images. Planetary and Space Science 57, 880–887 (2009)
3. Salamuniccar, G., Loncaric, S.: Gt-57633 catalougue of martian impact craters developed for evaluation of crater detection algorithms. Planetary and Space Science 56, 1992–2008 (2008)
4. Simovici, D.A., Jaroszewicz, S.: An axiomatization of partition entropy. IEEE Transactions on Information Theory 48, 2138–2142 (2002)
5. Simovici, D., Jaroszewicz, S.: Generalized conditional entropy and decision trees. In: Extraction et Gestion des connaissances – EGC 2003, Lavoisier, Paris, pp. 363–380 (2003)
6. Leavers, V.F.: Survey: which hough transform? Computer Vision Graphics and Image Processing: Image Understanding 58(2), 250–264 (1993)
7. Illingworth, J., Kittler, J.: Survey: a survey of the hough transform? Computer Vision Graphics and Image Processing 44(1), 87–116 (1988)
8. Duda, R.O., Hart, P.E.: Use of the hough transformation to detect lines and curves in pictures. Communications of the Association of Computin Machinery 15(1), 11–15 (1972)
9. Davis, E.R.: A modified hough scheme for general circle location. Pattern Recognition Letters 7, 37–43 (1987)
10. Yip, R.K.K., Tam, P.K.S., Leung, D.N.K.: Modification of hough transform for circles and ellipses detection using a 2-dimensional array. Pattern Recognition 25, 1007–1022 (1992)
11. Pao, D.C.W., Li, H.F., Jayakumar, R.: Shape recognition using the straight line hough transform: theory and generalization. IEEE Transactions on Pattern Analysis and Machine Intelligence 14(11), 1076–1089 (1992)