## NAME

rcomb - combine and convert matrices a row at a time

## SYNOPSIS

**rcomb** [ **−h** ][ **−w** ][ **−f[afdc]** ][ **−n nproc** ][ **−f file** ][ **−e expr** ][ **−C {symbols|file}** ][ **−c ce ..** ][ **−s sf ..** ] **m1 ..** [ **−m[t] mcat** ]

## DESCRIPTION

*Rcomb* combines inputs given on the command line, one matrix row or picture scanline at a time. By default, the result is a linear combination of the matrix elements or pixels transformed by *−c* specifications and scaled by *−s* coefficients, but an arbitrary mapping can be assigned with the *−e* and *−f* options, similar to the *pcomb(1)* and *rcalc(1)* commands. (The definitions in each *−f source* file are read and compiled from the RADIANCE library where it is found.)

If any *−c* or *−s* options follow the last input matrix, output results will be transformed and/or scaled accordingly. These operations are discussed in greater detail below. A single concatenation matrix may be applied after element operations using the *−m* option. If the option is given as *−mt* then the concatenation matrix will be transposed before it is applied. Matrix concatenation will happen before or after any trailing operations, depending on relative command line placement.

Each input file must have a header containing the following metadata:

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

The number of components indicates that each matrix element is actually composed of multiple channels, 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. Similarly, any of the inputs may be read from a command instead of a file by using quotes and a beginning exclamation point ('!').

In the case of Radiance picture files, the number of columns is the X-dimension of the picture, and the number of rows is the Y-dimension. The picture must be in standard pixel ordering, and the zeroeth row is at the top with the zeroeth column on the left. Any exposure changes that were applied to the pictures before *rcomb* will be undone, similar to the *pcomb −o* option. Radiance spectral pictures with more than 3 components are also supported. These are typically produced by *rtpict(1)* or *rfluxmtx(1)*.

Before each input, the *−c* and/or *−s* options may be used to modify the matrix elements. The *−c* option can "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.

Alternatively, a set of symbolic output components may be given to the *−c* option, with the following definitions:

R - red channel
G - green channel
B - blue channel
X - CIE X channel
Y - CIE Y channel (aka., luminance or illuminance)
Z - CIE Z channel
S - scotopic luminance or illuminance
M - melanopic luminance or illuminance

A - average component value

These letters may be given in any order as a single string, and if *-c RGB* or *-c XYZ* is specified for an input picture or the *-fc* option is given, the output will be written as a RGBE or XYZE picture. Note that conversion from a float or RGBE color space applies an efficacy factor of 179 lumens/watt (for CIE or melanopic output) or 412 (for scotopic output), and the inverse happens for conversion from XYZE input to RGB or RGBE output. Lower case versions of all these components are also supported, the only difference being that the efficacy factors are ignored.

If a matrix or picture file path is given to the *−c* option, then the color space of that file will be used, instead.

The *−C* option takes either a symbolic color space or an input file, and will be applied to all subsequent matrices that do not have their own associated *−c* option.

Additionally, 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 *after* application of any *−c* option for this input matrix or picture, even if the *−s* option appears first.

The number of components in all input matrices after applying any *-c* transform must agree. Similarly, the number of rows and columns of all results must match exactly. (The *rcrop(1)* utility may be used to trim inputs if necessary.)

If the *−e* or *−f* options are used to define a "co" variable or "co(p)" function, which will be evaluated for each output component from the current element. The "co" variable defines identical operations for all components, whereas "co(p)" may specify different operations for each component. The element position is defined by the "r" and "c" variables, where *r* goes from 0 to "nrows" minus one, and *c* goes from 0 to "ncols" minus one. (Note that "nrows" may be zero if unspecified in inputs, and this is a unique capability of *rcomb* to handle these.) Component p from input i is accessed with the "ci(i,p)" function, and the number of components is defined by the "ncomp" constant. If given as "ci(i)", the function returns the current component being evaluated by *rcomb*. A different component may be referenced using the second argument. For example, "ci(1,2)" accesses the second component from the first input. If the input is a picture, the the constants "R", "G", and "B" are conveniently defined as the channel numbers 1, 2, and 3, respectively. For color or spectral inputs, the function "wl(p)" gives the central wavelength for channel *p* in nanometers. For convenience and compatibility with *pcomb*, the functions "ri(i)", "gi(i)", and "bi(i)" are predefined as "ci(i,R)", "ci(i,G)", and "ci(i,B)", respectively. Accordingly, the "ro", "go", and "bo" variables may be used in place of "co(R)", "co(G)", and "co(B)", but all three must be defined for this substitution to take place. Finally, the total number of input files is set in the constant "nfiles".

Results are sent to the standard output. By default, the values will be written in the lowest precision format among the inputs, but the *−f[adfc]* option may be used to explicitly output components as ASCII (-fa), binary doubles (-fd), floats (-ff), or common-exponent colors/spectra (-fc). In the latter case, the actual matrix dimensions are written in the resolution string rather than the header. Also, matrix results will be written as standard Radiance pictures if they have either one or three components. In the one-component case, the output is written as grayscale. If more than 3 components are in the final matrix and *-fc* is specified, the output will be a Radiance spectral picture.

The *−h* option may be used to reduce the information header size, which can grow disproportionately, otherwise. The *−w* option turns off warnings about divide-by-zero and other non-fatal calculation errors.

The *−n* option specifies how many execution processes to employ, which may improve performance on multi-core architectures, especially for matrix multiplication and complex operations on long input rows.

## EXAMPLES

To convert two hyperspectral pictures to RGB color space, average them together, and write them out as a RADIANCE picture:

```
rcomb -C RGB -s .5 img1.hsr -s .5 -fc img2.hsr > avg.hdr
```

Divide one set of matrix elements by the Euclidean sum of two others:

rcomb -e "co=ci(1)/sqrt(ci(2)^2+ci(3)^2)" inp1.mtx inp2.mtx inp3.mtx > out.mtx

Compute the absolute and relative differences between melanopic and photopic values in a spectral image:

rcomb -C MY -e "abs(x):if(x,x,-x)" -e "co(p)=select(p,abs(ci(1,1)-ci(1,2)),(ci(1,1)-ci(1,2))/ci(1,2))" input_spec.hsr -fa > compare.mtx

Concatenate a spectral flux coefficient matrix with a spectral sky matrix to compute a set of melanopic lux values:

rcomb view_spec.mtx -m sky_spec.mtx -c M > melux.mtx

## NOTES

The *rcomb* tool was created to overcome some limitations of *rmtxop* and *pcomb*, whose capabilities somewhat overlap. The former loads each matrix into memory before operations, and element components are stored as double-precision. Very large matrices therefore present a problem with that tool. Furthermore, *rmtxop* does not allow arbitrary expressions, limiting what can be accomplished easily on the command-line. In contrast, *pcomb* is fully programmable and operates on its input using a scanline window, so it can handle much larger input dimensions. It also handles single- and three-component float matrices on input and output, but unlike *rmtxop*, *pcomb* has not been extended to handle RADIANCE hyperspectral images or more general matrix data.

The *rcomb* tool is a compromise that exceeds the capabilities of either of its predecessors in certain circumstances. In particular, very large matrices may be combined using arbitrary, user-defined operations, and the convenient color conversions of *rmtxop* are supported for both input and output. Finally, a single matrix may be concatenated after operations, permitting a flux transfer matrix with millions of rows to pass through. Generally speaking, *rcomb* should be preferred over *rmtxop* for any operations it can handle, which is everything except multiple matrix concatenations and transpose operations. The latter may be handled more efficiently by *rcollate(1)*. That said, there is no significant difference for simple operations on small matrices, and only *rmtxop* and *dctimestep(1)* accept XML files as inputs. Also note that the resizing function of *pcomb* is not supported in *rcomb*, and should instead be handled by *pfilt(1)*.

Similar to *rmtxop*, all calculations are peformed internally using 32-bit floating-point, so there is little benefit in either reading or writing 64-bit double data. This may be overridden at compile time using the macro "-DDTrmx_native=DTdouble".

## BUGS

The *rcomb* command currently ignores the "PRIMARIES" setting in input headers, and does not produce any on output, even in circumstances where it would make sense to.

## AUTHOR

Greg Ward

## SEE ALSO

dctimestep(1), icalc(1), getinfo(1), pcomb(1), pfilt(1), pvsum(1), ra_rgbe(1), ra_xyze(1), rcalc(1), rcollate(1), rcontrib(1), rcrop(1), rfluxmtx(1), rmtxop(1), rtpict(1), rtrace(1), vwrays(1)