## NAME

rmtxop - concatenate, add, multiply, divide, transpose, scale, and convert matrices

## SYNOPSIS

**rmtxop** [ **−v** ][ **−f[afdc]** ][ **−t** ][ **−s sf ..** ][ **−c ce ..** ][ **−rf|−rb** ] **m1** [ **.+*/** ] **..**

## DESCRIPTION

*Rmtxop* loads and concatenates or adds/multiplies/divides together component matrix files given on the command line. Each file must have a header containing the following variables:

NROWS={number of rows}
NCOLS={number of columns}
NCOMP={number of components}
FORMAT={ascii|float|double|32-bit_rle_rgbe|32-bit_rle_xyze}

The number of components indicates that each matrix element is actually composed of multiple elements, most commonly an RGB triple. This is essentially dividing the matrix into planes, where each component participates in a separate calculation. If an appropriate header is not present, it may be added with a call to *rcollate(1)*. A matrix may be read from the standard input using a hyphen by itself ('-') in the appropriate place on the command line.

Any of the matrix inputs may be read from a command instead of a file by using quotes and a beginning exclamation point ('!').

Two special cases are handled for component matrices that are either XML files containing BSDF data, or Radiance picture files. In the first case, the BSDF library loads and interprets the transmission matrix by default. Alternatively, the front (normal-side) reflectance is selected if the *−rf* option precedes the file name, or the backside reflectance if *−rb* is specified. (XML files cannot be read from the standard input or from a command.) In the second case, the RGBE or XYZE values are loaded in a 3-component matrix where the number of columns match the X-dimension of the picture, and the number of rows match the Y-dimension. The picture must be in standard pixel ordering, and the first row is at the top with the first column on the left. Any exposure changes that were applied to the pictures before *rmtxop* will be undone, similar to the *pcomb(1) −o* option.

Before each file, the *−t* and *−s* or *−c* options may be used to modify the matrix. The *−t* option transposes the matrix, swapping rows and columns. The *−s* option applies the given scalar factor(s) to the elements of the matrix. If only one factor is provided, it will be used for all components. If multiple factors are given, their number must match the number of matrix components. Alternatively, the *−c* option may be used to "transform" the element values, possibly changing the number of components in the matrix. For example, a 3-component matrix can be transformed into a single-component matrix by using *−c* with three coefficients. A four-component matrix can be turned into a two-component matrix using 8 coefficients, where the first four coefficients will be used to compute the first new component, and the second four coefficients yield the second new component. Note that the number of coefficients must be an even multiple of the number of original components. The *−s* and *−c* options are mutually exclusive, insofar as they cannot be applied together to the same input matrix.

If present, the second and subsequent matrices on the command line are concatenated together, unless separated by a plus ('+'), asterisk ('*'), or forward slash ('/') symbol, in which case the individual matrix elements are added, multiplied, or divided, respectively. The concatenation operator ('.') is the default and need not be specified. Note also that the asterisk must be quoted or escaped in most shells. In the case of addition, the two matrices involved must have the same number of components. If subtraction is desired, use addition ('+') with a scaling parameter of -1 for the second matrix (the *−s* option). For element-wise multiplication and division, the second matrix is permitted to have a single component per element, which will be applied equally to all components of the first matrix. If element-wise division is specified, any zero elements in the second matrix will result in a warning and the corresponding component(s) in the first matrix will be set to zero.

Evaluation proceeds from left to right, and all operations have the same precedence. If a different evaluation order is desired, pipe the result of one *rmtxop* command into another, as shown in one of the examples

below.

The number of components in the next matrix after applying any *-c* transform must agree with the prior result. For concatenation (matrix multiplication), the number of columns in the prior result must equal the number of rows in the next matrix, and the result will have the number of rows of the previous and the number of columns of the next matrix. In the case of addition, multiplication, and division, the number of rows and columns of the prior result and the next matrix must match, and will not be changed by the operation.

A final transpose or scaling/transform operation may be applied to the results by appending the *−t* and *−s* or *−c* options after the last matrix on the command line.

Results are sent to the standard output. By default, the values will be written in the lowest resolution format among the inputs, but the *−f* option may be used to explicitly output components as ASCII (-fa), binary doubles (-fd), floats (-ff), or RGBE colors (-fc). In the latter case, the actual matrix dimensions are written in the resolution string rather than the header. Also, matrix results written as Radiance pictures must have either one or three components. In the one-component case, the output is written as grayscale.

The *−v* option turns on verbose reporting, which announces each operation.

## EXAMPLES

To concatenate two matrix files with a BTDF between them and write the result as binary double:

```
rmtxop -fd view.vmx blinds.xml exterior.dmx > dcoef.dmx
```

To convert a BTDF matrix into a Radiance picture:

```
rmtxop -fc blinds.xml > blinds.hdr
```

To extract the luminance values from a picture as an ASCII matrix:

```
rmtxop -fa -c .265 .670 .065 image.hdr > image_lum.mtx
```

To scale a matrix by 4 and add it to the transpose of another matrix:

```
rmtxop -s 4 first.mtx + -t second.mtx > result.mtx
```

To multiply elements of two matrices, then concatenate with a third, applying a final transpose to the result:

```
rmtxop first.mtx \* second.mtx . third.mtx -t > result.mtx
```

To left-multiply the element-wise division of two matrices:

```
rmtxop -fd numerator.mtx / denominator.mtx | rmtxop left.mtx - > result.mtx
```

To send the elements of a binary matrix to *rcalc(1)* for further processing:

```
rmtxop -fa orig.mtx | rcollate -ho -oc 1 | rcalc [operations]
```

## NOTES

Matrix concatenation is associative but not commutative, so order matters to the result. *Rmtxop* takes advantage of this associative property to concatenate from right to left when it reduces the number of basic operations. If the rightmost matrix is a column vector for example, it is much faster to concatenate from the right, and the result will be the same. Note that this only applies to concatenation; element-wise addition, multiplication, and division are always evaluated from left to right.

## AUTHOR

Greg Ward

## SEE ALSO

cnt(1), getinfo(1), histo(1), neaten(1), pcomb(1), rcalc(1), rcollate(1), rcontrib(1), rcrop(1), rfluxmtx(1), rlam(1), rsplit(1), tabfunc(1), total(1), wrapBSDF(1)