Shape characterization with turning functions

Carlos F. S. Volotão, Rafael D. C. Santos, Guaraci J. Erthal and Luciano V. Dutra.
Image Processing Division and Computing and Applied Mathematics Associated Laboratory
National Institute for Space Research, São José dos Campos, Brasil
{volotao, dutra, guaraci}@dpi.inpe.br, rafael@lac.inpe.br

Abstract—The goal of image segmentation is to separate objects from a cluttered background. It is an important step on the identification of shapes contained in the image. A turning function is a 1D shape signature and can be used as a 2D shape representation. We propose a way to identify and modify classes of shapes by operating directly on the turning function space. In this work we present a formal definition of turning functions and a way to identify circles and lines in turning functions in order to find complex shape classes determined by user defined constraints.

Keywords - shape analysis; object recognition, shape modeling; shape matching, turning function, segmentation.

I. INTRODUCTION

Image segmentation is the process of identification of contiguous pixel regions (image segments) that may belong to the same object on the original scene. The classical approach to do this involves no prior knowledge about the objects [1]. Thresholding, edge extraction and region growing are three strategies to perform image segmentation. These strategies can be useful as a step on the extraction of objects from images toward shape representation/ identification and scene understanding. Shape analysis is usually performed on object contour data extracted from a segmented image and represented as a set of 2D points (or 2D vectors) in Euclidian space.

To improve the segmentation process we suggested [2] an object-based approach that cause image segments to properly match the geometric behavior expected of the shapes of each class of objects. We introduced the idea of using the turning function transform to model shapes and enhance segmentation capabilities.

During the segmentation process the extraction of edges of each image segment result in a closed polygon. There are several categories of procedures to recognize objects in images by their contours. The most common are direct approaches in which information is obtained directly from the 2D contour in raster or vector representation. These approaches use the contour itself with some preprocessing to filter, regularize or simplify it for further matching or classification. Some capabilities are enhanced in a transformed space, which maps from the original image space into a new space. The shape represented in a parametric space has a new geometry. Basic tools are necessary for the identification and handling of ways to explore the new properties inherent to the transformed shape. In a transformed space the same real world object have a diverse geometric configuration in which the extraction of edge information and the assignment of classes are done in a different way.

Shape characterization and classification are discussed by [3]. General principles and models of image analysis are presented and discussed by [4]. Signature functions can be used to characterize object contours for further shape classification and analysis [5, 6]. The method presented in this work is based in the transformed space of the turning function where the shape classes can be modeled by rules or constraints to match models.

II. FOUNDATIONS OF TURNING FUNCTION SPACE

In this section we present the foundations of the turning function space and explore some of its features and limitations. We define first turning function and then its feature space.

A. *Turning function*

An image segment resulting from a segmentation process is a closed shape that can be represented as a polygon. Any finite two-dimensional polygon represented in the Euclidean plane can be converted into a turning function.

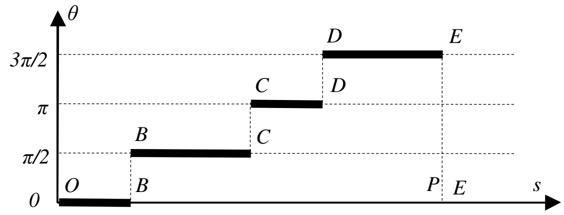

Figure 1. Turning function of a rectangle *OBCD*, where *O≡E*.

Turning functions, also called turning angle functions [7] and ψ-s curves [8, 9], are represented on a tangent angle versus arc length plot and consists on a 1D description of the 2D shapes. They are generated by traversing the boundaries and obtaining the angle between a reference line and the tangent of that point to the boundary. Fig. 1 shows a rectangle *OBCD* represented by its turning function. In this paper when we write an angle x we mean x radians.

B. Turning function space

Turning functions can be represented graphically as a mapping from Euclidean space to a space named turning function space and represented by Θ. The function is bijective and thus modifications in one space changes the other. The transformed space is a cylindrical projection. Polygons are represented by a finite number of horizontal line segments in Θ. The turning function $\theta(s)$ starts from a sample point on the contour and follows the curve counterclockwise [10].

C. Related methods

When an original image is transformed to other spaces certain techniques can be applied to obtain better results. Separable and orthogonal transforms, Fourier transform, Walsh and Hadamard transform, discrete cosine, Gabor, Wavelet, Hotelling, Radon and Hough are discussed in [8] as playing an important role in several image processing techniques.

Template matching can be used to detect edges in blobs or contours. That technique needs the shape to be known *a priori* and it needs the shape's estimated size and orientation. Template matching approaches use only known object models and can use several instances of the same object to select one with the best matching score.

Flexible shape extraction includes deformable templates, active contours, snakes, shape skeletonization and active shape models. The main goal of these methods is to evolve to the target solution or adapt their result to the data when it is not possible to model a shape with sufficient accuracy [12].

Turning function belongs to the category of shape signature functions, which is a shape descriptor as are the Fourier descriptors. Reference [7] use chain code and turning function to guide polygonal approximation of contours. Our approach differs on this by proposing modification on the space rather than in shape.

D. Considerations about turning functions

A turning functions is sensitive to noise, it has a computational cost associated with matching and it may cause errors that occur in matching when the border is slightly changed [11]. The arc length grows locally with noises resulting in the whole perimeter increase. The withdrawal of a speckle, for example, makes the right part of the function to be modified. In this proposed method the goal is to simplify some operations for certain types of shape, particularly for simple or complex shapes composed of lines and arcs of circles. Many remote sensing man-made urban objects have this property.

III. The Proposed Method

A. Preparing the turning function

A traditional preprocessing step is used to obtain the boundary polygon from the binary blob [13]. The polygon is then converted to the turning function representation.

Ordinarily shape detections are performed in the Euclidean space, but using the turning function space allows the exploration of the simplicity that exists in this representation. One can filter polygons, but at this stage of the studies we performed the filtering before obtaining the turning function and then categorize the shape. Each group of adjacent line segments of a turning function can be simplified by a single segment that has the same start and end points of the group (Fig 2). In this example each five segments form a group.

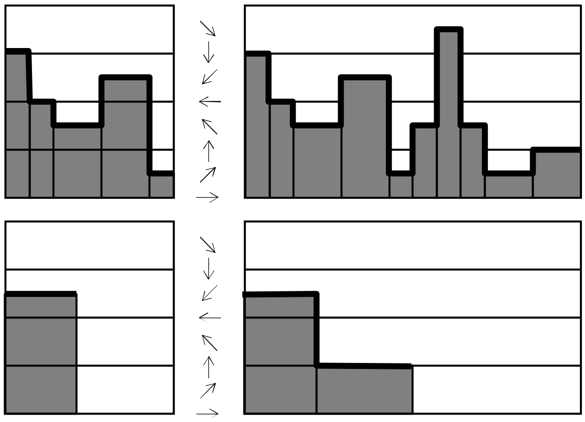

Figure 2. A schematic turning function representation (*upper right*) and its left side (*upper left*) and after substitutions (*bottom*).

Whenever we find a noisy line segment then it must be straightened by replacing it by the resulting vector. Some issues related to the segment's minimum size and noise tolerance are inherent to the definition of the model and are being discussed briefly in this paper. In practical terms we need to recognize whether it is a line segment or not.

Since shapes in digital images are discrete, any curve will be represented as a polygonal approximation. A polygon P can be defined as a set of n vertices, $P = \{p_0, p_1,\ldots, p_{n-1}\}$, where p_{i-1} is adjacent to $p_{i \bmod n}$ $i \in \{1,\ldots, n\}$. Can also de defined as a set of line segments, $P = \{\Delta s_1, \Delta s_2,\ldots, \Delta s_n\}$ where $\Delta s_i = s_{i \bmod n} - s_{i-1}$, is the line segment joining two adjacent vertices p_{i-1} and $p_{i \bmod n}$ in P. Further for each line segment Δs_i , $\theta_i \in [0,2\pi)$ is its slope angle. Furthermore if $u_i(s)$ is the indicator function ($u_i(s) = 1$ if $s \in [s_{i-1},s_i)$ and 0 otherwise) and given that s_0=0, the turning function can be defined by a step function θ, as defined in (1) .

$$\theta(s) = \theta_1 u_1(s) + \theta_2 u_2(s) + \cdots + \theta_n u_n(s) \qquad (1)$$

In order to determine the area of a closed polygon we may use (1). We can define s_i as the edge length from origin (point O, with length s_0=0) to a certain i^{th} position. In Fig. 1 the respective i^{th} positions for points B, C, D and E are 1,2,3 and 4. The area is A, Δs_i is the length of each of the n segments and θ_i is the cumulative deviation angle of the turning function.

$$A = \frac{1}{2}\left\{\sum_{i=1}^{n-2}\left[\Delta s_i \sum_{j=i+1}^{n-1}\left(\Delta s_j \sin\left(\theta_j - \theta_i\right)\right)\right]\right\} \qquad (1)$$

B. Line segment substitution

The operations of the Euclidean domain can be performed either graphically or analytically on the Θ space because there is no loss of information.

To find a line segment in turning function space we first calculate the equivalent line segment between the vertices t_0 and t_f using (3), and then find the signed horizontal component s_H by projecting all the sides on the range onto the axis.

$$s_H = \sum_{t=t_0+1}^{t_f} \Delta s_t \cos \theta_t \qquad (3)$$

Do the same for the vertical signed direction s_V (4). A negative term means an opposite direction and in Θ is equivalent to add π/2 in θ.

$$s_V = \sum_{t=t_0+1}^{t_f} \Delta s_t \sin \theta_t \qquad (4)$$

The orthogonal components (3) and (4) result in a new component which length is s_{Equiv} (5) and which angle is θ_{Equiv} (6).

$$s_{Equiv} = \sqrt{{s_H}^2 + {s_V}^2} \qquad (5)$$

$$\theta_{Equiv} = \arctan\left(\frac{s_V}{s_H}\right) \qquad (6)$$

These equations allow one segment to replace any sequence of line segments keeping the same start point and end point. Particularly in this approach this is very useful because noise on the binary blob greatly alter the graphical representation of the turning function. If instead of s_H and s_V, we have s_A and s_B that are not guaranteed to be orthogonal, we must use the generic equations (7) and (8) instead.

$$s_{Equiv} = \sqrt{{s_A}^2 + {s_B}^2 + 2 s_A s_B \cos(\theta_B - \theta_A)} \qquad (7)$$

$$\theta_{Equiv} = \arccos\left(\frac{s_{Eq}^2 + s_A^2 + s_B^2}{2 s_A s_{Eq}}\right) + \theta_A \qquad (8)$$

As stated, polygons in Euclidean space are represented by line segments in Θ, but we can find larger segments from the alignment of smaller segments and circles based on line geometry. Tolerance parameters are needed in implementation.

C. Arc detection

Once we found all trivial linear segments we could search for circle arcs. In Θ circle arcs are ladder-shaped. The height of each step corresponds to the accumulated value of the angle from the initial segment. If the radius of curvature is consistent with a circle and the angle is below a certain threshold then the figure is a circle arc, otherwise the figure is a polygon.

Fig. 3. presents circles and the turning function contains several stair-like sequences and, as the figure is relatively noiseless, we can visually recognize stair shapes of distinct slopes, meaning different curvatures and therefore, distinct circles or arcs. We can search for the tangent circle radius and compare its curvature radius for each point. The goal is to exploit the simplicity of the geometry.

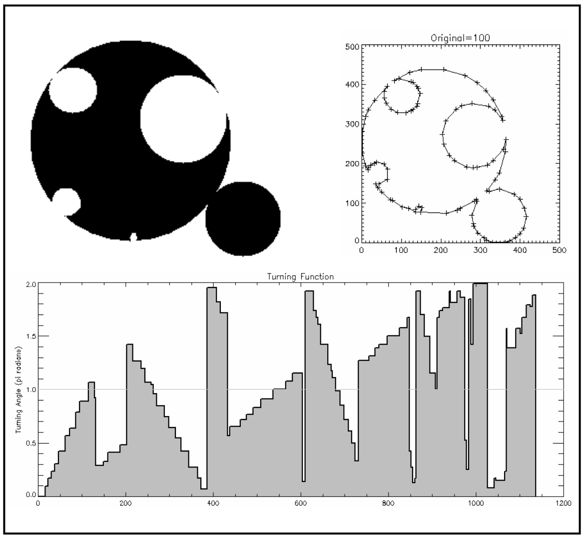

Figure 3. A binary image of an object, a vectorized contour and the turning function.

For a polygon with n sides, t is a position of one vector on the curve (t=1,2,…,n) and n is therefore the number of steps in the stair of the turning function, we first find the radius. Next we test if the hypothesis of vertex t belongs to the arc. If so, we store $t_{start}=t$, and $r_{arc}=r_t$. To find out if the next sequence is compatible and the last t is part of the same arc we need to use for each new vector the decomposition of s_H and s_V and find the resulting vector defined by s_{Equiv} and θ_{Equiv}. This vector is the segment between t_0 and t_f.

D. Circumference

The representation of a continuous circle in turning function space is a sloping straight line ranging from 0 to 2π and with the slope larger if the radius is smaller. Circles extracted from images have discrete number of vertices and therefore its turning function is a monotonically increasing stair-shaped function. We can reconstruct shapes in original position, scale and rotation. Turning functions convert circles to lines. Polygon exterior angle is the one formed by a side of a polygon and the extension of its adjacent side. In a turning function space it is represented as the discontinuity amount between two horizontal segments. This interruption sometimes is represented graphically as a vertical line. All points in a vertical alignment represent the same point.

A shape description to be used is a prior knowledge about the object category. The statements depict a class of objects and all possible variations, presenting fixed or non fixed and positive or negative allowances of: angles, curvature radius, sub shapes compounds, sizes, quantities, regularities, symmetry and so on.

E. Circle passing through three vertices

For any two adjacent line segments, $a_i=s_i-s_{i-1}$, and $b_i=s_{i+1}-s_i$, with turning angle $\Delta\theta_i$, the radius of curvature is (11). These segments can be obtained by substitution of other segments and refers to the i^{th} position vertex.

$$r_i(a_i, b_i, \Delta\theta_i) = \sqrt{\frac{a_i^2 + b_i^2 + 2a_i b_i \cos(\Delta\theta_i)}{2(1-\cos(2\Delta\theta_i))}} \quad (9)$$

Using exhaustive search to find all segments that matches similar values of radius and angle to the center of the inscribed circle with radius r_i. Testing one by one all the vectors are tested and, if we do not have an arc then we leave the segment as a polygonal line.

We can find the center position of the circle obtained from (9) by using (10), where $\Delta\beta_i$ is the noncumulative turning angle from a_i segment in the direction of the center of the circle. This is to be used if the process needs more accuracy.

$$\Delta\beta_i = \Delta\theta_i + \arccos\left(\frac{b_i}{2r_i}\right) \quad (10)$$

This leads us to circles and line segments and we need to make approximations to find a way to group the values, for example, with the distance to the mean of the value. With this and a model of constraints we can recognize all the shapes we need to classify, such as circles, parallelograms, triangles, complex shapes etc.

IV. CLASSIFICATION

Shape classification is the assignment of classes or categories to shapes according to their properties [6]. We are interested in the classification of shapes or regions of those shapes accordingly to predefined models. To do so we need to identify properties and patterns on the shapes and models. We need to know the turning function behavior and basically how to find lines and circles to be able to find both simple and complex shapes as the ones in Fig. 4.

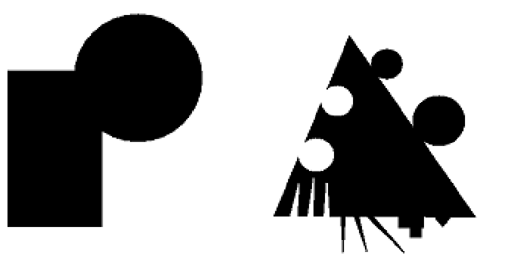

Figure 4. Example of two binary blobs.

The classification method being proposed needs some models and rules to define the expected features each shape's class. Some other alternatives could include the learning by examples and the definition of parameters for flexible template matching. To verify the feasibility of the method we used simple classes like circles and rectangles. Complex shapes can contain sub-shapes. We could find lines and arcs and classify them accordingly, whether the noise was not too large. It must be pointed that we do not want to classify the whole shape as belonging to one class, but also to classify the different subshapes which may represent extrusions, intrusions, shape overlaps, etc. This can be done with the methods presented in section III: for each known shape signature we can verify its occurrence on the main shape and try and determine the subshape parameters.

V. CONCLUSION

We presented a new approach to characterize shapes by the use of a signature-based 1D function. The main objective of this method is to show theoretically that it is feasible to work with turning functions and apply them to the classification of shapes and subshapes present in binary images. This method uses turning function space to find circles and lines being represented as piecewise aligned lines on a turning function transformed space. Turning functions are sensible to noise effect. This could be a negative aspect of this approach, but preliminary studies in this sense indicate that this same feature can help the identification of noises. We are still on the preliminary steps of the study of turning functions for shape and subshape classification, but the theoretical foundations and preliminary results are promising. We intend to keep researching this approach for application in model-based object recognition in images

REFERENCES

[1] Amit, Y., *2D object detection and recognition: models, algorithms, and networks*. 2002: The MIT Press.

[2] Volotão, C.F.S., Santos, R.D.C., and Dutra, L.V. *Proposta de segmentação de imagens baseada em objetos e uso de função de desvios para modelar formas*. in *XII Encontro de Modelagem Computacional*. 2009. Rio de Janeiro.

[3] Costa, L.F. and Cesar, R.M., *Shape Classification and Analysis: Theory and Practice*. 2nd. ed. 2009: CRC. 692.

[4] Winkler, G., *Image Analysis, Random Fields and Markov Chain Monte Carlo Methods: A Mathematical Introduction*. 2003: Springer Verlag.

[5] Gonzalez, R.C. and Woods, R.E., *Digital Image Processing*. 2007: Prentice Hall.

[6] Otterloo, P.J., *A contour-oriented approach to digital shape analysis*. 1988, Technische Universiteit Delft. p. 393.

[7] Rangayyan, R.M., et al., *Polygonal approximation of contours based on the turning angle function*. Journal of Electronic Imaging, 2008. **17**(2): p. 023016.

[8] Zhang, Y.J., *Image Engineering: Processing, Analysis, and Understanding*. 2009, Singapore: Tsinghua Univ P. 726.

[9] Ballard, D.H. and Brown, C.M., *Computer Vision*. 1982, New Jersey: Prentice-Hall. 543.

[10] Arkin, E.M., et al., *An efficiently computable metric for comparing polygonal shapes*. Pattern Analysis and Machine Intelligence, IEEE Transactions on, 1991. **13**(3): p. 209-216.

[11] Zhang, D. and Lu, G., *Review of shape representation and description techniques*. Pattern recognition, 2004. **37**(1): p. 1-19.

[12] Nixon, M.S. and Aguado, A.S., *Feature extraction and image processing*. 2008: Academic Press.

[13] Seul, M., O'Gorman, L., and Sammon, M.J., *Practical algorithms for image analysis: description, examples, and code*. 2000: Cambridge University Press.

[14] Les, M., *Shape Understanding System: The First Steps toward the Visual Thinking Machines*. 2008: Springer Verlag.