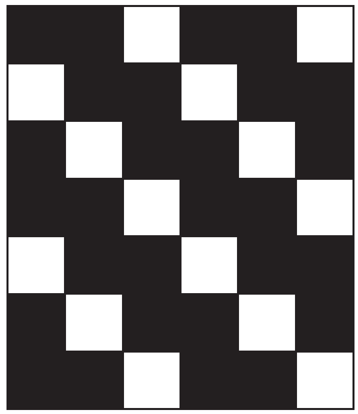
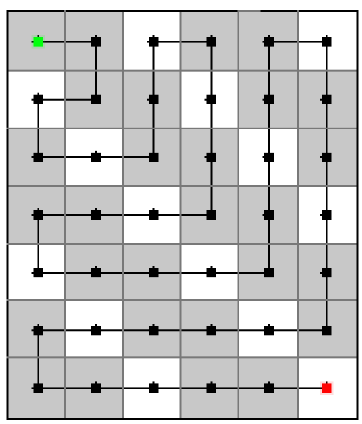
Pattern Tours

The patterns considered here are black and white and represented by a rectangular grid of cells. Here is a typical pattern: [Redundant]



A sequence of cell locations is called a *path*. A path that includes every cell of a grid exactly once is called a *tour*. The focus here is on tours. Here is an example of a tour on the pattern shown above:

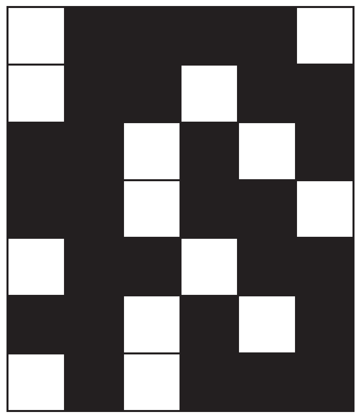


The green dot indicates the start of the tour and the red dot, the end.

The sequence of colors of the cells along a tour is called a *band*. Here is the band for the example above:

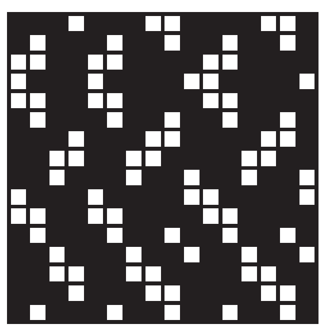
A tour and the corresponding band completely characterize a pattern in the sense that the tour and band can be used to construct the pattern.

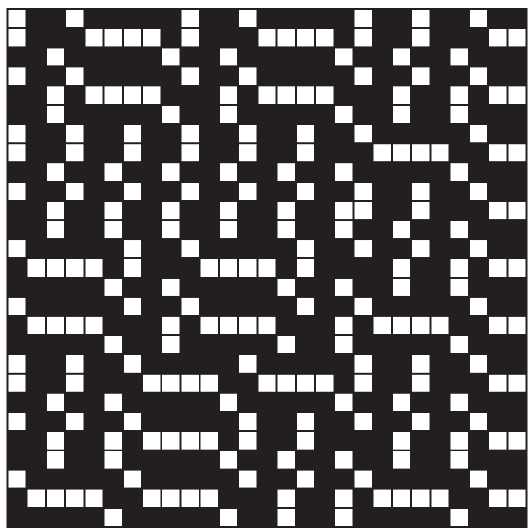
The process of producing a band from a pattern and a tour on it is called *reading out* the band. Conversely, given a blank grid, a tour, and a band, a pattern can be constructed by *reading in* the band. A band read out according to one tour and written in according to a different tour generally produces a different pattern. For example, here is the pattern that results from reading in the band show previously with the tour in reverse order:



Tours and bands can be constructed independently of any pattern. This allows the possibility of constructing interesting and perhaps unexpected patterns.

Here are three patterns produced by tours and bands created mathematically but independently.





These are just some results from the first experiments. They are by no means the most interesting patterns that can be created by the methods described here.

Perspective

A tour serves to distribute the colors of the band over all cells in a grid.

The number of possible tours for all but trivially small grids is enormous. If there are k cells, there are k possibilities for the first cell on the tour. This eliminates one cell but leaves $k - 1$ possibilities for the second cell. Therefore the number of possible tours for a grid of k cells is $k \times (k - 1) \times (k - 2) \times \ldots 3 \times 2 \times 1 = k!$ (k factorial). For an 8×8 grid, there are 64 cells and the number of possible tours

[Sidebar on factorials.]

is 64!, which is

126,886,932,185,884,164,103,433,389,335,161,
480,802,865,516,174,545,192,198,801,894,375,
214,704,230,400,000,000,000,000

[Sidebar here or elsewhere — or in in appendix — on immense numbers]

Obviously, tours must be chosen with some plan or concept in mind, for a tour is the geometry from which the eventual pattern is crafted. A random tour is extremely unlikely to produce attractive results.

The problem of tour design is complicated by the fact that it alone does not determine the pattern; the band plays a strong role in the final result. This makes tour design challenging and interesting.

There are certain things that can be looked for in tours. One is some kind of pattern in the tour itself. A jumbled, chaotic tour, even if far from random, is unlikely to produce attractive results *unless* the band is developed along with the tour. This is possible — take any pattern and any tour, however disorganized, and read out the band. This band, when read in by the tour, will reproduce the original pattern. Some uses for tours and bands produced in this way are described in a subsequent section, but the first concern is designing tours independently of bands.

While designing tours and bands independently requires both the application of some principles and some serendipity, it is the core of a process for getting attractive and unusual patterns.

If tours are taken alone, their design needs to be guided so that the tours themselves are attractive and interesting.

Tour Design

There are several properties to keep in mind when designing tours.

Symmetry: Symmetric designs are attractive to human beings (although no one knows exactly why). Various kinds of symmetries are possible for tours.

Repetition: Repetition of a unit within a design can be useful in tours just as it is in tilings, weaves, and other kinds of artistic constructions.

Variety: A little variety or an element of surprise can break an otherwise monotonous design and be aesthetically pleasing.

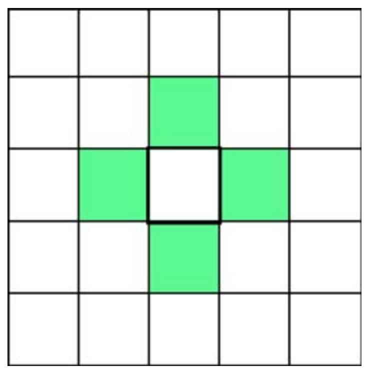
Continuity: Since a tour is a path in the general sense, there is some value in continuity. For example, the locations on a tour may be adjacent (their cells sharing a side). There are various kinds of continuity and technical names for them. The problem is discussed in another section in the context of constraints.

Tour Constraints

Constraints can be useful in design; they limit what is possible in a systematic way that prevents accidental aberrations, and they provide for a certain degree of regularity.

The kind of constraints here are local ones that allow tours to be built step-by-step. Such constraints might limit the distance from one location on the tour to the next or limit the possible directions in which the next location may lie.

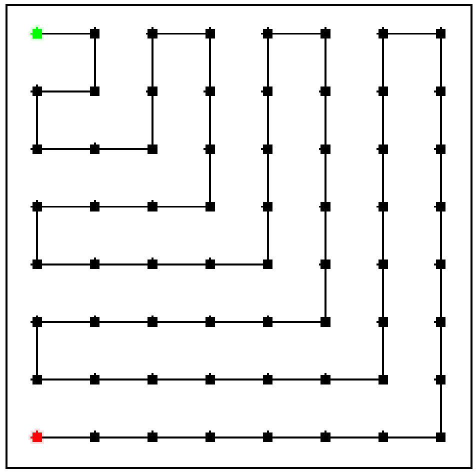
The neighborhood concept from cellular automata [2] is a useful model for this kind of constraint. For example, the von Neumann neighborhood looks like this:

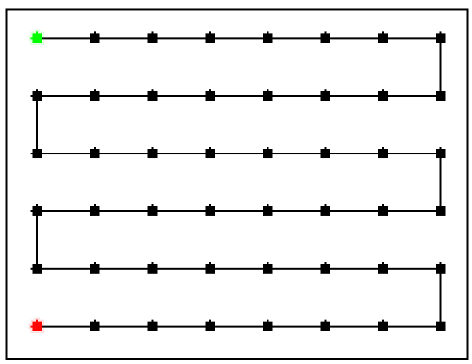


Viewed as a constraint, this neighborhood requires the location following the central location to be one cell away, horizontally or vertically. Staying put is not an option, since a location cannot appear twice in a tour. Locations that are off the grid, when a cell is at an edge, obviously are excluded. Similarly, locations that already are on the tour are forbidden.

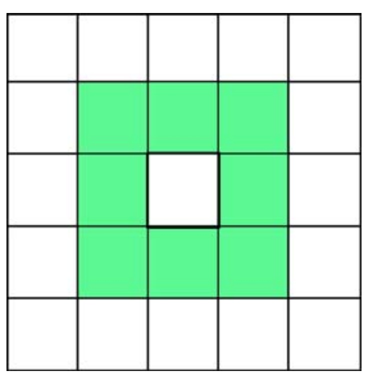
We'll call a tour constructed using this neighborhood a *von Neumann tour*. Von Neumann tours have a special kind of continuity, called *unicursal* in graph theory. Von Neumann tours also are *planar*, meaning there are no crossings on a line drawn along the tour [?].

Here are examples of von Neumann tours:

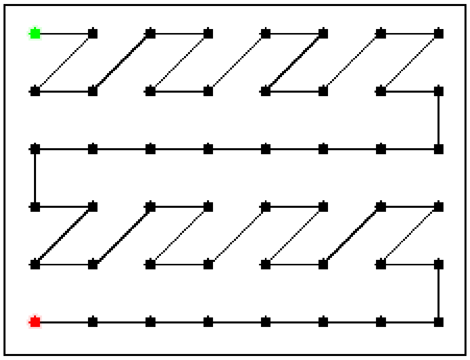


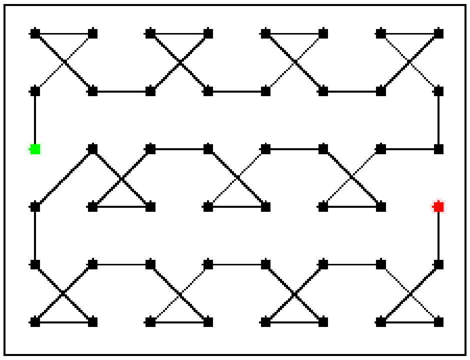


The Moore neighborhood allows more freedom of movement:

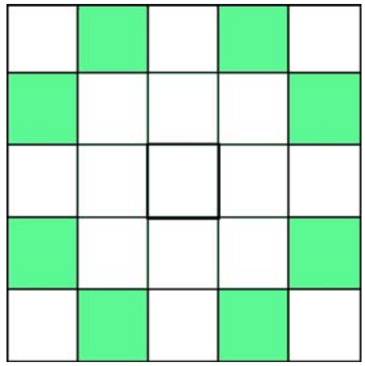


Moore tours include von Neumann tours as a subset, but they allow considerably more variety, including diagonal moves and non-planar tours. Here are some examples of Moore tours that are not von Neumann tours:



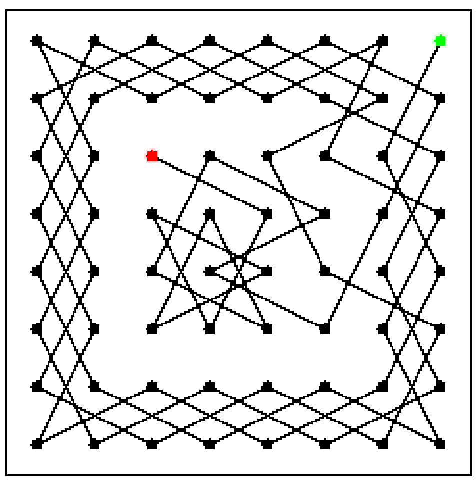


The legal moves of chess pieces can be described as neighborhoods. For example, the knight has the neighborhood



Note that the knight "jumps"; it cannot move to an adjacent cell.

Knight's tours are sufficiently interesting and difficult to design that they have occupied the attention of chess players and mathematicians for centuries. Here is an example:

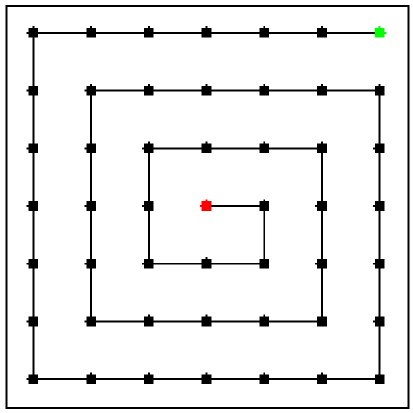


Neighborhoods provide constraints that limit the nature of tours. In constructing tours using neighborhoods, there are, of course, choices. One thing that can happen is getting into a situation in which no further move is possible. There are ways of dealing with this problem, which we'll discuss in a later section.

Tour Classification

Tours can be classified roughly according to the methods by which they can be constructed and places they can be found. In many cases, a tour will fit into more than one category, depending on how it is created or viewed.

Algorithmic Tours: These tours are constructed according to a fixed set of rules applied in a well-defined fashion. An example of an algorithmic tour is a square spiral:

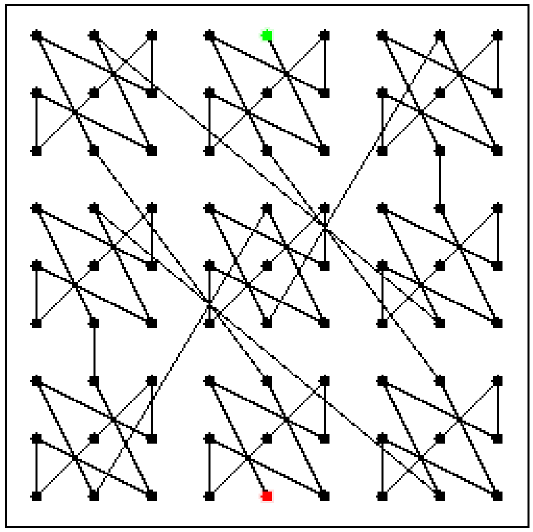


Note that this also is a von Neumann tour.

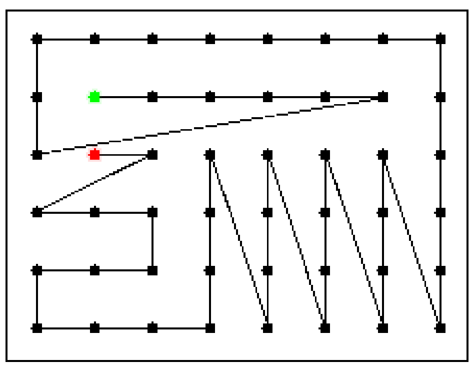
The geometry of this kind of tour is obvious, as are possible variations on it. You might find it illuminating to devise an algorithm that produces square-spiral tours.

Neighborhood-Constrained Tours: These tours are discussed above. They come in great variety and will be the subject of subsequent articles.

Numerically Derived Tours: These tours are derived from numerical problems and puzzles that are not directly related to tours but that nonetheless can be interpreted as tours. An example is this tour derived from a magic square, in which the sums of the rows, columns, and main diagonals are all equal:



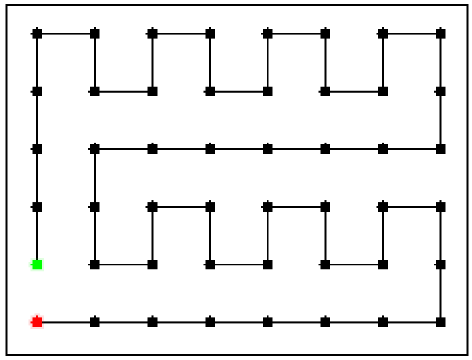
Miscellaneous: There is the inevitable "other" category containing tours that do not fit elsewhere. Here is an example of a tour that was constructed by hand:



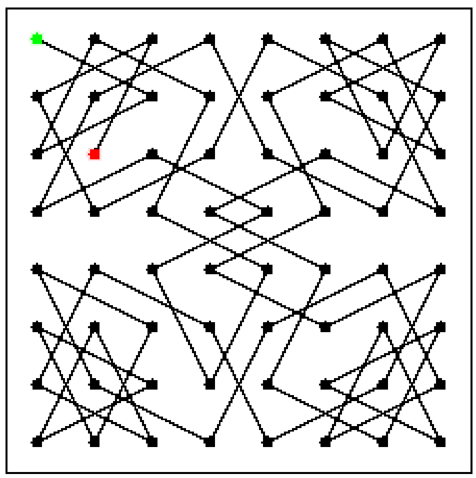
More Terminology

There are a few other terms related to tours that are important in some contexts:

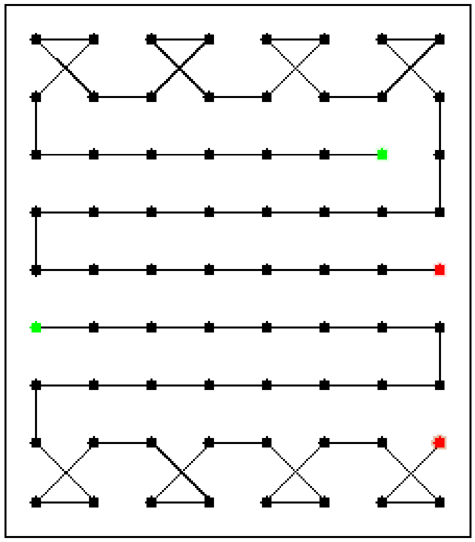
Re-Entrant Tours: A tour whose last location is within a legal move of its first location is called *re-entrant*. An example of a re-entrant von Neumann tour is:



Here is an example of a re-entrant knight's tour:



Piece-Wise Tours: A piece-wise tour is one composed of parts that satisfy some condition, but the whole does not. Here is an example of a piece-wise Moore tour:



Such tours can be made into regular tours by connecting the end of one piece to the beginning of the other, but it often is more useful to view the parts independently.

Incomplete Tours: A path that does not include every location in a grid is called *incomplete* or *open*. Incomplete tours can be used for components of piece-wise tours or alone in some applications.

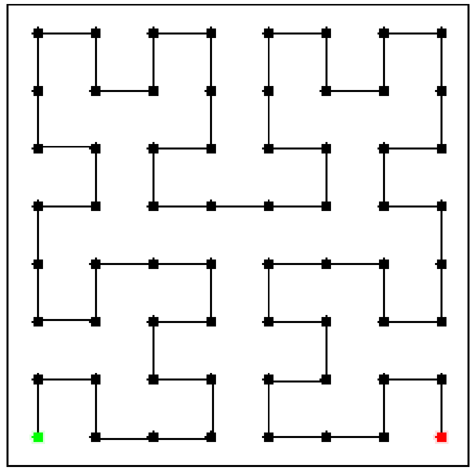
Overtours: A path that includes a location more that once is called an overtour. Overtours also have their applications, which are discussed in another article. [Watch references to non-existent sections, which if not written, should be turned into subjects for further study.]

Graphical Representations

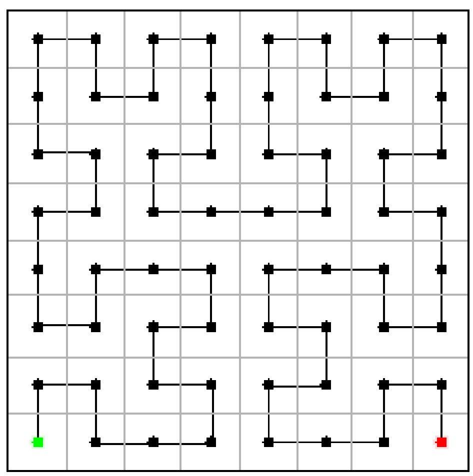
In order to deal with tours, both representations for understanding them and representations for creating and manipulating them are needed.

So far pattern tours have been represented by lines drawn though the cells in the order of traversal. The beginning of a tour was shown in red and the end in green for two reasons: for ease in identifying these important locations and for establishing the direction of transversal.

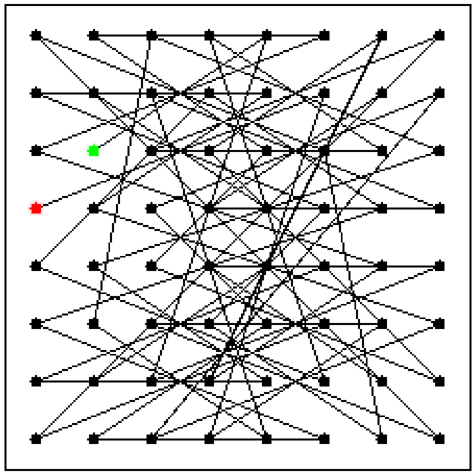
This method works well for many tours in giving an overall impression of their geometries. Here is an example:



Grid lines may help identify specific locations:



If, however, the tour is non-planar, line crossings may make the tour difficult to follow. And if the tour is chaotic and has many criss-crossing jumps, such a graphic representation may be a useless jumble:



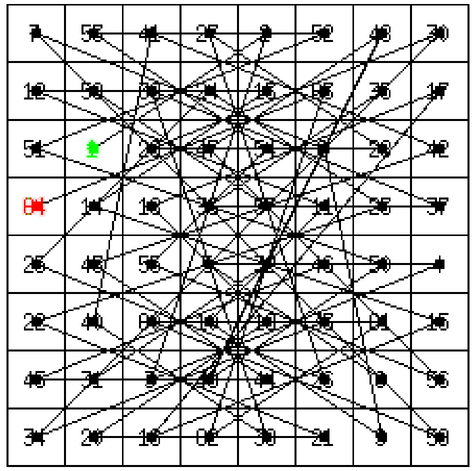
This representation also is ambiguous. There is no way to tell if a line through several cells includes all the cells on the line or jumps over some.

An alternative graphical method is to dispense with the lines and assign numbers to the cells that specify the order of traversal. Again, the ends can be highlighted by colors:

7	53	41	27	2	52	48	30
12	58	38	24	13	63	35	17
51	1	29	47	54	8	28	42
64	14	18	36	57	11	23	37
25	43	55	5	32	46	50	4
22	40	60	10	19	33	61	15
45	31	3	49	44	26	6	56
34	20	16	62	39	21	9	59

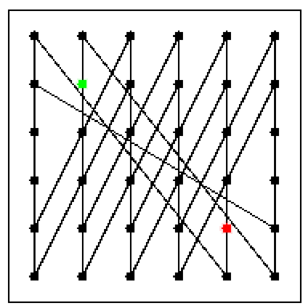
This numerical representation has the virtue of being precise and unambiguous. It may be difficult, however, to locate successive cells on a tour.

An alternative that is sometimes used is to overlay the line and numbered grid representations as in:



Except for small, simple tours, the result may be worse than either of the two forms separately. Both numbered grids and line drawings in situations are needed where a clear understanding of a tour is required. An example is:

28	10	31	13	34	16
19	1	22	4	25	7
29	11	32	14	35	17
20	2	23	5	26	8
30	12	33	15	36	18
21	3	24	6	27	9



Data Representations

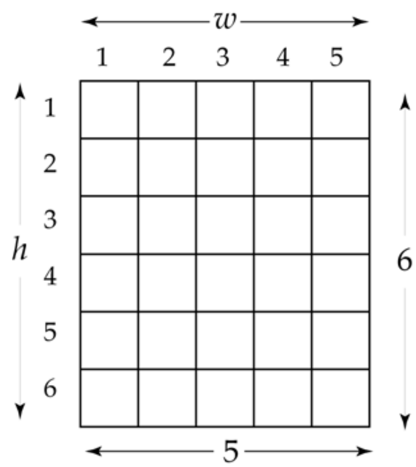
In order to write procedures for manipulating tours, data representations that a computer program can use are necessary. These are very different from the graphical representations that human beings find easy to understand.

Coordinate Lists

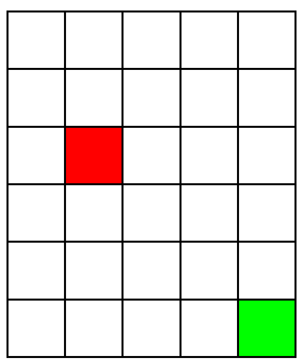
One data representation of a tour as a list of cell locations. For this, A systematic way of identifying locations on a grid is needed.

[Duplicates material given elsewhere.]

The dimensions of a grid in cells are given by two numbers, w for the width in cells and h:

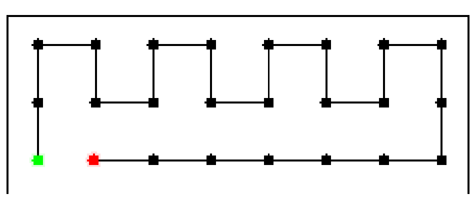


Rows and columns are numbered as indicated. The location of a cell is given by its coordinates, which is a pair (i, j) in which i is the column number of the cell and j is the row number of the cell. For example, in



the coordinates of the red cell are (2, 3) and the coordinates of the green cell are (5, 6).

A tour then can be represented by a list of coordinates in the order of traversal. For example, the tour



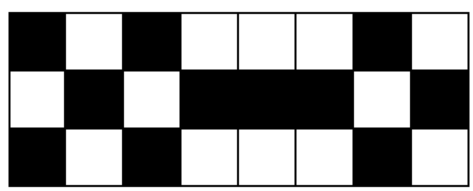
has the coordinate list

(1, 3), (1, 2), (1, 1), (2, 1), (2, 2), (3, 2), (3, 1),
(4, 1), (4, 2), (5, 2), (5, 1), (6, 1), (6, 2), (7, 2),
(7, 1), (8, 1), (8, 2), (8, 3), (7, 3), (6, 3), (5, 3),
(4, 3), (3, 3), (2, 3)

To apply a band to a tour represented by a coordinate list, it is only necessary to pair the coordinate list with the band and place the colors of the band, in order, at the designated succession of locations. For example, reducing the type size of the coordinate list for the tour above and drawing a band below it provide a complete specification for a pattern:

(1,3),(1,2),(1,1),(2,1),(2,2),(3,2),(3,1),(4,1),(4,2),(5,2),(5,1),(6,1),(6,2),(7,2),(7,1),(8,1),(8,2),
(8,3),(7,3),(6,3),(5,3),(4,3),(3,3),(2,3)

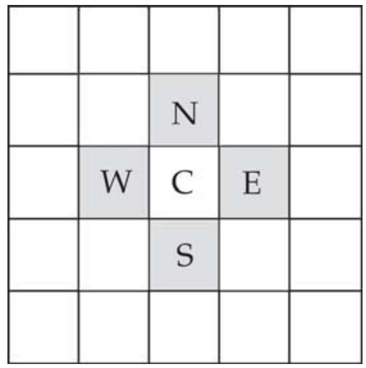
The resulting pattern is:



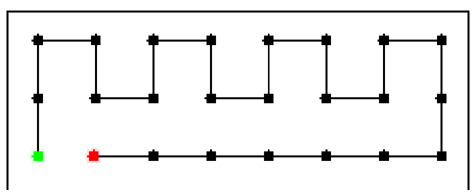
Navigational Representations

Methods for describing tours, which are particularly useful in some constructing some kinds of tours, use navigation rather than specific coordinates.

For von Neumann paths, a list of compass points that specify the direction of one cell to the next is sufficient. Given the neighborhood labels in terms of compass points, as in



the cell following C can be indicated by a letter that corresponds to the direction of movement. For example, for the tour

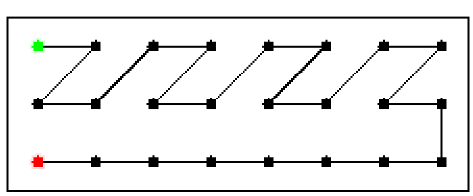


the sequence of directional moves is given by the string of letters

NNESENESENESENESSWWWWWW

Such a string and the location of the starting cell completely characterize von Neumann tours and in a much more compact way than coordinate lists.

As given above, direction strings are limited to von Neumann tours. There is, for example, no way to specify a diagonal move. The concept of direction string could be extended to include Moore tours by adding letters for the diagonal moves. A more general method, that can be used for all tours, is to provide a way for specifying passing over cells without including them on the tour. We'll use lowercase letters for this: nesw in addition to NESW. For example, the Moore tour



is described by the direction string

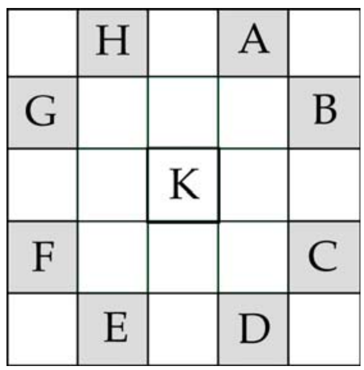
EsWEeNEEsWEeNEEsWEeNEsWES
WWWWWWWW

Of course sW is equivalent to wS, and so on.

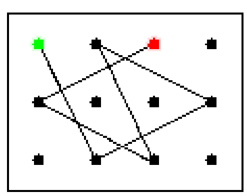
Using this navigational method, it is not necessary to specify the starting cell. Assuming the string starts at the upper-left corner, the first location on the tour can be reached by a string of nesw moves at the beginning.

Although this method is completely general, it is cumbersome for tours in which there are long jumps.

In cases of specialized neighborhoods, such as the knight's, a customized navigational system can be used. Here's the knight' neighborhood with the cells that can be reached lettered clockwise around the knight:



For this fragmentary tour



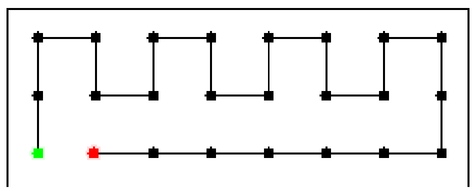
the navigational string is DBGDGB.

There are many possible navigational alternatives to these kinds of navigational strings. An attractive one is to use L-System notation in which the symbols used are interpreted to draw images [3]. Adapted to describing tours, these symbols are:

F move forward, including cell on tour
f move forward, not including cell on tour
+ turn right 90°
– turn left 90°

The initial direction is east and that the starting point is the upper-left corner.

For example, the von Neumann tour shown earlier:

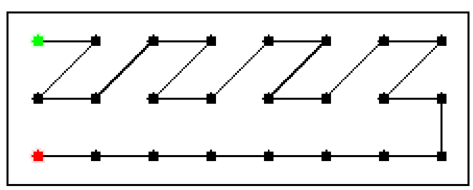


can be represented by the L-System string

+ffFF+F+F–F–F+F+F–F–F+F–F–F+FF+FFFFFF

The initial +ff gets to the first location on the tour.

The Moore tour shown earlier



can be represented by

F+f+Ff–F+f+Ff–F+f+F+f–F+f+F++F+
fFFFFFF

Again, diagonal moves can be represented in different ways. For example, f–F produces the same result as –fF.

An advantage of using the L-System method is that there is a large body of knowledge associated with them. And this material includes some interesting tours.