

How to differentiate your HD multimedia design

By Steve Leibson
Strategic Marketing Manager
Tensilica Inc.

There's an explosion occurring in consumer video. It's fueled by the adoption of digital video-compression standards, which deliver excellent picture fidelity and give consumers the ability to shape and mold video for their own viewing preferences. The popularity of digital video has revived the consumer TV market, boosting TVs from the low-margin, bargain-basement category to highly featured, wall-size, high-margin flat screens. At the same time, high-definition (HD) disc players, digital video recorders, cable, satellite and terrestrial broadcast STBs, and camcorders also lure their share of consumers' wallets and pocketbooks. After decades of slumber, video is once again a hot commodity in CE.

Differentiation

Consequently, product differentiation through video-enhancing features has also become important. Consumers directly compare picture quality between products that are on display in stores and make purchasing decisions based on perceived picture quality, image fidelity and color presentation. While digital video compression codecs are at the heart of the video revival as a hot consumer-product category, these codecs are now standardized for HD video products and do not offer much room for product differentiation. Instead, consumer-product developers have discovered that video pre- and post-processing algorithms (performed before video encoding and after video decoding) are the tools they need to make their HD offerings stand out in a very crowded market.

HD digital-video codecs are standardized to ensure interoper-

ability. Standard encoders must be able to drive standard decoders or interoperability suffers, as will sales for the entire category. Consequently, there's not much room for differentiation within the digital-video codecs. That's a potential disadvantage for vendors that wish to differentiate their products but it's an advantage for end-product designers because codec developers can design very efficient hardware to implement the codecs. The opportunity for product differentiation and the need for programmability now reside in the video pre- and post-processing blocks that improve upon the picture and color fidelity delivered by the digital HD codecs.

Pre-processing algorithms

Unsurprisingly, video doesn't stream off of a sensor in pristine condition. There is a long list of transformations that can be performed on this raw video to improve the image before it's encoded. Some of the pre-processing operations include the following:

- Pixel scan/data transfer—This operation simply gets the image off of the sensor.
- Bayer pattern deinterleaving—Modern video imaging places a three-color RGB Bayer filter over a monochromatic image sensor. The data streaming off of the sensor therefore contains red, green and blue information that must be separated so that the image can be converted to YCbCr luma and chroma representation.
- Noise filtering—Nothing electronic occurs in the absence of noise. The best place to filter or eliminate noise is before encoding.
- Shake detection and compensation—They can reduce

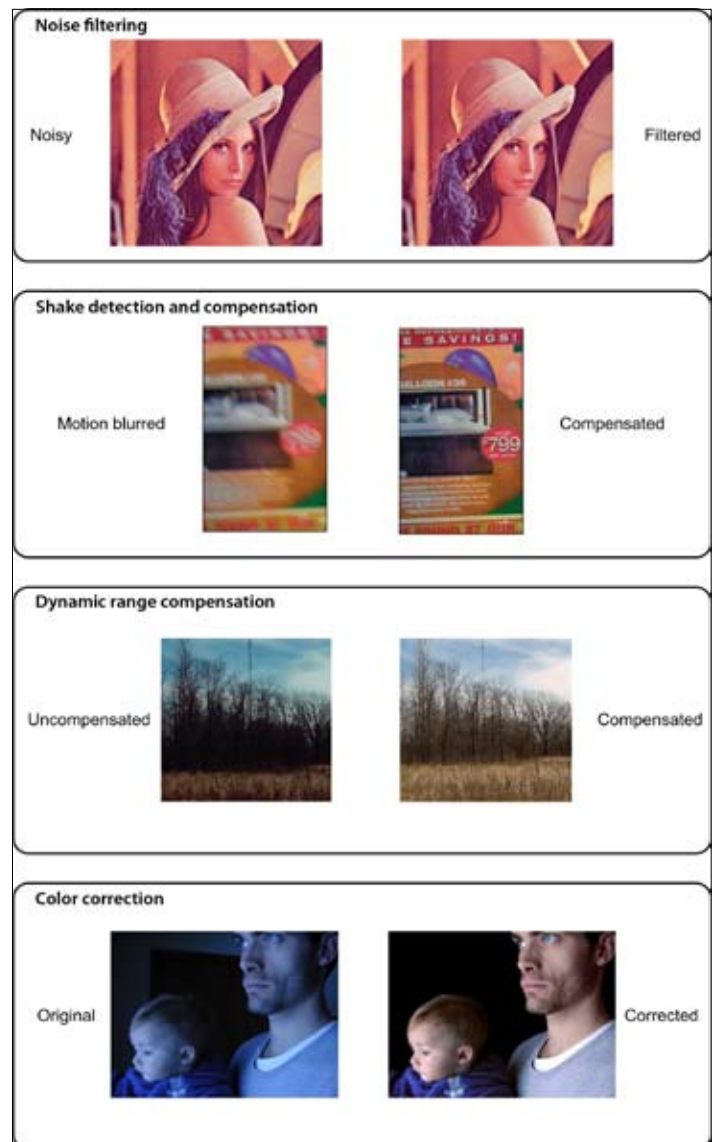


Figure 1: Shown are examples of four pre-processing algorithms: noise filtering, shake detection and compensation, dynamic range compensation, and color correction.

the image degradation caused by camera shake.

- Localized dynamic range compensation—An image's dynamic range can exceed that of a sensor operated in a particular capture mode. An appropriately equipped camera or camcorder can intelligently adjust exposure within a frame to expand the sensor's basic dynamic range.
- Focus adjustment (sharpening)—Image sensors do not

capture continuous images. Instead, they break the image into pixels, which must then be interpolated and re-assembled into an image at a specified resolution. These operations cause a loss of sharpness, which can be corrected with the appropriate pre-processing algorithm. In addition, an image that's slightly out of focus because of an improperly focused lens can be brought into

sharper apparent focus using a sharpening algorithm.

- Color correction—White light varies depending on color temperature, which affects the resulting captured video. In addition, different image-presentation systems alter the color in various ways. Video color correction compensates for these color-shifting factors, which are especially apparent with skin tones.
- Face detection—In images populated with people, the videographer generally wishes for the faces to be in focus. A video-capture device that can recognize faces can do a better job of setting the camera's or camcorder's focus.
- Stereoscopic imaging—It is possible to synthesize stereoscopic images from density information in a flat image. This sort of pre-processing can be used, for example, in mixed-reality systems that combine virtual and real imagery.

Figure 1 shows graphic examples of four pre-processing algorithms: noise filtering, shake detection and compensation, dynamic range compensation, and color correction.

Post-processing operations

Video emitted from a standardized video decoder can also have many imperfections, and post-processing algorithms can greatly enhance the resulting image. Products with visibly better images sell better for higher prices. Video post-processing operations include the following:

- Deblocking/deringing filters—Video compression and decompression break images into blocks, encode them and then put the blocks back together at the other end. These operations leave visible artifacts that can be reduced through deblocking/deringing filters.
- Edge detection—Edge-detection improves scaling, deinterlacing and other video-processing operations. In addition,

edge detection is important in video applications such as security and video surveillance, traffic management, and medical imaging.

- Scaling—With the explosion of new wall, desktop, and handheld video devices, screen sizes vary widely. Chances are good that an encoded video stream will not be the correct size for many of these video devices. Image scaling solves this problem by resizing the image size to fit the screen.
- Deinterlacing—Interlaced video is designed for TV viewing: each interlaced video frame is displayed as two consecutive fields, each field containing the odd- or even-numbered lines of video. Therefore, each field contains half the lines of the frame in an alternating fashion. However, video displayed on computer and TV LCDs needs to be progressively scanned where every line of video appears in each consecutive frame. Interlaced video must be de-interlaced before display on a non-interlaced screen.
- Frame rate conversion—Video frame-rate upconversion through interpolation can increase the perceived smoothness of motion in displayed video by creating interpolated video frames between the decoded frames emerging from the video decoder. This ability can also be important for high-end, 120fps displays and for upconverting 24fps film-based video to normal video rates.
- Noise filtering—All manner of noise sources can degrade video images. There are a variety of noise-filtering algorithms that improve the picture depending on the noise present.
- Video overlays/alpha blending—Many video systems generate imagery such as user interfaces that must be overlaid on top of live video.
- Color-space conversion/brightness/contrast/gamma correction—Different displays have different dynamic ranges

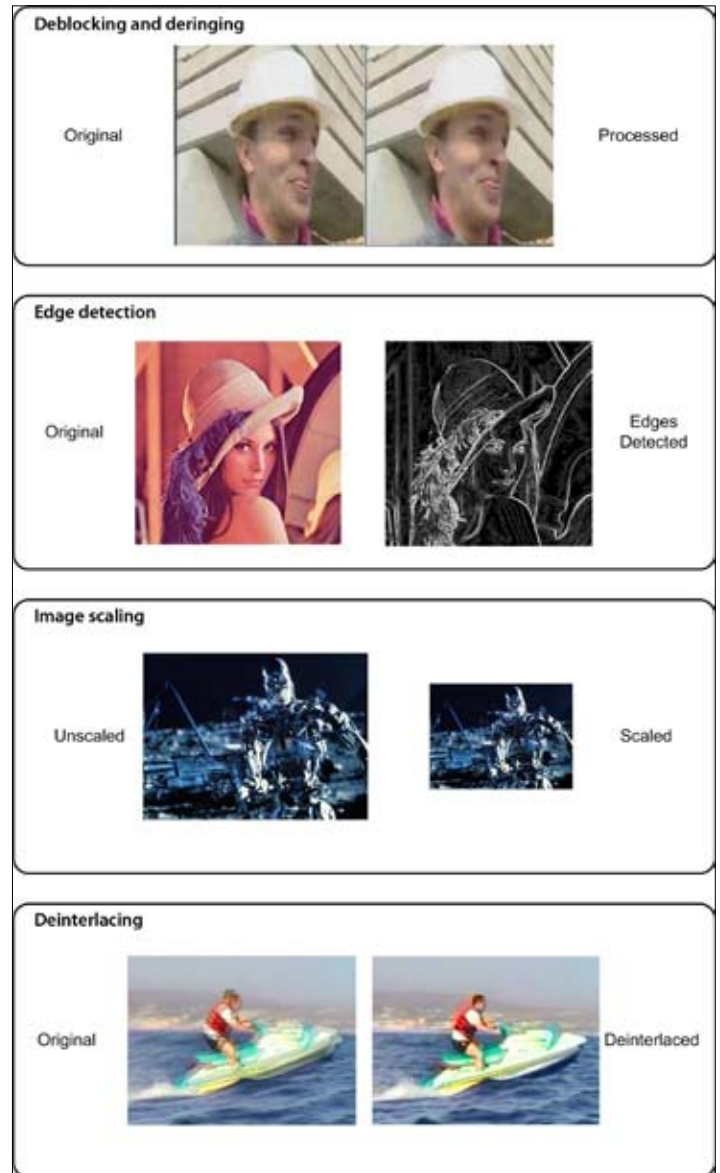


Figure 2: Shown are examples of four post-processing algorithms: deblocking and deringing, edge detection, image scaling, and deinterlacing.

and handle colors differently (different color transfer functions). Appropriate video post processing can make the most of a display when its characteristics are known. Gamma correction is another important processing step because LCD panels each have unique transfer functions.

Figure 2 shows examples of four post-processing algorithms: deblocking and deringing, edge detection, image scaling, and deinterlacing.

Implementation

Unlike video compression/decompression codecs, video pre- and post-processing algorithms

are not standardized. They vary by end product. In addition, these algorithms are constantly being updated and new ones regularly appear. As a result, hardware that implements video pre- and post-processing algorithms needs far more programmability than does video codec hardware. Processors that are specially configured to run these video pre- and post-processing algorithms are therefore a good design solution for such tasks.

You may not be familiar with the concept of configurable processors but you are no doubt familiar with fixed-ISA processor cores, which have been in use since the first days of SoCs. Fixed-ISA processor cores extend

the long tradition of using standard processor architectures, first established in 1971 with Intel's introduction of the 4004, the industry's first commercial single-chip microprocessor. The advent of SoCs, which are nothing more than ASICs with on-chip processor, in the mid 1990s changed the equation with respect to microprocessors. As long as the silicon is going to be custom-tailored for a specific application, the on-chip processors can be custom-tailored as well. However, the relative lack of processor designers and the complexity of developing and maintaining an associated tool chain have stopped designers from using custom-tailored processors in the past.

Automated tools are now available that allow logic designers and software developers—specifically not processor designers—to custom-tailor processors for specific on-chip SoC tasks. The customized processors are optimized to run targeted algorithms faster than can general-purpose processors and DSPs because new registers and instructions have been added so that multi-instruction sequences become single-instruction sequences. Therefore, it should not be surprising that it is possible to develop video-specific processor architectures that deliver substantial performance boosts to various pre- and post-processing video algorithms. There are several ways to tackle video algorithms. The first way is simple: take the algorithm written in C, compile it for a processor, run it, and profile it. It's very easy to see the performance difference between processor architectures using this approach.

The table shows the result of such an approach for five different video algorithms running on a RISC processor, on a DSP, on a processor that has been customized to efficiently run the five video algorithms using an automated processor customization tool, and for a processor that has been directly optimized by a design engineer. The RISC processor used in this example is a version

Benchmark	Results			
	Pure C	Vectra LX	XPRES	Direct customization
5x5 2D filter	163 cycles	7.1 cycles	4.7 cycles	0.96 cycles
5x5 threshold filter	338 cycles	7.4 cycles	7.1 cycles	1.5 cycles
Median filter	41 cycles	1.9 cycles	0.8 cycles	
Frame subsample	3.1 cycles	0.3 cycles	0.2 cycles	
Frame sum	4.4 cycles	0.8 cycles	0.4 cycles	<0.1 cycles
Area	0.25mm ²	0.63mm ²	0.59mm ²	0.70mm ² (see notes)

Notes:

- 8 cycles/pixel budget @ 500MHz to achieve 62Mpixels/s (1,080p video)
- Area values are for TSMC 65LP process, synthesized to 300MHz target
- 2D filter area increment 0.35mm² over base core
- Threshold filter area increment 0.09mm²
- Frame sum area increment 0.01mm²
- All area values are for TSMC65 LP, 300MHz target

Table: Video algorithm performance results for a RISC processor vs. DSP vs. processor customized with Tensilica's XPRES compiler are listed.

of Tensilica's Xtensa RISC core. The DSP is also a version of Tensilica's RISC core, but the core has been extended with a comprehensive set of DSP instructions including a four-way SIMD MAC and DSP ALU extensions that can perform four- and eight-way SIMD vector operations. The processor has also been augmented so that it can issue three independent instructions per clock cycle. This set of instruction extensions is called Vectra LX. Even though these two processors are quite different, they are both built on the same architectural foundation.

The video-processing algorithms listed in the table are:

- 5 x 5pixel 2D filter—Two-dimensional filters can be used for several operations including sharpening images, adding blur, or to add dithering.
- 5 x 5pixel threshold filter—Threshold filtering reduces image pixel values to one of two values—Zero (whiter than white) or One (blacker than black)—effectively creating 1bit pixels. The result is a strictly black-and-white image with no shades of gray and the resulting image is used, for example, by noise-detection algorithms.
- Median filter—This sort of filter is often used to reduce the effects of salt-and-pepper or Gaussian noise.

- Frame subsample—A simplistic image downscaler.
- Frame sum—This algorithm can be used to measure the amount of light in different portions of the video frame, which is used for noise reduction and exposure compensation, and to aid some types of video compression.

The SIMD/DSP-augmented version of the Xtensa processor is much faster than the unaugmented version. The performance improvement is about an order of magnitude or more (sometimes much more) for most of the video-processing algorithms. Yet the increase in silicon area is very small in absolute terms (about 0.4mm² for Taiwan Semiconductor Manufacturing Co. Ltd's 65LP process).

The column labeled "XPRES" provides performance numbers for a version of Tensilica's Xtensa processor that was customized to efficiently handle all five video algorithms using the XPRES compiler, which accepts C source code and automatically generates alternative processor architectures with ISA enhancements that accelerate performance for the submitted code. The resulting processor is slightly smaller than the Xtensa processor core augmented with VectraLX extensions yet it outperforms the VectraLX

version of the Xtensa processor by a factor of two or more on most of the algorithms.

The last column shows how the addition of custom instructions and enhanced I/O interfaces accelerates the processor core's performance for these particular tasks. Note that the 5x5 2D filter runs 170 times faster on the processor with directly customized instructions than on the unaugmented Xtensa processor core; the 5x5 threshold filter runs 225 times faster than on the unaugmented Xtensa processor core; the frame summation runs 44 times faster than on the unaugmented Xtensa processor.

Product differentiation through video-enhancing features has become critically important in consumer video markets. Consumers directly compare picture images and choose purchases based on the best perceived picture. While digital video-compression codecs are at the heart of the video revival as a hot consumer-product category, these codecs are standardized and cannot be used for product differentiation. Consequently, video pre- and post-processing algorithms can help video products stand out in a crowded market. Customized processor cores provide a fast, easy path to implement these algorithms with all of the performance your project requires.