

Normal Vector and Winding Number in 2D Digital Images with Application for Hole Detection

Franck Xia

Department of Computer Science
 Saint John's University
 Collegeville, MN 56321
 E-mail: fxia@cs.csbsju.edu

I. Introduction

Digital topology has been extensively investigated in the past [1]. Fundamental topology concepts such as connectivity and component have been widely applied in pattern recognition. For instance, preserving the connectivity of objects is a key issue for thinning; in order to identify objects in images, we need to detect connected components. Theoretically, a connected component is entirely enclosed and determined by its external and hole contours. If we can detect all contours of a component and differentiate external with hole contours, the extraction of the component is straightforward. However, distinguishing external and internal contours is not a trivial task as it appears.

There exist several techniques for distinguishing external and internal contours. Suzuki and Abe developed an algorithm being able to differentiate components from holes relying on the notion of adjacency tree of contours: background surrounds components, components may include holes and holes may, in turn, surround objects, etc. [4]. The method seems inefficient for component analysis because, even to extract a small component, all the contours surrounding it must be found first. Qian and Bhattacharya observed recently that hole contours behave differently than external contours under certain morphological operations. Relying on their empirical observation, they advanced an algorithm for hole detection using dilation [2]. However, morphological operations such as dilation could produce undesirable effects, for the topological property of images could be changed. Van Vliet and Verbeek proposed to separate external contours and holes using Euler number. The rationale is that Euler number can be calculated on an external or internal contour and the resulting value should be 2π for external contours and -2π for internal ones [3]. This method can certainly work when external contours are geometrically separated from internal contours but would generate erroneous results when internal contours intersect with external ones. Recently, Lee, Poston and Rosenfeld proposed a method to distinguish holes from components based on Gauss Mapping and winding number [5]. For any region, if we follow its boundaries under left-hand convention, the normal vector on any boundary will transit any given direction, say D in Fig. 1. A transition of normal vector has a sign, positive if the transition is counter-clockwise and negative clockwise. External and internal boundaries can then be differentiated by the sum of signed transits of normal vector on a boundary: 1 for external and -1 for internal boundary.

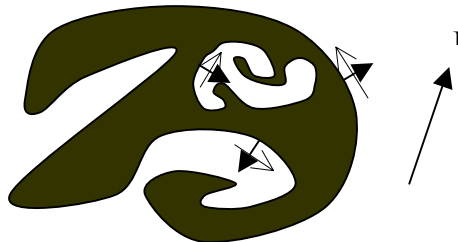


Figure 1: Movement of normal vector on external and hole boundaries

In contrast with others, Lee et al.'s method can handle all kinds of images and discriminate holes from components in both 2 and 3-D images directly based on a theoretic property of contours. In order to avoid the discontinuity of digital images, Lee et al. introduced a smoothing function to

replace digital lattice and calculated normal vector on rounded boundary. From a computational viewpoint, however, it is time consuming to calculate normal vector on rounded boundary with a smoothing function rather than on digital images.

In this paper, we show that the idea of Gauss mapping and winding number can be implemented in 2D images, directly. It is true that the discontinuity of Freeman codes of edge may cause problems for computing normal vector, and the discontinuity of normal vectors may render the detection of transition of normal vector erroneous too. However, we find that there are two types of discontinuity for Freeman codes and normal vectors, one reflecting the intrinsic discontinuity of digital images and another due to the limited range of Freeman codes. We will prove that the two types of discontinuity have distinct properties. While the first type of discontinuity is intrinsic in images, only the second one will cause errors when calculating normal vector or counting normal vector transition. We show that the errors can be avoided because we know how to distinguish the two types of discontinuity based on their properties. Following this idea, we propose an algorithm of hole detection by counting the transitions of normal vector on contours without introducing any smoothing operation. We show further that for 2-D images it is simpler to calculate the winding number of contours than to count the number of transits of normal vector, and a second algorithm is proposed. We will present the result of both our algorithms to illustrate their simplicity.

In the following, we will use 8-connectivity for objects and 4-connectivity for background. Image contours will be traced under left-hand convention. When following contours, edges are coded with Freeman codes. We denote P_k ($k \geq 0$) the k^{th} pixel traced during contour following and $E(k)$ the Freeman code of edge $P_k P_{k-1}$. For our theoretic results presented below, a rigorous proof will be provided in our full-length paper.

II. Discontinuity Property of Freeman Code

In order to demonstrate the properties of Freeman codes of edge and normal vector, we introduce a new coding scheme called continuous codes just for the purpose of our theoretic analysis, but not for the implementation of the two algorithms we are going to develop. Let $E'(k)$ be the continuous code of $P_k P_{k-1}$.

Definition 1: When tracing a contour of n pixels, set $E'(1) = E(1)$ for $P_1 P_0$. For $P_{k+1} P_k$ with $1 \leq k < n$, if $P_{k+1} P_k$ returns left $i\pi/4$ with respect to the direction of $P_k P_{k-1}$, $E'(k) = E'(k-1) + i$; if $P_{k+1} P_k$ turns right $i\pi/4$, $E'(k) = E'(k-1) - i$.

The key point in the definition is to maintain the continuity of edge codes in Z . A direct consequence is that there is no upper bound value for continuous codes. From the definition, it is easy to see that $E(k) = \text{Modulo}(E'(k), 8)$. In the following we will focus on the property of $E'(k+1) - E'(k)$ (denoted by $\Delta E'(k)$) and $E(k+1) - E(k)$ (denoted by $\Delta E(k)$).

Definition 2: The Freeman chain codes of the two adjacent edges at P_k have a discontinuity of first category when $E'(k-1) = E(k-1) + 8m$ $E'(k) = E(k) + 8m$.

Let D_1^e denote the set of contour pixels at which the Freeman codes of the two adjacent edges have a discontinuity of first category. Clearly, $P_k \in D_1^e$ can be defined alternatively as $\Delta E(k) = \Delta E'(k)$. We notice that not all contour pixels can satisfy $\Delta E(k) = \Delta E'(k)$. Consider the case when both Freeman code $E(k-1)$ and continuous code $E'(k-1)$ of edge $P_k P_{k-1}$ equal to 7. If $P_{k+1} P_k$ turns left $\pi/2$ with respect to the direction of $P_k P_{k-1}$, $E'(k) = 9$ by definition. With Freeman codes ranging from 0 to 7, $E(k) = 1$. So $\Delta E(k) \neq \Delta E'(k)$. This motivates the following definition.

Definition 3: The Freeman chain codes of the two adjacent edges at P_k have a discontinuity of second category if $E'(k-1) = E(k-1) + 8m$ $E'(k) \neq E(k) + 8m$.

Let D_2^e denote the set of contour pixels at which the Freeman chain codes of the two adjacent edges have a discontinuity of second category.

Proposition 1: $\forall P_k, -2 \leq \Delta E'(k) \leq 4.$

Proposition 2: $\forall P_k \in D_2^e, E'(k) = E(k) + 8m \quad E'(k+1) = E(k+1) + 8(m \pm 1), m \in Z.$

Theorem 1: $\forall P_k \in D_1^e, -2 \leq \Delta E(k) \leq 4.$

Theorem 2: $\forall P_k \in D_2^e, \Delta E(k) \geq 6$ or $\Delta E(k) \leq -4.$

Theorem 1 and 2 show that the ranges of $\Delta E(k)$ for pixels of the two types of discontinuity D_1^e and D_2^e have no intersection, we can therefore distinguish contour pixels of D_1^e and D_2^e directly by their $\Delta E(k)$. This is the key for calculating correctly normal vector in digital images.

III. Normal Vector in Z^2 and its Property

Definition 4: Given three consecutive contour pixels $P_{k-1}, P_k,$ and P_{k+1} . The normal vector at P_k is the segment P_kN , N being a neighbor of P_k and the direction of P_kN is determined by $\text{Modulo}([E'(k+1)+E'(k)] \text{ div } 2 - 2, 8), \text{div}$ standing for integer division.

Fig. 2 illustrates some normal vectors. In definition 4, $[E'(k+1)+E'(k)] \text{ div } 2$ represents an approximation of the tangent on P_k . The normal vector must be perpendicular to the tangent and point to the background of image. As contours are traced under left-hand convention, we subtract -2 from the tangent value which is equivalent of turning $\pi/2$ from the tangent toward the background. Therefore N is the closest pixel or one of the closest two in the background to the bisector of $\angle P_{k-1}P_kP_{k+1}$ in the 8-neighbors of P_k and P_kN indicates the direction of normal vector at P_k . Note that, in the definition, we use continuous codes rather than Freeman codes because the latter may have D_2^e discontinuity and could cause error. The Modulo function ensures that the value of normal vectors ranges between 0 and 7.

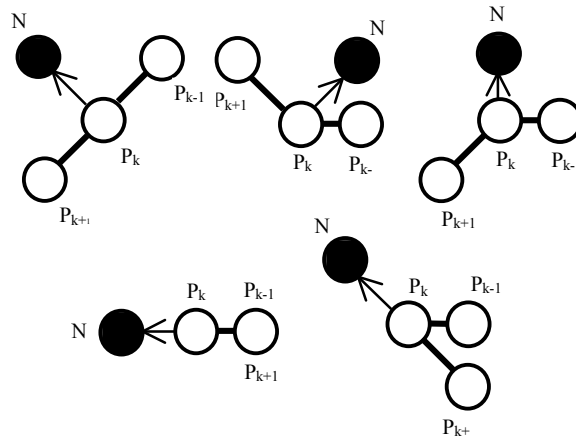


Figure 2: normal vectors in digital image

For convenience, we shall denote $\text{Modulo}([E'(k+1)+E'(k)] \text{ div } 2 - 2, 8)$ by $N(k)$ and $N(k+1) - N(k)$ by $\Delta N(k)$. To facilitate our analysis, we will use again continuous codes for normal vector $N'(k) = [E'(k+1)+E'(k)] \text{ div } 2 - 2$ and denote $N'(k+1) - N'(k)$ by $\Delta N'(k)$. The following shows

that edge codes and normal vectors have very similar properties of discontinuity.

Definition 5: The two normal vectors on $P_{k+1}P_k$ have a discontinuity of first category when $N'(k) = N(k) + 8m \quad N'(k+1) = N(k+1) + 8m.$

Definition 6: The two normal vectors on $P_{k+1}P_k$ have a discontinuity of the second category when $N'(k) = N(k) + 8m \quad N'(k+1) \neq N(k+1) + 8m$.

Let D_1^n denote the set of edges on which the two normal vectors have a discontinuity of first category and D_2^n the set of edges on which the two normal vectors have a discontinuity of second category.

Proposition 3: $\forall P_k, -2 \leq \Delta N'(k) \leq 4$.

Proposition 4: $\forall P_{k+1}P_k \in D_2^n, N'(k) = N(k) + 8m \quad N'(k+1) = N(k+1) + 8(m \pm 1), m \in \mathbb{Z}$.

Theorem 3: $\forall P_{k+1}P_k \in D_1^n, -2 \leq \Delta N(k) \leq 4$.

Theorem 4: $\forall P_{k+1}P_k \in D_2^n, \Delta N(k) \geq 6$ or $\Delta N(k) \leq -4$.

Theorem 3 and 4 show that the ranges of $\Delta N(k)$ for edges of the two types of discontinuity D_1^n and D_2^n have no intersection by theorem 3 and 4. Therefore we can distinguish edges of D_1^n and D_2^n directly by their $\Delta N(k)$ and overcome the discontinuity problem of normal vectors.

IV. Transition of Normal Vector

The meaning of the transition of normal vector is clear on a regular curve $C(t)$ with $\frac{d^2C(t)}{dt^2} \neq 0$ everywhere. Given a direction and in the neighborhood of a boundary point, the transit is negative if the normal vector passes the direction clockwise, and positive if counter-clockwise. This can be simulated in digital plane easily. However, when using Freeman codes and normal vectors which could have discontinuity of second category, a simple digital simulation may not work. For example, two consecutive normal vectors with value 1 and 7 do not represent a positive transit in direction 4, but a negative one in direction 0. Our solution is to refer to continuous codes for normal vectors, since they cannot have such kind of discontinuity.

But even with normal vectors of first category discontinuity, counting the transition of normal vector is still not that evident. Consider a piecewise regular curve containing a segment $C(t_1)C(t_2)$ with $\frac{d^2C(t)}{dt^2} = 0 \quad \forall t \in (t_1, t_2)$ (see Fig. 3). If the normal vector on this segment is parallel to given direction D , how should we count the transit at $C(t_1)$ and $C(t_2)$? Based on either $C(t_1)$ or $C(t_2)$, we cannot make such a decision. However, by checking both $C(t_1)$ and $C(t_2)$, we find there is one positive transition on $C(t_1) C(t_2)$ in Fig. 3.b but none in Fig. 3.a, as the normal vector joins direction D from one side and rotates back to the same side when passing through $C(t_1)C(t_2)$.

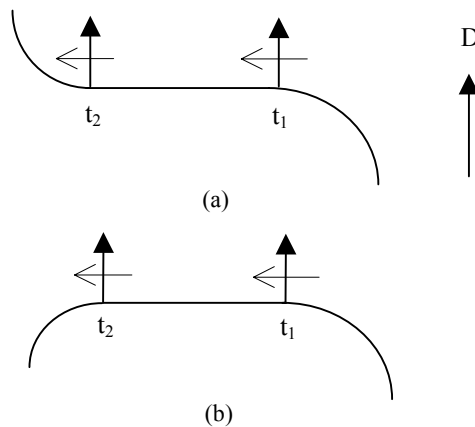


Figure 3: Transition of normal vector in piece-wise regular curves.

If such a case occurs in an image, we need to check all pixels between $C(t_1)$ and $C(t_2)$ in order to determine whether there is a transition at $C(t_1)$. But this is inconvenient from a computational viewpoint, for we may never know how many contour pixels there are between $C(t_1)$ and $C(t_2)$. The solution we propose is to consider no more than three consecutive pixels. Let $TNV(k)$ be the transition of normal vector at contour pixel P_k during contour tracing.

Definition 7: Given direction d with $0 \leq d \leq 7$ and for any integer m , $TNV(k) = 1$ if **P.1**) $N'(k-1) < d + 8m$ and $N'(k) > d + 8m$, or **P.2**) $N'(k-1) < d + 8m$, $N'(k) = d + 8m$, and $N'(k+1) \geq d + 8m$; $TNV(k) = -1$ if **N.1**) $N'(k-1) > d + 8m$ and $N'(k) < d + 8m$, or **N.2**) $N'(k-1) \geq d + 8m$, $N'(k) = d + 8m$, and $N'(k+1) < d + 8m$; otherwise $TNV(k) = 0$.

Proposition 5: Definition 7 considers all possible normal vector transitions.

Theorem 5: $\prod_{i=1}^n TNV(i) = 1$ when contour is external and -1 when internal.

Proposition 5 ensures that no potential transit could be missed by definition 7. As the definition takes only two or three consecutive normal vectors into consideration whereas between $C(t_1)$ and $C(t_2)$ in Fig. 3 there could be many pixels, we want to make sure that definition 7 will not generate any false transit. This is guaranteed by theorem 5.

V. Hole Detection Algorithm Using TNV

When tracing borders, edges are coded with Freeman codes and normal vectors are computed. Although normal vector at P_k is defined based on continuous code of edges $E'(k+1)$ and $E'(k)$, it can be calculated from Freeman codes $E(k+1)$ and $E(k)$ after a local transform. This avoids the potential problem of unlimited range of continuous codes. The computation of normal vector is described by the procedure below:

Procedure Normal_Vector()

Input: $E(k)$ and $E(k+1)$

Output: $N(k)$

If $E(k+1) - E(k) \leq -4$ Then

$E(k+1) := E(k+1) + 8$;

Else if $E(k+1) - E(k) \geq 6$ Then

$E(k) := E(k) + 8$;

Endif

$N(k) := \text{Modulo}((E(k+1) + E(k)) \text{ div } 2 - 2, 8)$;

End Normal_Vector

Proposition 6: Normal_Vector() returns $N(k)$.

The following procedure detects the transition of normal vectors at contour pixel P_k in given direction D through the Freeman code of two or three consecutive normal vectors. Again, it is interesting to note that although normal vector transition is defined on the continuous codes of normal vectors, it is just for our theoretic analysis, and we do not need to generate continuous codes for normal vectors. We only need to transform locally three consecutive normal vectors to ensure that they are continuous.

Procedure Normal_Vector_Transition()

Input: $N(k-1)$, $N(k)$, and $N(k+1)$

Output: TNV

$M := 0;$

If $N(k) - N(k-1) \leq -4$ Then

 If .not. $(N(k+1) - N(k) \geq 6)$ Then

$N(k+1) := N(k+1) + 8;$

 Endif

$N(k) := N(k) + 8; M := 1;$

Else if $N(k) - N(k-1) \geq 6$ Then

 If .not. $(N(k+1) - N(k) \leq -4)$ Then

$N(k+1) := N(k+1) - 8;$

 Endif

$N(k) := N(k) - 8; M := -1;$

Else

 If $N(k+1) - N(k) \leq -4$ Then

$N(k+1) := N(k+1) + 8; M := 1;$

 Else if $N(k+1) - N(k) \geq 6$ Then

$N(k) := N(k) + 8; M := 1;$

 Endif

Endif

TNV := 0;

For $s := 0$ to M do

 If $N(k) = D + 8s$ Then

 If $N(k-1) < D + 8s$.and. $N(k+1) \geq D + 8s$ Then

 TNV := 1;

 Else if $N(k-1) \geq D + 8s$.and. $N(k+1) < D + 8s$ Then

 TNV := -1;

 Endif

 Else if $N(k-1) < D + 8s$.and. $N(k) > D + 8s$ Then

 TNV := 1;

 Else if $N(k-1) > D + 8s$.and. $N(k) < D + 8s$ Then

 TNV := -1;

 Endif

Endfor

End Normal_Vector_Transition

Proposition 7: Normal_Vector_Transition returns TNV(k) in direction D.

When all the contours are traced and a list of normal vectors created for each contour, our algorithm of hole detection scans then the list to detect and accumulate all the transitions of normal vector in a given direction. By theorem 5, the sum of transitions of normal vector in a given direction must equal to 1 or -1, depending on whether the border is external or internal. Fig. 4, illustrates a result obtained by our algorithm implemented in C on a SUN Station. The numeric values on contour pixels represent the Freeman codes of normal vectors. A pixel with letter P or N stands for a positive or negative transition of normal vector in given direction 4.

Contour No.1: Normal Vector Pass = 1
 Contour No.2: Normal Vector Pass = -1

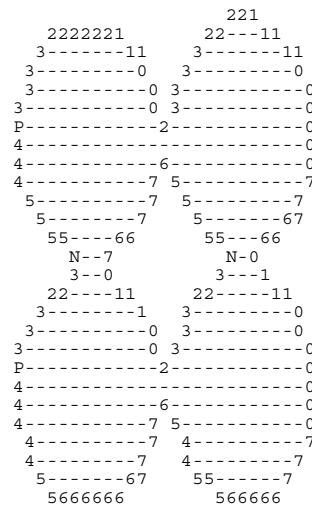


Figure 4: Results of hole detection via counting TNV

VI. Winding Number and Hole Detection

The method we presented in section IV and V is not really based upon but derived from winding number, as it integrates only the transitions of normal vector (first order derivative) in a given direction whereas winding number is the integral of normal vector differentials (second order derivative). But this theoretical difference is not our motivation to develop another algorithm based on winding number. The main reason is that our second algorithm has a better running time complexity and is easier to implement comparing to our first algorithm.

Definition 8: $\sum_{i=1}^n \Delta N'(i)$ is the winding number of C.

Theorem 6: $\sum_{i=1}^n \Delta N'(i) = 8$ when C is external and -8 when C is internal.

The algorithm accumulates the differences of two successive normal vectors in continuous codes along a contour. When a contour has been traversed, the sum of accumulated normal vector differentials must equal either to 8 or -8, depending on if it is an external or internal contour. Although winding number is defined on continuous codes of normal vectors, we do not need to use continuous codes for normal vectors in our program.

Procedure Delta_Normal_Vector()

Input: N(k) and N(k+1)

Output: $\Delta N(k)$

If $N(k+1) - N(k) \leq -4$ Then

$N(k+1) := N(k) + 8;$

Else if $N(k+1) - N(k) \geq 6$ Then

$N(k) := N(k) + 8;$

Endif

$\Delta N(k) := N(k+1) - N(k);$

End Delta_Normal_Vector

Proposition 8: Delat_Normal_Vector() returns $\Delta N^2(k)$.

Comparing with procedure Normal_Vector_Transition, Delat_Normal_Vector() is significantly simpler in terms of running time and code complexity. In Fig. 5, differentials of normal vector are represented on contours with numerals and letters. Note that letter a and b stand for -1 and -2.

Contour No.1: winding number = 8
 Contour No.2: winding number = -8

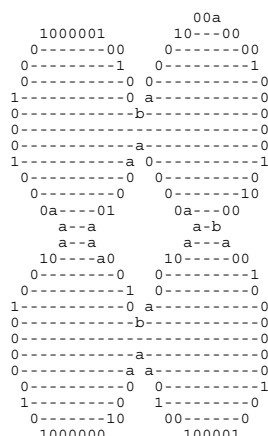


Figure 5: Result of hole detection via winding number

VII. Conclusion

Component is a fundamental concept in image analysis and pattern recognition. In order to analyze a component, we need to distinguish external and internal contours surrounding the component. In this paper we show that normal vector and winding number can be defined and implemented directly in 2D digital images for distinguishing external from hole contours. Our analysis demonstrates that Freeman code of edges and normal vectors in digital plane have two types of discontinuity, with distinct properties. Based on these properties, the discontinuity difficulty of Freeman codes and normal vectors can be overcome readily for calculating either the transitions of normal vector or winding number. We have presented two algorithms for hole detection in Z^2 which do not need any kinds of smoothing function suggested in [5]. We show that winding number is much more easier to implement, comparing to the method counting the sum of transitions of normal vector in a given direction.

References

[1] Kong T.Y. and Rosenfeld A., Digital Topology: Introduction and Survey, CVGIP, Vol. 48, 1989, pp. 357-393
 [2] Qian K. and Bhattacharya P., Determinin holes and connectivity in binay images, Comput. & Graphics, vol. 16, no. 3, 1992, 283-288
 [3] Van Vliet L.J. and Verbeek P.W., Better geometric measurement based on photometric information, Proc. 10th IMTC, Hamamatsu, Japan, 1994, 1357-1360
 [4] Suzuki S. and Abe K., Topological structural analysis of digital binary images by border following, CVGIP, vol. 30, 1985, 32-46
 [5] Lee C.N., Poston T. and Rosenfeld A., Winding and Euler numbers for 2D and 3D digital images, CVGIP: Graphical Model and Image Processing, vol. 55, no. 1, 1993, 20-47