

Installation and Usage

Why did you make Picturenaut?

A few years ago when I heard the first time the abbreviation DRI (Dynamic Range Increase), I was enthused from the potentials of this technology.

However, working with masks and levels was not what I had imagined myself, although one can obtain good results with this technology. I imagined an automatic process, which I found quickly in the Internet. The new magic word was called HDRI. But unfortunately the HDRI software at that time could not really convince me and so I developed my own HDRI software. That is how Picturenaut was born.

Picturenaut was freely available from the beginning.

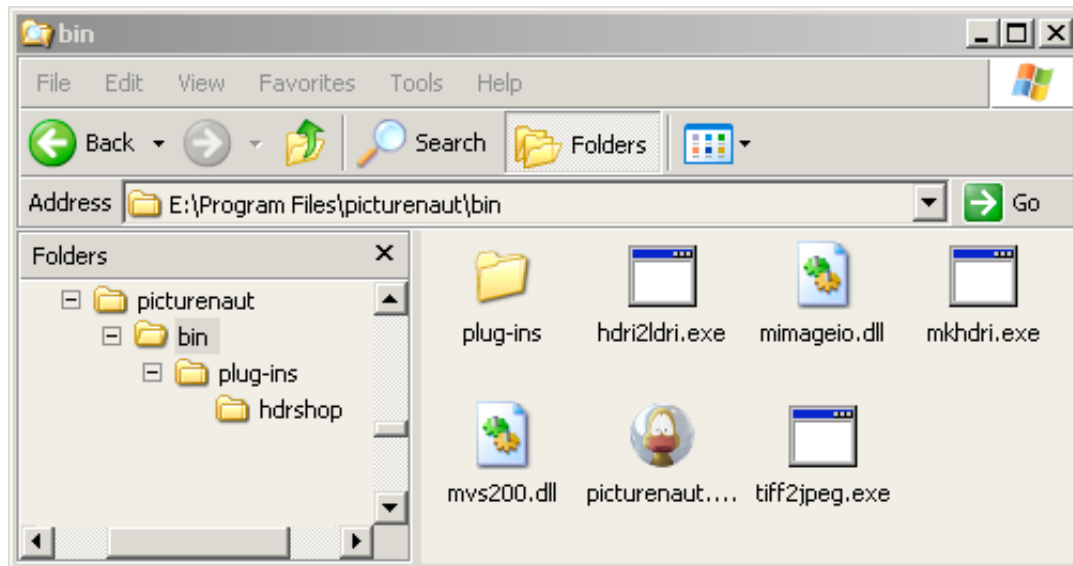
Marc Mehl

I have just downloaded Picturenaut, how do I install it now?

Picturenaut has no installer.

Unzip the picturenaut ZIP archive with it's path informations into a folder of your choice. **It is very important that the subfolder structure stays intact**, otherwise it will not work correctly.

For example, if you unzipped into "C:\Program Files", your directory structure should look like this:



Then you can start picturenaut directly. You will find it in the "bin" subfolder.

What image formats are supported by Picturenaut?

HDR Formats

- PFM (Portable Float Map)
- HDR (Radiance)
- EXR (OpenEXR)
- TIFF (32-bit Floating Point)
- TIFF (LogLuv)

LDR Formats

- JPEG
- TIFF (only RGB color space)
- TGA (Targa, no alpha channel)

All these formats are also supported by the included commandline programs MKHDRI and HDRI2LDRI.

What about RAW files?

Picturenaut does not load RAW files directly at the moment. This feature is on the list, but hasn't been implemented yet.

If you want to create an HDRI out of bracketed RAW files, you need to develop them in a RAW converter first. Use the same settings for white balance, and do not perform any exposure adjustments during this conversion. It's also recommended to turn off sharpening, highlight recovery, and other image enhancements. Make sure the EXIF data is retained, and save as TIFF files.

HDRI Creation

In the merging dialog, I saw optional "exposure correction". What is this doing?

When you combine images to produce an HDRI you add all pixel values - for example at position x:100 and y:100 - and divide the resulting value by the number of images. This is simply a value averaging. Sounds easy.

But before you can average pixel values you must align the image exposures. The image exposures are obtained by the EXIF informations in every image. In addition pixel value averaging only works correct on linear images. Linear means that images have no gamma correction. But normally images taken from a camera have a gamma correction. Picturenaut must compute the inverse gamma correction to applying an exposure correction. If the EXIF values are inexactly, Picturenaut computes a better exposure value based from the inexactly EXIF values for every image. This is what the switch "exposure correction" means.

What about the weighting, what to the options "Derivative, +hat 1, +hat2" mean?

Above I described the process of pixel value averaging. But there is still something to be observed.

An image have dark pixels and light pixels. In dark regions images are often noisy. The light regions could be over exposed. This kind of pixels should not be computed because they are useless. Instead of dropping those pixels, we assign a weight to all pixels (a value between 0 and

1 multiplied with the pixel value). The weight determines the trust for every pixel. One definition for the weight is that the weight is higher, the more the pixel value is in the middle of a camera curve (or gamma curve). From that definition we can construct a weight function. And for this, the weight function is a simple triangle where the pixel value 0.5 have the most trust. This simple weight function produces very good results for daylight HDRIs but have some disadvantages for night HDRIs. If you have moving lights in your image bracketing you get black holes at this postions.

In Picturenaut the weighting function is the derivation of the camera curve. This function produces very good night HDRIs but for daylight HDRIs the result not so rich in contrast as the triangle function described above. For that reason you can choose a hat function in Picturenaut. A hat function is an additional weight for the weight. At the edges of the hat function the weight is multiplied with a value near zero all other values of the hat function are 1. Hat function 2 produces the same contrasty images like the triangle weight function but have the same problem with moving lights in night HDRIs. For that you can choose hat function 1 or no hat function. I hope the explanation was not so difficult to understand.

And what is the "color balance" doing?

This means Picturenaut tries to align the HDRI colors to the source image colors. Color balancing in Picturenaut is a post process after all image computations. The HDRI generation can have little errors if the camera curve estimation is erroneous or inexactly. Color balancing is only an option which I used very rare.

Commandline Access and Automation

What are these other programs in the "bin" directory?

Picturenaut's architecture is modular. Its most important functions can all be accessed from a commandline. There are:

- o **MKHDRI.exe** ... Make HDRI from a series of bracketed images
- o **HDRI2LDRI.exe** ... The Tonemapping module, converting HDRIs into TIFFs or JPEGs

These programs can be useful for automating tasks like HDR creation or tonemapping. They can be called from a script or from a custom application.

In the most basic form, you just fire these programs from a commanline shell:

- o Go to your Start-Menu and select "Run"
 - o Type "cmd" and hit enter
 - o A commandline shell will open
-

How can I use MKHDRI.exe to make HDRIs from a commandline?

Let's say you have a folder full with numbered TIFF files, which have all been exported from your favorite RAW-converter **with the EXIF data intact**. Then you can generate an HDRI automatically with this command:

```
c:> mkhdri -a -co:myCurve.txt -out:myPic.tif myPics\*.tif
```

The commandline output would then look like this:

FAQ Manual

```
file 'myPics\DPV_0001.TIF' (8-BIT 3072x2048) opened
file 'myPics\DPV_0002.TIF' (8-BIT 3072x2048) opened
file 'myPics\DPV_0003.TIF' (8-BIT 3072x2048) opened
file 'myPics\DPV_0004.TIF' (8-BIT 3072x2048) opened
file 'myPics\DPV_0005.TIF' (8-BIT 3072x2048) opened
file 'myPics\DPV_0006.TIF' (8-BIT 3072x2048) opened
file 'myPics\DPV_0007.TIF' (8-BIT 3072x2048) opened
file 'myPics\DPV_0008.TIF' (8-BIT 3072x2048) opened
F-Stops : 5.584963
align images ...
summary of shifts against image 'myPics\DPV_0006.TIF'
image 'myPics\DPV_0007.TIF' shifted : xs = 0 : ys = 1
image 'myPics\DPV_0008.TIF' shifted : xs = 0 : ys = 1
estimate brightness transfer functions ...
estimate camera curve for channel 1 ...
estimate camera curve for channel 2 ...
estimate camera curve for channel 3 ...
writing camera curve file 'curve' ...
combining images ...
dynamic range : 4382.434065
max radiance : 13.497492
writing image file 'myPic.tif' ...
success
```

In this example there were 8 images in the folder "mypics". The result is an HDRI in the LogLuv TIFF format.

Hint: You can also supply several source folders or files, separated by a space. The order of the images is irrelevant, MKHDRI automatically sorts them by exposure.

What are the other options for MKHDRI.exe?

You can see all options by calling the help with:

```
c:> mkhdr -?
```

This will show a listing of available parameters:

```
-out:[file]          output file name
-fno:[float]         F-Number overwrite
-tr:[float=1.0]     trim ratio to reject side areas with vignetting
-sl:[float=1.00]    saturation level
-nl:[float=0.01]    noise level
-cesl:[float=0.85] curve estimation saturation level
-cenl:[float=0.02] curve estimation noise level
-ci:[file]          read curve file (skips curve estimation)
-co:[file]          write curve file
-a                  enable automatic image alignment
-eo                 estimate curve only
-ec                 disable exposure correction
-ep                 disable pixel based exposure correction
-cb                 enable color balancing
-ab                 ask for exposure bias
```

FAQ Manual

```
-linear          linear input files (skips curve estimation)
-f32            use 32-bit IEEE floating point for HDRI TIFFs
-verbose       disable verbosity
-?            help
```

Most of these options are preset with default values, and are optional. In most cases they don't need to be specified. Note, that automatic image alignment is OFF by default, and optionally needs to be activated with "-a".

Hint: The parameter "-co" saves a response curve, which can be re-used for tonemapping with HDRI2LDRI.exe.

What options can I use for HDRI2LDRI.exe?

HDRI2LDRI.exe is the tonemapping module. It will in effect convert your HDR image to an LDR image.

Commandline options are:

```
-g:[float]      gamma value (default is gamma 2.2)
-rg            disable Rec. 709 gamma
-c:[file]      camera curve file
-ic:[file]     inverse camera curve file
-cplot:[file]  create curve plot as TIFF or JPEG
-cpo          create curve plot only
-heq          disable histogram equalization
-tm           disable tone mapping
-b:[float]     tone mapping bias [0.0 - 1.0] (default is 0.9)
-sb:[float]    shadow bias [-2.0 - 2.0] (default is 0.0)
-ct:[float]    contrast [-1.0 - 1.0] (default is 0.0)
-e:[float]     exposure multiplier +/- (default is -2.2 if
               tone mapping is enabled; otherwise 0.0)
-xs:[int]     output width
-ys:[int]     output height
-sf:[filter]   resample filter name (default is lanczos)
               filter names are: lanczos, catmull-rom, triangle,
                               hermite, mitchell, cubic-bspline
                               bell
-q:[quality]   quality if output file is a JPEG (default is 85)
-8            write 8-bit TIFF (default is 16-bit TIFF)
-verbose      disable verbosity
-?           help
```
