Decode HDR image

Submit Solution

Write a C++ command line program that accepts the following options:

```
1. hdr_decode <input file .HDR> <output file .PAM>
```

The program must manage the command line and load a file in HDR format, apply a Global Tone Mapping Operation, and finallay save it in PAM format.

The HDR format, originally known as the Radiance picture format (.hdr, .pic), was first introduced as part of the Radiance lighting simulation and rendering system in 1989, and has since found widespread use in the graphics community, particularly for HDR photography and image-based lighting. A somewhat criptic format description is provided in file `filefmts.pdf` from page 28 to page 33.

The file starts with a line based header. Each line is terminated by a single `\n`. The first line must be `"#?RADIANCE"`.
Then a list of "variables" is defined as `<VARIABLE_NAME>=<some text>`. A line starting with `#` is a comment and should be skipped. The end of the header is indicated by an empty line. For this exercise, only the `FORMAT` variable is important and it must be the string `"32-bit_rle_rgbe"`. The solution must cope with multiple comments and variables just skipping them.

After the header end (the empty line) there is a single line (again terminated by `\n`) called "Resolution String". For this exercise, only the format `-Y N +X M` must be supported. `N` is the number of rows (Y increases from top to bottom) and `M` is the number of columns (X increases from left to right). Both are encoded as ASCII text and pixel `(0,0)` is the top left one.

Now three floating point R,G,B values are encoded with 4 bytes as 3 mantissa values (1 byte each) and a single common exponent (1 byte). To reconstruct the channels floating point values, the following equations are used:

$$R_{float} = \frac{R + 0.5}{256} 2^{E-128}$$

$$G_{float} = \frac{G + 0.5}{256} 2^{E-128}$$

$$B_{float} = \frac{B + 0.5}{256} 2^{E-128}$$

where R,G,B,E are the four bytes read from the file.

There are different ways to store these values, but in this exercise we will use only the "New run-lenght encoding". In this format each image line (called *scanline*) is stored in a non-interleaved fashion, that is first `M` values of R are encoded, then `M` values of G are encoded, then `M` values of B are encoded, then `M` values of E are encoded.

The scanline starts with two bytes equal to 2 then the number of columns is written with 16 bits big-endian. For example a scanline of an image with 2141 columns will start with the four bytes: `02 02 08 5D`. `085D` is 2141 in hexadecimal.

The values are stored with an adaptive run-length encoding (similar to the PackBits encoding). There are two options: run or copy. Each option starts with an 8 bit unsigned integer `L`:
1) if `L <= 127`, the next `L` bytes of the input should be sent to the output (copy);
2) if `L > 127`, the next byte of the input should be sent to the output `L - 128` times (run length);

After decoding the HDR image, you need to map linear luminance values to the range `[0, 255]`. For this reason, you must find the gloabl minimum (min) and maximum (max) values in the decoded numbers. You don't need a minimum for each channel, but the global minimum. If the minimum `R` is 0.332, the minimum `G` is 0.011 and the minimum `B` is 0.153, the global minimum is 0.011. The same for the maximum. To obtain the final value the formula is:

$$C_{8bit} = 255 \cdot \left(\frac{C_{float} - min}{max - min} \right)^{0.45}$$

where C_{float} is the three channel vector of the decoded HDR image, while C_{8bit} is the output 3 bytes RGB image.

The final image must be saved in PAM format. The PAM format is documented in the file `PAM format specification.html` and PAM images can be viewed with XnView. The output image will be a PAM file with DEPTH 3 and TUPLTYPE RGB.