3D-Curve Representation by Means of a Binary Chain Code

E. Bribiesca
Department of Electrical Engineering
Universidad Autónoma Metropolitana
Unidad Iztapalapa, México, D.F.
ernesto@leibniz.iimas.unam.mx

(Received November 2002; accepted January 2004)

Abstract—A binary chain code for representing 3D curves is presented. 3D curves are digitalized and represented as 3D discrete curves. 3D discrete curves are composed of constant straight-line segments using only orthogonal directions. The proposed binary notation is based on the orthogonal direction change chain code. Thus, using the proposed binary chain code it is possible to represent any 3D curve by only two states ("0" and "1"). The proposed binary chain code produces a curve descriptor which is invariant under translation, rotation and may be starting point normalized. Also, we define the complementary chain of a 3D curve which produces the mirror image of the curve.

Keywords—3D curves, Chain coding, Binary chain code, Complementary chains, 3D discrete curves.

1. INTRODUCTION

The 3D-curve representation is an important topic in computer vision and pattern recognition. This work deals with 3D-curve representation by means of a binary chain code. The first approach for representing 3D curves using chain coding was introduced by Freeman [1] in 1974. The Freeman's chain code depends on rotation, this is due to the fact that only absolute directions are used. The notation proposed in this work uses only relative directions which produces a notation invariant under rotation. A canonical shape description for 3D stick bodies was presented by Guzmán [2]. Chain-code techniques are widely used because they preserve information and allow considerable data reduction. Many shape features may be computed directly from chain-code notation. Chain codes are the standard input format for numerous shape analysis algorithms. For instance, the 2D contour following algorithms may be represented by means a binary chain code.

The simplest contour following algorithms were presented by Papert [3] (Papert's turtle) and Duda et al. [4]. Rosenfeld [5] presented a slightly more elaborate algorithm of contour following.
which generates the boundary pixels exactly. Ledley [6] analyzed digitalized pictures of chromosomes by means of contour following. Thus, using the algorithm proposed by Duda et al. [4], it is possible to represent 2D shape contour by only two states: right turn (represented by “0”) and left turn (represented by “1”). Figure 1a illustrates an example of a flat region and its contour. Figure 1b shows the contour following algorithm on the region presented in (a). This contour was obtained using the algorithm proposed by Duda et al. [4] whom state it as follows:

"Scan the picture until a figure cell is encountered.
Then:
If you are in a figure cell turn left and take a step.
If you are in a ground cell turn right and take a step.
Terminate when you are within one cell of the starting point."

Figure 1c shows the discrete contour and its elements. Finally, Figure 1d illustrates the binary chain of the discrete contour.

The above-mentioned algorithm produces a chain composed of only binary elements. Now, the question is as follows: is it possible to represent 3D curves using only two states (“0” and “1”)? Yes, it is possible to represent 3D curves using only two states. In the content of this work, we present a method which allows us to represent 3D curves by means of binary chains. This method is based on the concepts of the orthogonal direction change chain code [7]. In this chain code, there are only five possible chain elements for representing any 3D curve. In order to have binary chains for representing 3D curves, we have only selected two previous defined chain elements of the orthogonal direction change chain code, and we have called the binary chain code.
2. THE BINARY CHAIN CODE

Any 3D continuous curve can be digitalized and represented as a 3D discrete curve. This discrete curve is composed of constant straight-line segments using only orthogonal directions. Two contiguous straight-line segments of a 3D discrete curve define a direction change and two direction changes define a chain element. In the content of this work, the length $l$ of each straight-line segment is considered equal to one.

**DEFINITION 1.** A chain $A$ is an ordered sequence of elements, and is represented by

$$A = a_1a_2\ldots a_n = \{a_i : 1 \leq i \leq n\},$$

where $n$ indicates the number of chain elements.

![Figure 2](image)

Figure 2. The binary chain code: (a) an example of a curve; (b) the digitalized version of the curve shown in (a); (c) the binary chain elements; (d) the first chain element; (e) the second chain element; (f) the last chain element; (g) the binary chain.

Figure 2a illustrates an example of an open continuous curve. In the content of this work, curves are represented as ropes. This improves the understanding of the figures. Figure 2b shows the digitalized version of the curve presented in (a). Note that this discrete curve is composed of constant straight-line segments using only orthogonal directions.

**DEFINITION 2.** An element $a_i$ of a chain, taken from the set $\{0, 1\}$, is obtained by calculating the relative orthogonal direction changes along the curve.

Each element of the chain, labels a vertex of the 3D discrete curve and indicates the orthogonal direction changes of the polygonal path in such a vertex. There are only two possible orthogonal direction changes for representing any curve. Figure 2c illustrates these orthogonal direction changes:

1. the element "0" indicates a direction change to the right,
2. the element "1" indicates a direction change to the left.

Formally [8], if the consecutive sides of the reference angle have respective directions $b$ and $c$ (see Figure 2c), and the side from the vertex to be labeled has direction $d$ (from here on, by
direction, we understand a vector of length 1), then the chain element is given by the following function,

\[
\text{chain element } (b, c, d) = \begin{cases} 
0, & \text{if } d = b \times c, \\
1, & \text{if } d = -(b \times c),
\end{cases}
\]

where \( \times \) denotes the cross product.

Figure 2d shows the first element of the chain which corresponds to the element "1". Note that the first direction change (which is composed of two contiguous straight-line segments) is used only for reference. The origin of the above-mentioned curve is considered at the lower side. Figure 2e illustrates the next element obtained of the chain, which is based on the last direction change of the first element; the second element corresponds to the element "1", too. Figure 2f presents the last element of the curve, which corresponds to the element "0". Finally, Figure 2g illustrates the binary chain "110" of the above-mentioned curve.

2.1. The Method of Digitalization

We propose an algorithm to digitalize 3D curves using only two states ("0" and "1"). In order to explain the proposed algorithm, we have generated an easy example. Figure 3a shows an example of a 3D curve. Figure 3b presents the superimposition of the 3D curve on a part of a
3D grid. All 3D discrete curves considered in this work are inside of a 3D grid or a 3D array of voxels. Thus, the origin of the 3D array of voxels is defined by the centroid of one voxel which corresponds to the previous defined beginning of the curve, this is shown in Figure 3b. The proposed algorithm for digitalizing 3D curves is as follows.

1. Select a vertex of the starting voxel as the origin (see Figure 3c) of the discrete curve.
2. In order to construct the discrete curve, select the appropriate binary element which is closer to the 3D curve (this is calculated using the Euclidean distance between the computed elements of the discrete curve and the 3D curve). In this case, the first element corresponds to the binary element “0”. The second element which is closer to the 3D curve corresponds to the binary element “1”, and so on. Figure 3c shows the 3D curve and the construction of the discrete curve.
3. In some cases, when you are selecting the binary elements. It is possible that both elements (“0” and “1”) have the same Euclidean distance to the 3D curve. In these cases, we recommend that the binary element “0” be used.
4. Terminate, when you have been computed all distances. Figure 3d illustrates the discrete curve and its binary elements. Finally, Figure 3e presents the binary chain of the discrete curve.

When we are generating the discrete curve, this algorithm does not prevent us to have loops. In these cases, additional criteria must be included. Also, in order to have the same starting point of the discrete curve invariant under rotation, we have to include more restrictions.

This algorithm does not detect curves that are sharp compared to the length of the straight-line segment. Due to the fact that only binary elements are used, this notation does not have sharp returns and this may produce, as a result, that the discrete curve moves further away from the 3D curve. In order to improve the above-mentioned cases, we recommend that segment length be decreased. The length of the straight-line segment defines the digitalization resolution. If resolution is high, the number of chain elements is increased linearly depending on the characteristics of the curve.

2.2. The Inverse of a Chain

**Definition 3.** The inverse of a chain of a 3D discrete curve is another chain formed of the elements of the first chain arranged in reverse order, i.e., the chain obtained by traveling the curve in one direction is just the reverse of the chain obtained traveling the same curve in the opposite direction.

![Figure 4. The inverse of a chain: (a) the 3D discrete curve, its elements and its chain; (b) the inverse of the chain of the above-mentioned curve.](image-url)
Figure 4 illustrates an example of the inverse of a chain. In order to present the example of the inverse of a chain, Figure 4a shows the discrete curve which was presented in Figure 3d again. Finally, Figure 4b shows the inverse of the chain presented in (a), its elements and its chain. Notice that the chain elements of the inverse of the chain shown in Figure 4b are arranged in reverse order.

2.3. Starting-Point Normalization for Open Curves

Using the definition of the inverse of the chain, this binary chain code may be starting-point normalized by choosing the starting point so that the resulting sequence of elements forms an integer of minimum magnitude. For example, the chain of the open curve illustrated in Figure 4a represents an integer number; and the chain shown in (b) represents another integer number, respectively. Thus, the integer of minimum magnitude corresponds to the chain illustrated in Figure 4a. Therefore, this chain is starting-point normalized.

2.4. Starting-Point Normalization for Closed Curves

Also, closed curves may be starting-point normalized, by choosing the starting point so that the resulting sequence of chain elements forms an integer of minimum magnitude. Figure 5 illustrates an example of a closed curve. Figure 5a shows the binary chain elements, again. Figure 5b presents an example of a closed curve. Figure 5c shows the digitalized version of the curve presented in (b) and its elements (the origin of this 3D discrete curve is represented by a sphere). Finally, Figure 5d illustrates the binary chain of the above-mentioned curve. Thus, the chain of the 3D curve presented in Figure 5 may be invariant under starting point by rotating the digits until the number is minimum. In special cases, there may exist two or more minimum numbers, in these cases, additional criteria must be included. The chain of the 3D curve shown in Figure 5c is already invariant under starting point.

2.5. Invariance Under Rotation

In shape recognition is very important to have a notation invariant under rotation. The rotation transformation is convenient in studying curves and surfaces, since it does not alter them intrinsically (preserves shape and size). This binary notation is invariant under rotation, this is due to the fact that only relative direction changes are used. Figure 6 illustrates the independence of rotation. Figure 6a shows a rotation of the curve presented in Figure 5b, its
3D-Curve Representation

2.6. Complementary Chains

DEFINITION 4. The complement of a chain of a 3D discrete curve is another chain (termed complementary chain) whose elements "0" and "1" are replaced by "1" and "0", respectively.

The complementary chain of a 3D discrete curve corresponds to the mirror image of the curve. Figure 7 illustrates examples of complementary chains. In Figure 7a, the mirroring plane is aligned with the standard plane "XY", notice that the chain elements "0" and "1" of the com-
Figure 7. Examples of complementary chains: (a) the mirroring plane is aligned with the standard plane "XY"; (b) with the plane "XZ"; and (c) with the plane "YZ", respectively. Notice that the definition of complementary chains is valid for all possible orthogonal mirroring planes. We do not prove this, only illustrate it.

3. RESULTS

To illustrate the capabilities of the binary chain code proposed here, we present some examples of classification of simple closed curves. The study of simple closed 3D curves is of importance in computer vision [9]. In a simple closed curve, the point of the beginning of the curve is equal to the point of the end. Also, a simple closed curve has no inner crossings. It is clear that the same closed curve gives rise to several chains. But, given \( n \), the chain of order \( n \) of that curve is unique. So, the order of a chain is the number of digits that the chain contains [10]. It is always even, because the curve is closed.

By evaluating all possible combinations of binary chain elements of curves, selecting those whose initial and final points are equal, then eliminating those with inner crossings, one obtains the desired family of closed curves. Independence of starting point for discrete curves is considered. Also, mirroring curves are considered because they have different chains. Figure 8 illustrates
different families of all mirror-symmetric closed curves, their continuous representations and their chains at orders 6, 12, and 18, respectively. In order to obtain these results, we translate chains into strings and use parsers for curve classification. Thus, parsing the language \(\{\alpha \alpha' : \alpha \in \Sigma^*\}\) detects all mirror-symmetric curves in a given family, where \(\Sigma = \{0, 1\}\) and \(\alpha'\) is the complement of \(\alpha\). Figure 8a illustrates the family of all mirror-symmetric closed curves of order 6. Notice that, in this family, there is only one mirror-symmetric closed curve. Figure 8b shows the family of all mirror-symmetric closed curves of order 12. Finally, Figures 8c–8e present the family of all mirror-symmetric closed curves of order 18. Note that, in this case, there are only three mirror-symmetric closed curves.

![Figure 8](image)

We consider the problem of eliminating redundancy and slight differences of the shape-of-the-curve (due perhaps to noise in the digitalization of real-world curves). Thus, it is possible to apply transformations to chain elements using grammatical techniques.

3.1. Conversion from the 3D Binary Chain Code to a 2D Chain Code

The conversion from 3D to 2D chain codes is an important topic in curve representation. In this section of results, we present an example of the orthogonal projection of a 3D discrete curve onto the three orthogonal planes, this produces three plane curves. Figure 9a illustrates an example of a 3D continuous open curve. Figure 9b shows the discrete version of the curve shown in (a), its elements and its chain. Figure 9c illustrates the three orthogonal planes: "ZX", "YZ", and "XY" represented by "E", "F", and "G", respectively.

Any direction change which is composed of two contiguous straight-line segments defines a plane. Thus, any sequence of chain elements of a discrete curve generates an ordered sequence of planes of the curve. In a discrete close curve, the plane of the beginning of the curve corresponds to the plane of the end of the curve. Figure 9d illustrates the discrete curve and its
corresponding planes. Notice that the ordered sequence of planes is independent of the selected binary chain element. In this manner, we always know what element impacts on what plane. Finally, Figure 9e shows the ordered sequence of planes and the chain of the above-mentioned curve. Figure 10 presents an example of a 3D discrete curve and its projections onto the planes. This view is presented in perspective. The binary chain of the curve shown in Figure 10 is “0111100000011111111”. The projections onto the planes represent three plane curves which may be describes using a 2D chain code [1]. In order to improve the encoding efficiency for representing planar curves, we recommend that the orthogonal direction change chain code [7] be used.

4. CONCLUSIONS

We have presented a method for representing 3D curves by only two states: right turn (represented by “0”) and left turn (represented by “1”). Thus, this method produces a chain composed of only binary elements. This binary chain produces a 3D-curve descriptor, which is invariant under translation, rotation, and may be starting-point normalized. Also, the mirroring transformation may be obtained with ease by means of the use of the concept of complementary chains.
REFERENCES