IFIG

Research

Report

Institut für Informatik

Improving Raster Image Run-Length Encoding Using Data Order

Markus Holzer Martin Kutrib

IFIG Research Report 0105

July 2001

Institut für Informatik
JLU Gießen
Arndtstraße 2
D-35392 Giessen, Germany
Tel: +49-641-99-32141
Fax: +49-641-99-32149
mail@informatik.uni-giessen.de
www.informatik.uni-giessen.de

Justus-Liebig-Universität Giessen

IFIG Research Report
IFIG Research Report 0105, July 2001

Improving Raster Image Run-Length Encoding Using Data Order

Markus Holzer[1]

Institut für Informatik, Technische Universität München
Arcisstraße 21, D-80290 München, Germany

Martin Kutrib[2]

Institute für Informatik, Universität Giessen
Arndtstr. 2, D-35392 Giessen, Germany

Abstract. We examine the technique of run-length encoding in combination with data order, where our attention is focused on good performance of image operations such as, e.g., rotation, reflection, and zooming. To this end we develop a new type of data order that supports these operations well and allows to perform them on a variant of a double-queue automaton directly on the compressed data stream. Because of its shape we call this data order shamrock or S-order. To confirm our theoretical results on S-order we have performed some experiments on sample data using various data orderings that appear in the literature.

CR Subject Classification (1998): E.4, F.1, G.2.1, I.4.2, I.5.3

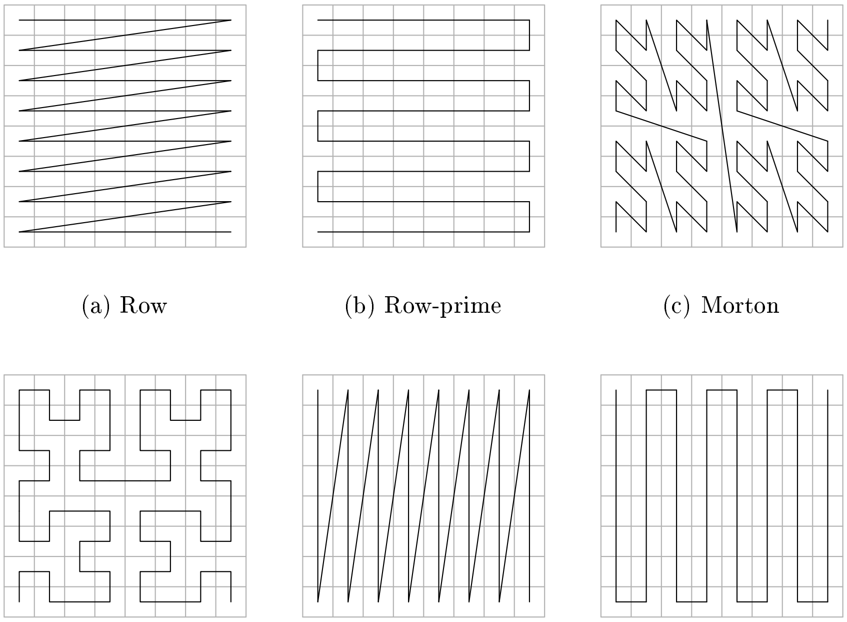
[1]E-mail: holzer@informatik.tu-muenchen.de
[2]E-mail: kutrib@informatik.uni-giessen.de

1 Introduction

Run-length encoding (RLE) is one of the most simplest lossless type of data compression schemes and is based on a simple principle of encoding data; see, e.g., Salomon [9]. This principle is that every partial stream which is formed of the same data values, i.e., sequence of repeated data values, is replaced by a single value and a count number—in the terminology of data compression repeating values are called a *run*. This intuitive principle works best on certain data types in which sequences of repeated data values can be noticed. Therefore RLE is usually applied to files that contain large numbers of consecutive occurrences of the same byte pattern. Good examples are text files which contain multiple spaces for indentation and formatting paragraphs, tables, and charts, as well as digitized signals as, e.g., monochrome images or sound files. Although RLE can be easily implemented and quickly executed it cannot achieve great compression ratios.

Many researchers, including many in the database community, have proposed clustering data using orderings to improve compression ratios. Since the computation of the stream, which can be compressed using the above described principle, is the heart of RLE, it is natural to pay special attention to these data orderings. One of the most common form of grided data representations as, e.g., raster images, is row order, where cells are traversed in sequence by row and, within row, by column. That RLE may benefit from using different data ordering was shown by several authors, especially geographers. One of the first experiments with data orderings are often credited to Morton [8] in the mid 1960's. Later further research into this topic was undertaken by Goodchild and Grandfield [5]. In this paper four data orders were tested to determine their compression capability in combination with RLE. Besides the already mentioned row ordering also row-prime, Morton order, and Hilbert order—see Figure 1(a)–(d)—were considered. Note that Morton order is also called Z-order and plays a crucial role in spatial databases to cluster multi-dimensional data; for further details we refer to Gaede and Günther [4]. It turned out, that storage could be reduced up to 60% using Hilbert order, and 25% using Z-order, as opposed to using row order, for (monochrome) images with high spatial homogeneity. For images with low spatial homogeneity the tests resulted in a 5% reduction from Hilbert order and a 5% increase from Z-order, over row order. Similar results were obtained by a comparative analysis by Abel and Mark [1]. Intuitively, these results tell us that using a Hilbert scan the coherence will be improved by avoiding the discontinuities found at the end of row order scan lines; adjacent pixels from a Hilbert scan share the same area because of its fractal nature, and are thus more likely to be similar than pixels in horizontally scanned lines. These factors improve the performance of RLE.

Digital image compression deals with the problem of finding such a succinct representation for raster images. Over the years quite a few compression techniques have been developed that aim to eliminate redundancy from raster images; see, e.g., Kou [7]. Images are preferably maintained in compressed form not only to save storage space, but also to perform operations faster on the compressed



(a) Row (b) Row-prime (c) Morton

(d) Hilbert (e) Column (f) Column-prime

Figure 1: Some standard data orderings.

image. This is a realistic objective because in compressed form less data has to be processed. Because it must be expected that different properties of the various orderings are reflected in differing degrees of suitability for various tasks we examine the technique of RLE in combination with data order, where our attention is focused on good performance of image operations such as, e.g., rotation, reflection, and zooming.

Intuitively speaking, a data order should possess certain desirable properties: (1) Pass once through each point in the image, (2) easy description and computation of the order, (3) be stable when the image becomes larger, and (4) provide variable resolution. For our purpose we have to develop a new sort of data order which supports these image operations, since non of the existing orders are useful. Because of its shape we call it shamrock or S-order, for short. More precisely we will see that the support of the image operations rotation and reflection clashes with condition (1). Hence, S-order must access certain image points more than once, which might be some disadvantage compared to Hilbert order. Nevertheless, our experiments show that S-order is still comparable with Z-order and mostly dominates it. Moreover, the S-order can be computed by a recursive algorithm as in the case of the Hilbert order and the order is actually so simple that the mentioned image operations can be performed by a variant of a double-queue-automaton.

The paper is organized as follows. In the next section we introduce the necessary notations. Section 3 introduces the new sort of data order under the light that non of the existing orderings support rotation and reflection well. Then in Section 4 we briefly describe some algorithms performing raster image operations on the encoded data in linear time with respect to the input size. Finally, Section 5 summarizes experiments, which we have performed recently.

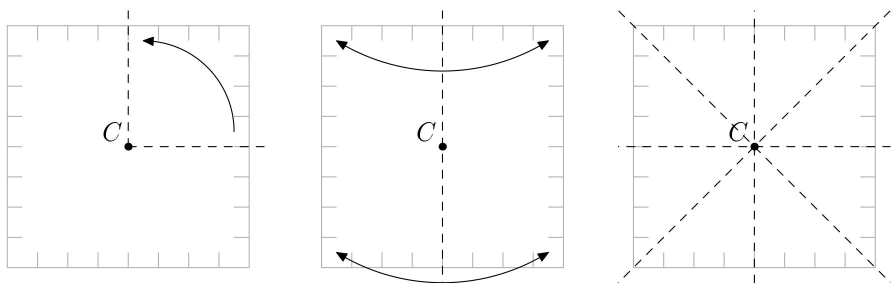
2 Definitions

In this section we give the necessary notations on raster images and data order. Let n be a power of two. An $n \times n$ raster image is defined to be a quadratic array or grid of same size built from regularly sampled values, known as pixels—picture elements. In a raster image each pixel has a coordinate (i, j), with $1 \leq i, j \leq n$, and a number associated with it, generally specifying a color which the pixel should be displayed in. The pixels with coordinates $(1, 1)$ and (n, n), respectively, lie in the lower left and upper right corner, respectively. Throughout the paper we only make use of monochrome raster images, hence restricting the colors to *black* and *white*.

The image operations rotation and reflection are nothing other than certain permutations of the pixels in the raster image. Let d refer to the $\frac{\pi}{2}$ rotation around the center C of the image, and s to the reflection at the vertical line intersecting the center—see Figures 2(a) and (b). Then d and s are permutations operating on the set of pixel coordinates $P_n = \{\, (i, j) \mid 1 \leq i, j \leq n \,\}$, satisfying $d^4 = id$, $s^2 = id$, and $dsd = s$. From these equations one sees that d and s generate the dihedral group D_4 containing

$$D_4 = \{id, d, d^2, d^3, s, sd, sd^2, sd^3\}.$$

Observe, that D_4 has order 8. For further information on dihedral groups we refer to Weinstein [10]. The four axis of symmetry induced by D_4 are shown in Figure 2(c). We say that a raster image is closed under the operations of the dihedral group D_4 if and only if the picture is unchanged by the transformations in D_4.



(a) Rotation d. (b) Reflection s. (c) Axis of symmetry.

Figure 2: Basics on the dihedral group D_4.

Finally, we make our intuition on data orderings more precise. The definition of a data order reads as follows. Again, let n be a natural number. A *data order* φ on an $n \times n$ grid is a one-to-one mapping

$$\varphi : \{1, \dots, n^2\} \to \{1, \dots, n\} \times \{1, \dots, n\}.$$

A data order φ is called *continuous* if and only if $d_2(\varphi(i), \varphi(i+1)) = 1$ holds for all $1 \leq i < n^2$, where d_2 denotes the Euclidean distance d_2, and a continuous data order φ is *cyclic* if and only if $d_2(\varphi(1), \varphi(n^2)) = 1$. We associate an undirected graph $D(\varphi)$ with a data order φ in the obvious way, i.e., $D(\varphi) = (V, E)$, where

$$V = \{1, \dots, n\} \times \{1, \dots, n\}$$

and

$$E = \{\, (\varphi(i), \varphi(i+1)) \mid 1 \leq i < n^2 \,\} \cup \{(\varphi(n^2), \varphi(1))\}.$$

We consider two data orders two be equal if and only if they induce the same graph. This is particularly useful because, e.g., data order φ and the data order where every image point is moved one step along the underlying graph or in other words

$$\varphi'(i) = \begin{cases} \varphi(i+1) & \text{if } 1 \leq i < n^2 \\ \varphi(1) & \text{otherwise,} \end{cases}$$

are equivalent. Thus, the starting point of a data order is not important. Moreover, if φ is continuous, then so is φ'. The some holds in case of cyclic data orders.

A data order φ is *closed* under the dihedral group D_4 if and only if the graph $D(\varphi)$ remains unchanged by the transformations in D_4, i.e., if one applies the appropriate permutation operation on the edges. Note, that none of the data orderings shown in Figure 1 are closed under the dihedral group D_4.

Observe, that a continuous data order traverses a grid making unit steps and turning only at right angles. Sometimes a continuous data order in our sense is called *discrete space-filling curve* in the literature. We have already seen the most typical example of a continuous data ordering, namely Hilbert ordering, which is based on the equally named space-filling curve [6]. It is created by starting with an initial shape, looking like a *staple*, that is copied and rotated four times with connecting lines inserted to fill a square area. The first few stages are shown in Figure 3.

Its simplicity and beauty derive from the fact that it progressively subdivides a square array down into an array of four sub-squares. The final curve is created by repeating and rotating the copying infinitely. This is yet another feature of continuous data orderings since they are usually *self-similar*, i.e., the data order or curve can be generated by putting together identical units, only applying rotation and reflection to these units. Observe, that although Z-order has a flavour of self-similarity as one can image from Figure 1(c), it is a non-continuous data order in our sense. In the sequel we are only interested in continuous data orderings.

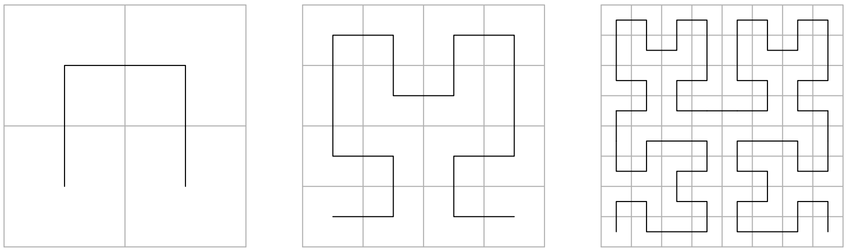


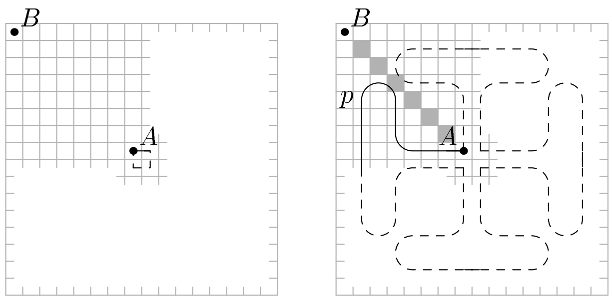
Figure 3: First four stages of the Hilbert space-filling curve.

3 Data Order and its Generalization

In this section we examine the question, whether there exists a data order which is closed under rotation and reflection. Up to now we only have seen one example with this property, namely the Hilbert order on a 2×2 grid—see Figure 3. It is clear, that Hilbert order on larger grids is not closed under rotation and reflection from the dihedral group due to the "sticky" ends of the data order. In general, it is easy to see that non-cyclic continuous data orderings can not be closed under rotation and reflection. The next theorem shows, that the situation is even worse, since for large enough grids even no cyclic data order with the property we are looking for exists.

Theorem 1 *Let n be a power of two greater than or equal to four. Then there is no cyclic data order $\varphi : \{1, \ldots, n^2\} \to \{1, \ldots, n\} \times \{1, \ldots, n\}$ which is closed under the dihedral group D_4.*

Proof. For the sake of a contradiction assume that a cyclic data order φ on an $n \times n$ grid which is closed under the dihedral group D_4 exists. Divide the $n \times n$



(a) 1st case.

(b) 2nd case.

Figure 4: An assumption on the connectivity of cells is drawn with a solid line, while connectivity induced by rotation and reflection is drawn with a dashed line.

grid in four equally sized sub-grids and consider cells A and B with coordinates $(\frac{n}{2}, \frac{n}{2}+1)$ and $(1, n)$, respectively, i.e., the lower right and upper left cell, respectively, of the upper left sub-grid. Then we have to consider two cases. First, if A is connected to one of its neighbouring sub-grids, then we immediately obtain a contradiction. This is due to the fact that all to cell A corresponding cells—by rotation and reflection—are connected, and build a cycle without entering all cells in the grid. This is not possible in a cyclic data order. The situation described is seen in Figure 4(a).

Next consider the case when A is connected to the interior of the leftmost upper sub-grid. Then starting the indexing process in cell A leads us to a path p through that sub-grid eventually entering another sub-grid. Without loss of generality we may assume that the first step starting in A leads to the west and that we may enter the leftmost lower sub-grid. This already induces a cyclic ordering, which must be consistent with φ. The situation described is depicted in Figure 4(b). If path p doesn't include the cells on the diagonal between A and B—these cells are shaded in Figure 4(b), then the induced ordering is *not* space-filling. Thus a contradiction. On the other hand, if at least one of the cells on the diagonal between A and B is part of the path p, then we have to consider six cases how the path goes through this cell. The six possibilities are shown in Figure 5.

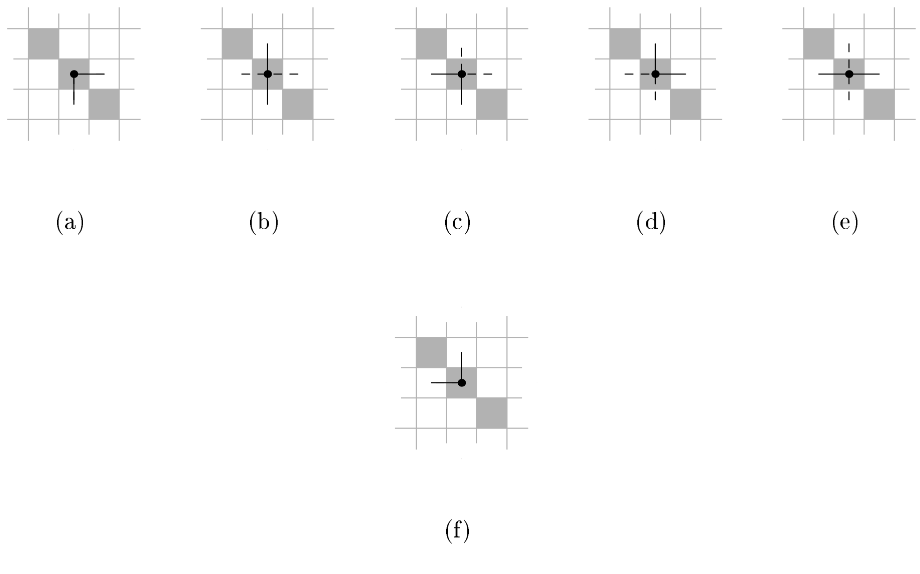


Figure 5: All possible connections of a gray shaded cell. An assumption on the connectivity of cells is drawn with a solid line, while connectivity induced by rotation and reflection is drawn with a dashed line.

Obviously, cases (a) and (f) are not possible since this would contradict our assumption that p must enter another sub-grid. The remaining four cases (b)–(e) immediately lead to a contradiction since by rotation and reflection the cell under consideration gets degree four. Thus, this is also a contradiction, and therefore shows that no cyclic data order closed under the dihedral group D_4 exists. $\square$

It is worth mentioning that the previous theorem can be strengthened such that n is any natural number greater than or equal to three. The theorem above easily generalizes to even numbers, while the missing odd case is seen as follows: In an $n \times n$ grid the interior cell A with coordinates $(\lceil \frac{n}{2} \rceil, \lceil \frac{n}{2} \rceil)$ must be connected to one of its neighbouring cells, and hence by rotation and reflection to all its four neighbours with Euclidean distance one. But then the cell under consideration has degree four, which is a contradiction since every point $\varphi(i)$ has degree two in the graph $D(\varphi)$. Thus, we have shown the following corollary.

Corollary 2 *Let $n \geq 3$. Then there is no cyclic data order φ on an $n \times n$ grid which is closed under the dihedral group D_4.* $\square$

The main argument in the previous proofs was that the closure under rotation and reflection implies that a grid graph must have vertices with degree four. Since this is not possible with an ordinary data order, we have obtained our contradiction. On the other hand this also shows that we must allow multiple access to grid point since a degree four vertex is traced twice. This immediately leads to generalized data orderings, which are defined as follows.

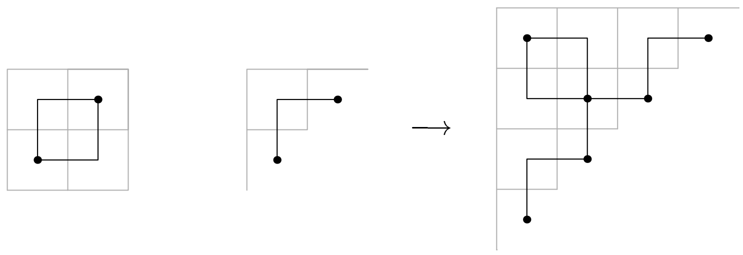
Let m and n be natural numbers. A *generalized data order* ψ on an $n \times n$ grid is a onto mapping

$$\psi : \{1, \ldots, n^2 + m\} \to \{1, \ldots, n\} \times \{1, \ldots, n\}.$$

As in case of ordinary data orderings we can associate a graph $D(\psi)$ with ψ. A generalized data order ψ is called *continuous* if and only if (1) $d_2(\psi(i), \varphi(i + 1)) = 1$ for all $1 \leq i < n^2 + m$, and (2) the associated graph $D(\psi)$ can be drawn in one stroke without hitting an edge twice. A continuous generalized data order φ is *cyclic* if and only if $d_2(\varphi(1), \varphi(n^2)) = 1$. By definition an ordinary (cyclic, continuous, respectively) data order is a (cyclic, continuous, respectively) generalized data order, but not necessarily *vice versa*. The next lemma shows that the graph $D(\psi)$ of a generalized data order fulfills the necessary condition to be closed under the dihedral group D_4. We omit the straight-forward proof of this lemma.

Lemma 3 *Let ψ be a generalized data order. Then every vertex in the graph $D(\psi)$ has even degree.* $\square$

At this point it is still not clear whether a generalized data order satisfying our requirements exists. In the sequel we answer this question positively by constructing such a data order, which we call *Shamrock* order or for short S-order, due to its shape. Analogously to Hilbert order we obtain in an inductive manner a generalized data order for $n \times n$ grids, where n is a power of two. However, there is a slight decisive difference. Whereas in Hilbert order the basic building blocks are squares (cf. Figure 3), the construction of the S-order is easier to describe using triangles to which only rotation is applied.



(a) Axiom

(b) Rule

Figure 6: Axiom and rule of the 0L type grammar to construct the S-order.

We visualize the construction by means of an 0L type grammar, hence forcing the rewriting to take place simultaneously at every nonterminal. A similar approach to describe the construction of space-filling curves was undertaken by Asano *et al.* [2]. A nonterminal in the grammar represents an upper triangle of an $n \times n$ grid together with a line segment. The start and endpoints of a line segment are marked by a dot to control the rule application and for better orientation. Hence to obtain the entire area of an $n \times n$ grid one has to put two triangles in an up-side down manner together. The axiom is depicted in Figure 6(a). The rewriting process goes through a number of iteration, where each iteration rewrites all non-overlapping triangles according to an application of a rule in our grammar as shown in Figure 6(b). Observe, that the 0L type grammar is allowed to rotate triangles. Figure 7 displays the first few iterations of the S-order starting with a single triangle instead of a whole square; the drawings are properly scaled such that each has same size. Moreover, in the first and second iteration we have indicated how to draw the line in one stroke without hitting an edge twice.

Hence we have shown the following theorem.

Theorem 4 *Let n be a power of two. Then there is a cyclic generalized data order $\psi : \{1, \ldots, n^2 + m\} \to \{1, \ldots, n\} \times \{1, \ldots, n\}$ which is closed under the dihedral group D_4.* □

It remains to compute the number m for S-order since this is the important parameter that will influence the compression ratio of RLE. Let n be a power of two for some constant $k \geq 1$. Then let E_k denote the number of degree-two points that hit the diagonal of a triangle that fits on an $n \times n$ grid and I_k be the number of intersecting points in a triangle that fits on an $n \times n$ grid. From Figure 7 one obtains that the first few values of (E_k, I_k) are $(0, 0)$, $(2, 1)$, $(6, 7)$, etc. In general $2 \cdot I_k + E_k$, which equals the number of intersecting points of the whole grid of appropriate size, obeys the recurrence

$$2 \cdot I_1 + E_1 = 0,$$

and

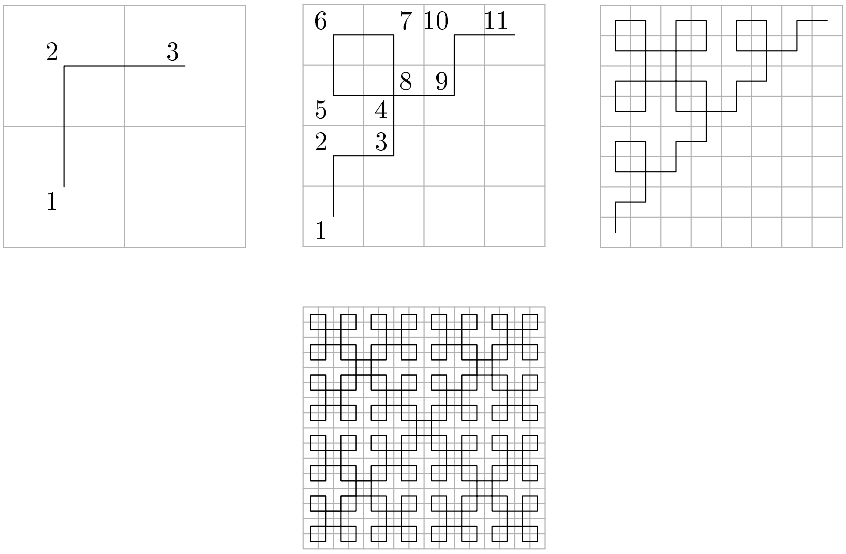


Figure 7: The upper left triangle part of the first three stages of the S-order and its final shape.

$$
\begin{aligned}
2 \cdot I_k + E_k &= 8 \cdot I_{k-1} + 4 \cdot E_{k-1} + 4 \\
&= 4 \cdot (2 \cdot I_{k-1} + E_{k-1}) + 4,
\end{aligned}
$$

for $k \geq 2$, which has the solution

$$
2 \cdot I_k + E_k = \frac{4^k - 4}{3}.
$$

Comparing this number with the overall number of pixels on the corresponding $n \times n$ grid results in an surplus of

$$
m = \frac{n^2 - 4}{3}
$$

visited points. Thus, roughly a third of the points is used to obtain our closure under the dihedral group D_4.

At this point the question arises, whether the constructed order is best possible with respect to m, i.e., the number of points visited more than once. In fact, for generalized data orderings constructed in a recursive manner by an 0L type grammar with nonterminals in triangle form, we can show that one cannot do better than $m = \frac{n^2-4}{3}$. This is seen as follows. Consider the triangle of a right-hand side of a rule. Partition this triangle counter clock-wise into a smaller triangle, a square, and a smaller triangle again. These three substructures have to be connected by edges in order to obtain a generalized data order. Moreover, the endpoint of these edges under consideration have degree four by Lemma 3. Degree two is not possible, because the drawing in the square area without outside connecting edges must be already closed under the dihedral group D_4 and thus has no vertices with degree one. Then the recurrence that determines m reads as

$$2 \cdot I_k + E_k = 4 \cdot (2 \cdot I_{k-1} + E_{k-1}) + c,$$

for $k \geq 2$ and some constant c. Here c is the overall number of vertices that are visited twice in a complete square built by two right-hand sides of the rule under consideration. By our argumentation above, c must be at least four, because one may place the connecting edges of the substructures near the midpoint of the hypotenuse, which results in four additional points (instead of eight). This shows that the S-order is best possible with respect to m.

4 Image Operations and S-Order

We discuss how to implement certain image operations under the assumption that RLE in combination with S-order is used to store raster images. It is seen that a certain variant of a double-queue automaton is already suitable to perform rotation, reflection, zooming, and inversion on the raster image only using its encoded form. Due to the lack of space we only briefly describe how to perform rotation and reflection.

First consider a 4×4 grid as shown in Figure 7. Then one easily observes that a rotation by $\frac{\pi}{2}$ changes the numbering of the appropriate S-order equal to the permutation

$$d = \begin{pmatrix} 1 & 2 & \dots & 15 & 16 & \dots & 20 \\ 6 & 7 & \dots & 20 & 1 & \dots & 5 \end{pmatrix}$$

on 20 points. This nicely generalizes to

$$d = \begin{pmatrix} 1 & 2 & \dots & n^2-1 & n^2 & \dots & \frac{4n^2-4}{3} \\ \frac{n^2-1}{3}+1 & \frac{n^2-1}{3}+2 & \dots & \frac{4n^2-4}{3} & 1 & \dots & \frac{n^2-1}{3} \end{pmatrix}$$

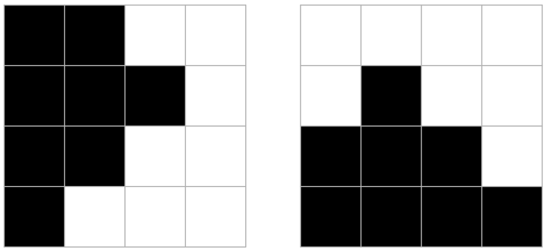
for $n \times n$ grids, which can be verified by induction on n. Analogously, we find that the reflection s on $n \times n$ grids is mimicked by the permutation

$$s = \begin{pmatrix} 1 & 2 & \dots & n^2 & n^2+1 & \dots & \frac{4n^2-4}{3} \\ n^2 & n^2-1 & \dots & 1 & \frac{4n^4-4}{3} & \dots & n^2+1 \end{pmatrix}.$$

¿From the formal language point of view, the rotation corresponds in a certain sense to an operation that maps a word uv to a word vu, while a reflection maps uv to $u^R v^R$, where w^R denotes the mirror image of w. Based on this easy observations we develop a pseudo code for a variant of a double-queue automaton that operates on the RLE of the raster image. To this end we have to make some assumptions on the RLE.

Without loss of generality we assume that the encoded data consist of a head and a body section, where the head contains information on the raster image size and the body the actual RLE of the image using S-order. The body is a sequence of tuples of the form $(\blacksquare : i)$ or $(\square : i)$ indicating that a data stream with i repeated values of $\blacksquare$ or $\square$, respectively, was seen during the

S-order like traverse of the raster image. Consider the raster image shown in Figure 8(a). A S-order traverse, partially depicted in the second drawing of Figure 7, reads out the pixel colors and results in the 20 element color sequence ■■■■■■■■■□□□■□□□□□■□, which is simply encoded as (■ : 9)(□ : 3)(■ : 1)(□ : 5)(■ : 1)(□ : 1). The RLE of the rotated raster image as shown in Figure 8(b) equals the tuple sequence (■ : 4)(□ : 3)(■ : 1)(□ : 5)(■ : 1)(□ : 1)(■ : 5).



(a) Original. (b) Rotated.

Figure 8: An example of a 4×4 monochrome raster image.

Now we are ready to give the pseudo code for a one-way double-queue automaton with additional registers to perform simple arithmetic tasks such as addition and multiplication of integers. The double-queue is manipulated by the functions first, rest, last, least, prefix, and postfix, which have the intuitive meanings. The isempty function checks whether the double-queue is empty or not–for further information on these operations we refer to Bauer and Goos [3]. Operation next acts on the input. It gives back a tuple of the form (■ : i) or (□ : i) and moves the input head to the right. The below given pseudo-code algorithm for the raster image rotation is based on our previous observations and uses the implicitly defined self-explaining and pre-initialized variables *dqueue* and *input*. It reads as follows:

```
                                        initialization and simple calculations
position := 0; surplus := 0;
cutpoint := (n^2-1)/3;
                                        search for the cut-point (n^2-1)/3
while position ≤ cutpoint do
        element := next(input);
        dqueue := postfix(dqueue, element);
        position := position + element.number od;
                                        remove last stored item
dqueue := least(dqueue);
                                        store remaining part
surplus := position − cutpoint;
dqueue := postfix(dqueue,
    (element.color : element.number − surplus));
```

```
                                              write overhanging pixels
write (element.color : surplus);
                                              write remaining input
while ¬iseof(input) do
        write next(input) od;
                                              write previously stored part
while ¬isempty(dqueue) do
        write first(dqueue) od;
```

The reader may have notice, that the algorithm is not optimal, since it doesn't verify whether it is possible to glue tuples with same color together after the cutting and recombination. For instance, rotating the raster image shown in Figure 8 twice, should result in the RLE $(\square : 3)(\blacksquare : 1)(\square : 5)(\blacksquare : 1)(\square : 1)(\blacksquare : 9)(\square : 1)$ but our algorithm applied twice to the RLE of the original raster image produces the sequence $(\square : 3)(\blacksquare : 1)(\square : 5)(\blacksquare : 1)(\square : 1)(\blacksquare : 5)(\blacksquare : 4)(\square : 1)$ instead. This is not a serious problem and can be fixed with slight changes to the above given pseudo code.

Similar algorithms can be given for the other raster image operations mentioned at the beginning of this section. Only in case of zooming a slight enhancement of the double-queue automaton is necessary in order to be able to construct the S-order for larger or smaller sized images. We have to omit the details and refer to a longer version of this paper. In all cases the running times of the algorithms are linear in the input size.

5 Experiments

We have conducted experiments with data order, especially S-order, on monochrome raster images. First of all to experimentally verify that S-order can compete against already existing data orderings and secondly to implement image operations as discussed in the previous sections. A screenshot of the Java 1.3 program used to perform our test runs on Intel Celeron 400 MHz and AMD Athlon 1 GHz machines and some sample input is shown in Figure 9—unfortunately, the buttons on the screenshot are labelled in German. Both machines use Windows 98 as an operating system and have 256 MB RAM. The input to our program are files in bmp-format.

As test data sets we have chosen a wide range of images from standard examples like text fragments and artificial images as, e.g., chess boards and fractals, up to non-common ones as, e.g., silhouettes of islands inspired by some of our geographical references, and modern paintings. We compare the various data orderings paying special attention to S-order by their compression ratio, which is the size of the output data divided by the size of the input data. The results are summarized in Figures 10 and 11 and can be interpreted as follows.

Figure 10(a) shows the results for Hilbert order compared to the traditional row order and column order. As expected Hilbert order is slightly better than the others, since most of the dots are above the diagonal that splits the first

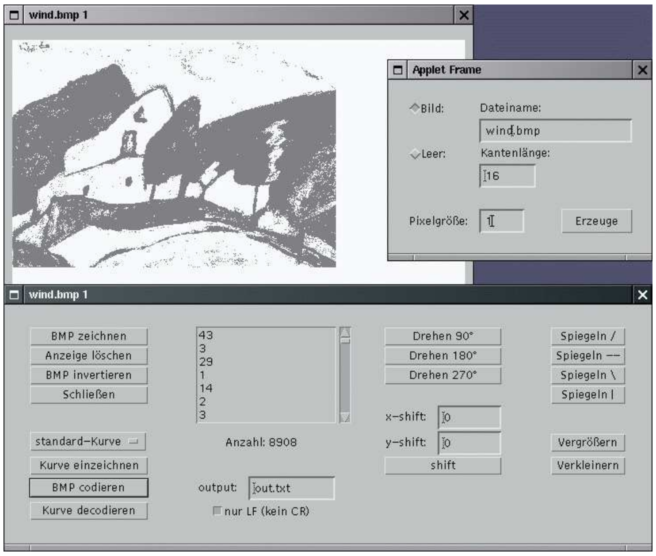
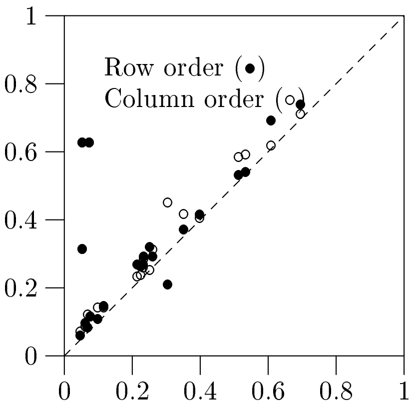


Figure 9: Screenshot of the Java 1.3 program written to perform some experiments and a sample image—"Wind and Weather" (1910) of G. Münter.

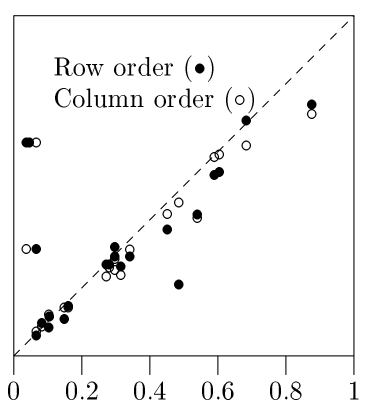
quadrant into half. The results for Z-order are nearly completely the other way around as shown in Figure 10(b). There, maybe due to the non-local nature of Z-order, most of the dots are below the diagonal. Hence row and column order mostly supersedes Z-order. Finally in Figure 10(c) the results for S-order are shown. Surprisingly S-order is weaker than the traditional orderings, which comes from the surplus of roughly a third of pixels points. The gray shaded area in Figure 10(c) indicates the region of compression ratio induced by these additional points. Taking a closer look, one observes, that the dots cover nearly the same region as in case of Z-order and in fact the overall impression is like in the Hilbert order case, only slightly shifted to the lower end in the gray shaded region. The outliers at the left hand sides of the diagrams are generated by chessboard-like images.

Nevertheless, the dominance of the space-filling Hilbert order and Z-order is not as significant as one might have expected from previous results in the literature. To confirm this intuition, we have to continue our experiments to clear this up.

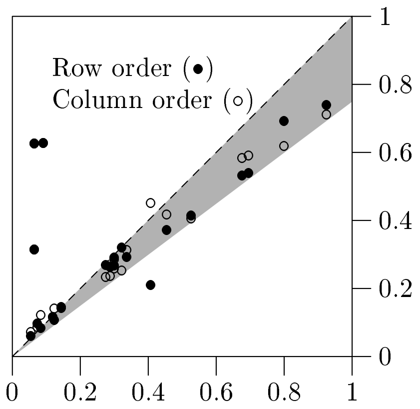
Finally in Figures 11(a)–11(c) row order, column order, and S-order are compared to Hilbert order and Z-order. For the former two figures we come to the same conclusion as for Figures 10(a) and 10(b), respectively. It remains to consider Figure 11(c) with the results for S-order. As in Figure 10(c) the gray shaded region is induced by the additional points of the S-order traverse. One observes, that S-order is roughly comparable to Z-order, which nicely fits to the interpretation of Figure 10(b). Interestingly, most of the dots for the Hilbert order lie exactly on the lower end of the gray shaded region, which shows that Hilbert order and S-order are similar with respect to their compression ability and the difference in compression ratio is determined by the surplus of pixels, only. This is exactly the result which one can await from the theoretical investigations in Section 3.



(a) Hilbert order.



(b) Z-order.

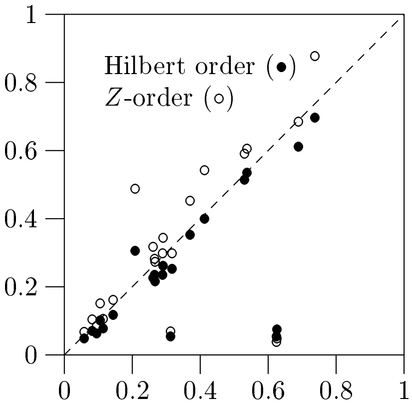


(c) S-order.

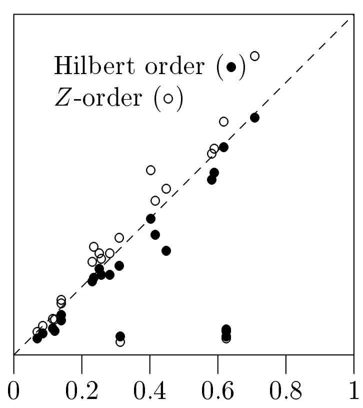
Figure 10: Compression ratio of Hilbert order, Z-order, and S-order (x-axis) compared to row and column order (y-axis).

6 Conclusions

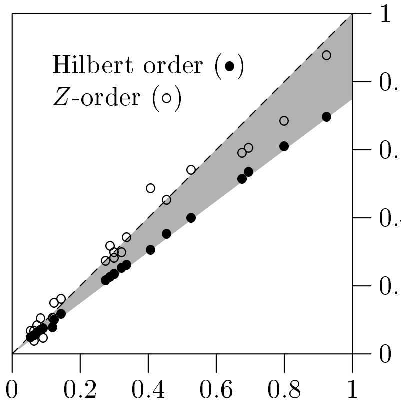
The results on the S-order nicely fit to already existing experimental investigations on data orderings and look very promising. We have seen that S-order is comparable to Z-order from the compression point of view, although the size of the domain of S-order is $\frac{4n^2-4}{3}$ instead of n^2 as in case of traditional data orderings. Moreover, our experiments suggest that in most cases Hilbert order is not better than $\frac{3}{4}$ of S-order, and that in addition S-order has the advantage to support certain image operations like rotation, reflection, and zooming well, since it was especially designed for this purpose. To overcome the slight obstacle on compression ratio of S-order one might relax the condition on the closure under the dihedral group D_4, in order to get rid of twice visited pixel



(a) Row order.



(b) Column order.



(c) S-order.

Figure 11: Compression ratio of Row order, Column order, and S-order (x-axis) compared to Hilbert order and Z-order (y-axis).

points, but still supporting image operations well. Future research will reveal to what applications and to what degree S-order or a relaxed variant can be advantageous.

7 Acknowledgments

We thank Björn Fay who has written the Java 1.3 program and conducted the experiments on various data orderings for us.

References

[1] D. J. Abel and D. M. Mark. A comparative analysis of some two-dimensional orderings. *International Journal of Geographical Information Systems*, 4(1):21–31, 1990.

[2] T. Asano, D. Ranjan, Th. Roos, E. Welzl, and P. Widmayer. Space-filling curves and their use in the design of geometric data structures. *Theoretical Computer Science*, 181(1):3–15, 1997.

[3] F. L. Bauer and G. Goos. *Informatik—Eine einführende Übersicht II.* Springer, 1984

[4] V. Gaede and O. Günther. Multidimensional access methods. *ACM Computing Surveys*, 30:170–231, Juni 1998.

[5] M. F. Goodchild and A. W. Grandfield. Optimizing raster storage: an examination of four alternatives. In *Proceedings of the 6th International Symposium on Automated Cartography*, pages 400–407, Ottawa, Ontario, Canada, 1983.

[6] D. Hilbert. Über die stetige Abbildung einer Linie auf ein Flächenstück. *Mathematische Annalen*, 38:459–460, 1891.

[7] W. Kou. *Digital Image Compression: Algorithms and Standards.* Kluwer, 1995.

[8] G. M. Morton. A computer oriented geodetic data base, and a new technique in file squencing. Technical report, IBM, Ottawa, Ontario, Canada, 1966.

[9] D. Salomon. *Data Compression.* Springer, 1997.

[10] M. Weinstein. *Examples of groups.* Polygonal Publishing House, Passaic, New Jersey, 1977.