



[Home \(/\)](#) > [Docs \(/docs/\)](#) > [Encoder Parameters](#)

VP8 Encode Parameter Guide

This document details some of the encoder controls that are available in VP8.

Show Contents

Different encoders or tools will map these controls in different ways, but knowing what is available should at least give you some idea of what to look for. In this document we use the parameter names defined in the sample encoder, `vp8enc.exe`. A summary of the command line usage and parameter set is given at the end.

1. The Basics

This section describes the basic parameters for codec selection, setting image dimensions, output frame rate, encoder speed profile and target bitrate.

```
--codec=<arg>
```

The video codec to use, `vp8` or `vp9`. Default is `vp8`.

```
--width=<arg> (or -w <arg>)  
--height=<arg> (or -h <arg>)
```

Image dimensions (width and height). Only required for "raw" yuv input. The recommended input format is Y4M files, as these will set the dimensions and frame rate automatically.

```
--fps=<arg>
```

Output frame rate, expressed as a fraction. For example for 29.97 frames per second you could specify 30000/1001. Only required for "raw" yuv input. The recommended input format is Y4M files, as these will set the dimensions and frame rate automatically.

```
--target-bitrate=<arg>
```

What bitrate per second should the encoder try and target (in `vp8enc` the number is assumed to be in kbits/second). The bitrate chosen will obviously have a big effect on quality. In general the larger this number the better the quality but the bigger the output file size. However, it is worth noting that this number is only a guideline to the encoder. How strictly the encoder tries to adhere to the value that you set, either on a frame by frame basis, or averaged over the duration of the clip, is controlled by other parameters.

2. Encode Quality vs. Speed

In general, the more time the encoder spends coding each frame the better the job it will do, though it is very much a case of diminishing returns.

<code>--best</code>	This usually gives the best quality output but is extremely slow. In general this is not a recommended setting unless you have a lot of time on your hands.
<code>--good</code>	This will probably be what most users use most of the time. Within the scope of "good" quality there are 6 further speed steps that are set through the <code>--cpu-used</code> parameter (values from 0 to 5). Setting <code>--good</code> quality and <code>--cpu-used=0</code> will give quality that is usually very close to and even sometimes better than that obtained with <code>--best</code> but the encoder will typically run about twice as fast. Setting <code>--cpu-used=1</code> or <code>--cpu-used=2</code> will give further significant boosts to encode speed, but will start to have a more noticeable impact on quality and may also start to effect the accuracy of the data rate control. Setting a value of 4 or 5 will turn off "rate distortion optimisation" which has a big impact on quality, but also greatly speeds up the encoder.
<code>--rt</code>	<p>Real-time mode allows the encoder to auto adjust the speed vs. quality trade-off in order to try and hit a particular cpu utilisation target. In this mode the <code>--cpu-used</code> parameter controls the %cpu target as follows:</p> <div>$\text{target cpu utilisation} = (100 * (16 - \text{cpu-used}) / 16) \%$</div> <p>Legal values for <code>-cpu-used</code> when combined with <code>--rt</code> mode are (0-15). It is worth noting that in <code>--rt</code> mode the encode quality will depend on how hard a particular clip or section of a clip is and how fast the encoding machine is. In this mode the results will thus vary from machine to machine and even from run to run depending on what else you are doing.</p>
<code>--cpu-used</code>	The meaning depends on the mode above. Negative values are for debug and force specific internal speed configurations.

3. Rate Control

This section describes more advanced rate control parameters and 1-pass vs. 2-pass encoding.

VP8, CBR and CQ Mode

VP8 offers VBR (variable bitrate) and CBR (constant bitrate) encoding options, and a VBR variant called CQ (constrained quality) mode.

<code>--end-usage=<arg></code>	(vbr, cbr, cq)
<code>--cq-level=<arg></code>	(valid values 0-63, default 10)

CBR attempts to keep the bitrate more constant, though in most implementations CBR does not actually try to force all frames to be exactly the same size, as this tends to harm video quality. Rather, in CBR mode, the codec tries to remain within given buffering constraints. It can spend a few more bits on one frame or short section, but cannot sustain a higher than average data rate for too long, as its notional buffers will run empty. Likewise, it can choose to "save up" bits during an 01/08/2015 06:48 PM section, but only up to a certain upper limit. If the user sets CBR mode but gives very loose buffer

The WebM Project - VBR Encode Parameter Guide <http://www.webmproject.org/docs/encoder-parameters/>
restrictions, then the result will start to resemble VBR. At the opposite extreme, if the restrictions are very tight, then this mode will move towards true CBR where all frames are encoded as near as possible at the same size.

CQ mode is a special variant of VBR. It is designed as a fire-and-forget mechanism for encoding a large set of clips such that, as much as possible, the output stays within given quality and size constraints across the set. CQ mode exposes an additional parameter (`--cq-level`), and the meaning of the `--target-bitrate` parameter changes to be the "target maximum rate".

In CQ mode the encoder will try to encode normal frames (all frames apart from key frames, golden frames and alternative reference frames) at a quantizer / quality level of `--cq-level` , provided that this does not cause the bitrate to rise above the target maximum value. Key frames, golden frames and alt ref frames may be coded at a lower "q" value, but the minimum is still linked to the user-selected value, and in all cases `--min-q` and `--max-q` are treated as hard limits. In practice this means that easy clips may undershoot the target maximum bitrate, because they are constrained by the CQ level, but harder clips will be bounded by the target maximum data rate and will increasingly revert to standard VBR behavior.

CQ mode is available for one-pass encodes, but is generally intended for two-pass. For one-pass, CQ applies the user `cq-value` , but can't adapt to a higher value if the clip is difficult.

In the two-pass variant of CQ mode there is a further refinement. If the first pass analysis suggests that a clip is too difficult to be encoded at the user-selected `--cq-level` , then rather than code part of the clip at this level and the rest at a much lower quality, it tries to pick a sustainable "auto-cq" level. Under no circumstances will this "auto-cq" value drop below the user-selected value.

```
--buf-initial-sz=<arg>
--buf-optimal-sz=<arg>
--buf-sz=<arg>
```

These three parameters set (respectively) the initial assumed buffer level, the optimal level and an upper limit that the codec should try not to exceed. The numbers given are in 'milliseconds worth of data' so the actual number of bits that these number represent depends also on the target bit rate that the user has set. Typical recommended values for these three parameters might be `4000` , `5000` and `6000` ms, respectively.

```
--undershoot-pct=<arg>      (valid values 1-100)
--overshoot-pct=<arg>       (no longer valid)
```

This parameter causes the codec to try and deliberately undershoot its normal data rate target for each frame in order to cause a notional decoder buffer to fill up. In effect it forces the codec to try and save bits if it can, ready for more difficult sections that it may encounter later. This is occasionally useful in 1-pass CBR mode but should generally be ignored or set to 100 for 2-pass encodes and when using VBR mode.

VBR attempts to distribute the bits between different frames or sections in order to maximize quality. Typically hard sections will be allocated more bits to ensure that the quality in these sections does not drop too low, at the expense of easy sections that will still look good even if coded with a lower than average number of bits per frame. Even in VBR mode though, there typically have to be some constraints on how skewed the distribution of bits can be.

Like many other codecs VP8 offers both one-pass and two-pass encoding. In some situations the choice is obvious. For example a video conferencing or live streaming application can't use two-pass, though for the latter case we are working on a sort of 1.5-pass solution that we call lagged compress which will give some of the benefits of a two-pass encode with a lag or latency of only a few frames.

In general two-pass encoding results in better quality and more accurate data rate control. The idea is that the encoder makes a first pass through the video data and collects statistics about each frame that can then be used to better allocate bits between different frames or sections of the video. Many two-pass (or even multi-pass) encoders do a full encode in the first pass and create a valid output video and this certainly has some advantages, but at the moment the VP8 two-pass encoder only does a partial encode in the first pass that results in a small set of statistics for each frame. In contrast a true one-pass encoder never knows what is coming next so it has to base its encoding decision on recent history. For example when deciding how big to make a key frame at a scene cut it does not know how well the key frame is going to predict subsequent frames (e.g. is it going to be a static scene or is there a lot of motion) or how long it will be until the next key frame. Similarly, because one section is easy to encode does not mean that a later section will also be easy (or vice versa) so it is difficult for a one-pass encoder to distribute more bits to hard sections at the expense of easier sections (see discussion of CBR vs. VBR)

Additional 2-Pass Rate Control Parameters

```
--minsection-pct=<arg>          (recommended value 0-20)
--maxsection-pct=<arg>          (recommended value 200-400 CBR or 400-800 for
VBR)
```

These two parameters set a nominal target bitrate range within which the VBR and CBR algorithms should try and remain when allocating bits to frames or sections. The numbers represent a percentage of the average allocation per frame. The restrictions are less stringent in VBR and in particular are relaxed for certain types of frame (for example key frames or golden frame updates).

```
--bias-pct=<arg>                (recommended value 50)
```

This parameter is misleadingly named in `vpxenc` as it does not tie directly to any sort of percentage. Basically it controls how the two-pass algorithm distributes bits between easier and harder sections or frames, based on complexity statistics gathered for each frame during the first pass. If you select a value of 100 then the allocation will be linear based on the relative complexity value for each frame when compared to the average for the clip (within the limits set by `--minsection-pct` and `--maxsection-pct`). For values of less than 100 the allocation does not increase (or decrease) as sharply in response to a frames relative complexity and a value of 0 means that the complexity is ignored completely when allocating bits. We usually recommend a value of 50.

The following parameters apply to both CBR and VBR modes

```
--min-q=<arg>                    (valid values 0-63, recommended value 0-4)
--max-q=<arg>                    (valid values --min-q to 63, recommended value
50-63)
```

The WebM Project | VP8 Encode Parameter Guide <http://www.webmproject.org/docs/encoder-parameters/>

These two parameters define the range of quantizers that the rate control algorithm may use. A lower number equates to higher quality but more bits (note, however, that these are not real quantizer values just control values). In effect these two parameters can trump all the other rate control parameters. For example if you have set a maximum of 10 then the encoder will never use a quantizer greater than the value represented by 10, even if it massively overshoots the target bit rate. They are useful however, because they allow the user to set upper and lower quality limits for a clip.

4. Key Frame Spacing

VP8 supports automatic detection of scene cuts and insertion of key frames. However the user can also specify a maximum interval between key frames (in frames, so for example at 30 fps 120 would be every 4 seconds).

```
--kf-max-dist=<arg>
--kf-min-dist=<arg>          (not currently supported)
```

5. The Alternate (or Constructed) Reference Frame

The alternate or constructed reference frame is currently only available for two-pass encodes. This frame buffer can be populated with arbitrary data by the encoder and updated in the bitstream but it is never displayed.

```
--auto-alt-ref=<arg>          (0= disabled, 1=enabled <default 0>)
--lag-in-frames=<arg>         (0-25 : recommended value 16)
```

When `--auto-alt-ref` is enabled the default mode of operation is to either populate the buffer with a copy of the previous golden frame when this frame is updated, or with a copy of a frame derived from some point of time in the future (the choice is made automatically by the encoder).

The `--lag-in-frames` parameter defines an upper limit on the number of frames into the future that the encoder can look.

However, many other options are possible and one alternative that has been implemented uses a temporally filtered image derived from a group of future frames. The extra control parameters for this are:

```
--arnr-maxframes=<arg>       (number of frames to filter over 0-25)
--arnr-strength=<arg>         (strength of the temporal filter 0-6)
--arnr-type=<arg>             (not currently supported)
```

Use of `--auto-alt-ref` can substantially improve quality in many situations (though there are still a few where it may hurt). Temporal filtering is experimental and is disabled by default.

6. Multi-threaded Encode and Decode

VP8 supports the use of multiple threads in the encoder and decoder.

```
--threads=<arg> (or -t <arg>) (recommended value : number of real cores - 1)
```

The `--threads` parameter determines the number of threads that will be allocated to the encode process. VP8 supports a mechanism whereby rows of macro-blocks can be simultaneously encoded

The WebM Project - VP8 Encode Parameter Guide <http://www.webmproject.org/docs/encoder-parameters/>
on different threads. However, the entropy encoding stage is limited to 1 thread unless a second parameter, `--token-parts`, is set. It is worth noting that if the threads number is set to `> 1` then the results of repeat encodes will not always be exactly the same.

```
--token-parts=<arg>          (0-3 : recommended 0 for small images, 2 or 3 f  
or HD))
```

Setting the `--token-parts` argument to a non 0 value directs the encoder to split the coefficient encoding across multiple data partitions that can be encoded and decoded independently. At the moment this parameter is interpreted as follows (0 = 1 coefficient partition, 1 = 2 partitions, 2 = 4 partitions, 3 = 8 partitions)

The decoder will usually automatically use an appropriate number of threads according to how many cores are available but it can only use multiple threads for the coefficient data if the encoder selected `--token-parts > 0` at encode time.

7. Temporal and Spatial Resampling

VP8 supports both temporal and spatial resampling. These are specialist parameters and are not generally recommended. Temporal resampling is only used in CBR mode and causes the encoder to drop frames if it cannot prevent its notional buffer from running empty in any other way. Spatial resampling involves scaling the image down to a smaller size in the encoder (as an alternative method for reducing the number of bits per frame to increasing the quantizer) and then scaling it back up in the decoder. Note that frames can be dropped at any time but the encoder can only change its spatial re-sampling ratio on a key frame.

```
--drop-frame=<arg>          (0=disabled to 100)
```

The drop frame parameter specifies a buffer fullness threshold at which the encoder starts to drop frames as a percentage of the optimal value specified by `--buf-optimal-sz`. If it is set to 0 then dropping of frames is disabled.

```
--resize-allowed=<arg>      (0 disabled, 1 enabled)  
--resize-down=<arg>         (0-100)  
--resize-up=<arg>           (--resize-down-100)
```

The resize up and down parameters are high and low buffer fullness "watermark" levels at which we start to consider changing down to a smaller internal image size, if the buffer is being run down, or back up to a larger size if the buffer is filling up again. The numbers represent a percentage of `--buf-optimal-sz`.

8. Video Conferencing

```
--error-resilient=<arg>     (0 disabled, 1 enabled <default 0>)
```

In error resilient mode encoder context tables are updated to a fully defined state not just on key frames but also whenever a "Golden Frame" is encoded (a special kind of frame that is usually encoded at a higher quality, that updates the "Golden Frame reference buffer". This frame can then be used to quickly recover if frame packets are dropped without the need to code a full key frame. Error resilient mode is *not* recommend for other scenarios.

`--static-thresh=<arg>`

The static threshold imposes a change threshold on blocks below which they will be skipped by the encoder. This can be used to suppress signal noise and enhance the encode speed in situations where there are low levels of real movement. Values of above 1000 are not recommended and any non zero value runs the risk of introducing artifacts caused by regions of the image not being updated. In most scenarios this value should be set to 0.

9. Miscellaneous

`--profile` (0-3: default and recommended value = 0)

This parameter sets the encoder profile. For non-zero values the encoder increasingly optimizes for reduced complexity playback on low powered devices at the expense of encode quality. For example using 1 tells the encoder only to use only bi-linear sub pixel filtering and a simplified loop filter. In general most users will want to set a value of 0 or ignore this parameter unless they are encoding high resolution content and require playback on very low power devices.

`--sharpness=<arg>` (0-7 : default and recommended value = 0)

This parameter affects the loop filter. Anything above 0 weakens the deblocking effect of the loop filter.

`--noise-sensitivity=<arg>` (0-6: default and recommended value = 0)

The encoder includes a crude temporal noise filter. There are better filtering options available in specialist pre-processing products, so unless you are encoding a very noisy source and have no easy alternatives this should be set to 0. Non-zero values equate to increasingly strong filtration.

`--tune=<arg>` (psnr, ssim: default = psnr)

Optimize output for PSNR or SSIM quality measurement. Certain input data or modifications to `vp8enc` benefit PSNR and harm SSIM, or *vice versa*. The `--tune` parameter can be used to explicitly optimize for one or the other index.

`--timebase=<arg>` (default = 1/1000)

The desired precision of timestamps in the output, expressed in fractional seconds. Default is `1/1000` (1 ms).

10. Sample Command Lines

In each case parameters that are particularly relevant to the scenario are highlighted.

2-Pass Best Quality VBR Encoding

```
vpxenc input_1280_720_30fps.yuv -o output_vp8.webm \  
--codec=vp8 --i420 -w 1280 -h 720 -p 2 -t 4 \  
--best --target-bitrate=2000 --end-usage=vbr \  
--auto-alt-ref=1 --fps=30000/1001 -v \  
--minsection-pct=5 --maxsection-pct=800 \  
--lag-in-frames=16 --kf-min-dist=0 --kf-max-dist=360 \  
--token-parts=2 --static-thresh=0 --drop-frame=0 \  
--min-q=0 --max-q=60
```

Alternative option to `--best` use `--good --cpu-used=0`

2-Pass Faster VBR Encoding

```
vpxenc input_1280_720_30fps.yuv -o output_vp8.webm \  
--codec=vp8 --i420 -w 1280 -h 720 -p 2 -t 4 \  
--good --cpu-used=1 --target-bitrate=2000 --end-usage=vbr \  
--auto-alt-ref=1 --fps=30000/1001 -v \  
--minsection-pct=5 --maxsection-pct=800 \  
--lag-in-frames=16 --kf-min-dist=0 --kf-max-dist=360 \  
--token-parts=2 --static-thresh=0 \  
--min-q=0 --max-q=60
```

2-Pass VBR Encoding for Smooth Playback on Low-end Hardware

```
vpxenc input_1280_720_30fps.yuv -o output_vp8.webm \  
--codec=vp8 --i420 -w 1280 -h 720 -p 2 -t 4 \  
--good --cpu-used=0 --target-bitrate=2000 --end-usage=vbr \  
--auto-alt-ref=1 --fps=30000/1001 -v \  
--minsection-pct=15 --maxsection-pct=400 \  
--lag-in-frames=16 --profile=1 \  
--kf-min-dist=0 --kf-max-dist=360 --static-thresh=0 \  
--min-q=4 --max-q=63
```

2-Pass CBR Encoding for Limited-bandwidth Streaming

```
vpxenc input_640_360_30fps.yuv -o output_vp8.webm \  
--codec=vp8 --i420 -w 640 -h 360 -p 2 \  
--good --cpu-used=0 --target-bitrate=400 --end-usage=cbr \  
--undershoot-pct=95 \  
--buf-sz=6000 --buf-initial-sz=4000 --buf-optimal-sz=5000 \  
--drop-frame=70 --fps=30000/1001 -v \  
--kf-min-dist=0 --kf-max-dist=360 --static-thresh=0 \  
--min-q=4 --max-q=63
```

It may be necessary to use `--drop-frame` (perhaps set to around 25) and/or `--resize-allowed` (see section on temporal and spatial resampling above) if hitting buffer constraints is an **absolute** requirement.


```
vp8enc input_1280_720_30fps.yuv -o output_vp8.webm \
--codec=vp8 --i420 -w 1280 -h 720 -p 2 -t 4 \
--good --cpu-used=0 --target-bitrate=2000 --end-usage=vbr \
--auto-alt-ref=1 --fps=30000/1001 -v \
--minsection-pct=5--maxsection-pct=800 --lag-in-frames=16 \
--kf-min-dist=0 --kf-max-dist=360 \
--token-parts=2 \
--min-q=4 --max-q=60 \
--arnr-maxframes=5 --arnr-strength=3
```

1-Pass Good Quality VBR Encoding

```
vp8enc input_1280_720_30fps.yuv -o output_vp8.webm \
--codec=vp8 --i420 -w 1280 -h 720 -p 1 -t 4 \
--good --cpu-used=0 --target-bitrate=2000 --end-usage=vbr \
--fps=30000/1001 -v \
--kf-min-dist=0 --kf-max-dist=360 \
--token-parts=2 --static-thresh=0 \
--min-q=0 --max-q=63
```

1-Pass Fast VBR Encoding

```
vp8enc input_1280_720_30fps.yuv -o output_vp8.webm \
--codec=vp8 --i420 -w 1280 -h 720 -p 1 -t 4 \
--good --cpu-used=3 --target-bitrate=2000 --end-usage=vbr \
--fps=30000/1001 -v \
--kf-min-dist=0 --kf-max-dist=360 \
--token-parts=2 --static-thresh=1000 \
--min-q=0 --max-q=63
```

Real-time CBR Encoding and Streaming

```
vp8enc input_640_480_15fps.yuv -o output_vp8.webm \
--codec=vp8 --i420 -w 640 -h 480 -p 1 -t 4 \
--rt --cpu-used=4 --end-usage=cbr --target-bitrate=500 \
--fps=15000/1001 --undershoot-pct=95 \
--buf-sz=6000 --buf-initial-sz=4000 --buf-optimal-sz=5000 -v \
--kf-max-dist=999999 \
--min-q=4 --max-q=56
```

It may be necessary to use `--drop-frame` (perhaps set to around 25) and/or `--resize-allowed` (see section on temporal and spatial resampling above) if hitting buffer constraints is an **absolute** requirement.

11. vpxenc Parameter Summary

Usage

```
vp8enc <options> -o dst_filename src_filename
```

Options

-D,	--debug	Debug mode (makes output deterministic)
-o <arg>,	--output=<arg>	Output filename
	--codec=<arg>	Codec to use (vp8 or vp9)
-p <arg>,	--passes=<arg>	Number of passes (1/2)
	--pass=<arg>	Pass to execute (1/2)
	--fpf=<arg>	First pass statistics file name
	--limit=<arg>	Stop encoding after n input frames
	--skip=<arg>	Skip the first n input frames
-d <arg>,	--deadline=<arg>	Deadline per frame (usec)
	--best	Use Best Quality Deadline
	--good	Use Good Quality Deadline
	--rt	Use Realtime Quality Deadline
-q,	--quiet	Do not print encode progress
-v,	--verbose	Show encoder parameters
	--psnr	Show PSNR in status line
	--ivf	Output IVF (default is WebM)
-P,	--output-partitions	Makes encoder output partitions. Requires IVF output!
	--q-hist=<arg>	Show quantizer histogram (n-buckets)
	--rate-hist=<arg>	Show rate histogram (n-buckets)

Encoder Global Options

	--yv12	Input file is YV12
	--i420	Input file is I420 (default)
-u <arg>,	--usage=<arg>	Usage profile number to use
-t <arg>,	--threads=<arg>	Max number of threads to use
	--profile=<arg>	Bitstream profile number to use
-w <arg>,	--width=<arg>	Frame width
-h <arg>,	--height=<arg>	Frame height
	--stereo-mode=<arg>	Stereo 3D video format mono, left-right, bottom-top, top-bottom, right-left
	--timebase=<arg>	Output timestamp precision (fractional seconds)
	--fps=<arg>	Stream frame rate (rate/scale)
	--error-resilient=<arg>	Enable error resiliency features
	--lag-in-frames=<arg>	Max number of frames to lag

Rate Control Options

--drop-frame=<arg>	Temporal resampling threshold (buf %)
--resize-allowed=<arg>	Spatial resampling enabled (bool)
--resize-up=<arg>	Upscale threshold (buf %)
--resize-down=<arg>	Downscale threshold (buf %)
--end-usage=<arg>	Rate control mode vbr, cbr, cq
--target-bitrate=<arg>	Bitrate (kbps)
--min-q=<arg>	Minimum (best) quantizer
--max-q=<arg>	Maximum (worst) quantizer
--undershoot-pct=<arg>	Datarate undershoot (min) target (%)
--overshoot-pct=<arg>	Datarate overshoot (max) target (%)
--buf-sz=<arg>	Client buffer size (ms)
--buf-initial-sz=<arg>	Client initial buffer size (ms)
--buf-optimal-sz=<arg>	Client optimal buffer size (ms)

Twopass Rate Control Options

--bias-pct=<arg>	CBR/VBR bias (0=CBR, 100=VBR)
--minsection-pct=<arg>	GOP min bitrate (% of target)
--maxsection-pct=<arg>	GOP max bitrate (% of target)

Keyframe Placement Options

--kf-min-dist=<arg>	Minimum keyframe interval (frames)
--kf-max-dist=<arg>	Maximum keyframe interval (frames)
--disable-kf	Disable keyframe placement

VP8-Specific Options

--cpu-used=<arg>	CPU Used (-16..16)
--auto-alt-ref=<arg>	Enable automatic alt reference frames
--noise-sensitivity=<arg>	Noise sensitivity (frames to blur)
--sharpness=<arg>	Filter sharpness (0-7)
--static-thresh=<arg>	Motion detection threshold
--token-parts=<arg>	Number of token partitions to use, log2
--arnr-maxframes=<arg>	AltRef Max Frames
--arnr-strength=<arg>	AltRef Strength
--arnr-type=<arg>	AltRef Type
--tune=<arg>	Material to favor psnr, ssim
--cq-level=<arg>	Constrained Quality Level
--max-intra-rate=<arg>	Max I-frame bitrate (pct)

VP9-Specific Options

```

--cpu-used=<arg>          CPU Used (-16..16)
--auto-alt-ref=<arg>      Enable automatic alt reference
                           frames
--noise-sensitivity=<arg>  Noise sensitivity (frames to blur)
--sharpness=<arg>         Filter sharpness (0-7)
--static-thresh=<arg>     Motion detection threshold
--tile-columns=<arg>      Number of tile columns to use, log2
--tile-rows=<arg>         Number of tile rows to use, log2
--arnr-maxframes=<arg>    AltRef Max Frames
--arnr-strength=<arg>     AltRef Strength
--arnr-type=<arg>         AltRef Type
--tune=<arg>              Material to favor
                           psnr, ssim
--cq-level=<arg>          Constrained Quality Level
--max-intra-rate=<arg>    Max I-frame bitrate (pct)
--lossless=<arg>         Lossless mode
--frame-parallel=<arg>    Enable frame parallel decodability
                           features

```

ABOUT

About WebM
[\(/about/\)](/about/)
 FAQ [\(/about/faq/\)](/about/faq/)
 Discuss [\(/about/discuss/\)](/about/discuss/)
 Supporters
[\(/about/supporters/\)](/about/supporters/)



MORE

Tools [\(/tools/\)](/tools/)
 Hardware
[\(/hardware/\)](/hardware/)
 Licenses
[\(/license/\)](/license/)
 Downloads

DEVELOPER

Overview [\(/code/\)](/code/)
 Contribute [\(/code/contribute/\)](/code/contribute/)
 Submitting
 Patches [\(/code/patches/\)](/code/patches/)
 Code Reviews
[\(/code/contribute/code-reviews/\)](/code/contribute/code-reviews/)
 Workflow [\(/code/contribute/workflow/\)](/code/contribute/workflow/)
 Conventions
[\(/code/contribute/conventions/\)](/code/contribute/conventions/)
 Bug Reporting
[\(/code/bug-reporting/\)](/code/bug-reporting/)
 Build Prerequisites
[\(/code/build-prerequisites/\)](/code/build-prerequisites/)
 Repository Layout
[\(/code/repository-layout/\)](/code/repository-layout/)

DOCS

libvpx API [\(/docs/vp8-sdk/\)](/docs/vp8-sdk/)
 RFC 6386: VP8 Data Format

 WebM Container Format
[\(/docs/container/\)](/docs/container/)
 VP8 RTP Proposal (Draft) 
 Encoder Examples [\(/docs/encoder-parameters/\)](/docs/encoder-parameters/)
 Wiki
[\(http://wiki.webmproject.org/\)](http://wiki.webmproject.org/)

Copyright 2010 - 2014

The WebM Project



Follow @WebM

webmaster@webmproject.org
[\(mailto:webmaster@webmproject.org\)](mailto:webmaster@webmproject.org)

