Behavior Research Methods & Instrumentation
1978, Vol. 10 (2), 170-176

Computer generation of symmetric, semisymmetric, antisymmetric, and asymmetric patterns

FRED L. ROYER
Veterans Administration Hospital (Brecksville Division), Cleveland, Ohio 44106

A computer algorithm generates selected symmetric, semisymmetric, antisymmetric, and asymmetric patterns on a square field, using random number generation as the seed. All possible combinations of the geometric operations of 90-deg rotation and reflection have exactly equivalent operations on the "bytes" and "words" that make up the binary descriptions of the horizontal and vertical arrangements of visual information. Dot, checkerboard, oblique line, orthogonal line segment, and other more complex pattern types can be generated using this algorithm.

The study of the effects of symmetry on pattern perception has a venerable history dating back to Mach (see Goldmeier, 1972). In recent years, as information-processing paradigms have been used increasingly in the study of perceptual and cognitive problems, there has been a resurgence of interest in symmetry. As stimulus displays become more complex, the importance of the effects of various kinds of stimulus structure, such as symmetry, becomes more evident.

Many studies have addressed themselves to the symmetry of patterns per se or to a related conceptualization of redundancy, rotation and reflection equivalence set size (Garner, 1962, 1974). Symmetry makes patterns easier to recall, recognize, and discriminate (Garner, 1962). Redundancy based on the identity of patterns under reflection and 90-deg rotation operations affects goodness of form, encoding time, backward maskability and masking efficiency, latency of pattern associates, and discrimination time (Bell & Handel, 1976; Garner, 1974; Garner & Sutliff, 1974). Royer and his associates (Royer, 1971a, 1971b, 1977; Royer & Friedman, 1973; Royer & Holland, 1975a; Royer & Weitzel, 1977) demonstrated the effect of this measure of pattern redundancy in performance in intelligence test tasks and in clinical use (Royer & Holland, 1975b).

Although early studies used very simple dot and line patterns, more recent work has required patterns of increasing complexity. These recent studies show that symmetry is an important attribute of a pattern, which interacts with other stimulus attributes that affect perceived pattern complexity (Chipman, 1977), pattern analytic processes (Royer & Weitzel, 1977), and symmetry judgment (Julesz, 1971; Royer, Note 1). As the size of arrays, configurational complexity, and the need to study interactions of symmetry and other stimulus attributes increase, so does the need for rapid, random generation of patterns with the desired symmetrical properties.

The purpose of this paper is to present a simple algorithm that permits the generation of patterns with specified symmetrical properties. Although there are alternative approaches to the mathematics of symmetry in two-dimensional spaces, the present paper is based on the work of Prokhovnik (1959). We do this for two reasons: (1) Prokhovnik's work provides a logical link between the work flowing from Garner's set theoretic approach to pattern redundancy and the work of others studying symmetry directly, and (2) Prokhovnik provides general combinatorial solutions for determining the number of different patterns which can be constructed from a particular proportion of binary alternatives and which have a particular symmetry (as well as proving which patterns cannot be constructed given certain stimulus parameters).

EQUIVALENCE SETS AND GROUPS

Any total set of binary patterns on a square field can be partitioned into subsets so that all elements of the subset are equivalent under operations of reflection and 90-deg rotation. These subsets, or rotation and reflection equivalence sets, vary in size (i.e., the number of elements in the subset): 1, 2, 4, or 8. Each size taken with respect to the size of the total set constitutes a measure of redundancy.

The algorithm described herein distinguishes among various types of symmetries rather than distinguishing rotation and reflection equivalence sets because, as Prokhovnik (1959) showed, patterns with equal equivalence set size may have different symmetrical properties. Such patterns can differentially affect performance

Requests for reprints should be sent to Fred L. Royer, PhD, VA Hospital, Psychophysiological Research Laboratory (Brecksville Division), 10701 East Boulevard, Cleveland, Ohio 44106. This work was performed as part of VA Research Project 7600-04.

(Royer, Note 1, Note 2). Patterns can be classified into groups of different orders determined by the group of operations which leaves the pattern unchanged. In the set theory approach, a set of operations on a pattern generates all other members of the equivalence set. In the group theory approach, a group of operations on a pattern under which the pattern is invariant determines the group order of the pattern. The operations are: I, identity; R, 90-deg rotation; R^2, 180-deg rotation; R^3, 270-deg rotation; S, reflection; SR, reflection and 90-deg rotation; SR^2, reflection and 180-deg rotation; and SR^3, reflection and 270-deg rotation.

There are four groups of operations characterized by Prokhovnik (1959) as G_8, symmetric; G_4, semisymmetric; G_2, antisymmetric; and G_1, asymmetric. The subscript identifies the order of the group. The equivalence set size of patterns in the orders are 1, 2, 4, and 8, respectively. These groups are defined in Table 2. Prokhovnik showed that there are three subgroups of Order 4 and five subgroups of Order 2; because of identical configurations in G_{22}-G_{23} and G_{24}-G_{25}, the five subgroups of Order 2 are treated as three.

Illustrations of patterns of each group are shown in Figure 1. Each group is characterized by a particular type of symmetry. G_8 patterns have symmetry on horizontal, vertical, and both diagonal axes, and they have centric symmetry as well. G_{41} patterns have 90-deg centric symmetry, in which the information is repeated every 90 deg around the central pivot. G_{42} patterns have symmetry on both the horizontal and vertical axes. G_{43} patterns have symmetry on both diagonal axes. G_{21} patterns have 180-deg centric symmetry; information is repeated every 180 deg around the central pivot. G_{22} patterns have symmetry around the vertical axis, and G_{23} patterns have symmetry around the horizontal axis. G_{24} patterns have symmetry around the diagonal of positive and G_{25} patterns around the diagonal of negative gradient.

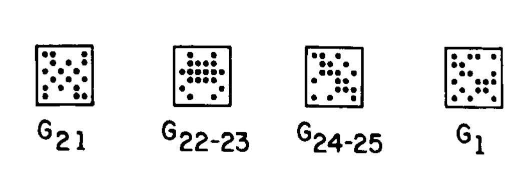

Figure 1. Examples of pattern symmetries: G_8, symmetric; G_4, semisymmetric; G_2, antisymmetric; and G_1, asymmetric.

THE ALGORITHM

Background Concepts

Any visual pattern composed of two alternative elements in a square field of n by n dimension can be described by a binary pattern. Such a pattern has two descriptions: the horizontal ordering of alternatives and the vertical ordering of alternatives.

The basic concept of the algorithm is that the identity of the description of a binary pattern is preserved in certain reverse orderings of the description—in particular, certain reflections of the description, in which the operation of reordering is exactly equivalent to the geometric operations of reflection and rotation. The binary descriptions are designated herein by the familiar computer terms "words" and "bytes." These will be used in their general sense and not in the restricted sense of a standard bit length.

By convention, the full description of the horizontal arrangements of alternatives is, for our purposes, designated as a word, H, having a length of n by n bits, where n is the dimension of the square field or matrix. A similar description of the vertical arrangements is the V word, having equal length. By convention, the word, for our purposes, is composed of a number of *bytes*, each byte having a length of n bits. Thus, each word contains n bytes, each of which has a length of n bits. The bytes of the H word correspond to the rows of the field; the bytes of the V word correspond to the columns of the field. For example, the G_8 pattern shown in Figure 1 is a 6 by 6 pattern. The words, composed of six 6-bit bytes, are

$$H = 110011\ 101101\ 011110\ 011110\ 101101\ 110011$$

and

$$V = 110011\ 101101\ 011110\ 011110\ 101101\ 110011.$$

The algorithm is based completely on performing a reflection operation, in which the order of the binary elements is reversed. A word may be reflected, its bytes may be reflected, or its bytes may be reflected followed by a reflection of the word. These operations are symbolized in Table 1. Symbols using V apply to operations on the V word.

If a reflection operation is performed on the H or V word and the word remains unchanged, the operation is directly equivalent to identity under the operation of reflection or rotation of the pattern. We have expressed the appropriate operations in terms of logical equalities in Table 2. Thus, for example, the expression H=HH means that reflection of the H word produces a binary pattern that is identical to the H word. In Table 2 are all the appropriate operations that produce patterns of a particular group or subgroup.

There are certain characteristics of words and bytes,

Table 1
Computer Operations and Their Symbols

Operation	Symbol	Binary Description
Identity H	H	010 110 001
Reflection of H	HH	100 011 010
Reflection of bytes of H	H×H	010 011 100
Reflection of H with bytes reflected	H(H×H)H	001 110 010

related to the reflection operations on descriptions, that are useful and convenient for understanding the generating algorithm. To satisfy the test that H=HH or V=VV, a word must be symmetrical. To satisfy the test that H=HXH or V=VXV, the bytes of the word must be symmetrical. To satisfy the test that H=H(HXH)H or V=V(VXV)V, the bytes, whether asymmetrical or not, must be symmetrically ordered. That is, the first asymmetrical byte is placed without reflection in the last byte, the second in the next to last, and so on. It is thus possible to categorize all symmetry group and subgroup patterns according to (1) whether they are composed of asymmetric or symmetric bytes and (2) whether the bytes are asymmetrically or symmetrically placed within the word. The generation of patterns, then, can be accomplished through appropriate sequencing of subroutines that build (1) asymmetrical or symmetrical bytes and (2) asymmetrical or symmetrical words by arranging the appropriate bytes in the proper order within the word.

Generating Patterns

The algorithm implements the set of equations listed in Table 2 for each group or subgroup. Subroutines generate a symmetric byte, an asymmetric byte, or either depending on the needs of the algorithm. Since patterns with asymmetric bytes need only a single asymmetric byte to satisfy tests of Table 2, random generation requires that either symmetric or asymmetric bytes be generated, subject to the restriction that at least one asymmetric byte is mandatory. Since there are 2^q symmetrical bytes, where $q=n/2$ or $q[(n-1)/2]+1$, depending on whether n is even or odd, respectively, and since there are $2^n - 2^q$ asymmetrical bytes, the probability of generating an asymmetric byte grows very rapidly as the field size increases. In practice, it is necessary to include a step which ensures generation of at least one asymmetric byte only when the pattern's field is very small.

If a symmetric byte is required, a subroutine randomly generates a number between 0 and $2^p - 1$, where $p = n/2$ or $p = (n - 1)/2$. This half byte is reflected in another subroutine, and the full byte is constructed with a randomly selected central bit if n is

Table 2
Geometric and Computer Reflection Operations

R & R ESS	Group	Identity Operations	Description Equalities	Symmetry Description
1	G_8	I	H=H, V=V	Horizontal
		R	H=V(V×V)V, V=H×H	Vertical
		R^2	H=HH, V=VV	Positive Diagonal
		R^3	H=V×V, V=H(H×H)H	Negative Diagonal
		S	H=H×H, V=V(V×V)V	Centric
		SR	H=VV, V=HH	
		SR^2	H=H(H×H)H, V=V×V	
		SR^3	H=V, V=H	

2	G_{41}	I, R, R^2, R^3	H=HH=V×V=V(V×V)V
			V=VV=H×H=H(H×H)H
				Centric 90 deg

2	G_{42}	I, S, SR^2, R^2	H=HH=H×H=H(H×H)H
			V=VV=V×V=V(V×V)V
				Horizontal
				Vertical

2	G_{43}	I, SR, R^2, SR^3	H=HH=V=VV
			V=VV=H=HH
				Positive Diagonal
				Negative Diagonal

4	G_{21}	I, R^2	H=HH
			V=VV
				Centric 180 deg

4	G_{22}	I, S	H=H×H
			V=V(V×V)V
				Vertical

4	G_{23}	I, SR^2	H=H(H×H)H
			V=V×V
				Horizontal

4	G_{24}	I, SR	H=VV
			V=HH
				Positive Diagonal

4	G_{25}	I, SR^3	H=V
			V=H
				Negative Diagonal

8	G_1	I	H=H
			V=V
				Asymmetric

odd. If a mandatory asymmetric byte is requested, the same procedure is repeated, except that instead of reflecting the half byte, a different half byte is randomly generated and the full byte is constructed. If either a symmetric or asymmetric byte is in the call to the subroutine, a number between 0 and $2^n - 1$ is randomly generated.

If a symmetric word (H=HH or V=VV) is requested, the i^{th} byte is reflected and placed in the j^{th} position in the word, where i is the index of successive bytes, ranging from 1 to n/2 or (n – 1)/2, and $j = n - i + 1$. The bytes of an asymmetric word are compiled serially without reflection. If any equality involves both the H and V words, for example, H=V, the descriptions are mutually constraining. Therefore, if a byte is selected for H and is placed in V, certain bits in V then limit the possible bytes which can be generated in the next position. Subroutines are used to map the descriptions from one word to another. When the mapping is done, the bit width of the next byte must be reduced appropriately.

Generality of the Algorithm

The algorithm can generate many different types of patterns. It is appropriate for any type of "counter" which can occupy a cell of an n by n field, whether dot, filled cell, color, or some other attribute, other cells being either unoccupied or occupied with a different alternative. In fact, the alternatives do not have to be binary; the only requirement is that the "counter" not be perceptibly different under rotational or reflection operations unless certain transformations are accomplished during the compiling of the word. It is possible to extend the algorithm to "counters" which require transformations to preserve symmetry relations; this is illustrated in Figure 2. The alternatives here are the cells with 45-deg diagonals. Symmetry exists when the two halves of a byte are complimentary. Therefore, the appropriate bytes and words are generated first and the appropriate complimentary transformation is then applied.

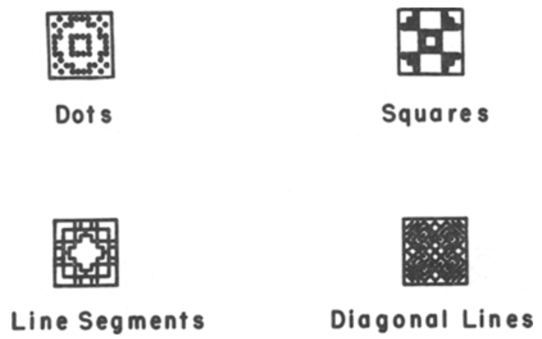

Figure 2. Examples of different types of G_8 patterns generated by simple adaptations of a common algorithm.

The algorithm can be extended to generate patterns in which two different binary alternatives can occupy the same space. I have used such patterns in studies of the block design task (Royer, 1977; Royer & Weitzel, 1977). One set of binary alternatives is the direction of the diagonals; the other set is the part of the cell (above and below the diagonal) that is filled. The set of diagonal alternatives requires byte complimentation; the other requires complimentation of the word.

The algorithm can be extended to generate patterns on the grid instead of within the cells of an n by n field. The bit length of a description is n – 1. Vertical and horizontal information are independent, not completely redundant, as in the case of patterns consisting of "counters" in cells. The algorithm applies to the construction of both the H and V words as before. In case of dot, squares, and diagonal line patterns, it is sufficient to satisfy either of the two equalities listed in Table 2 in constructing the pattern because the H and V words are redundant. In the case of line segments, since the H and V words are independent, both equalities must be met in constructing the pattern. An example of such line segment patterns is shown in Figure 2.

In practice, the appropriate binary words are constructed and drawing programs are called to construct particular types of patterns. For patterns that are redundant in horizontal and vertical descriptions, only the horizontal word (H) is used to draw the pattern. Both H and V words are used for patterns having independent horizontal and vertical descriptions.

ALGORITHM

The algorithm for constructing patterns of particular groups and subgroups composed by placing "counters" in cells is presented first below. The details of modification of the algorithm to generate patterns on the grids is presented later. There is a hierarchical structure in the relation of pattern groups and subgroups, with two orders of patterns differing by only a single restriction. Since the patterns are randomly generated, it is possible to generate a pattern of a higher order (smaller rotation and reflection equivalence set size) by chance. Appropriate checks and subroutines to alter certain bytes may be used to ensure that the pattern is in the desired Prokhovnik class. Of course, the probability of this decreases as the size of the field increases.

Throughout this presentation, n is the size of the field, i is the index of the byte or bit, $j = n - i + 1$ is the index of the reflected byte or bit, and p is n/2 or (n – 1)/2 for even and odd ns, respectively.

Patterns With Symmetrical Words Composed of Symmetrical Bytes

Symmetric G_8 patterns. These patterns are characterized by combined horizontal, vertical, diagonal, and centric symmetry. In addition to having symmetric bytes and words, the H and V words are identical. One example is an X-like design on a 3 by 3 matrix: H=101 010 101, V=101 010 101. To generate these patterns:

1. Generate a symmetric byte for the first byte in the H word.

2. Since the word is symmetrical, place the same byte in the last of n bytes in the H word, that is, the nth byte.

3. Since the V word is identical to the H word, place these bytes in corresponding positions of the V word.

4. Map the resultant H and V words onto one another according to the relation $V_{ji} = H_{ij}$. (Since H = V, the relation $H_{ij} = H_{ji}$ can be used directly to accomplish the mapping. This simplifies the procedure.)

5. The mapping has reduced the choices available in selecting the next byte by determining parts of other bytes. A symmetric word is generated by now choosing a number between 0 and $2^{p-i+1} - 1$, where i is the index of the byte being constructed.

6. Place this selection in the ith and the jth byte to maintain word symmetry, preserving the already determined bit(s) resulting from the mappings of H and V.

7. Continue Steps 3 through 6 for each ith byte and each (n – i)th byte until i = p. With each new byte, the choice of the half byte is reduced by an additional bit because of the mapping. It is clear that the last step for odd matrices will be the choice of a single bit.

Semisymmetric G_{42} patterns. These patterns are characterized by symmetry on both the horizontal and vertical axes but, in contrast to G_8, not on the diagonal axes. An example of the pattern is an H-like figure on a 3 by 3 matrix. The descriptions are: H=101 111 101, V=111 010 111. To generate the G_{42} patterns:

1. Perform Steps 1 and 2 listed above for constructing G_8 patterns.

2. Continue generation of symmetric bytes in the H word, placing the ith byte also in the jth byte until i = p or i = p + 1 if n is even or odd, respectively.

3. When the word is completed, V is composed from H. If $H \neq V$, an appropriate pattern has been generated. Note that the only difference in procedure from that used to construct G_8 is the lack of mapping H onto V, hence random generation of G_8 is possible in small fields.

Patterns With Symmetrical Words Containing at Least One Asymmetrical Byte

There are several pattern classes having these characteristics. They have either diagonal or centric symmetry. The first restriction is that in odd matrices the choice of the middle byte, H_i, $i = [(n - 1)/2] + 1$, must always be a symmetric byte to preserve the symmetry of the word. In the following it will be assumed that this choice will always be the last one for odd matrices and that step will be omitted from discussion.

Semisymmetric G_{41} patterns. These patterns are characterized by centric symmetry. If the pattern is divided into quadrants (ignoring the middle row and column of odd matrices), the pattern in each successive quadrant moving clockwise is a 90-deg clockwise rotation of the preceding quadrant. The patterns exist only for matrices in which $n > 3$. Such symmetry is also present in G_8 patterns, but G_{41} patterns have neither diagonal or horizontal-vertical symmetry. The characteristics that generate this pattern, according to the logical tests presented earlier, are that each byte of the H word must be reflected in the corresponding byte of the V word and that both H and V are symmetrical.

A restriction must be observed in selections of bytes for generating the pattern. A successive 90-deg rotation of a quadrant about a central point results in the repeat of all diagonal elements in the quadrant. As bytes (h) are generated for the word (H), bits in the diagonal positions must be equal. Let i = 1, 2 . . . p index the byte being generated and j = 1, 2 . . . p and k = n – i + 1, . . . p + 1 index the bits in the byte, then $h_j = h_k$ for all j and k (in odd matrices the symmetry of the middle byte assures that the restriction is not violated). An example of this class of pattern is a swastika-like design on a 5 by 5 matrix:

H = 11101 00101 11111 10111 10111

V = 10111 10111 11111 00101 11101.

The way to generate the G_{41} pattern is:

1. If n = 4, generate an asymmetric byte. If n > 4, generate either an asymmetric or symmetric one for the first byte of H.

2. Test whether the ith byte has the same element in the ith and jth bits. If not, randomly alter the byte at one of the proper bit positions.

3. Construct reflection of byte. Place in last byte of H to create symmetry. Also place in first byte of V. To create symmetry in V, place the first byte of H in the last byte of V.

4. Map H_{ij} to V_{ji} and V_{ij} to H_{ji} ($H_{ji} = H_{ij}$). Generate the next byte by reducing the length of the half byte by 2k, since mapping has now determined the first and last bits of the byte. Place the resultant symmetric or asymmetric byte into H, preserving the already determined bits.

5. Continue Steps 2 to 4 until i = p.

6. At least one asymmetric byte must be chosen before the final step.

Antisymmetric G_{21} patterns. These patterns are characterized by centric symmetry; however, instead of a single subpattern in a quadrant being repeated in 90-deg rotation, there are two different subpatterns repeated in 180-deg rotations, so that each subpattern and its 180-deg rotations are located in diagonally opposite quadrants. Since identity under a 180-deg rotation defines the group, only the symmetry of the descriptive word is required.

If $n > 3$, either symmetric or asymmetric bytes may be chosen, providing at least one asymmetric byte is present. The restriction on choice of symmetric bytes that applies to G_{41} patterns applies to G_{21} as well. There are no restrictions on the bits representing the diagonals of the matrix. As an example, suppose the pattern is one-half of a swastika-like design on a 3 by 3 matrix; the description would be H=110 010 011, V=100 111 001. To generate these patterns:

1. If $n = 3$, generate an asymmetric byte. If $n > 3$, generate either an asymmetric or symmetric byte.
2. Construct reflection of the byte. Place in last byte to create symmetry of H.
3. For $n > 3$, repeat Steps 1 through 2 until $i = p$.
4. By chance it is possible to generate a G_{41} pattern. Check and correct, if so.

Semisymmetric G_{43} patterns. The characteristics of these patterns are symmetry on both diagonals and an absence of centric or horizontal-vertical symmetry. In addition to word and byte characteristics, H and V are identical. To generate the pattern:

1. If $n = 3$, generate an asymmetric byte. If $n > 3$, generate either an asymmetric or symmetric byte for the first byte of H.
2. To preserve symmetry, construct the reflection of the byte and place in last byte of H and V.
3. Since the H and V descriptions are identical, map H_{ij} to V_{ji} and V_{ij} to H_{ji} ($H_{ji} = H_{ij}$).
4. Generate next byte by reducing length of byte by 2k, since mapping has now determined the first and last bits of the byte. Place the resultant symmetric or asymmetric bytes into H, preserving the already determined bits.
5. Continue Steps 2 to 4 until $i = p$. At least one asymmetric byte must be chosen in the process before the last step is completed.

Asymmetric Words Composed of Symmetric Bytes

Antisymmetric G_{22}-G_{23} patterns. The patterns are characterized by symmetry on either the vertical or horizontal axis but not both. Symmetry on one of these axes is present when all bytes are symmetrical. Therefore, all bytes of either the H or V description must be symmetrical, although the word will not be. As an example, the descriptions of a U-like pattern on a 3 by 3 matrix are: H=101 101 111, V=111 001 111.

The patterns are generated by the following steps:

1. Successively generate symmetrical bytes in H until $i = n$.
2. This procedure always produces a pattern with vertical symmetry (G_{22}). To produce a pattern with horizontal symmetry (G_{23}), construct V first and then compose H from it or compose H, set V = H, and then reconstruct H from V prior to drawing.

Patterns With Asymmetric Words Composed of Asymmetric or Symmetric and Asymmetric Bytes

Antisymmetric G_{24}-G_{25} patterns. These patterns are characterized by symmetry on one diagonal. One subgroup, G_{24}, has symmetry on the diagonal with positive gradient. The H and V descriptions are reflections of each other. As an example, the description of an L pattern on a 3 by 3 matrix would be H=100 100 111, V=111 001 001. The other subgroup, G_{25}, has symmetry on the diagonal with negative gradient. For example, the description of gamma-like design on a 4 by 4 matrix is H=1111 1000 1000 1000, V=1111 1000 1000 1000. In addition to the word and byte characteristics, the H and V descriptions are identical to each other. If the matrix is odd, the byte for the middle, $[(n - 1)/2] + 1$, position must be symmetrical. It is assumed that the step of selecting a symmetrical byte for this position will be the last step. At least one asymmetric byte must occur in the pattern which, of course, will generate other asymmetric bytes when placed in the appropriate byte in V with V mapped back to H again.

The steps for generating the pattern are:

1. If $n = 3$, generate an asymmetric byte. If $n > 3$, generate a symmetric or asymmetric byte. Place the byte in the first position of H.
2. Depending on whether the pattern is to be symmetrical on the diagonal axis with positive or negative gradient: (a) If the negative gradient is selected, place the byte in the first byte of V. (b) If the positive gradient is selected, reflect the byte and place in the last byte of V.
3. Map H_{ij} to V_{ji} and V_{ij} to H_{ji} ($H_{ji} = H_{ij}$).
4. Generate the next byte by reducing length of the byte by one (or k) bit ($k = j$), since mapping has now determined the first (or i^{th}) bit of the byte. Place the resultant symmetric or asymmetric byte into H, preserving the already determined bits.
5. Repeat Steps 2 to 4 until $i = p$, ensuring that at least one asymmetric byte has been selected.
6. By chance, a G_{43} pattern may be generated. Check and correct if necessary.

Asymmetric G_1 patterns. Patterns of this type have no symmetry of any type; therefore, their descriptions are asymmetrical and composed of asymmetrical or symmetric and asymmetric bytes. Both H and V descriptions are different but are not symmetric ($H \neq HH$) and are not reflections of each other ($H \neq VV$); neither are reflections of the bytes of one the same as those of the other ($H \neq VXV$, $V \neq HXH$). As an example, the description of an F-like pattern on a 5 by 5 matrix would be H=11111 10000 11111 10000, V=11111 10100 10100 10100 10100.

These patterns are generated as follows:

1. Successively generate asymmetric or symmetric bytes until $i = n$, so that at least one byte is asymmetric.
2. By chance, a pattern of any of the other groups

may be generated. Check and correct if necessary.

Generating Line Segment Patterns on an n by n Grid

The above algorithm may be extended to generating line segment patterns on a square-grid system rather simply. First, byte length is n – 1 bits and the word is composed of n bytes. The alternatives are the presence or absence of lines on the grid. Since the alternatives on the vertical grid lines are independent of the alternatives on the horizontal grid lines, both V and H words must be constructed and drawn separately. While only one of the sets of equalities for H and V words shown in Table 1 needed to be satisfied with patterns generated on cells, both sets of equalities must apply to patterns generated on grids. To generate patterns in which H = V, that is, G_8, G_{43}, G_{25}, or in which V = HH, V = HXH, of V = H(HXH)H, that is, G_{41}, G_{43}, G_{24}, no extensions of the algorithm are necessary. The two descriptions are mutually constraining. Extensions of the algorithm are necessary, then, only for patterns G_{42}, G_{21}, G_{22}, and G_{23}. Since the same type of logical equalities are required for both the V and the H words in G_{42} and G_{21}, the steps used to generate H are repeated exactly to generate V. In the case of G_{22} and G_{23} patterns, an additional subroutine is needed to select bytes so that at least one is asymmetric and to reposition the bytes symmetrically but without reflection [H(HXH)H or V(VXV)V operations] in the appropriate word.

Testing Generated Patterns

As is evident in the discussion above, it is possible to generate a pattern of higher order than that which is desired, simply because of random generating processes. One needs to test for this possibility with small fields. The probability of generating a pattern other than the one desired diminishes very rapidly, however, as the field size increases. For example, if the construction rule requires at least one asymmetric byte, the probability of selecting a symmetric byte when the asymmetric one is desired is .5 for n = 3, .25 for n = 5, .125 for n = 7, and .06 for n = 9. The probability of constructing a symmetric word when an asymmetric one is desired (as would be the case for G_1), for example, is .06 for a 3 by 3 matrix, .008 for a 4 by 4, .0002 for a 5 by 5, and $.05 \times 10^{-8}$ for an 8 by 8 matrix. Clearly, it becomes a waste of time to make exhaustive checks of patterns of very large size; it becomes more economical to generate additional patterns.

IMPLEMENTATION OF THE ALGORITHM

The algorithm can be implemented most easily in a system with a high-level language by using doubly subscripted arrays, each element of which has a value of zero or one. The number returned by a random number function is converted to binary and each digit of the binary representation is placed in the corresponding element of the array. If implemented by machine language, reflection of the generated random number's bit pattern can be accomplished by logic-shift instructions into the carry bit and out of the carry bit into the appropriate register through rotate instructions.

REFERENCE NOTES

1. Royer, F. L. *Detection of symmetry in Prokhovnik patterns.* Unpublished manuscript.

2. Royer, F. L. *Stimulus variables affecting performance in the block design task.* Unpublished manuscript.

REFERENCES

Bell, H. H., & Handel, S. The role of pattern goodness in the reproduction of backward masked patterns. *Journal of Experimental Psychology: Human Perception and Performance* 1976, **2**, 139-150.

Chipman, S. F. Complexity and structure in visual patterns. *Journal of Experimental Psychology: General*, 1977, **106**, 269-301.

Garner, W. R. *Uncertainty and structure as psychological concepts.* New York: Wiley, 1962.

Garner, W. R. *The processing of information and structure.* New York: Halsted Press, 1974.

Garner, W. R., & Sutliff, D. The effect of goodness on encoding time in visual pattern discrimination. *Perception & Psychophysics*, 1974, **16**, 426-430.

Goldmeier, E. Similarity in visually perceived forms. *Psychological Issues*, 1972, **8**, Monograph 28.

Julesz, B. *Foundations of cyclopean perception.* Chicago: University of Chicago Press, 1971.

Prokhovnik, S. J. Pattern variants on a square field. *Psychometrika*, 1959, **24**, 329-341.

Royer, F. L. Information processing of visual figures in the digit symbol substitution task. *Journal of Experimental Psychology*, 1971, **87**, 335-342. (a)

Royer, F. L. Spatial orientational and figural information in free recall of visual forms. *Journal of Experimental Psychology*, 1971, **91**, 326-332. (b)

Royer, F. L., & Freidman, S. Scanning time of schizophrenics and normals for visual designs. *Journal of Abnormal Psychology*, 1973, **82**, 212-219.

Royer, F. L., & Holland, T. R. Rotations of visual designs in psychopathological groups. *Journal of Consulting and Clinical Psychology*, 1975, **43**, 546-556. (a)

Royer, F. L., & Holland, T. R. Rotational transformation of visual figures as a clinical phenomenon. *Psychological Bulletin*, 1975, **82**, 843-868. (b)

Royer, F. L., & Weitzel, K. Effect of perceptual cohesiveness on pattern recoding in the block design task. *Perception & Psychophysics*, 1977, **21**, 39-46.

Royer, F. L. Information processing in the block design task. *Intelligence*, 1977, **1**, 32-50.