

# Image Permutations

In a contribution to Complex Weavers' Mathematics and Textiles Study Group, William J. Jones described a method of permuting images to obtain new images. He called his method MultiReduce.

A permutation (rearrangement) can be described by assigning successive numbers to the objects to be permuted and then giving the order of the objects after the permutation is applied. For example, if there are eight objects

1 2 3 4 5 6 7 8

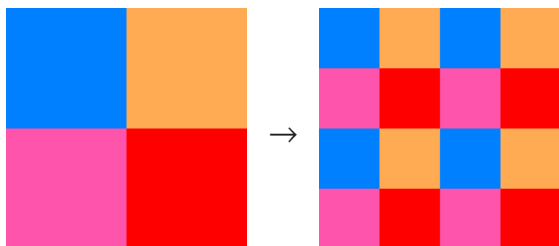
the permutation

8 7 6 5 4 3 2 1

reverses the order of the objects.

MultiReduce uses permutations of the columns and rows of an image to "quarter" it. First all the even-numbered columns are moved to the left half, while all the odd-numbered columns are moved to the right half. Then the same thing is done to the rows. This process is then repeated.

Here's an example image and the result of applying MultiReduce to it:

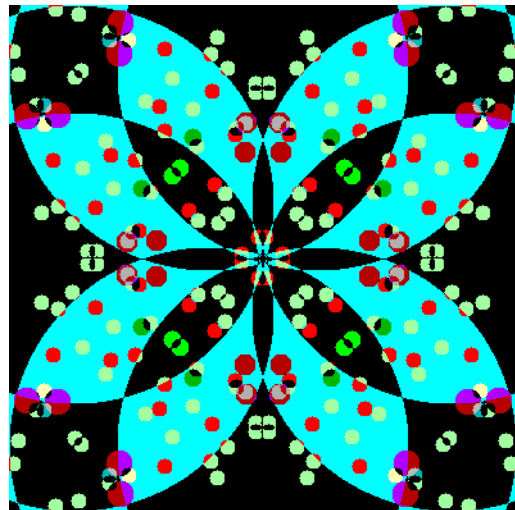


## Choosing Images for MultiReduce

Jones remarked that simple images generally produce more interesting results than complex ones. That certainly is true, although there are exceptions.

Generally speaking, applying MultiReduce to photo-realistic images with a many of different colors do not produce interesting results. What seems to matter most is the degree of organization in the image — a concept that eludes precise definition but nonetheless is understandable. For example, images composed completely at random, which

have little aesthetic appeal, produce more of their kind. (If they don't, the original image wasn't really random.) On the other hand, the application of MultiReduce to symmetric images often produces attractive results. This is due, at least in part, to the fact that MultiReduce preserves symmetry to some extent. Here's an example:



Almost all MultiReduce permutations of this image are attractive — for example, they would make interesting decorative tiles.

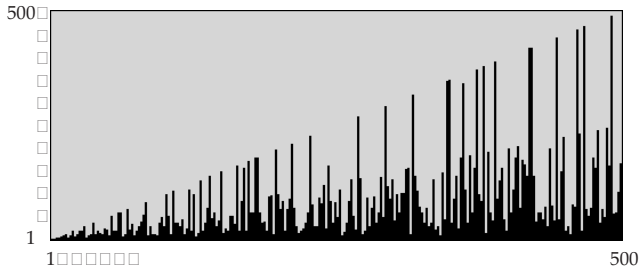
Large images tend to produce more interesting results under MultiReduce than small images. This is due in large part to the fact that large images disintegrate into pixel dust less quickly than small images.

By the nature of MultiReduce, its application to images consisting of vertical or horizontal stripes produces vertical or horizontal stripes, respectively, usually with many interesting variations on the original images. Checks, plaids, and so forth also are good candidates.

When the same permutation is applied to an image over and over, the original image eventually is produced. The number of permutations required to accomplish this is called its period. The size of an image strongly affects the period. If the period is very large, it may be impractical to get all the images, and sometimes the most interesting images occur near the end of the period. If the period is small, on the other hand, there may not be enough im-

ages to be worthwhile.

For square images, the period for MultiReduce increases, on average, with the size, but by no means in a regular fashion. For example, the period for a  $100 \times 100$  pixel image is 30, but for a  $128 \times 128$  pixel image, it is 7. Here's a histogram of MultiReduce periods for square images of sizes 1 to 500:

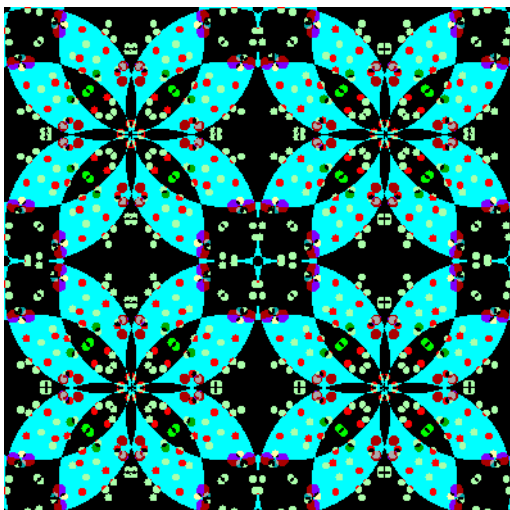


For images that are not square, the period, which is the least common multiple of the vertical and horizontal periods, can be very large. For example, the period for an  $80 \times 84$  image is 3,198.

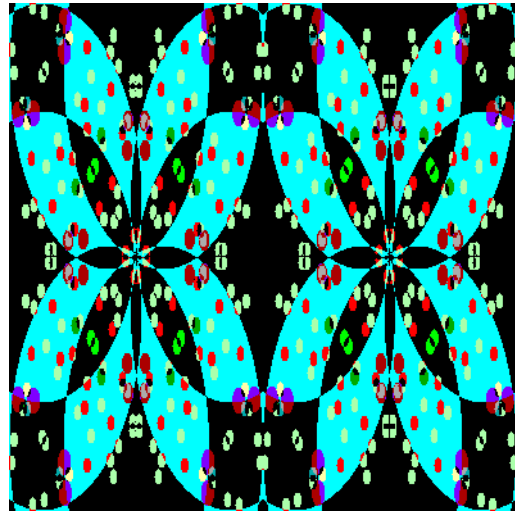
### Variations on MultiReduce

The basic MultiReduce permutation is  $2 \times 2$ ; that is, it divides the image in half, horizontal and vertically on each application. Other divisions, such as  $1 \times 2$ ,  $2 \times 3$ ,  $4 \times 4$  and so on, produce different images that are no less interesting.

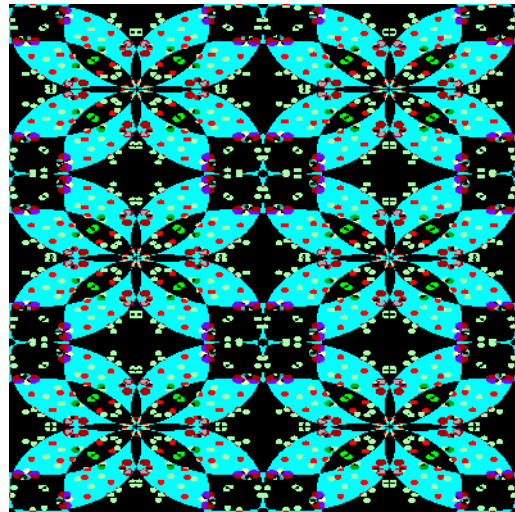
Here are some examples of the first permutation of the symmetric image shown earlier for different variations on MultiReduce:



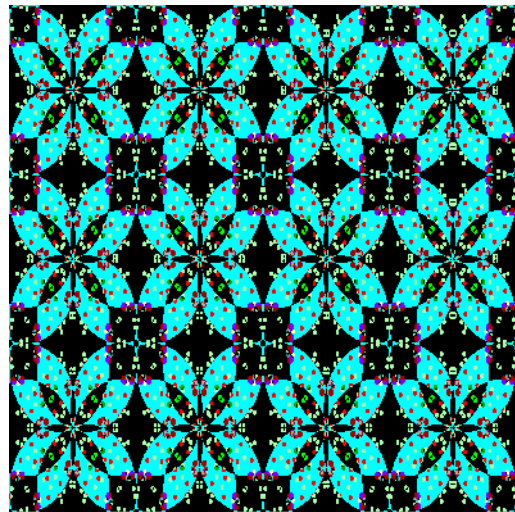
$2 \times 2$



$1 \times 2$



$2 \times 3$



$4 \times 3$

## MultiReduce and Weavable Images

Permutations preserve loom-controlled weavability as described in Reference 2. That is, if an image is weavable, any permutation of it is also. Conversely, if an image is not weavable, then neither is any permutation of it.

A permutation of a weavable image requires the same loom resources — shafts and treadles — as the original image.

Permutations can be viewed as pattern generators and hence as tools for weave design.

Here are some examples of MultiReduce applied to a weavable image that is used as a border for Web pages related to weaving [3]. The original image is at the upper left.



## Other Permutations

The number of different permutations of  $n$  objects becomes astronomical as  $n$  gets large. To make any sense of it, it's necessary to focus on *kinds* of permutations. Some, like reversal and rotation, may be useful for some purposes but not produce much in the way of variety.

One kind of permutation with some promise permutes blocks of pixels while leaving the pixels within a block unchanged. For example, a 32-column image might be divided into 4 blocks of 8 columns each, labeled A, B, C, and D:

A	B	C	D
1 2 ... 7 8	9 10 ... 15 16	17 18 ... 23 24	25 26 ... 31 32

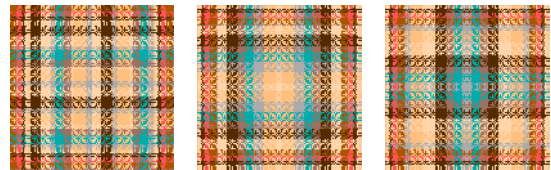
These blocks then might have the permutation B D C A, putting columns 9 through 16 first, followed by columns 25 through 32, and so on.

The blocks, of course, need not be of the

same length and the number of blocks may vary.

Such block permutations, involving only a few “elements”, have short periods. The best results for this kind of permutation occur for images that are themselves “blocky” or at least rectilinear.

Here are some block permutations applied to the image shown in the last section:



An extension of this idea would be to specify (nontrivial) permutations within the blocks.

## Mutations

Permutations rearrange objects without deleting or duplicating any. A more general form of “mutation” allows deletion and/or duplication of objects as well. The permutation notation extends naturally to mutations. For example, suppose there are 9 columns in an image. Then the mutation

9 8 7 6 6 5 4 3 2

reverses the order of the columns, while duplicating 6 and discarding 1. The mutation

1 1 2 2 3 3 4 4 5 5 6 6 7 7 8 8 9 9

magnifies the image by 2, while

1 3 5 7 9

deletes the even-numbered columns.

Mutations preserve weavability. Can you think of a use for them in weave design?

## Animated Permutations on the Web

Most permutations have periods that are too long to allow all the images to be shown on a printed page.

A way to show the results for the full period is to bundle the images from a permutation as an animation. There are a few examples

at:

<http://www.cs.arizona.edu/patterns/weaving/mtsg/Animations/>

### Speaking of the Web ...

One of the problems with permutations is how can you actually use them in weaving.

Most permutations are too complicated to do by hand. If you are a programmer, you may be able to write a program for producing permutations. But then how do you get them into, say, a weaving program? Most weaving programs have very limited facilities for importing data. The one method most weaving programs support is the importation of WIFs [4].

A CGI application could take specifications of a permutation and produce a WIF—or at least WIF sections for threading and treadling sequences. Any takers?

### Yet More Ideas

Here are some ideas for further work:

- Investigate the inverses of MultiReduce permutations. The first result of an inverse applied to an image would be the next-to-last result of the MultiReduce permutation — which provides an easy way to get to the images near the end of the MultiReduce period.
- Apply permutations directly to threading and treadling sequences. This doesn't require as much mechanism as image manipulation.
- Add an element of randomness into the creation of permutations.
- Apply different permutations in succession rather than using the same one repeatedly.
- Investigate applications of the kinds of permutations used in change ringing as described in a recent article in *Handwoven* [5].
- Design a language for describing/implementing the construction of permutations. Now *there's* a challenge.

### Acknowledgment

I am indebted to William J. Jones for the ideas that led to these explorations and for subsequent e-mail exchanges about the subject.

### References

1. "Iteration with Permutations: A 'Magic' Filter", William J. Jones, *Mathematics and Textiles Study Group Newsletter* 3, October 2000.
2. *Weavable Color Patterns*, Ralph E. Griswold, 2000:  
[http://www.cs.arizona.edu/patterns/weaving/weavedocs/gre\\_cwev.pdf](http://www.cs.arizona.edu/patterns/weaving/weavedocs/gre_cwev.pdf)
3. <http://www.cs.arizona.edu/patterns/weaving/>
4. <http://www.mhsoft.com/wif/wif.html>
5. "Oranges & Lemons Say the Bells of St. Clement's", Pauline Drake, *Handwoven*, September/October 2000, pp. 32-34.

Ralph E. Griswold  
Department of Computer Science  
The University of Arizona  
Tucson, Arizona

[ralph@cs.arizona.edu](mailto:ralph@cs.arizona.edu)

© 2001, 2002 Ralph E. Griswold