J. Vis. Commun. Image R. 29 (2015) 8–15

Contents lists available at ScienceDirect

J. Vis. Commun. Image R.

journal homepage: www.elsevier.com/locate/jvci

A universal chain code compression method ☆

Borut Žalik *, Domen Mongus, Niko Lukač

University of Maribor, Faculty of Electrical Engineering and Computer Science, Smetanova 17, SI-2000 Maribor, Slovenia

ARTICLE INFO

Article history:
Received 12 September 2014
Accepted 18 January 2015
Available online 4 February 2015

Keywords:
Image processing
2D shapes
Contours
Chain codes
Compression
Run length encoding
Dictionary coding
Data modelling

ABSTRACT

This paper introduces a new approach for lossless chain code compression. Firstly, the chain codes are converted into the binary stream, independent on the input chain code. Then, the compression is done using three modes: RLE^0, $LZ77^0$ and *COPY*. RLE^0 compresses the runs of the 0-bits, $LZ77^0$ is a simplified version of *LZ77* and handles the repetitions within the bit stream, whilst *COPY* is an escape mode used, when the other two methods are unsuccessful. This method has been tested on the Freeman chain code in eight and four directions, the Vertex chain code, the Three OrThogonal chain code, and the Normalized angle difference chain code. The experiments confirmed better compression ratios on various benchmark datasets in comparison to the state-of-the-art lossless chain code compression methods.

1. Introduction

Chain codes are popular representation methods for rasterised shapes within various scientific and engineering disciplines [1–9]. The chain code can be regarded as a sequence of commands, which control the movement of a virtual walker throughout all the boundary pixels of a shape. The most intuitive chain code was proposed by Freeman back in 1961 [10] using pixels' 8-connectivity (Fig. 1a) known as *Freeman 8-directional chain code* (i.e. *F8*). The movement through the boundary pixels is encoded with an alphabet $\Sigma(F8) = \{0,1,2,3,4,5,6,7\}$, where each element $\sigma \in \Sigma$ represents a $45^\circ \times \sigma$ angle from the positive direction of the x coordinate axis. Freeman also determined that the boundaries of digitalised shapes can be described by the pixels' 4-connectivity (*Freeman 4-directional chain code* (F4)) by encoding $90^\circ \times \sigma$ angle using the shorter alphabet $\Sigma(F4) = \{0,1,2,3\}$ (see Fig. 1b). Nunes et al. [11] encoded the boundary edges of the shape instead of the pixels and therefore the walking direction should be specified. They proposed *differential chain code* (*DCC*), the first chain code with only three elements. Namely, the *DCC* alphabet $\Sigma(DCC) = \{R, L, S\}$, where R stands for right, L for left, and S for straight. In 1999 Bribiesca discovered a new, nowadays very popular chain code, named the *vertex chain code* (VCC) [12]. *VCC* determines the number of boundary pixels in the raster junctions (see Fig. 1b) and also has an alphabet with only three codes $\Sigma(VCC) = \{1,2,3\}$. The *Three OrThogonal* (3OT) chain code was introduced in 2005 [13]. Its codes are determined as follows:

- if the current coding direction is the same as the coding direction of its predecessor, $\sigma = 0$;
- if the current coding direction is equal to its first predecessor whose coding direction is different than the direction of its predecessor, $\sigma = 1$;
- $\sigma = 2$ otherwise.

The 3OT alphabet therefore consists of three symbols $\Sigma(3OT) = \{0,1,2\}$ (see Fig. 2b).

Chain codes are used for describing geometrical shapes, which may take a lot of memory storage on today's high-resolution devices, hence various domain-specific chain code compression methods have been developed. Sánchez-Cruz and Rodríguez-Diaz [14] showed that chain code based compression of bi-level images can achieve even better results than JBIG [15].

This paper is structured as follows: the next section provides a brief overview of existing chain code compression methods. Section 3 introduces a new universal lossless chain code compression approach. The efficiency of the proposed approach is demonstrated in Section 4 using benchmark shapes. Section 5 concludes the paper.

2. Related work on chain code compression

The pioneering work in the chain code compression has been done by Nunes et al., who used Huffman codes on *DCC* [11]. Their

☆ This paper has been recommended for acceptance by Prof. M.T. Sun.

* Corresponding author. Fax: +386 2 220 7272.
E-mail address: borut.zalik@um.si (B. Žalik).
URL: http://gemma.uni-mb.si (B. Žalik).

http://dx.doi.org/10.1016/j.jvcir.2015.01.013
1047-3203/

compression was quasi-lossless (i.e. near-lossless), as some of the original data were lost at the expense of better compression. Another Huffman-based compression was proposed by Liu and Žalik [16] on their directional difference chain code (*DDCC*). *DDCC* encodes eight angular differences of *F8* (a modification of *DDCC* – a *Normalised angle difference* was used in our test and explained in Section 3). Liu et al. later [17] introduced three methods for *VCC* compressing, namely: extended *VCC* (*E_VCC*), variable *VCC* (*V_VCC*), and compressed *VCC* (*C_VCC*). The first method encodes symbol combinations $\{1,3\}$ (or $\{3,1\}$) as a single code employing the unused bit combination of three *VCC* code, *V_VCC* encodes symbol 2 with one bit, and the remaining two symbols with two bits, whilst *C_VCC* provides even further compression by utilising Huffman codes for the more frequent combinations of *VCC* symbols. Sánchez-Cruz et al. [18] introduced a modified *3OT* (*M_3OT*) chain code, which achieves higher compression ratios by introducing symbols $3, 4$, and 5 in order to encode frequent combinations of *3OT* chain code symbols 0 and 1. Soon after they improved the *3OT* and *DDCC* chain codes [14], where arithmetic coding was used instead of statically determined Huffman codes. Sánchez-Cruz [19] also proposed a newer compression method for *F8* by finding repetitive substrings within the chain code. Liu et al. [20] have recently proposed another quasi-lossless chain code compression method, which replaces less frequent angular differences (i.e. 135° and 180°) in *DDCC* with pairs of more frequent ones. They have also introduced a run-length encoding (*RLE*) scheme in order to further compress repetitive runs of 0° angular differences. Recently, Žalik and Lukač [21] developed a new lossless chain code compression that does not rely on statistical assumptions (i.e. Huffman coding tables), but rather use a move-to-front transform (*MTFT*) to reduce the information entropy and then encode the transformed symbols with adaptive *RLE* (*ARLE*). To date, this method provided the most efficient compression of *VCC*, *3OT*, and normalised angle difference (*NAD*). Chain codes can also be used for the compression of 3D voxelised shapes, as recently demonstrated by Lemus et al. [22].

3. The proposed lossless compression method

The algorithms for chain code compression developed to date and briefly summarised in Section 2, have been dedicated to spe-

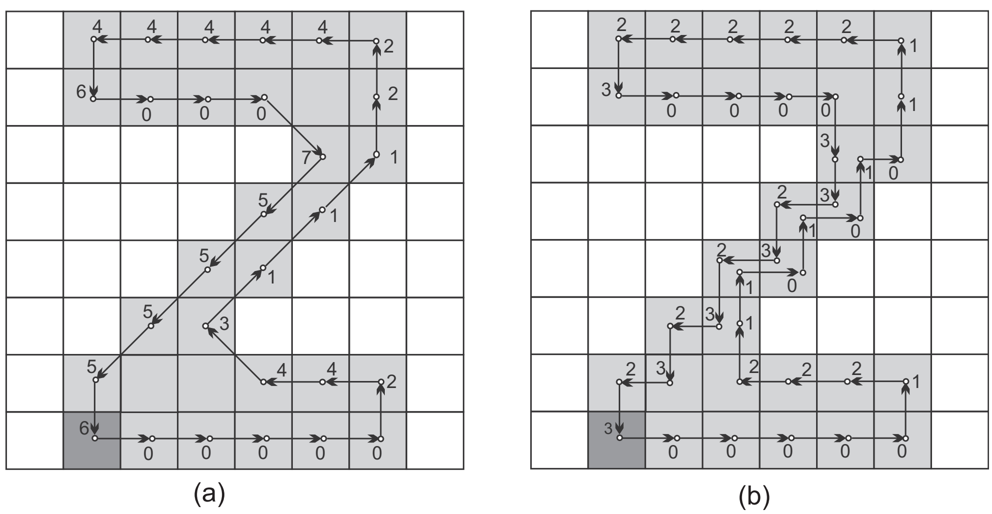

Fig. 1. Freeman chain codes: (a) *F8* and (b) *F4*.

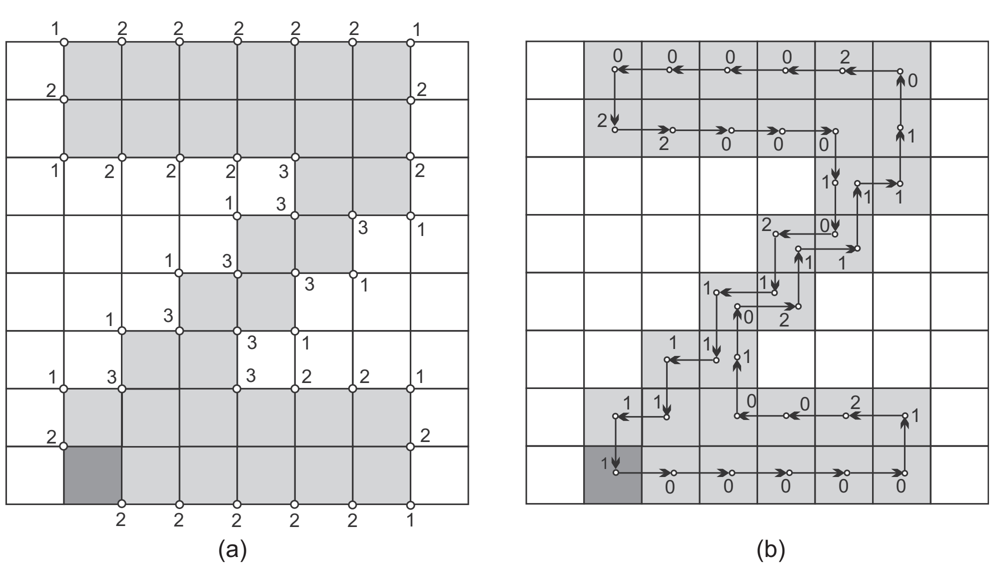

Fig. 2. Chain codes: (a) *VCC* and (b) *3OT*.