TRELLIS-BASED CIRCLE DETECTION

Kenneth E. Hild II[1], *Santosh Mathan*[2], *Deniz Erdogmus*[1], *Misha Pavel*[1]

[1]Department of Biomedical Engineering
Oregon Health and Science University, Portland, OR, USA
[2]Human Centered Systems Group
Honeywell Labs, Seattle, WA, USA
k.hild@ieee.org, santosh.mathan@honeywell.com, derdogmus@ieee.org, pavel@bme.ogi.edu

ABSTRACT

We introduce a new method for finding circles in images. The proposed method assumes a given pixel is the center of a prospective circle, attempts to fit a circle at that location to the data, and then it scans over all possible pixels. The score at each assumed center location is found by traversing a trellis structure. The trellis allows for gaps in the prospective circles and it enforces a global all or none constraint. The proposed method is compared to a binary matched filter using real visible-spectrum satellite images, where the targets are the circular-shaped silos of surface-to-air missile sites. The proposed method performs noticeably better than the binary matched filter for the data used in this study.

1. INTRODUCTION

Computer vision is the field of study in which the goal is to develop algorithms that obtain information from images. Example applications include, but are not limited to, automatic target recognition [1], target classification [2], document recognition [3], pose estimation [4], tracking [5], event detection [6], and autonomous vehicles [7]. One approach to full-scale computer vision is to build models for a set of primitive shapes, such as circles and rectangles. Herein we introduce a method for finding circles that uses a trellis to compute the cost of fitting a circle to the data.

2. TRELLIS-BASED CIRCLE DETECTION

An overview of the proposed trellis-based circle detector is given in Table I In this table $w = 2r - 3$ is the width of the search window (hence $r = \frac{w+3}{2}$), B is the set of pixels of the original image that that pertain to the current search window, (x_o, y_o) denotes the center of the search window (which is assumed to be the center of a prospective circle), J_{min} is the cost of the least cost path through the trellis (this value depends on, among other things, the location of the search window), l_x is the number of rows of pixels in the original image, and l_y is the number of columns of pixels in the original image.

TABLE I. PSEUDO-CODE FOR THE PROPOSED METHOD

1. Let $x_o = r$
2. Let $y_o = r$
   (a) Let $B = \{(x, y) \mid |x_o - x| < r, |y_o - y| < r\}$
   (b) Tessellate the elements of B (see Table II)
   (c) Find the outer border of each B_n (see Table III)
   (d) Find the least cost trellis path (see Table IV)
   (e) Let $J(x_o, y_o) = J_{min}$
   (f) Let $y_o \leftarrow y_o + 1$
   (g) Go to Step 2(a) until $y_o + r > l_y$
3. Let $x_o \leftarrow x_o + 1$
4. Go to Step 2 until $x_o + r > l_x$

The tessellation step, outer border determination step, and trellis step are explained further in the subsequent sections. The raw output of this algorithm is a map of the least cost values. These values are then sorted so that the smallest cost appears first in the list. The first (estimated) positive corresponds to the lowest cost. Second and subsequent positives are added by proceeding down the list. A prospective positive is added to the list only if it does not exceed a maximum amount of overlap with all the previously selected positives (each of which is represented by an appropriately centered $(w \times w)$ window). Additions to the list of positives are made until a stopping criterion is met. One option for the stopping criterion is to use a maximum cost threshold. A second option is to find a pre-determined number of positives per image.

2.1. Tessellation

The tessellation step divides the portion of the image contained in the search window, B, into $N + 1$ mutually exclusive and collectively exhaustive blobs, where the pixels belonging to a single blob are chosen based on proximity, similarity of grayscale values, and correct approximate size.

The motivation behind this step is that, in the absence of significant contextual information, human observers tend to group neighboring pixels having similar grayscale values.

The tessellation is performed using the pseudo-code shown in Table I, where $p_{i,j}(C = c)$ is the grayscale value of pixel (i, j) having class c, $p_{x,y}$ is the grayscale value of pixel (x, y) having an unknown class, the neighbors to pixel (i, j) are given by $(i - 1, j)$, $(i + 1, j)$, $(i, j - 1)$, $(i, j + 1)$, σ is the standard deviation of the original image, and δ is a user-defined scale parameter.

TABLE II. PSEUDO-CODE FOR TESSELLATION

1. Let $c = 1$
2. Find an unlabeled pixel (i, j) and label it with class c
   (a) Find all unlabeled neighbors to all class c pixels
   (b) Label each neighboring pixel (x, y) with class c if

$$|p_{i,j}(C = c) - p_{x,y}| < \delta\sigma$$

   (c) Go to Step 2(a) until all neighbors are checked
3. Let $c \leftarrow c + 1$
4. Go to Step 2 until all points have been labeled
5. Discard blobs that are too small or that extend to any edge of the search window

The tessellation produces N mutually exclusive blobs, which are denoted B_n for $n = 1, \ldots, N$. The N^{th} blob, by definition, equals the empty set. It is used below to allow for gaps in the target circle. Using B_{N+1} to denote all the pixels from discarded blobs we have,

$$B = \bigcup_{n=1}^{N+1} B_n \tag{1}$$

After the blobs have been defined we then find the outer border of each blob.

2.2. Determination of Outer Border

The outer border of each blob is a set of pixels, with at most one pixel per radial zone (described below), that are farthest from the assumed center, (x_o, y_o). Hence, the outer border depends on the location of the search window. The symbol Θ is used to denote the set of pixels that constitute the current search window. Determination of the outer border also depends on M mutually exclusive radial zones, which are denoted Θ_m for $m = 1, \ldots, M$. The set of pixels that constitute a single radial zone depends on the minimum-allowed radius, r_{min}, the width of the (square) search window, w, and a user-defined radial resolution parameter, M. The radial zones are defined such that the angles between the leading edge of the radial zones and the abscissa are equispaced from 0 to 2π radians. An example where $M = 40$ and $w = 67$ pixels is shown in Figure 1. The extent of the black circle in the center of this figure roughly corresponds to twice the minimum-allowed radius (this circle becomes distorted as the minimum-allowed radius is reduced). Using Θ_{M+1} to denote the mutually exclusive set of pixels that constitute the circular center of the search window we have,

$$\Theta = \bigcup_{m=1}^{M+1} \Theta_m \tag{2}$$

The appropriate range of values for M depends on r_{min} and w. Increasing the value of M increases the radial resolution of the search. However, the resolution cannot be increased indefinitely since the pixels have a non-zero extent. The values of r_{min} and w should be chosen based on prior knowledge, or lack thereof, of the expected range of radii of the target circles. For maximal resolution and ignoring computational complexity, the value of M should be chosen to be the largest value such that all M radial zones extend contiguously from Θ_{M+1} to one of the edges of the search window.

The outer border of each blob, R_n, is determined using the pseudo-code shown in Table II, where R_n is the set of pixels constituting the outer border for blob B_n.

TABLE III. PSEUDO-CODE FOR OUTER BORDERS

1. Let $n = 1$
2. Let $R_n = \emptyset, m = 1$
   (a) Compute $A_{m,n} = \Theta_m \cap B_n$
   (b) Find pixel (x_m,y_m) having the largest $d(\Theta_m,B_n)$

$$d(\Theta_m, B_n) = \max_{(x,y)\in A_{m,n}} \left\{(x - x_o)^2 + (y - y_o)^2\right\}$$

   (c) $R_n \leftarrow \{R_n \cup (x_m, y_m)\}$
   (d) Let $m \leftarrow m + 1$
   (e) Go to Step 2(a) until $m > M$
3. Let $n \leftarrow n + 1$
4. Go to Step 2 until $n > N$

The parameter $d(\Theta_m, B_n)$ represents the distance between the assumed center of the prospective circle and the farthest pixel that is in both Θ_m and B_n.

2.3. Trellis

It is expected that each target is represented by more than one blob. Hence, we need to select a set of up to N blobs. Due to combinatorial explosion it is impractical to try all possible combinations for large N. For this reason we use a greedy approach. Our approach uses a trellis to calculate the total cost for many of the possible paths. It allows for the existence of gaps in the target circle and it is constrained so that a blob is either always used or never used. This

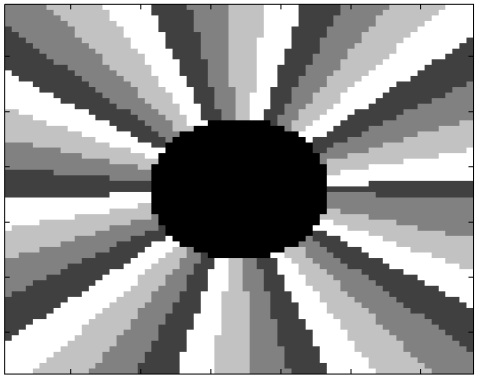

Fig. 1. Example of the radial zones Θ_m, where $M = 40$ and $w = 67$.

approach is similar to maximum-likelihood sequence detection (MLSD) [8] and the hidden Markov model (HMM) [9]. The main differences are the introduction of a gap cost and the all or none constraint, the latter of which prevents the proposed approach from considering all possible paths through the trellis since it is a global constraint (as opposed to a local constraint, the cost of which can be determined at each step of the trellis).

The costs of many of the possible paths are determined using the pseudo-code shown in Table IV. In this table J_m is the vector of N costs at the m^{th} row of the trellis, $J_m(n)$ is the n^{th} element of J_m (which corresponds to the path that passes through the border of blob B_n located in radial zone Θ_m), $T(\Theta_m,B_n,\Theta_{m-1},B_k)$ is the transition cost, and T_{gap} is the cost of a gap.

TABLE IV. PSEUDO-CODE FOR TRELLIS

1. Let $m = 2$ and $J_1 = [0\ 0\ \cdots\ T_{gap}]$
2. Calculate the new cost vector

$$J_m(n) = \min_k \{J_{m-1}(k) + T(\Theta_m,B_n,\Theta_{m-1},B_k)\}$$

$$T(\Theta_m,B_n,\Theta_{m-1},B_k) = \begin{cases} [d(\Theta_m,B_n) - d(\Theta_{m-1},B_k)]^2 & (n<N) \\ T_{gap} & (n=N) \\ \infty & \text{(invalid path)} \end{cases}$$

3. Let $m \leftarrow m + 1$
4. Go to Step 2 until $m > M$
5. Let $J_{min} = \min_k J_M(k)$

As can be seen from the table the n^{th} element of the current cost, $J_m(n)$, equals the sum of one element of the most recent previous cost, $J_{m-1}(k)$, and the cost of transitioning from $J_{m-1}(k)$ to $J_m(n)$, the latter of which is denoted $T(\Theta_m,B_n,\Theta_{m-1},B_k)$ and equals the squared difference between successive radii (when $n < N$). The infinite transition cost is used to enforce the all or none constraint.

3. RESULTS

The testing is performed using 7 real, grayscale, visible-spectrum satellite images. The typical size of the images is $(3 * 10^4 \times 2.7 * 10^4)$ pixels. Each image has one surface-to-air (SAM) site and each SAM site contains 6 silos. Due to the size of the images both methods were performed in two stages. The first stage uses data that were first down-sampled by a factor of 16 (4 along each axis). The second stage uses the full-resolution image, but it only evaluates the $L_{10\%}$ best locations found in the first stage, where $L_{10\%} = 0.10(l_x l_y)/w^2$ is 10% of the ratio of the number of pixels in the image to the number of pixels in the search window. For the proposed method the minimum blob size is 5 in the first stage and 15 in the second stage, no overlap is allowed among all the positives, and δ is chosen as the value that produced a mean blob size closest to 20 for the first stage and 140 for the second stage. This choice of stopping criterion allows the two methods to be directly compared even though they use different metrics.

The binary matched filter method first makes the image contained in the search window binary and then it estimates the correlation of the resulting image with a binary circular filter (having a mean of 0 and a standard deviation of 1). More specifically, for each location of the search window two new images are created, A_1 and A_2. For both A_1 and A_2 the image pixels contained in the search window are converted to either α_0 or $alpha_1$, where $\alpha_1 > \alpha_0$ and the two values are chosen so that the resulting image has a mean of 0 and a standard deviation is 1. The difference between the two is that A_1 assigns α_1 to the set of pixels that are at least β standard deviations above the mean and α_0 to all other pixels, whereas A_2 assigns α_1 to the set of pixels that are at least β standard deviations below the mean. The final output of the method is a map of the maximum of the two correlation coefficients (the first correlation coefficient is the correlation between A_1 and the circular filter and the second is the correlation between A_2 and the circular filter). We used $\beta = 1$ to create the figures below.

Figure 2 shows an example where the search window is centered on a silo. This figure also shows the outputs of several different steps of the proposed trellis-based method. The bottom right subfigure represents the union of R_6 and R_{12}, which happens to correspond to the lowest cost path through the trellis for this particular search window. Figure 3 shows the silo detection ROC curve for both stages of both methods. The performance of stage 2 necessarily equals the performance of stage 1 (for the same method) when stage 2 uses all of the positives found in stage 1. Figure 4 shows the SAM site detection ROC curve. For this figure we conclude

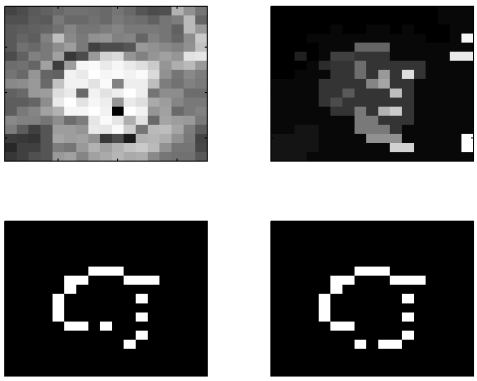

Fig. 2. Upper left: example silo (downsampled by 16). Upper right: image formed after the pixels are classified. Each color represents a different class. Lower left: the border of the 6^{th} blob, R_6. Lower right: the union of R_6 and R_{12}.

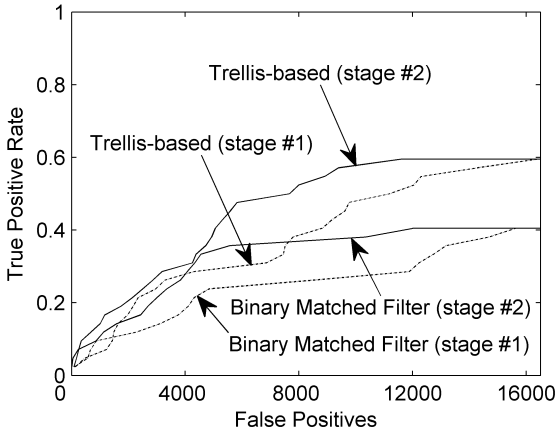

Fig. 3. Silo detection ROC curve.

that the SAM site has been found is there are any positives found within the SAM site, e.g., if 1 or more of the 6 silos that constitute the SAM site are found.

4. CONCLUSIONS

We have introduced a trellis-based method for finding circles. The proposed method performed better than a standard binary matched filter on the satellite data that we tested. The most noticeable weakness with the proposed method is that it also found multiple square-shaped objects, such as houses. We expect that the method can be improved by either adding a square/rectangle detector or by including information on the distribution of the error between the shape that corresponds to the lowest cost and the best-fit circle since the distribution of the error between a circle and a square is much different than the distribution of the error between a circle and a noisy circle.

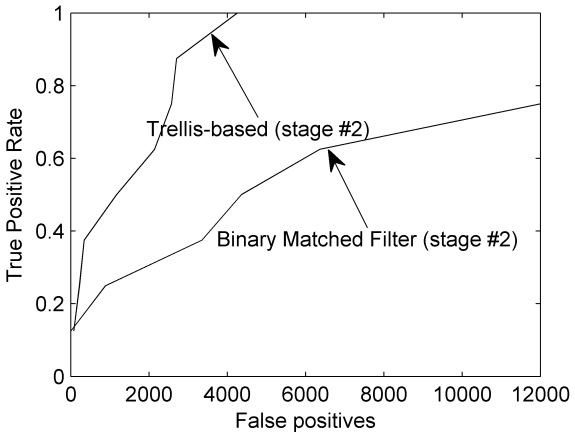

Fig. 4. SAM site detection ROC curve.

5. REFERENCES

[1] W.E. Pierson, T. Ross, "Automatic target recognition (ATR) evaluation theory: a survey," Proc. of SPIE, Orlando, FL, Vol. 4053, pp. 666-676, Apr. 2000.

[2] I. Bilik, J. Tabrikian, and A. Cohen, "GMM-Based Target Classification for Ground Surveillance Doppler Radar," IEEE Trans. on Aerospace and Electronic Systems, Vol. 42, No. 1, pp. 267-278, Jan. 2006.

[3] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-Based Learning Applied to Document Recognition," Proc. IEEE, Vol. 86, No. 11, pp. 2278-2324, Nov. 1998.

[4] C. Unsalan, "A model based approach for pose estimation and rotation invariant object matching," Pattern Recog. Letters, Vol. 28, No. 1, pp. 49-58, Jan. 2007.

[5] T. Zhang and D. Freedman, "Tracking objects using density matching and shape priors," IEEE Intl. Conf. on Computer Vision, Vol. 2, pp. 1056-1062, Oct. 2003.

[6] T. Gandhi and M.M. Trivedi, "Motion analysis for event detection and tracking with a mobile omnidirectional camera," Multimedia Systems, Vol. 10, No. 2, pp. 131-143, Aug. 2004.

[7] G. Bradski and A. Kaehler, "Robot-vision signal processing primitives," IEEE Signal Proc. Magazine, Vol. 25, No. 1, pp. 130-133, Jan. 2008.

[8] J.G. Proakis, Digital Communications, 3rd ed., McGraw-Hill, Inc., Boston, MA, 1995.

[9] L.R. Rabiner " A tutorial on hidden Markov models and selected applications inspeech recognition," Proc. of the IEEE, Vol. 77, No. 2, pp. 257-286, Feb. 1989.