©2016 Society for Imaging Science and Technology
DOI: 10.2352/ISSN.2470-1173.2016.2.VIPC-227

Sony ARW2 Compression: Artifacts And Credible Repair

Henry Gordon Dietz; Department of Electrical and Computer Engineering, University of Kentucky; Lexington, Kentucky

Abstract

Although most cameras produce JPEG-encoded images ready for viewing, many also offer the ability to save the digitized, but otherwise unprocessed, sensor data in a file so that more sophisticated processing can be applied later. For recent Sony cameras, the raw data is encoded in a format known as ARW (Alpha RaW). Controversially, ARW version 2.0 and later employ lossy compression for 32x1-pixel blocks, combining a non-linear reduction of value range and a form of delta encoding. This compression has been associated with occasionally severe visual artifacts. In this paper, the artifacts observed with real cameras using this compression scheme are characterized and several algorithms for credible repair of those artifacts, including computational texture synthesis, are presented and experimentally evaluated.

Introduction

Image compression and resulting image quality have been well studied for various schemes. Perhaps the most detailed characterizations of compression artifacts appear in the literature about image anti-forensics[1], where algorithms are developed primarily to make artifacts from prior JPEG compression undetectable for forensic use. In general, there is a large body of work investigating removal of blocking compression artifacts, but again, primarily with respect to JPEG DCT blocking.

Sony's ARW2 compression in particular has not been the target of much academic research, but has been of great interest to the public. Articles[2][3][4] have explained how this compression causes artifacts and how to manually identify the artifacts within images. In contrast, this paper characterizes properties of ARW2 artifacts and an effective repair strategy is developed and evaluated.

Our work suggests that the lossy compression itself is not the entire problem. Much of the problem seems to be rooted in the class-leading dynamic range of Sony sensors, which inspires extreme lifting of shadows that makes artifacts much more visible – and interpolation schemes like AMaZE tend to enhance the accidental edges produced by posterization. It also seems that under certain circumstances the data recorded for some pixels might simply be wrong.

The repair algorithm presented here is quite novel. It uses a form of texture synthesis, but is actually driven by modeling the error bounds for each pixel value. The repair also is unusual in that it is implemented by directly processing a raw image file to output a repaired raw.

A brief history of lossy ARW2 compression

The lossy-compressed ARW2 format that is the subject of this paper was introduced in 2007 in the APS-C DSLR-A700 and the full-frame DSLR-A850 and A900. In those cameras, the user was actually given a choice between lossy compression (which the manual called "cRAW") and a simple uncompressed format ("RAW"). The analog to digital conversion hardware of those three cameras produced just 12 bits per pixel, so the uncompressed format simply packed 12-bit samples. The very clever cRAW encoding produces files are only about 2/3 the size of RAW files – roughly one byte per pixel.

The smaller cRAW file size speeds camera operation, especially time to write an image to a memory card. It also effectively increases image memory capacity. With very few people saying they saw a difference in image quality, it is not surprising that Sony dropped the uncompressed option in their next set of camera bodies. In fact, complaints about compression artifacts continued to be rare until very recently, and Sony continued to use their lossy compression algorithm in at least the following camera models spanning from 2007 to current production:

Full Frame E-mount: NEX-VG900 and ILCE-7, 7R, 7S, 7M2, 7RM2, 7SM2
Full Frame A-mount: DSLR-A850, A900, and SLT-A99
APS-C E-mount: NEX-3, 5, 5N, 5R, 5T, 6, 7, C3, F3, VG20, VG30, and ILCE-3000, 3500, 5000, 5100, 6000, and QX1
APS-C A-mount: DSLR-A450, A500, A550, A560, A580, A700, and SLT-A33, A35, A37, A55, A57, A58, A65, A77, and ILCA-77M2
Cyber-shot: DSC-RX100, RX100M2, RX100M3, RX100M4, RX10, RX10M2, RX1

It is important to note that while the cRAW encoding did not change, properties of the camera sensor datapath did. Dynamic range of the sensors has increased notably. In addition, 14-bit analog to digital conversion is now common, so the compression is now relatively more aggressive: the effective compression factor went from 8/12 to 8/14. Although hundreds of user images from a wide range of Sony camera models have been processed with the KARWY repair tool described later in this paper, lossy-compression artifacts are not obvious in any of the images from the older 12-bit cameras.

Perhaps the real issue making artifacts more visible is the concept of ISO-less exposure, which we formally introduced at Electronic Imaging 2015[5] and was independently popularized by DPReview as "ISO invariance"[3]. Fundamentally, a camera is ISO-less if the image data captured is of sufficiently high quality at low film-speed settings so that increasing the ISO setting used (i.e., the analog gain) does not result in any improvement in the quality of the post-processed final result. ISO-less exposure methods thus deliberately bias exposure to take advantage of the full dynamic range of the sensor, by conventional standards often grossly underexposing and then digitally boosting the brightness

©2016 Society for Imaging Science and Technology
DOI: 10.2352/ISSN.2470-1173.2016.2.VIPC-227

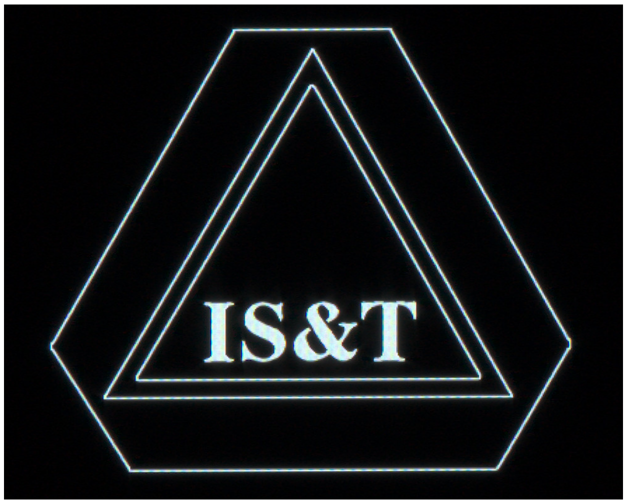


Figure 1. *Normal processing of compressed capture of logo*

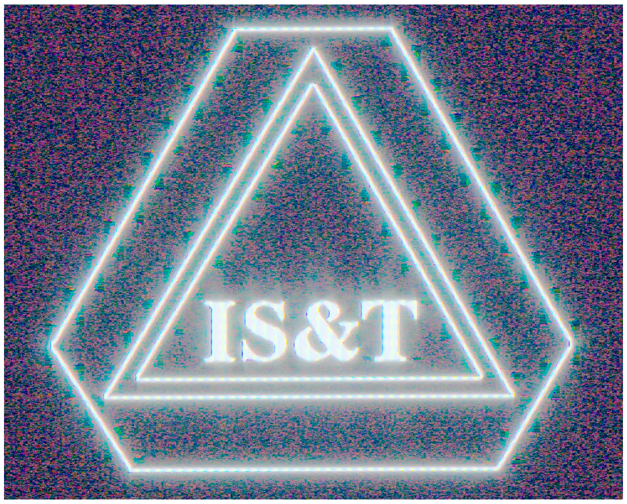


Figure 2. *Boosted compressed capture of logo*

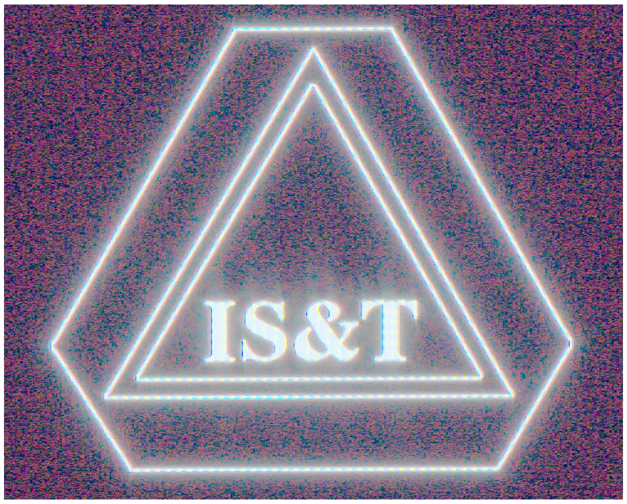


Figure 3. *Boosted KARWY repaired compressed logo*

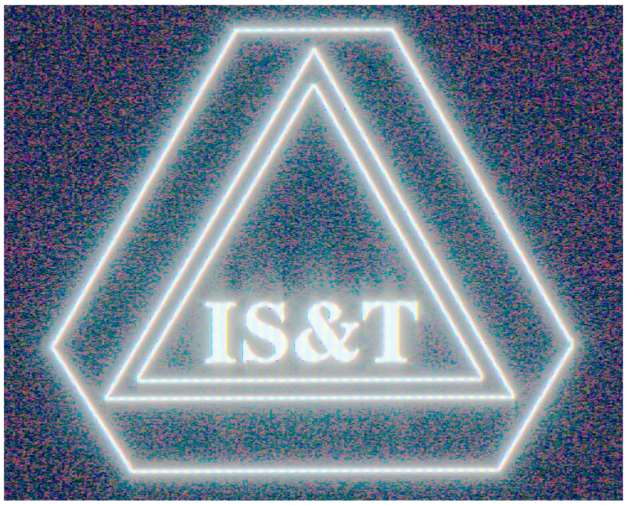


Figure 4. *Boosted uncompressed capture of logo*

of the captured raw image. Boosting by as much as 6 stops or more has been advocated, and this greatly increases the visibility of the artifacts associated with lossy-compression.

In Fall 2015, DPReview ran a series of articles praising Sony's ISO-less performance and cursing Sony's use of lossy-compressed raw for the artifacts the digital exposure boosting made visible. At the same time, we produced KARWY – the tool described in the current paper which performs credible computational repair of the offending raw artifacts. Within a month, Sony announced that they would be providing a firmware update for the ILCE-7RM2 offering the option of using an uncompressed raw format, and this was soon followed by release of the new ILCE-7SM2 including this support and a firmware update providing the same functionality for the somewhat older ILCE-7M2.

There are many millions of lossy-compressed ARW2 files already taken, and even some current models (e.g., the ILCE-7 and ILCE-6000) do not provide an uncompressed option, so there still is good reason to computationally improve lossy-compressed ARW2 images. However, the introduction of an uncompressed raw option offers the opportunity to judge repair quality not only by appeal of the image, but by direct comparison with an uncompressed capture of the same scene.

To demonstrate the artifacting problem, we prepared a test target which is a version of the IS&T logo edited to have thin lines with extreme contrast – the type of scene structure known to be prone to lossy-compression artifacts. This target was photographed as displayed on a monitor with a 5000:1 contrast ratio. Figure 1 is a 1000x800 pixel crop from a normally-processed image that was captured using the lossy-compressed raw format of the ILCE-7M2; it shows some texture from the pixels of the monitor, but no compression artifacts are apparent. However, digitally boosting that image by about 6 stops produces Figure 2, revealing "boxy" artifacts along all high-contrast lines that have a vertical component. There is also some glow around the lines and a heavily-colored noise pattern in the dark areas, but those are not due to the compression – they also are visible in Figure 4,

DOI: 10.2352/ISSN.2470-1173.2016.2.VIPC-227

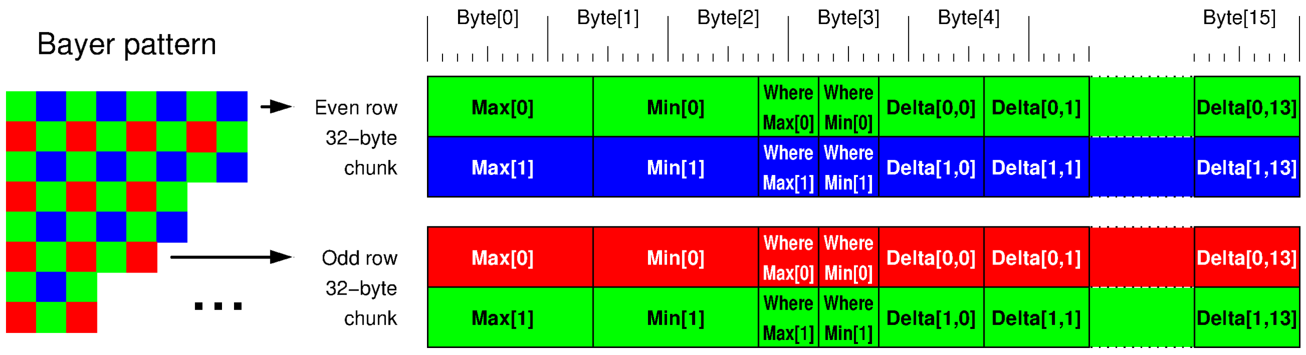


Figure 5. *Encoding of 11-bit pixel values in lossy-compressed ARW2 pixel data*

which was created from an uncompressed raw capture similarly boosted. Figure 3 shows how KARWY computationally repaired the artifacts in the lossy-compressed capture.

The ARW2 lossy compression algorithm

A wide variety of schemes are used to encode raw data in digital cameras. Perhaps most common is direct use of the image frame buffer data, which might or might not be packed. For example, Sony's new uncompressed 14-bit raw format (also called ARW2) is not packed, but byte-aligned, so that each 14-bit sample is padded with two 0 bits to fit a 16-bit word. Canon and Nikon both use schemes that compress the dynamic range and then apply a secondary compression method. Leica uses a very simple scheme in which each linear-gamma sample is simply encoded as an 8-bit log value.

Although Sony has published very little about the details of ARW2 compression, it is relatively straightforward to determine how compression was done by examining the code used for decompression. Most of the following is deduced from the decoding logic used in dcraw[6]. Our discussion here is also consistent with the description of ARW2 compression given by the author of RawDigger[2] and by Jim Kasson[4].

The ARW2 compression algorithm has many desirable properties. Of course, the most important property is that it should not introduce visible artifacts and, despite the recent fuss, the truth is that artifacts are rarely visible even with the latest cameras and were virtually never seen with older cameras. Both compression and decompression require only a modest amount of computational processing. Although Sony sensors produce data with as many as 14 bits per pixel sample, the compression algorithm achieves an impressive, and roughly constant, compression ratio of about 60% the size of the 14-bit raw coding – basically 8 bits per pixel. Better still, the compressed form is randomly addressable: values for individual pixels can be extracted, or even changed, without impacting more than a 16-byte block of the image data.

The first stage of the Sony ARW2 compression algorithm is tone mapping that approximates a logarithmic encoding. This is done by lossy piecewise-linear mapping of the raw pixel values to values with lower precision. The actual number of original bits per pixel sample depends on the camera model and mode used for capture, but the linear values are essentially scaled to a 14-bit range and a fixed black level offset (e.g., 512) is added in to form the input value for mapping. The precision of these linear values is then reduced to 11-bit using a five-linear-segment curve defined by the thresholds at which the spacing between distinguished original values doubles. Metadata in the ARW2 file records the number of linear values before this "step size" changes from 1 to 2, 2 to 4, 4 to 8, 8 to 16, and 16 to 32. This process is conceptually very similar to the tone mappings used in other camera makes, and is usually considered to be relatively harmless.

The more clever, and more controversial, encoding is the second stage. In this stage, the 11-bit per pixel tone-mapped image data is lossy compressed, using a form of delta encoding, into 32-byte blocks each describing a contiguous sequence of 32 pixels in a row. The Bayer color filter pattern implies that a 32-pixel sequence actually will contain spatially interleaved pixels of two colors, either 16 green and 16 red or 16 blue and 16 green (as shown in Figure 5). The compression algorithm is not really sensitive to which color channel is being represented by each pixel, but essentially treats the two groups of 16 same-colored pixels within each 32-pixel block as independent compression problems.

As illustrated in Figure 5, each group of 16 same-color pixels is encoded as a vector of 18 values. The maximum (Max) and minimum (Min) 11-bit values are directly stored, followed by the pixel positions (as 4-bit values, 0-15) they occupy within the 16 same-colored pixels of this block. The values of the remaining 14 pixels are encoded, in left-to-right order, as scaled 7-bit deltas between the pixel value and the recorded minimum. If the difference between the maximum and minimum values is less than 128, the scaling of the delta is 1 and this encoding step is essentially lossless. However, if the difference between the maximum and minimum is greater, then the delta is scaled by a sufficiently large power of two for the delta to fit in 7 bits. For example, if the maximum is 1518 and the minimum is 1000, the difference is 518, and the scaling factor will round 518/127 (which is approximately 4.08) up to the next power of 2 – which is 8. This scaling of already tone-mapped values can cause significant posterization, and is thus the most controversial aspect of ARW2 compression.

Forcing scaling factors to be powers of two in both stages of the compression algorithm is not logically necessary, and was

©2016 Society for Imaging Science and Technology
DOI: 10.2352/ISSN.2470-1173.2016.2.VIPC-227

probably done to simplify the computation. After all, scaling by a power of two is implementable by simply dropping bit positions, rather than requiring computationally expensive integer division.

It is easy to understand why Sony engineers would consider this power-of-two simplification to be relatively harmless to image quality. When this compression scheme was first introduced, usable sensor dynamic range was only around 11 bits/pixel. There also is a good argument that even if the sensor has a large dynamic range, it takes a special type of scene and a very good lens to render anywhere near the full contrast range within a small group of adjacent pixels from the same color channel – thus, one would expect visible artifacting to be rare. No doubt this was a carefully considered engineering decision; there were several camera models that provided both uncompressed and this lossy compressed raw format before the uncompressed option was dropped. It is only with the pairing of new cameras having larger dynamic ranges with lenses offering very high microcontrast that the compression artifacts became a major topic of public discussion.

Design Of KARWY

KARWY (pronounced car-we) is the program we created to credibly repair visible artifacts in Sony ARW2 lossy-compressed raw files. Logically, it is a wrapper, written at the University of Kentucky (KY), that reprocesses a Sony alpha raw (ARW) file to produce an improved raw file.

The approach used in KARWY is the result of trying over 75 different repair algorithms. The artifacts in the lossy-compressed ARW2 files are subtle and closely tied to the compression process. This makes them exceptionally difficult to recognize and repair by simply examining the recovered image data. However, directly operating on the compressed form makes the artifacts much easier to identify. The primary disadvantage is that this approach sacrifices the ability to repair a JPEG rendered from an ARW2, or even a DNG-format raw created from an ARW2.

Constructing An Error Model

The key to credible repair of compression artifacts is being able to reliably identify the artifacts. With very subtle artifacts, identification can be very unreliable, and repairs may change pixel values that were not incorrect while failing to correct faulty values. Many of our earliest attempts at repair demonstrated these flaws.

Instead, the latest repair algorithm begins by using the raw decoding process not only to decode the compression, but to directly construct upper and lower bounds for the possible value of each pixel. In effect, the decoded value is not treated as the value, but rather as an initial estimate of the actual value.

The first step is determination of the possible value range for a pixel's 11-bit tone-mapped value before block compression. This is a somewhat subtle problem, for which there are three cases:

- A pixel that is used as the reference maximum or minimum in its block is stored as a precise 11-bit value.
- A pixel in a block for which that color channel's maximum is less than 128 more than the minimum is able to be reconstructed as a precise 11-bit value.

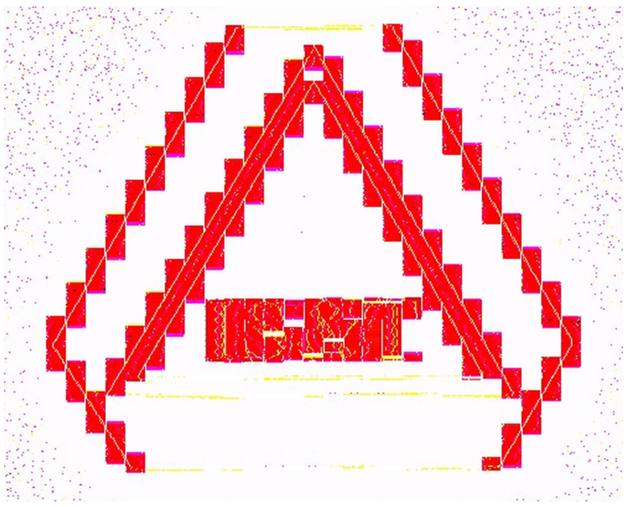


Figure 6. *Bits valid in compressed capture of logo*

- All other pixels have an error bound which is determined by the delta scaling factor for that block. This brings some odd properties; for example, two same-value pixels in the same color channel of the same block generally cannot have the same value in the encoding if one of them is the maximum. Reusing our example with a maximum value of 1518 and a minimum of 1000, if there is more than one pixel with the value 1518, all but the one coded as the reference maximum will actually be assigned the value 1512.

Determining the possible value range for the inverse tonal mapping is straightforward and the same transformation is applied to all pixels. The error bounds are determined by the step size in that linear region of the tone-mapping curve. Of course, a pixel whose 11-bit tone-mapped value is uncertain from the block encoding actually gets the original value range that is the convex hull of the ranges for the top and bottom of the 11-bit range.

However, there is yet another complication: even before compression, pixel values are subject to noise. If the approximate original value is known, it is possible to improve (appropriately expand) the error bounds by incorporating an error model for the camera electronics, sensor, and photon shot noise. These noise bounds can be difficult to model precisely across various cameras and shooting conditions, but the primary impact of an overly generous estimate of the noise is that it allows a wider range to be considered in searching for the correct value. In contrast, not accounting for noise-induced error can lead to similarities between pixels not being recognized, causing less information to be available for reconstruction of highly ambiguous pixel values.

In the current version of KARWY, the decode logic from dcraw is used to compute an error model for each pixel. The error model is actually larger than the image itself, and is directly output to a file.

Figure 6 illustrates the properties of a KARWY-constructed error model. This figure is derived from the exact same image shown in Figures 1, 2, and 3. Each pixel is assigned a value based on the number of bits known to be accurate in its value: the brighter a pixel, the more precisely that pixel's value is known.

©2016 Society for Imaging Science and Technology
DOI: 10.2352/ISSN.2470-1173.2016.2.VIPC-227

The block-with-a-line-through-it pattern is a direct result of the delta compression, with the white line corresponding to the pixels that were identified as the minimum/maximum in their block: their 11-bit values were directly encoded. Note that portions of the scene that do not contain vertical edges are not very ambiguous. However, there is also a general speckle pattern, and this is the result of modeling the loss of accuracy associated with the reduction to 11-bit values combined with the simple fact that not all pixels are sufficiently well lit to provide much coverage of the dynamic range: a pixel that was exposed by only a few photons really cannot have a precisely-known and accurate value.

DNG Conversion

There is a fundamental problem with production of a raw file correcting the artifacts introduced by ARW2 compression: the compressed ARW2 file format does not provide a way to encode the repaired image data without re-introducing the same artifacts. Since Sony has now provided the alternative of an uncompressed ARW2 format, it might be possible to rework KARWY to output the repaired raw data in that format... but that format did not exist at the time KARWY was created. For that reason, KARWY's output is a DNG raw file.

Although DNG is a standard, the truth is that the combinations of metadata field values expected in DNGs by Adobe tools is essentially undocumented, and hence very difficult to synthesize. Thus, the second step in KARWY is to use the free Adobe DNG Converter[8] to convert the lossy-compressed ARW2 file into an uncompressed DNG with appropriate metadata.

The plan was to use DNG Converter only for the metadata, with KARWY overwriting the image data with its own version. However, that did not work. Apparently, DNG Converter does not produce a DNG with the same pixel values obtained by applying the ARW2 compression algorithms used in other tools (such as dcraw[6]). In fact, in many cases, DNG Converter does not even generate the same pixel resolution, having either more or fewer pixels than the ARW2 file encoded!

The result is that we cannot simply replace the DNG image data with a repaired version derived by other means. Instead, KARWY reads the generated DNG and error model files, adjusts the error model to match the DNG data, and then modifies the DNG image data in place. Indeed, this is not an entirely satisfactory solution either, as encoding the repaired raw data using the DNG wrapper appears to impose a slight color shift, as is perhaps most obvious in Figure 3. In practice, this color shift is easily corrected in post-processing, but to keep the comparisons in this paper faithful to the actual data, such correction has not be applied here.

Smoothing

After adjusting the error model (range data) computed from the ARW2 file to be applicable to the data in the DNG file created by Adobe DNG Converter, the DNG data is essentially taken to be the initial estimates of the pixel values that we wish to improve. These estimates are important in that they strongly bias how textures are synthesized, but it is really the error model that constrains the final raw pixel values assigned. Thus, it is relatively harmless for even very aggressive smoothing to be applied to these initial estimates. To reduce the impact of specific types of visual artifacts, KARWY performs two types of stochastic smoothing operations on the the pixel estimates.

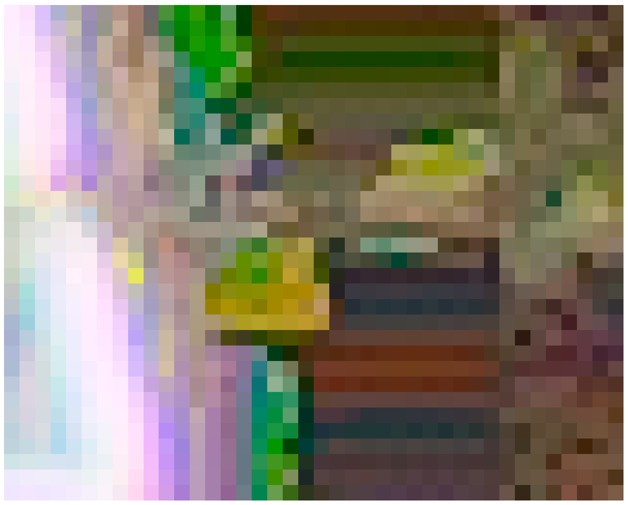


Figure 7. "Blondie artifacts" – colored parallel lines

Despite the care taken in computing the error model, it seems visually apparent that a tiny fraction of pixels should have values that fall outside of their computed error bounds. This seems to happen not in blocks that have large error bounds, but immediately adjacent to such blocks. At this writing, it remains unclear why these apparently wrong pixel values occur. The issue could be due to noise reduction or analog signal interference or hysteresis before encoding, rounding during encoding, or even simple one-off programming errors causing one block's encoding to affect that of a neighboring block. It is impossible to be certain of the cause without access to Sony source code. What is clear is that, although these presumably incorrect pixel values are very rare, when they occur, the artifact tends to be quite visible.

To reduce the impact of these pixels, KARWY provides three smoothing parameters. The three different parameters are all values that range from 0% to 100%, and differ only in to which pixels they are applied: pixels that are in blocks that suffered loss of accuracy by delta compression, blocks immediately adjacent to high-loss blocks, and blocks that have neither issue. In the KARWY interface, these are refered to as bad, near-bad, and other regions. The smoothing logic looks for large (as compared to noise) variations between the nearest same-color-channel pixels and reduces the variation by a random amount bounded by a noise model and scaled by the percentage specified for that type of region. That arithmetic certainly sounds like smoothing that will blur detail, but the pixel value it sets is really nothing more than the initial estimate of the pixel value that is used to seed the texture matching. Thus, the effect is more of a biasing that favors textural matches that lean toward smoother transitions.

A second type of smoothing-like operation is always applied in KARWY. This algorithm is intended to smooth what we have been calling "Blondie" artifacts: parallel line patterns that occur in the horizontal (long-side-of-the-frame) orientation in bad, and sometimes near-bad, regions. Figure 7 shows a clear exam-

©2016 Society for Imaging Science and Technology
DOI: 10.2352/ISSN.2470-1173.2016.2.VIPC-227

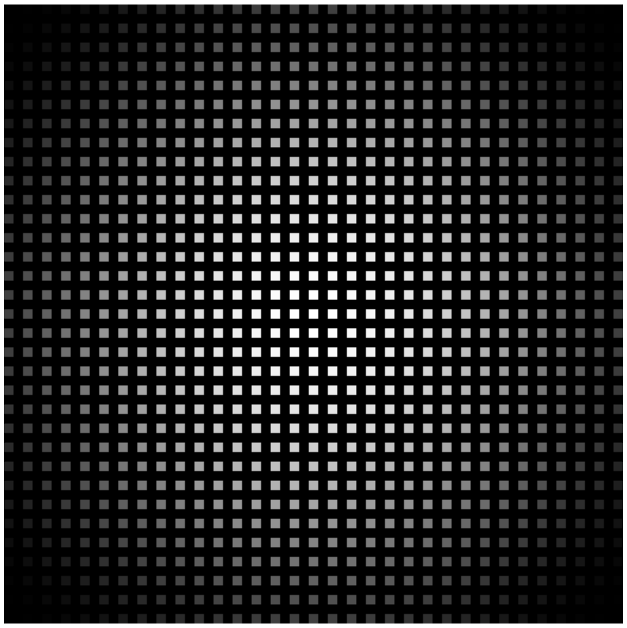


Figure 8. *Spiral weighting pattern for texture synthesis*

ple of this type of patterned artifact (in what is actually a much tighter 40x32 pixel crop from the right side of the Finnegan's sign in Figure 10). Basically, the line pattern comes from the consistent biasing of the brightness, and even moreso of the color, due to the posterized values within a block being offset by noise in the maximum value. The code in KARWY recognizes these "Blondie" artifacted areas using 3x3 arrays of the nearest same-color-channel pixels and again adds randomized noise to the pixel estimates in order to bias the texture matching process in favor of creating smoother textual matches.

Texture Synthesis

Texture synthesis is the process of creating a texture that "looks like it belongs" and inserting it in an image. Normally, the unit of synthesis is one or more pixels, as was the case in the texture synthesis we implemented in DeOrbIt[7]. In contrast, here we only seek to synthesize refinements of uncertain pixel values. Thus, the goal is to synthesize the most credible texture while keeping all pixel values within their computed error bounds.

The texture synthesis algorithm used in KARWY is fairly straightforward, based on weighted averaging of values of nearby pixels with similar surroundings. For each pixel with an ambiguous value, pixels from the same Bayer-filter color channel are examined in a spiralling-out sequence centered at the pixel in question, with positions outside the image boundaries ignored. If the value of spiral pixel being examined is not as ambiguous, then its value is added to a weighted sum. The weighting applied to each value is computed as a function of:

- The number of bits valid in the pixel value.
- The similarity of nine neighboring pixels to the corresponding neighbors of the pixel to be improved, where similarity is measured by overlap of computed value error bounds. It is primarily this similarity metric that causes credible textures to be synthesized. Note that for the red and blue channel pixels, the only one of the nine pixels coming from the same color channel as the pixel being improved is the one in the center.
- The base distance weighting for that position in the spiral. The base distance weightings in the spiral decrease with distance in the pattern shown in Figure 8.

The complete spiral samples 1089 same-color-channel pixels to compute the weighted pixel value estimate. However, the total quality of the pixel value estimate is incrementally updated in the spiral pattern, and the search is terminated early if quality is sufficient. To avoid echoing posterization from other pixels, the weighted average value is then adjusted by random noise bounded to approximately half the uncertainty in the value. Finally, the pixel value being improved is adjusted towards the weighted average value.

Of course, it is possible that a particular pixel has sufficiently unusual neighbors that none of the 1089 spiral samples yields a match and no weighted average is produced. This is prevented by initializing the weighted sum for each pixel to favor the current pixel value.

In some versions of KARWY, the spiral texture synthesis process above was allowed to repeat up to three times. The idea was that repeating the process could iteratively tighten the error bounds. However, the improvement did not seem large enough to justify the extra execution time, so the iteration was disabled in the version of KARWY posted online and used for all the examples in this paper.

Final pixel data adjustment

Although the above processing does appropriately incorporate noise, many pixel values are essentially copied by the above process, leaving a small amount of posterization from rounding bias in the normal encode/decode handling. Thus, as a final adjustment, random noise below the detail floor is added to each pixel value and the value of each pixel is clipped to fall within its computed error bound range.

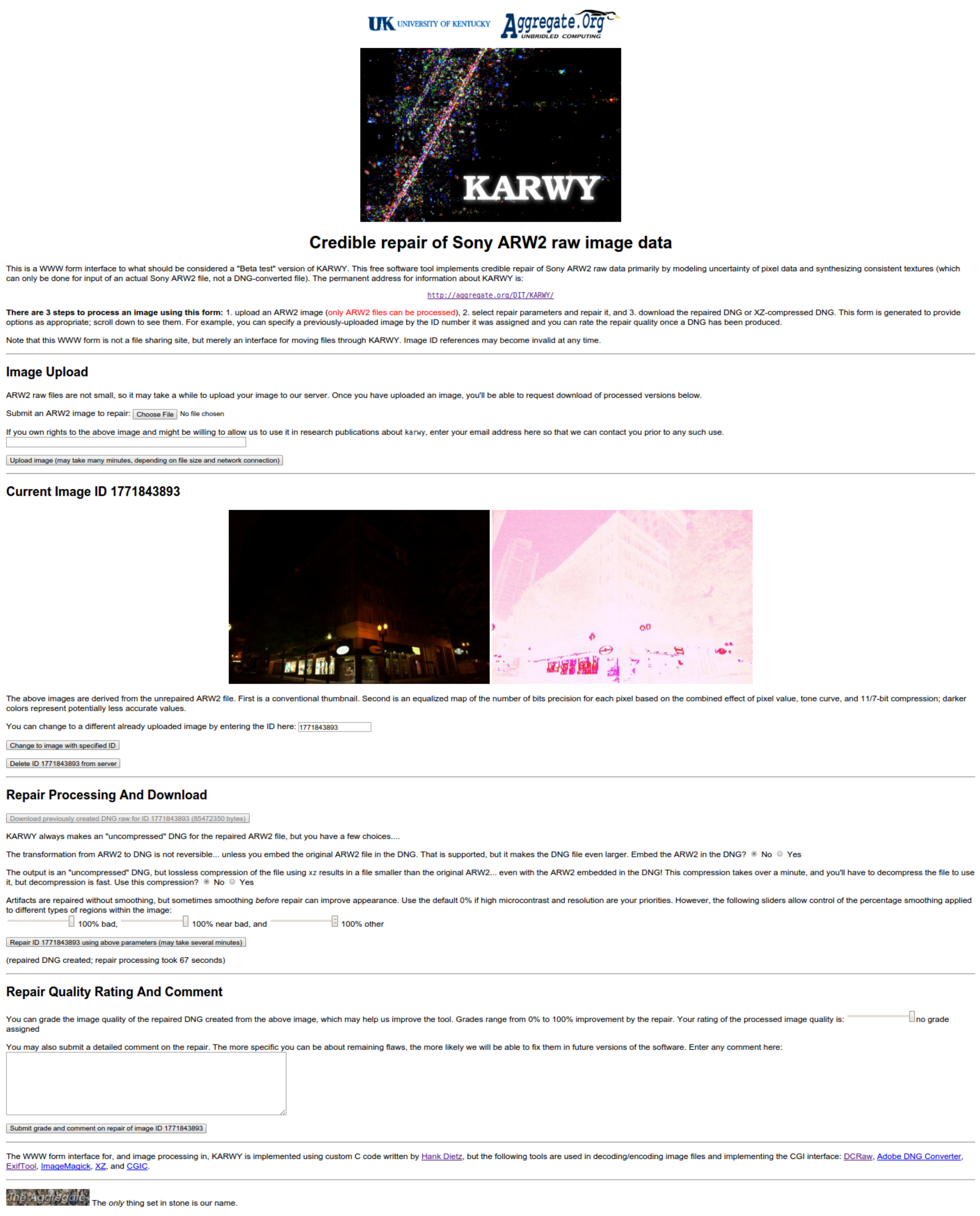
KARWY user interface

In order to facilitate others using KARWY, the tool was implemented as a CGI service accessed via a WWW form that could be used by any browser. This form interface is shown in Figure 9. Because moving large files can take a while and some browsers might thus time-out, processing an image is done in three separate steps:

1. Upload your lossy-compressed ARW2 file (or select one uploaded previously)
2. Set repair parameters and initiate repair processing
3. Download the resulting DNG

The user interface also provides for reprocessing an uploaded file without retransmitting the file. The smoothing levels can be adjusted, as well as enabling the original ARW2 file to be enbedded in the output DNG or enabling xz compression of the DNG. It is also possible to submit comments on repair quality along with a numerical rating from 0 to 100.

©2016 Society for Imaging Science and Technology
DOI: 10.2352/ISSN.2470-1173.2016.2.VIPC-227



Credible repair of Sony ARW2 raw image data

This is a WWW form interface to what should be considered a "Beta test" version of KARWY. This free software tool implements credible repair of Sony ARW2 raw data primarily by modeling uncertainty of pixel data and synthesizing consistent textures (which can only be done for input of an actual Sony ARW2 file, not a DNG-converted file). The permanent address for information about KARWY is:

http://aggregate.org/DIT/KARWY/

There are 3 steps to process an image using this form: 1. upload an ARW2 image (only ARW2 files can be processed), 2. select repair parameters and repair it, and 3. download the repaired DNG or XZ-compressed DNG. This form is generated to provide options as appropriate; scroll down to see them. For example, you can specify a previously-uploaded image by the ID number it was assigned and you can rate the repair quality once a DNG has been produced.

Note that this WWW form is not a file sharing site, but merely an interface for moving files through KARWY. Image ID references may become invalid at any time.

Image Upload

ARW2 raw files are not small, so it may take a while to upload your image to our server. Once you have uploaded an image, you'll be able to request download of processed versions below.

Submit an ARW2 image to repair: Choose File No file chosen

If you own rights to the above image and might be willing to allow us to use it in research publications about karwy, enter your email address here so that we can contact you prior to any such use.

Upload image (may take many minutes, depending on file size and network connection)

Current Image ID 1771843893

The above images are derived from the unrepaired ARW2 file. First is a conventional thumbnail. Second is an equalized map of the number of bits precision for each pixel based on the combined effect of pixel value, tone curve, and 11/7-bit compression; darker colors represent potentially less accurate values.

You can change to a different already uploaded image by entering the ID here: 1771843893

Change to image with specified ID

Delete ID 1771843893 from server

Repair Processing And Download

Download previously created DNG raw for ID 1771843893 (85472350 bytes)

KARWY always makes an "uncompressed" DNG for the repaired ARW2 file, but you have a few choices....

The transformation from ARW2 to DNG is not reversible... unless you embed the original ARW2 file in the DNG. That is supported, but it makes the DNG file even larger. Embed the ARW2 in the DNG? [✓ No] Yes

The output is an "uncompressed" DNG, but lossless compression of the file using xz results in a file smaller than the original ARW2... even with the ARW2 embedded in the DNG! This compression takes over a minute, and you'll have to decompress the file to use it, but decompression is fast. Use this compression? [✓ No] Yes

Artifacts are repaired without smoothing, but sometimes smoothing *before* repair can improve appearance. Use the default 0% if high microcontrast and resolution are your priorities. However, the following sliders allow control of the percentage smoothing applied to different types of regions within the image:
100% bad, 100% near bad, and 100% other

Repair ID 1771843893 using above parameters (may take several minutes)

(repaired DNG created; repair processing took 67 seconds)

Repair Quality Rating And Comment

You can grade the image quality of the repaired DNG created from the above image, which may help us improve the tool. Grades range from 0% to 100% improvement by the repair. Your rating of the processed image quality is: no grade assigned

You may also submit a detailed comment on the repair. The more specific you can be about remaining flaws, the more likely we will be able to fix them in future versions of the software. Enter any comment here:

Submit grade and comment on repair of image ID 1771843893

The WWW form interface for, and image processing in, KARWY is implemented using custom C code written by Hank Dietz, but the following tools are used in decoding/encoding image files and implementing the CGI interface: DCRaw, Adobe DNG Converter, ExifTool, ImageMagick, XZ, and CGIC.

The Aggregate The *only* thing set in stone is our name.

Figure 9. *KARWY user interface*

©2016 Society for Imaging Science and Technology
DOI: 10.2352/ISSN.2470-1173.2016.2.VIPC-227

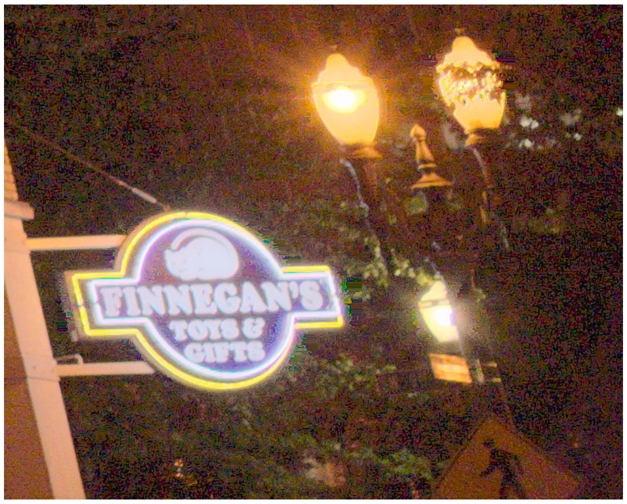


Figure 10. *ISO invariance lossy-compressed ARW2 test image*

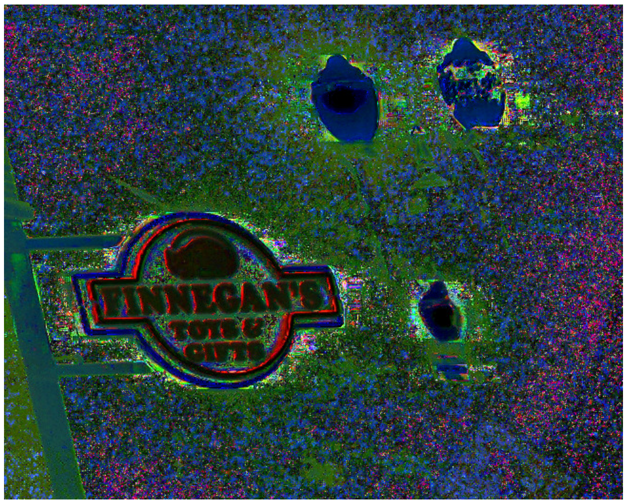


Figure 12. *Enhanced difference between Figures 10 and 11*

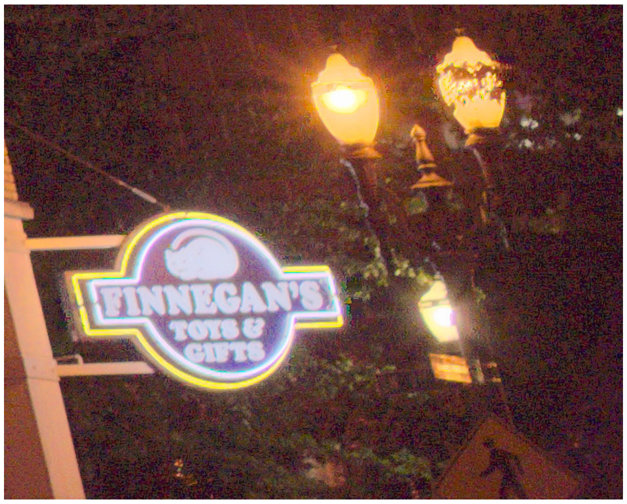


Figure 11. *ISO invariance test image processed by KARWY*

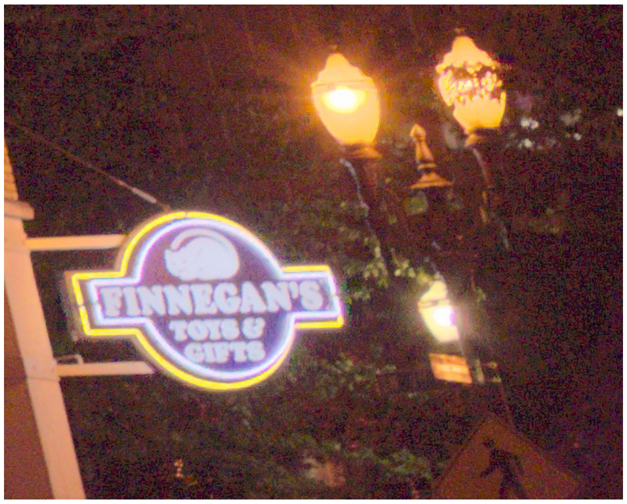


Figure 13. *ISO invariance test image with KARWY smoothing*

Results

The current version of KARWY is quite effective in credibly removing the disturbing artifacts associated with lossy-compressed ARW2 files. An excellent example from a 24MP full-frame ILCE-7M2 (also known as the a7II) appears in Figure 3. However, that example used an artificially-created scene built specifically for the purpose of showing artifacts. Thus, it is useful to consider more natural scenes.

Real-world examples

Writing for DPReview, Rishi Sanyal tested the "ISO invariance" of the current top-of-the-line 42MP full-frame BSI sensor Sony ILCE-7RM2 (also known as the a7RII)[3]. The nightime street scene he used to demonstrate the concept of grossly underexposing at low ISO and then boosting exposure in post-processing provides several excellent examples of artifacting in a natural scene. In fact, this is the same scene shown in the KARWY user-interface seen in Figure 9. A 650x520 pixel crop from the lower right side of the 5-stop boosted ISO100 shot is given in Figures 10, 11, 12, 13.

All three of these images come from the same lossy-compressed ARW2 file. Figure 10 is made from original ARW2 file, which (as discussed earlier) shows very significant "Blondie" artifacts (parallel lines running in the long dimension of the image) to the right side of the Finnegan's sign as well as around the light in the upper right. There are also more subtle posterization artifacts within the background of the Finnegan's sign and around the other two lights. Figure 11 is from the repaired DNG created by KARWY, and very dramatically reduces most of the artifacting. The enhanced difference image, Figure 12, makes the large scope of the improvements clear. However, Figure 13 shows an even more significant improvement. The image in Figure 13 was created by enabling KARWY's so-called smoothing option at 100% strength in bad, near-bad, and other regions.

©2016 Society for Imaging Science and Technology
DOI: 10.2352/ISSN.2470-1173.2016.2.VIPC-227

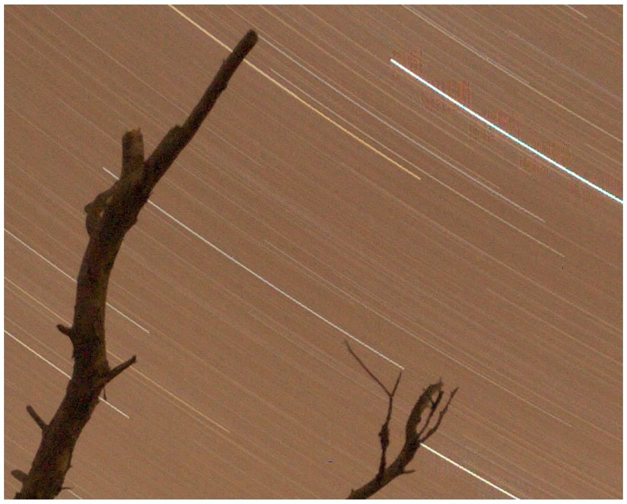


Figure 14. *Star trails lossy-compressed ARW2 test image*

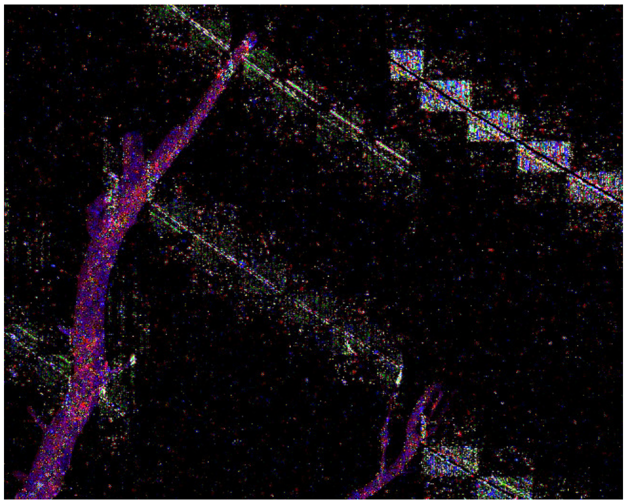


Figure 16. *Enhanced difference between Figures 14 and 15*

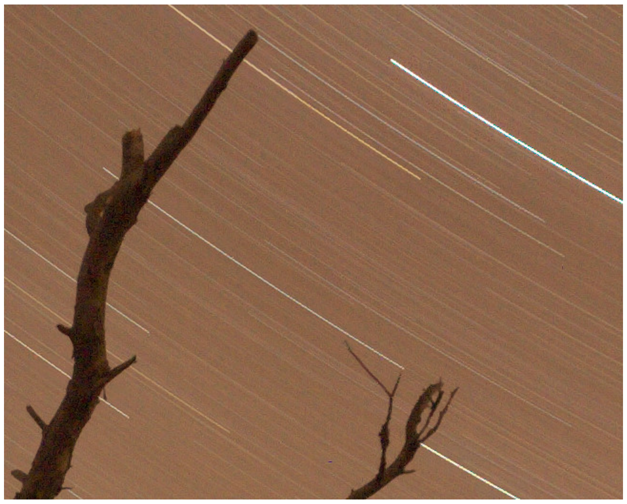


Figure 15. *Star trails image processed by KARWY*

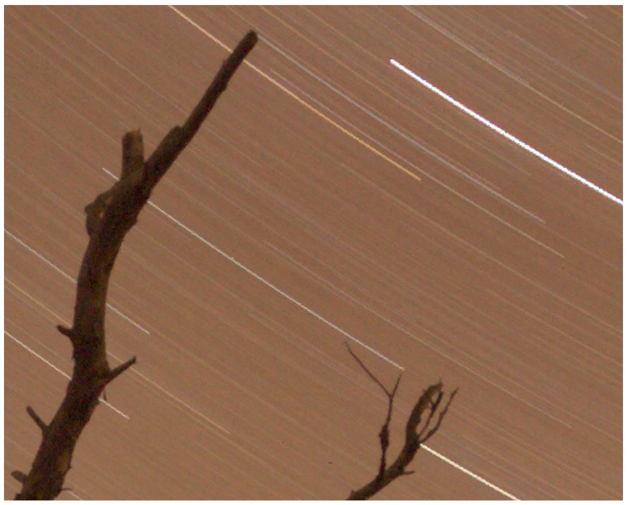


Figure 17. *Star trails image by KARWY with 100% smoothing*

Perhaps the best-known example of ARW2 artifacting is Matti Koski's "star trail" image, which has appeared in various places online – it was used for the examples in the RawDigger article about ARW2 artifacts and the original ARW2 file is freely distributed with that article[2]. This image was shot using an ILCE-7 (a7) in portrait orientation, so the 32-pixel blocks, and artifacts, run vertically instead of horizontally. A 650x520 pixel crop from near the center is presented in Figures 14, 15, 16, and 17. These respectively show the original image, KARWY reprocessed, the enhanced difference between the original and KARWY reprocessed images, and the KARWY reprocessed image with 100% smoothing.

User evaluation

Although KARWY has been freely available online at `http://aggregate.org/DIT/KARWY` since October 1, 2015, the server logs show only about 200 users. This is a shockingly low number given the level of concern expressed about lossy-compression ARW2 artifacts in various online forums, but it is clear that once Sony announced that they would provide a firmware upgrade offering an uncompressed option, concern quickly dissipated. Although the KARWY user interface makes it very convenient, only 15 users bothered to submit ratings for how well the repair worked. From lowest to highest, these percentage rating scores are summarized in Figure 18.

One of the low ratings complained of diagonal lines. Those were caused by Adobe DNG Converter changing the image dimensions, literally adding/dropping pixels when encoding a raw image. Specifically, Adobe DNG Creator makes a DNG file for a Sony SLT-A99 ARW2 that is missing 16 masked edge pixels. It only took a day to modify KARWY so that mismatches in the pixel array dimensions are handled correctly. The other very low ratings often complained that KARWY did nothing... which it turns out is exactly what it should have done, because the images did not have artifacts from lossy compression. In fact, the inability of people to judge when their image is defective in a particular

DOI: 10.2352/ISSN.2470-1173.2016.2.VIPC-227

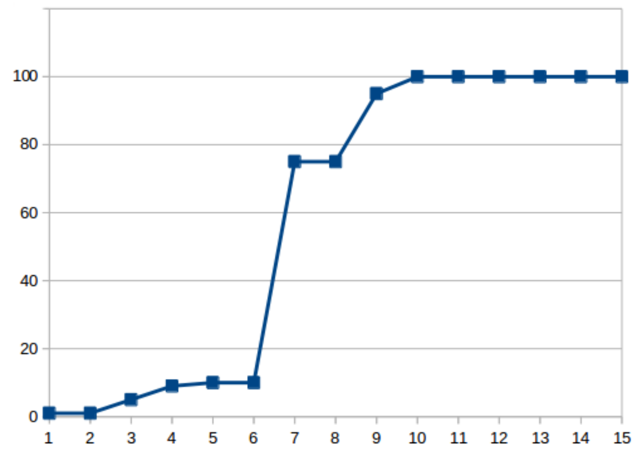


Figure 18. *User ratings of KARWY repairs*

way is a recurring problem; this phenomenon also happened with users of DeOrbIt[7], where obvious "white orb" blooming artifacts went unnoticed by some as others mistakenly thought every near-saturation or round-shaped region was a "white orb" bloom.

The in-between ratings noted that there were still traces of some artifacts visible – which certainly can happen, especially in images that lack appropriate textures to sample from near the artifacted area. The 100% ratings speak for themselves with comments like: "I think the repair tool has done a perfect job" and "The previous compression artifact is actually gone. The final image is a little bit less sharp, but just a little bit." KARWY should not have a significant impact on resolution, but can reduce micro-contrast, thus slightly reducing sharpness.

Conclusion

This paper explains how Sony's lossy compression for ARW2 files works and characterizes the artifacts it can create. More importantly, a novel repair algorithm is presented that combines concepts of texture synthesis with modeling of pixel value error bounds. The repair algorithm, as implemented by a free software tool, is also briefly evaluated and shown to be quite effective.

Although interest in credible repair of ARW2 artifacts faded quickly after Sony released firmware that offers the option of using an uncompressed format, that firmware does not work with the majority of Sony cameras, nor can it help improve the quality of photos already taken. Perhaps most significantly, the approach of using texture synthesis to improve the accuracy of pixel values has been proven viable, and it offers the possibility that properties like dynamic range could be improved for other cameras – even ones that use uncompressed raw formats. Our preliminary experiments attempting to use this approach to improve dynamic range have been promising, but inconclusive.

As a final question, is a fixed-compression, random pixel block access, compression scheme usable for future cameras? Clearly, the Sony ARW2 scheme is of sufficient quality to satisfy most current users shooting under most circumstances. Beyond that, the repair technology in KARWY easily could be embedded in the raw file decoders in image editors, leaving very few cases in which the image quality is visibly limited by this encoding. However, the convenient alignments with each 32-pixel block using 256 bits (8 4-byte words) are broken as soon as even a single bit higher precision is used, and camera dynamic range is improving with 16-bit pixel samples already common in industrial and medium-format cameras. It would be possible to improve the image quality without expanding the format by allowing non-power-of-two scaling factors, but that would greatly increase the computational complexity in compression. Sony's new uncompressed ARW2 format currently uses 16 bits to store each 14-bit sample, so perhaps having that option will allow the existing lossy-compressed ARW2 scheme to last as an option for less-critical use for some time to come?

References

[1] Matthew C. Stamm and K. J. Liu, *Anti-forensics of digital image compression*, Information Forensics and Security, IEEE Transactions on 6.3 (2011): 1050-1065.

[2] lexa, RawDigger: detecting posterization in SONY cRAW/ARW2 files, *http://www.rawdigger.com/howtouse/sony-craw-arw2-posterization-detection* (2014).

[3] Rishi Sanyal, Sony Alpha 7R II: Real-world ISO invariance study, *http://www.dpreview.com/articles/7450523388/sony-alpha-7r-ii-real-world-iso-invariance-study* (2015).

[4] Jim Kasson, A tough test of Sony raw compression, *http://blog.kasson.com/?p=4838* (2014).

[5] Henry Gordon Dietz and Paul Selegue Eberhart, *ISO-less?*, Proc. SPIE 9404, Digital Photography XI, 94040L (February 27, 2015); doi:10.1117/12.2080168.

[6] Dave Coffin, Decoding raw digital photos in Linux, *http://www.cybercom.net/d̃coffin/dcraw/* (2015).

[7] Henry Gordon Dietz, *FUJIFILM X10 white orbs and DeOrbIt*, Proc. SPIE 8660, Digital Photography IX, 866005 (February 4, 2013); doi:10.1117/12.2004411.

[8] Adobe DNG Converter 9.1.1, *https://www.adobe.com/support/downloads/detail.jsp?ftpID=5919* (2015).

Author Biography

Henry (Hank) Dietz earned his PhD at Polytechnic University and joined the faculty at Purdue University's School of Electrical and Computer Engineering in 1986. Since 1999, he has been the Hardymon Chair in Networking at the University of Kentucky. Although best known for his work in compilers and parallel supercomputing using Linux PCs, he also works to improve cameras as computing systems.