

Reference Manual

DiscretePhoton H.264 encoder

Reference Manual

About DiscretePhoton H.264 encoder

DiscretePhoton H.264 encoder Windows version is provided as DirectX Media Object (DMO) in 32-bit or 64-bit binary. It can be used directly or in DirectShow environment.

DiscretePhoton H.264 encoder supports up to 64 simultaneous threads. But actual number of threads is determined by the number of CPU-cores (or hyper-threads) of your system and the frame size. You can only decrease it.

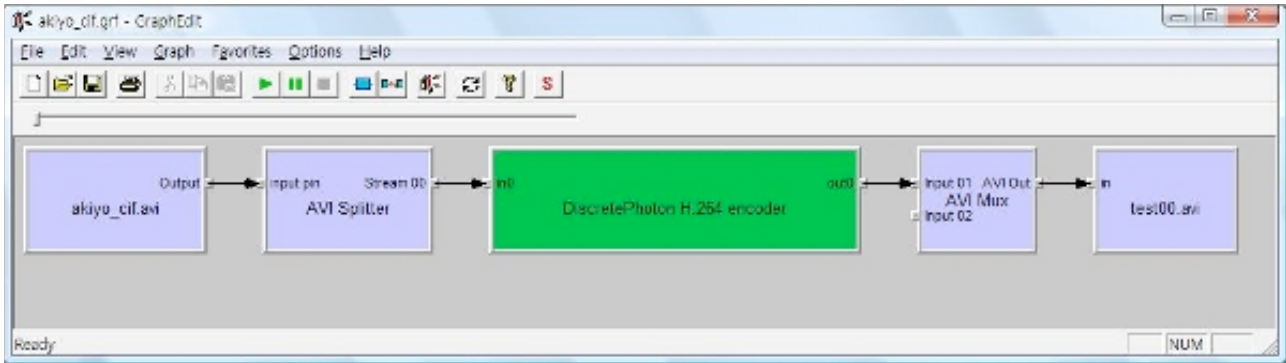
DiscretePhoton H.264 encoder's multi-threading method is based on wave-front. All the worker threads are concentrated on the most recent input frame, which could result in very low encoding latency without any penalty on final encoded quality or bit-size. Also, its CBR rate-control is tuned for low latency scenario. So, DiscretePhoton H.264 encoder is very well fitted for time-critical applications such as video conferencing, as well as other types of applications.

Low encoding latency also means low latency variation, which could result in extremely low frame-drop rate for live encoding.

Some performance figures can be found from [the latest video codec comparison by MSU](#).

For more information and evaluation versions, visit www.discretephoton.com.

Using GraphEdt.exe



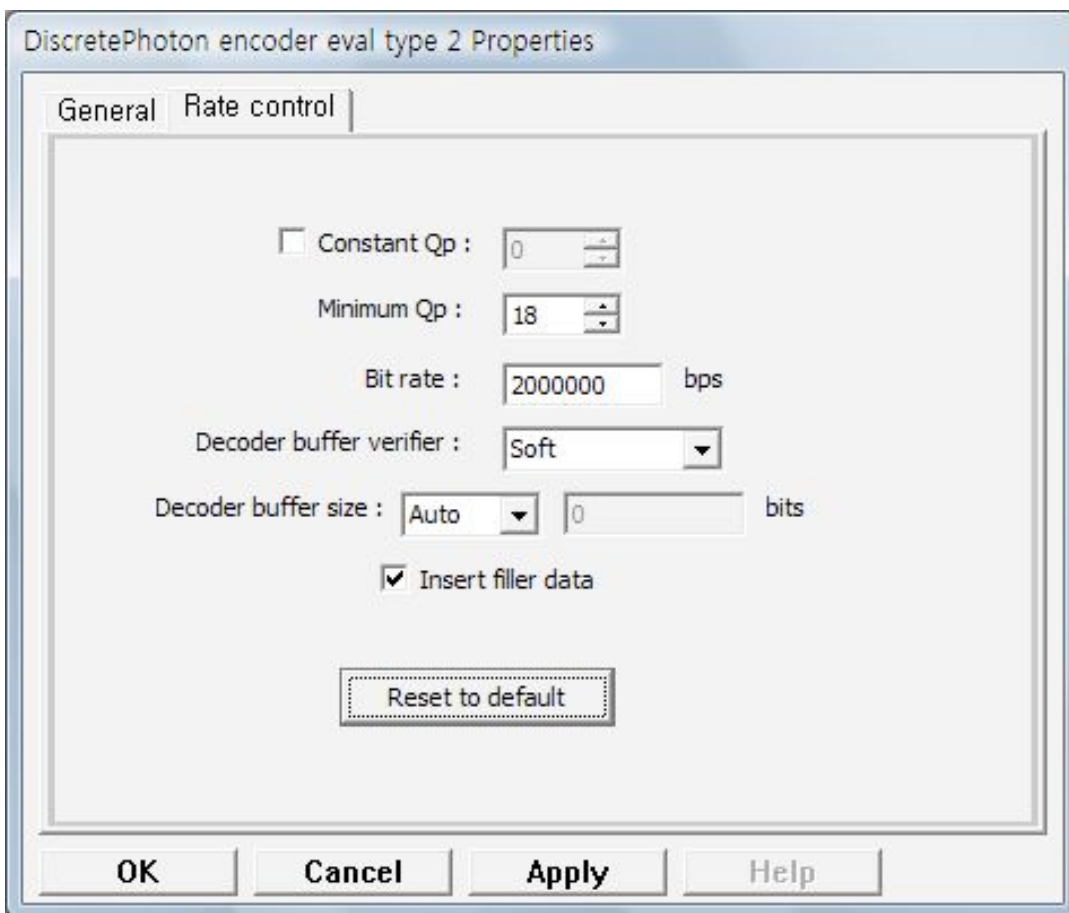
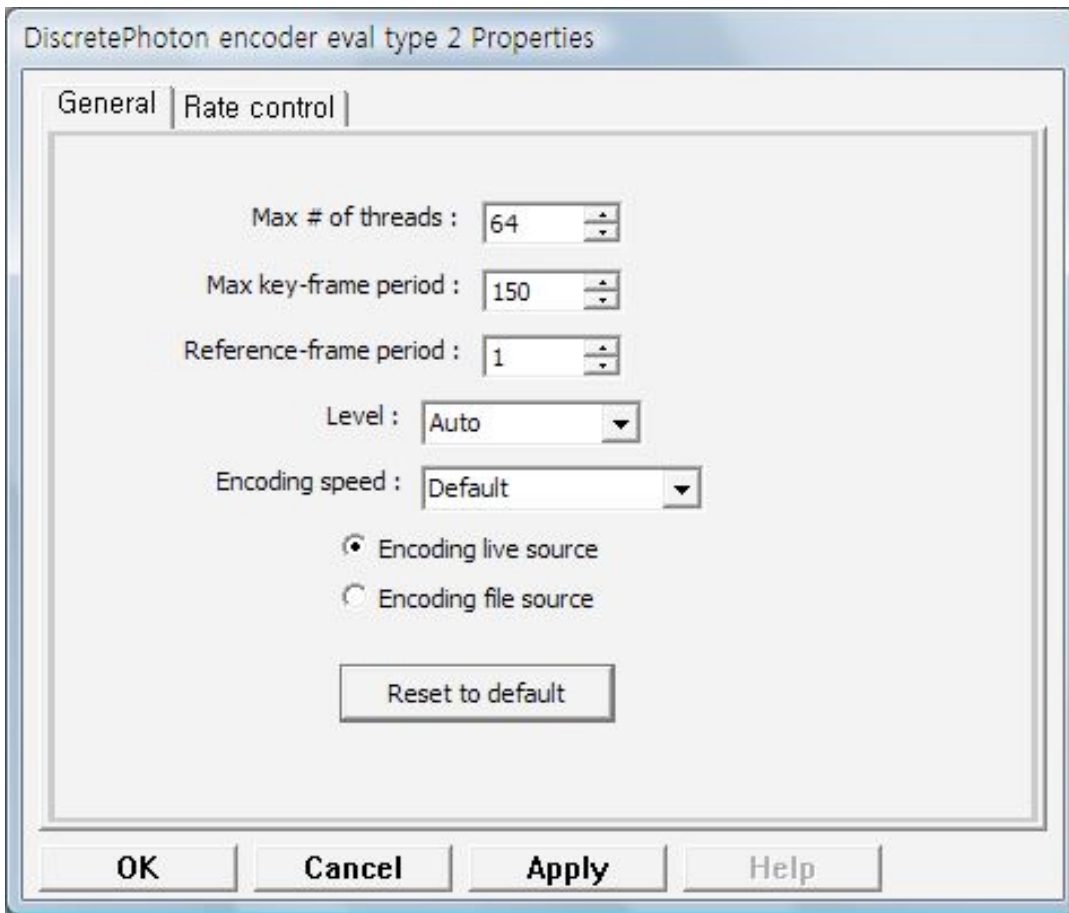
GraphEdt.exe is contained in Windows SDK. You can use it to build and test DirectShow filter graph visually.

After installation of 32-bit version of DiscretePhoton H.264 encoder, you can find it in GraphEdt.exe at Graph -> Insert Filters... -> Video Compressors from the menu bar.

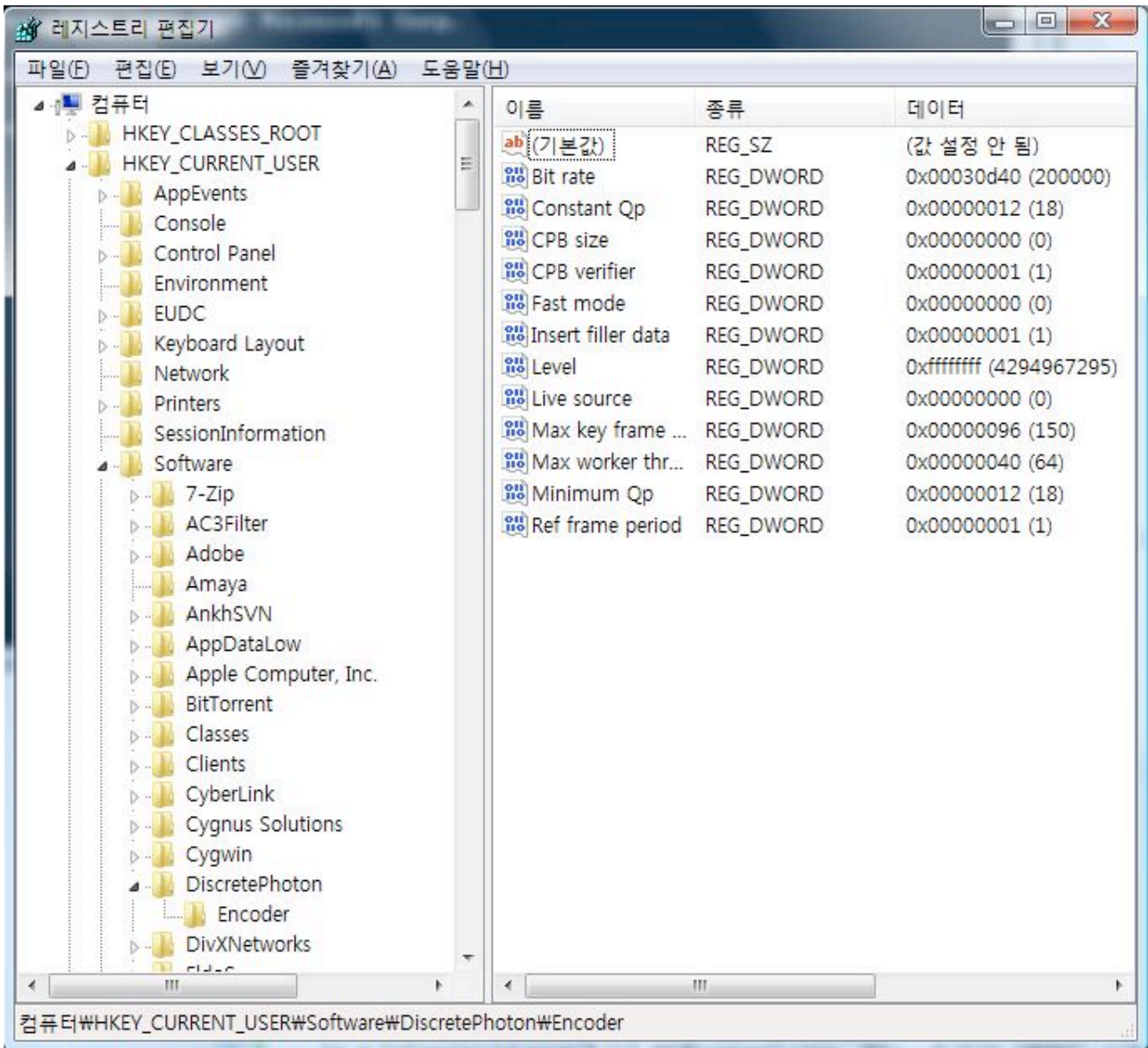
The green color of DiscretePhoton H.264 encoder node indicates that it is actually not a DirectShow filter, but a DMO(DirectX Media Object) with DirectShow wrapping.

DiscretePhoton H.264 encoder accepts **I420, YV12, UYVY, YUYV, RGB24, RGB32** as its input video format. And it outputs H.264 byte stream (FourCC: **H264**).

If you right click on DiscretePhoton H.264 encoder node, property page will be shown as follows.



There are two tabs, 'General' and 'Rate control'. After you change any value and apply it, the changed value will be remembered in the Windows registry (at HKEY_CURRENT_USER\Software\DiscretePhoton\Encoder). You may read or write those registry values in your program.



Meaning of each value field is as follows.

Max # of threads: ("Max worker threads" on registry.) Maximum number of threads that can be used during encoding process. The number is not actual but maximum because the actual number will be determined by the number of CPU cores (or Hyper threads) of your system and the frame size. You can decrease this number below your CPU cores if you do not want to utilize 100% of CPU power during encoding process.

Max key-frame period: ("Max key frame period" on registry.) Here, key-frame means IDR frame of H.264. Its value is also maximum and not actual

because actual key-frames could appear more frequently for encoding efficiency. Minimum key-frame period is half of this value.

Reference-frame period: ("Ref frame period" on registry.) By default, every frame will be used as reference frame. But by increasing this value (so, not every frame will be used as reference frame) and by decreasing Max key-frame period, fault tolerance during video transmission over unreliable network (such as UDP) might be achieved with the cost of decreased compression-ratio or overall quality.

Level: (On registry, -1:Auto, 0:Level_1, 1:Level_1b, ...) It is H.264 encoding level (as given in annex A.3 of the specification). Normally you can set it as 'Auto'. Then the level will be determined automatically with other parameters and input video.

Encoding speed: ("Fast mode" on registry. 0:Default, 1:Fast, 2:Even faster) Faster encoding speed can be achieved by sacrificing some video quality. So, if you are using rather high output bit-rate and want faster encoding speed, you can set it as 'Fast' or 'Even faster'. But if you are using low output bit-rate, then the 'Even faster' mode might produce blocky output video.

Encoding live source / Encoding file source: ("Live source" on registry. 0:file source, 1:live source) If your source video is live (such as from webcam or TV tuner card, etc.), set 'Encoding live source'. Otherwise, set 'Encoding file source'.

Constant Qp: (On registry, 0 value deactivates it.) If you set this value, output video quality will be almost constant throughout entire output video, but output bit-rate will fluctuate severely. That is, output bit-rate will not be controlled at all. So it is not good for video which is transmitted over network. Valid value range is 18 - 51. Larger value means more compression-ratio and lower video quality.

Minimum Qp: Qp (quantization parameter) values which are lower than this will not be applied during rate-control.

Bit rate: Output bit-rate in bits per second.

Decoder buffer verifier: ("CPB verifier" on registry. 0:Hard, 1:Soft, 2:None) It sets rigidity of decoder buffer verifier. Here, decoder buffer corresponds to CPB(coded picture buffer) of annex C of H.264 specification. (NOTICE: even 'Hard' does not mean perfect).

Decoder buffer size: ("CPB size" on registry.) Size in bits. If it is set to 'Auto', it will be set to maximum allowed value (of VCL HRD).

Insert filler data: If decoder buffer reached near overflow and Qp reached near its minimum value, filler data may be inserted into output bit stream to sustain output bit-rate. So normally it should be set.

Programming with DirectShow

I believe there are some good references about programming in DirectShow environment.

You can also download simple example programs that make use of DiscretePhoton H.264 encoder from

<http://www.discretephoton.com/php/downloader.php?f=Examples.zip> for 32-bit version or from

http://www.discretephoton.com/php/downloader.php?f=Examples_x64.zip for 64-bit version.

For building C++ version, DirectShow base class library is needed which is contained in Windows SDK. And C# version requires DirectShow.Net. More information can be found in README.htm.