AUDIOVISUAL QUALITY CONTROL AND PRESERVATION CASE STUDIES FROM LIBRARIES, ARCHIVES, AND MUSEUMS

Julia Kim, Digital Projects Coordinator, National Library Service for the Blind and Print Disabled at the Library of Congress, USA
Eddy Colloton, Project Conservator of Time-Based Media, Hirshhorn Museum and Sculpture Garden, USA
Dan Finn, Time-Based Media Conservator, Smithsonian American Art, USA
Rebecca Fraimow, Digital Archives Manager, WGBH Media Library and Archives, USA
Shu-Wen Lin, Collections Care Initiative Fellow for Time-Based Media, Smithsonian American Art Museum, USA
Crystal Sanchez, Media Archivist, Smithsonian Institution, USA
Annie Schweikert, Digital Archivist, Stanford University Library, USA

Abstract
Digital audiovisual workflows are complex. They can hinge on a breadth and depth of knowledge that is difficult to find within a single team or institution. The areas of knowledge called on can range from obscure and obsolete audiovisual carriers, to all the components in a digitization workflow chain, as well as new and evolving community resources and digital competencies for discovering errors during the quality control process. While there are many standardized audiovisual workflows, as this paper illustrates, QC work can be difficult even with a high level of training and experience; and problems, when caught, are often resource-intensive to diagnose and address. This paper details six distinct audiovisual case studies in which different digital preservation obstacles that are difficult to qualify, fully understand, and document are discussed; as well as, when possible, their solutions. They are all unique, but also unexceptional: we expect there are comparable situations, perhaps not-yet discovered or addressed in many audiovisual archives. This paper will underscore difficulties, and guide readers through some of the processes -- both formal and informal -- used to further analyze audiovisual file problems. Ultimately, in addition to helping other staff with similar problems, this paper should emphasize to administrators the special resource needs of audiovisual files and the staff responsible for them.

Introduction
Online resources like QCTools and the A/V Artifact Atlas go a long way in creating a growing knowledge base about artifacts and technological issues that can impact digital audiovisual files.[1] But how can these tools be best utilized by non-specialized staff or those who only intermittently work with digital audiovisual workflows? How can we diagnose and recover from errors discovered in a collection of audiovisual files? What is an artifact and what is "business as usual," especially in cases in which the date of transfer has long past, the vendor is far away, and there's limited information available? With audiovisual files, these distinctions can be very nuanced. Despite the many shared resources available, practical decisions are often made in a gray area of shared emerging practices.

1 "QCTools, an Open Source application created by MediaArea to allow for quantitative analysis of video files in order to enable a more thorough evaluation of digital video." https://mediaarea.net/QCTools. A/V Artifact Atlas is a shared resource for documenting audiovisual errors. https://bavc.org/preserve-media/preservation-tools/av-artifact-atlas.

With digital files, the conversation surrounding quality control most often focuses on layers of fixity at the frame and file level. Fixity information isn't always available from the time of a file's creation, however, and doesn't provide a detailed understanding of the issues that may impact files at the bit level. What are the errors that we should look for and understand collectively? How have our colleagues worked through some of these errors; and how can we build shared quality control models, practices, and language as a field?

This paper will document and share unique audiovisual digital case studies that required investigating these questions in the context of libraries, archives, museums, and broadcast organizations. In each case study presented, archivists worked back from an initial symptom to diagnose issues ranging from errors with digital file transfers, to problems in a digitization chain, to challenges with encoding and decoding proprietary or complex digital file codecs. All of these scenarios proved uniquely challenging to knowledgeable, formally trained moving image archivists and conservators. While not all errors could be fully explained, through investigation and knowledge sharing, all resulted in changes to archival workflows or practice to better avoid the issue in future. Examples submitted by the authors of the present article were initially presented by Julia Kim during her "Questionable File Show and Tell" session at the No Time to Wait 4 conference in 2019.[2]

1. The Checksums Match but the Files Are Bad

Crystal Sanchez, Julia Kim

In 2015, the Smithsonian's National Museum of African American History and Culture received copies of a collection of approximately 100 video interview files from the Library of Congress (LC) to accession into the Museum's collection. As a shared collection, preservation master files were to be stored at both institutions. Upon receipt at the Smithsonian Institution (SI), files were copied to a central staging location and then validated successfully. The checksums matched; thus, the fixity was confirmed. However, we later discovered that fatal errors caused some files to be unopenable or unplayable, and others to exhibit ephemeral digital artifacts during playback (Figure 1), prompting an analysis to identify potential points of failure along the workflow.

2 An archived version of the conference session is available at: https://www.youtube.com/watch?v=A-mm5Mijszk.

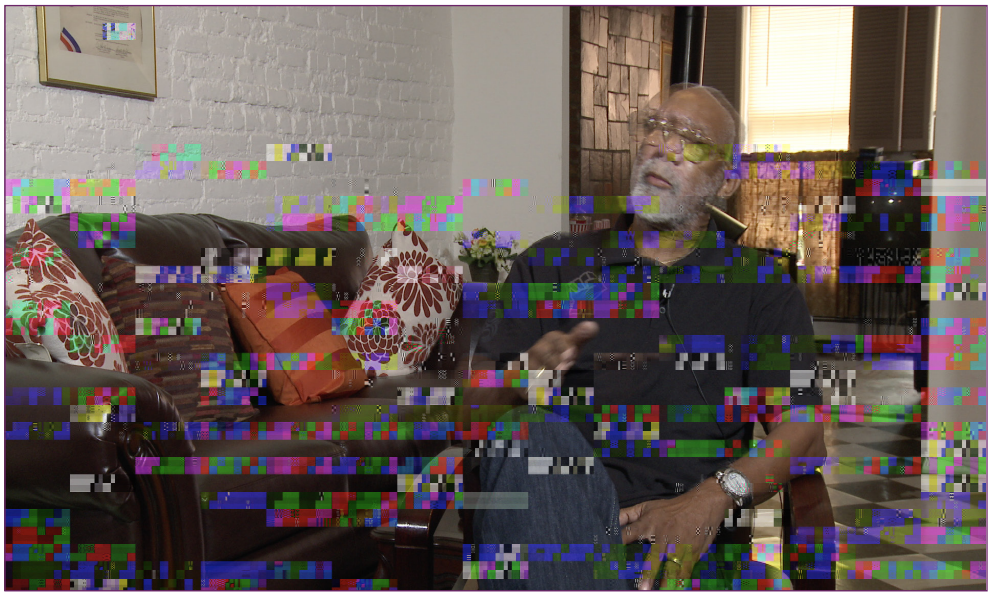


Figure 1. Video artifact in playback.

During the course of the approximately decade long project, digital files were created around the country and then transferred in batches to LC 's American Folklife Center, where they were migrated off of SD cards and drives to a local RAID. They were then transferred to archives staff and minimally processed and bagged upon ingest to long-term tape preservation storage. Each interview was also packaged inside the Bagit specification and copied to external hard drives for transfer to SI.[3] Once at SI, the "bags" were validated before moving forward with storage in SI's Digital Asset Management System (SI DAMS)[4]. Because of the large size of the files, the scale of the collection, and with the limited resources available, not all of the files were visually QC'd before ingest.[5]

Once the files were ingested to the SI DAMS repository, file fixity information was verified again, confirming that the files ingested into the repository were a bit-level copy of the files delivered from LC.

As part of the SI DAMS ingest process, files are automatically sent to transcoders to create proxy images and keyframes for easy reference. The transcoder step not only allows for reference to the files without accessing master files, but also provides another step of quality control for the files submitted to the repository. Transcoder log errors draw attention to any failed files that cannot be decoded, indicating a problem with playback of the original file.

3 Bagit is a standard packaging specification for easy receipt and validation of files when transferring them from one place to the other, https://tools.ietf.org/id/draft-kunze-bagit-14.txt.
4 The SI DAMS is managed by SI's Office of the Chief Information Officer, www.si.edu/dams.
5 See more on the born digital video specifications chosen for this collection on FADGI Digitization Guidelines Initiative's Creating and Archiving Born Digital Video: Part II. Eight Federal Case Histories. (2014), http://www.digitizationguidelines.gov/guidelines/FADGI_BDV_p2_20141202.pdf.

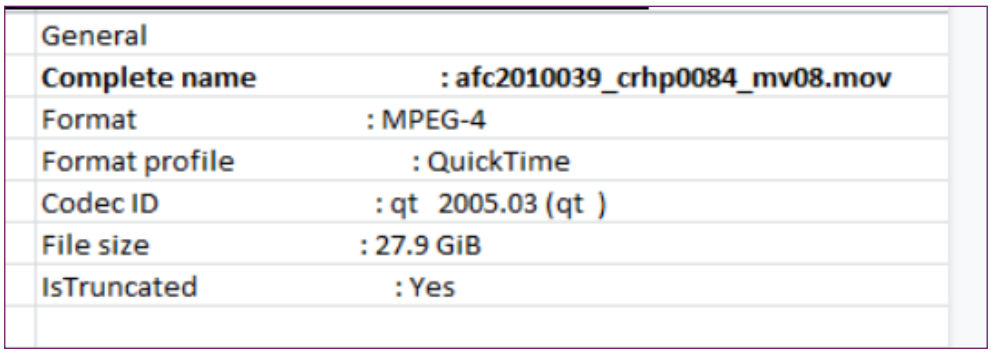


Figure 2. Sample corrupt file with minimal top-level metadata available.

Two major errors were widespread. Out of over 750 video files, 4 of the files failed with an unclear error in the system's Rhozet transcoder: "Process terminated for unknown reason." Further investigation with MediaInfo and Exiftool[6] showed no stream info available in the files, only top-level format data (Figure 2). Consequently, several common media players could not open the files, including VLC, QuickTime, Windows Media Player, and ffplay. FFmpeg provided a more specific error: "moov atom not found" (Figure 3).

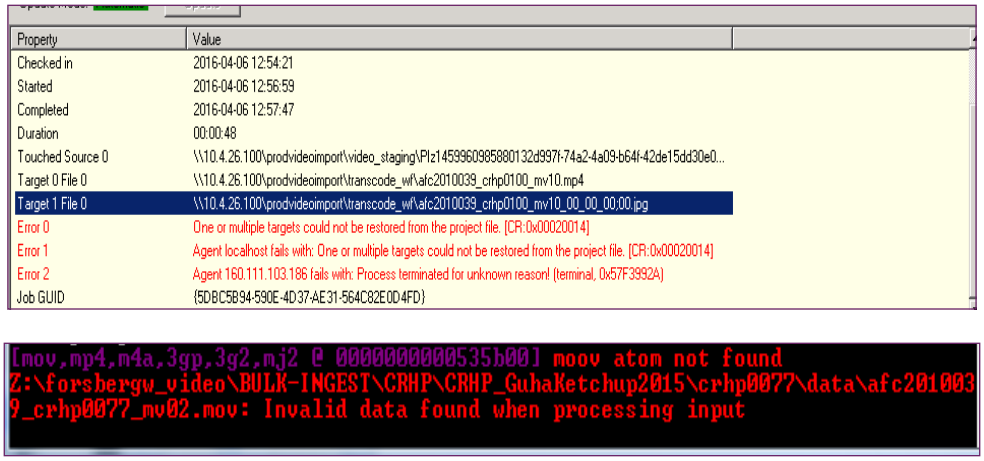


Figure 3. Rhozet error (above); FFmpeg error (below).

The moov atom in the QuickTime format contains information crucial to the decoding and playback of the file. Without this atom, the decoder cannot locate the stream data and has no instructions on how to play the file. Grasping for language to describe the problem, we defined the files as "incomplete" in our institutional correspondence.

6 MediaInfo by MediaArea, https://mediaarea.net/en/MediaInfo; Exiftool by Phil Harvey, https://exiftool.org/.

A second error presented itself only in the Rhozet transcoder, allowing us to target the exact time-stamped location of the error in the video. VLC was able to open the files and play them, but at the point of error, the video erupted in loud screeching sounds and presented colorful glitched blocks, signaling corrupted bits in the video stream. Decoding the file with FFmpeg output hundreds of the same error: "Error while decoding stream" (Figure 4).

```
frame=   57 fps= 15 q=-0.0 size=   33164kB time=00:00:01.76 bitrate=153626.2kbit
frame=   65 fps= 15 q=-0.0 size=   44858kB time=00:00:02.03 bitrate=180544.8kbit
[prores @ 000000000036bd360] invalid plane data size
    Last message repeated 299 times
[prores @ 000000000036bd360] error decoding picture
Error while decoding stream #0:2: Operation not permitted
[prores @ 000000000036bd360] invalid frame header
Error while decoding stream #0:2: Invalid data found when processing input
frame=   67 fps= 13 q=-0.0 size=   44858kB time=00:00:02.10 bitrate=174813.3kbit
[prores @ 000000000036bd360] invalid frame header
Error while decoding stream #0:2: Invalid data found when processing input
[prores @ 000000000036bd360] invalid frame header
Error while decoding stream #0:2: Invalid data found when processing input
[prores @ 000000000036bd360] invalid frame header
Error while decoding stream #0:2: Invalid data found when processing input
[prores @ 000000000036bd360] invalid frame header
Error while decoding stream #0:2: Invalid data found when processing input
[prores @ 000000000036bd360] invalid frame header
Error while decoding stream #0:2: Invalid data found when processing input
[prores @ 000000000036bd360] invalid plane data size
    Last message repeated 453 times
[prores @ 000000000036bd360] ac tex damaged 1024, 1024
[prores @ 000000000036bd360] invalid plane data size
    Last message repeated 2 times
[prores @ 000000000036bd360] error decoding picture
```

Figure 4. FFmpeg errors.

After discovering the errors, we began investigating where in the long path from producer's camera to LC to SI the errors may have occurred and to see if earlier copies of the files were available. Since the LC bags validated at SI, we checked the fixity values against the LC repository values, which matched. We then took a look at the files on the LC staging RAID. These files did not match our fixity values and could be opened and played fully. We were able to replace corrupt files with complete files from the LC RAID; all of the "missing moov" atom files and almost all of the glitch files were replaced.

We concluded that the files were accidentally corrupted in the transfer from the steps and transfers involved from migrating from LC RAID staging to hard drives for the archives staff for ingest into LC's repository when fixity information was created. Creating fixity information at the earliest point possible after the creation of a file is a best practice, because errors can occur in the workflow at any point and *it is possible to validate checksums on corrupted files*. Part of the administrative challenge with this then, is educating non-archival partners and content producers to create fixity values before the collections are ever accessioned and received in the archive.

Discovering the point of failure in the files was also a challenge, as the errors were only easily findable with some tools and not others. With the size of this collection, errors could have easily gone undetected for a while, even past the point of recovery. Based on our investigation of the workflow, the transfer protocols and fixity creation step were amended to mitigate future errors.

2. Static Image for Expected Duration

Rebecca Fraimow

In 2014, WGBH embarked on a project to retrieve all the audiovisual files from our digital asset management system (now deprecated) and send them to a vendor for transcoding and inclusion in the American Archive of Public Broadcasting. These files were stored on LTO-4 tapes administered through Sun Storage Archive Manager's (SAMFS-QFS)[7] server with an automatic tape robot; the files were retrieved in batches from the SAMFS-QFS server and downloaded onto a series of hard drives.

Initially, WGBH staff didn't QC the files after download. However, shortly after the beginning of the project, the vendor began to notice significant failure rates on the drives they were receiving from WGBH. These failures fell into several distinct types: files that could not be characterized as media files and, when analyzed with ffprobe,[8] reported the error message "moov atom not found"; files that could be characterized as media files but, when analyzed with ffprobe, reported the error message "Could not find codec parameters for stream 0"; and files that generated accurate metadata when analyzed by ffprobe, and could be played back with ffplay,[9] but showed visual signs of corruption. These last files would play correctly for a period of time, until the video stopped on a single frame—either a corrupted image from the content, or a green or black screen—that remained static throughout the remainder of the file's duration (Figure 5).

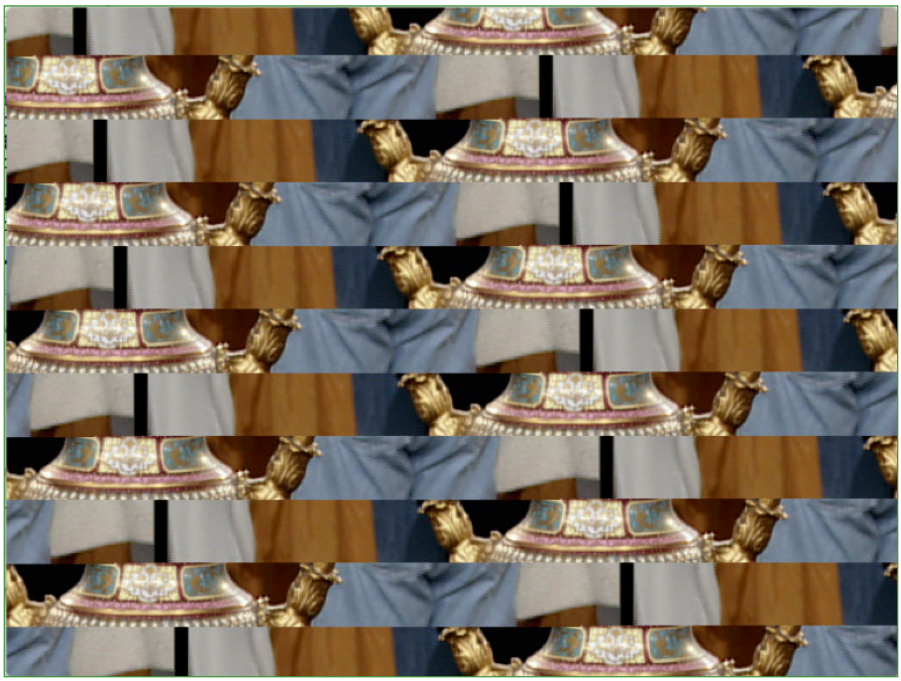


Figure 5. An example of one of the frozen images from a corrupted file.

7 The SAMFS-QFS is an Oracle product with documentation available here: https://docs.oracle.com/cd/E22586_01/html/E22570/glebg.html.
8 For more information about the file analysis tool ffprobe, visit https://ffmpeg.org/ffprobe.html.
9 For more information about the test playback tool ffplay, visit https://ffmpeg.org/ffplay.html.

As a result, WGBH began instituting extensive QC measures when downloading files from the server. Unfortunately, checksums had not been generated for most of these files before they were uploaded to the SAMFS-QFS system, making full file integrity checks impossible. Instead, WGBH started running automated checks on each downloaded file for size match and readability by file characterization tools. If a file did not match the size recorded for it and could not be read by FFmpeg, it was deleted from the drive. In order to detect more subtle failures, WGBH developed a script that automatically created image galleries of thumbnails taken at regular intervals throughout each video. WGBH staff could then identify issues by quickly reviewing the image galleries.

Many of the files that failed on the initial download were later downloaded successfully, which allowed us to compare the successful version with the corrupted one. Analysis with Apple's Atom Inspector, a tool for viewing and editing atom resources in QuickTime and MP4 files, confirmed that despite the different "symptoms," all the files were failing in the same way: a portion of the file was transferring correctly over the network, but at some point, the bit transmission became corrupted.

Each failure type represented differences in the structure of the original files. As discussed in the previous case study, most of the information that instructs a media player in how to process the contents of a QuickTime video file is contained in a data unit called the moov atom, generally located at the end of the file. The files that reported back "moov atom not found" were all QuickTime files structured in this way; when downloading, they became corrupted before the moov atom could be transferred, resulting in a file that could not be interpreted.

Files that reported correct media information but could not be played due to "incorrect codec parameters" were all MP4 files, which are structurally similar to QuickTime files but have the moov atom at the beginning of the file. However, the mdat atom—which stores the actual data content of the file—did not appear in the molecular tree of these files when they were analyzed; the data stream seems to have become corrupted almost immediately after the successful transfer of the moov atom. A media player presented with these files tries to decode the data in accordance with the track information provided in the moov atom, but finds no data to decode.

The files that demonstrated visual corruption were also structured with the moov atom at the beginning, and both mdat and moov atoms transferred successfully. Atom Inspector analysis shows that the original files and the corrupted files match each other exactly on the bit level until the point when corruption sets in, after which the data in the corrupted files appears to become randomized (Figure 6).

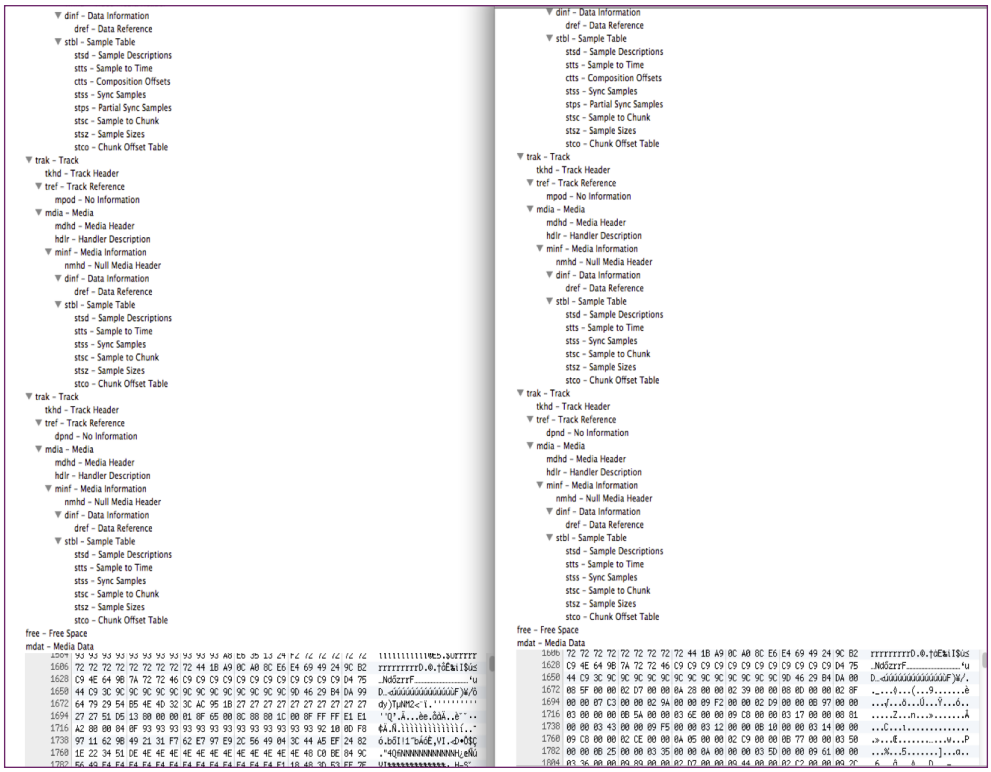


Figure 6. Left: hex data from a successful file. Right: hex data from a corrupted version of the same file. The hexes match until line 1650, after which the corrupted file deviates.

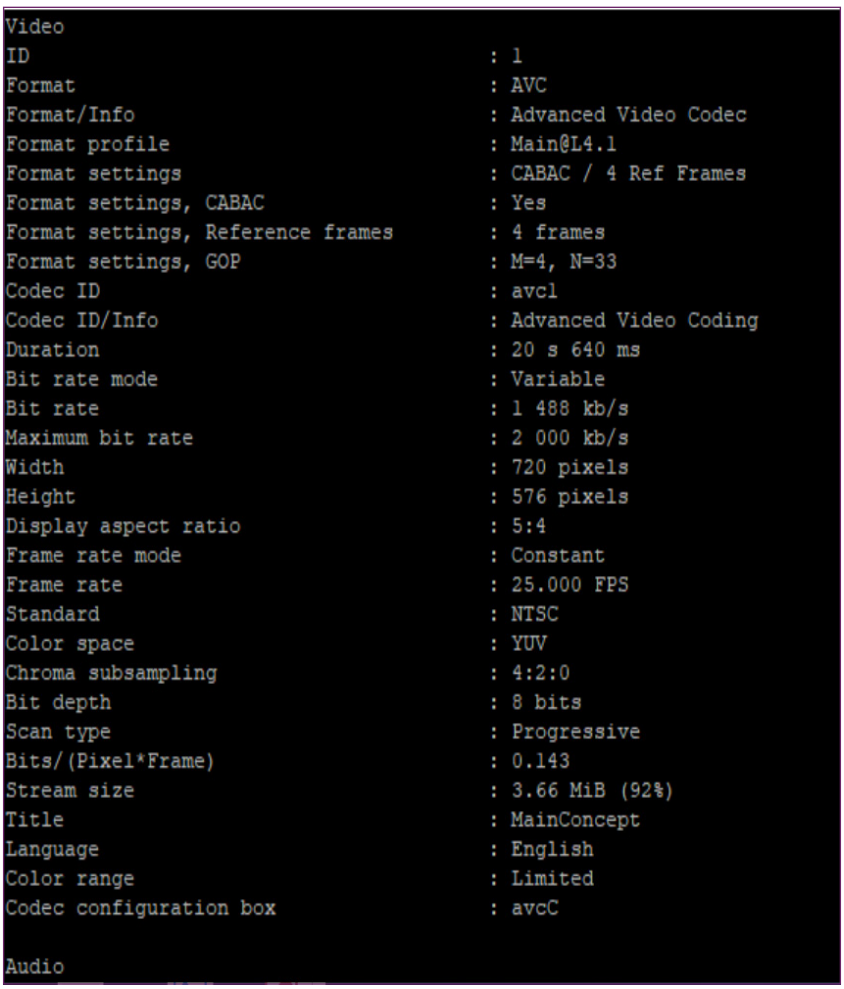
Because many of the corrupted files could eventually be downloaded successfully, we believe that the problems are an artifact of the transfer from the server to the hard drives. We're no longer using the SAMFS-QFS for backup storage, but we've also become more rigorous about fixity checking to ensure we'll catch these issues at the source going forward.

3. PAL and NTSC

Julia Kim

In 2019, an archival staff member at the Media Archive for Central England re-reviewing previously digitized collections noted an unknown artifact potentially created through the mediating digitization hardware and software used. Due to the artifact's uniqueness, she publicized it on Twitter and the A/V Artifact Atlas (AAA), an online resource for anyone to share, learn, and identify potential errors found in digitization QC workflows. The files containing the artifact had all been created through vendor digitization of 1-inch open-reel tape in 2012. While closer scrutiny of several other 1-inch transfers reveals the very same visual QC flaws, at this point in time, it's impossible to definitively trace and correct for the error without re-reviewing all aspects of the 2012 workflow.

During playback, an unusual "horizontal artifact" can easily be noted, which appears as an up and down rippling of the video image. The artifact can be likened to the motion of a wave or unfurling of cloth that shifts the video information up and down.[10] Still images of this playback issue do not reproduce the artifact. A closer inspection of the file's technical metadata (Figure 7) revealed potentially related and unexpected information, inviting further analysis.



```
Video
ID                                       : 1
Format                                   : AVC
Format/Info                              : Advanced Video Codec
Format profile                           : Main@L4.1
Format settings                          : CABAC / 4 Ref Frames
Format settings, CABAC                   : Yes
Format settings, Reference frames        : 4 frames
Format settings, GOP                     : M=4, N=33
Codec ID                                 : avc1
Codec ID/Info                            : Advanced Video Coding
Duration                                 : 20 s 640 ms
Bit rate mode                            : Variable
Bit rate                                 : 1 488 kb/s
Maximum bit rate                         : 2 000 kb/s
Width                                    : 720 pixels
Height                                   : 576 pixels
Display aspect ratio                     : 5:4
Frame rate mode                          : Constant
Frame rate                               : 25.000 FPS
Standard                                 : NTSC
Color space                              : YUV
Chroma subsampling                       : 4:2:0
Bit depth                                : 8 bits
Scan type                                : Progressive
Bits/(Pixel*Frame)                       : 0.143
Stream size                              : 3.66 MiB (92%)
Title                                    : MainConcept
Language                                 : English
Color range                              : Limited
Codec configuration box                  : avcC

Audio
```

Figure 7. Conflicting broadcast standard technical metadata

Most of the technical characteristics found with ffprobe and MediaInfo indicate that the file is in the PAL broadcast standard, matching its broadcast specifications and image size of 720 by 576 pixel resolution and 25 fps (frames per second). This correlates with the audiovisual collection's British geographic broadcast history. However, this is contradicted by tools like MediaInfo, which reports in the file's "Standard" metadata field that the NTSC broadcast standard is used instead (Figure 7). Broadcast audiovisual files

10 See video and error: https://archive.org/details/jowhite_one_inch_tape.

are not created to adhere to a mix of different broadcast standards and this issue may be a contributing factor to the "horizontal artifact" noted with visual QC. In my analysis, distinguishing whether or not this is a glitch with the file itself and/or an issue with verbosity and depth of the existing technical reporting tools available added another level of complexity. In other words, is this case of a PAL standardized file somehow legitimately exhibiting other characteristics that would mistakenly identify it as also NTSC that are less easy to detect with software tools?[11] Or, is it just a "glitchy" file?[12]

In investigating this further, I explored other related technically significant qualities and surprises between the way the file is meant to display information, as noted in the display aspect ratio (DAR), screen aspect ratio (SAR), and pixel aspect ratio (PAR); three mathematically related ratios that help determine how software plays back the file through manipulating, for example, pixel size and ratios (PAR) from non-square (not 1:1 ratio) to square (1:1). The DAR, the most commonly understood and referred to of the 3 interrelated characteristics, is how the file's aspect ratio is resolved for playback for viewers. It's often limited to values such as 4:3, which is referred to commonly as "standard definition," or 16:9 which is referred to as "high definition." The digitized files in question exhibit DAR values and play back with a 4:3 ratio, as expected. This value is related mathematically to the other values,[13] and the screen aspect ratio (SAR) is 1:1, which would mean that while the PAR should be 4:3 or non-square, it's noted as square. Through sharing this case study more widely at the 4th annual No Time to Wait conference with digital audiovisual archivists, developers, and other experts, while no one in the room had experience with the "PAL and NTSC" metadata problem itself, this PAR issue was dismissed as a frequent "red herring" from archivists experienced with this discrepancy.[14]

When this case study was raised for discussion at the NTTW 4 conference, other potential explanations were raised. The "horizontal artifact" error shared was, at least to a 1-inch digitization expert, visually familiar from his experience with similar artifacts created through the digitization and transfer process specific to 1-inch tape, specifically with a faulty time base corrector (TBC), a key component in audiovisual digitization workflows that is used to buffer and stabilize the video signal coming off the original magnetic tape. If this were the case, it seems that the best way to correct for the batch error would necessitate going back to the original analog carrier to redo capture, potentially researching alternative TBCs to use or even treating the media itself for the possibility of improved capture. Other suggestions were that the confounding broadcast metadata is a result of an intervening piece of software with defaults set to the NTSC broadcast standard. This default setting somehow may have gotten saved into the file itself, creating the conflicting metadata.

11 For other reports of both PAL and NTSC Mediainfo reports, see: https://forums.creativecow.net/docs/forums/post.php?forumid=24&postid=988654&univpostid=988654&pview=t.

12 For some further reported PAL and NTSC files and metadata reports, see https://sourceforge.net/p/mediainfo/discussion/297610/thread/14cb90cf/.

13 For a narrative explanation on, see Nagels. "PAR, SAR, and DAR: Making Sense of Standard Definitions (SD) Video Pixels." https://bavc.org/blog/par-sar-and-dar-making-sense-standard-definition-sd-video-pixels.

14 Thank you to Kieran O'Leary for his contributions to this case study during the NTTW 4 "Questionable File: Show and Tell" presentation.

QC in the future could involve checking for this type of broadcast discrepancy to support partial visual QC of batch digitization projects. These types of errors are still relatively difficult to diagnose, in part, thankfully, due to their seeming rarity. The NTTW 4 event, however, brought together specialists who were able to offer feedback based on their otherwise often discretely held specialized knowledge bases with, for example, a particular analog format's characteristics, specific digitization workflow tools, or the inner workings of software checkers in use. While this error, now discovered, can be diagnosed relatively easily, the only practical solutions—to redo the tape transfers or restage the workflow—are not feasible due to resource constraints, which underscores the need to resource and support QC as early in the digitization transfer as possible.

4. HDCAM Output Displayed Through the QCTools Bit Plane Filter

Eddy Colloton

In July of 2019, the Hirshhorn Museum and Sculpture Garden contracted with a vendor to migrate content of one HDCAM tape and one HDCAM SR tape off of their carriers to file-based formats for preservation. HDCAM is a high-definition (HD) digital videotape format released in 1997, used as a professional format for mastering HD productions, followed up by HDCAM SR in 2003, with a higher bit-depth and data rate. We requested delivery of uncompressed 10-bit (v210) QuickTime video files with 24 bit-depth 48 kHz linear PCM audio. I subsequently reviewed the files at the museum. Both files passed all of the museum's quality control (QC) procedures, which involve playing back all files in real time, fixity checks, and review in QCTools.

During a presentation at the 2019 Association of Moving Image Archivists (AMIA) annual conference, Morgan Morel of the Bay Area Video Coalition (BAVC) referenced issues he had observed in certain preservation workflow procedures that resulted in a lack of information in the 9th and 10th bits in a 10-bit depth video file.[15] Morel demonstrated the truncated data in the 9th and 10th bits of a video by using the bit plane filter in QCTools. Viewing video information through bit planes can help to illustrate how the information in the file is stored, with the first bit plane containing the most significant data, and less significant, more granular visual information contained in each successive bit. The QCTools bit plane filter allows a user to view each bit plane of either the Y, U, or V signal individually, or, using the "10 slice" option to view them side by side.

Following the conference, I chose to review our most recent transfers to see if the issues Morel had identified had gone overlooked in my initial QC of the files. Figures 8 and 9 display the Y-channel of the HDCAM and HDCAM SR transfers viewed through the Bit Plane 10 Slices filter in QCTools.

15 Morel, M., Blanche, J., Rice, D., and Hopfauf, L. (2019) Known Issues or Non Issues with AV Preservation Equipment. Association of Moving Image Archivists Conference.

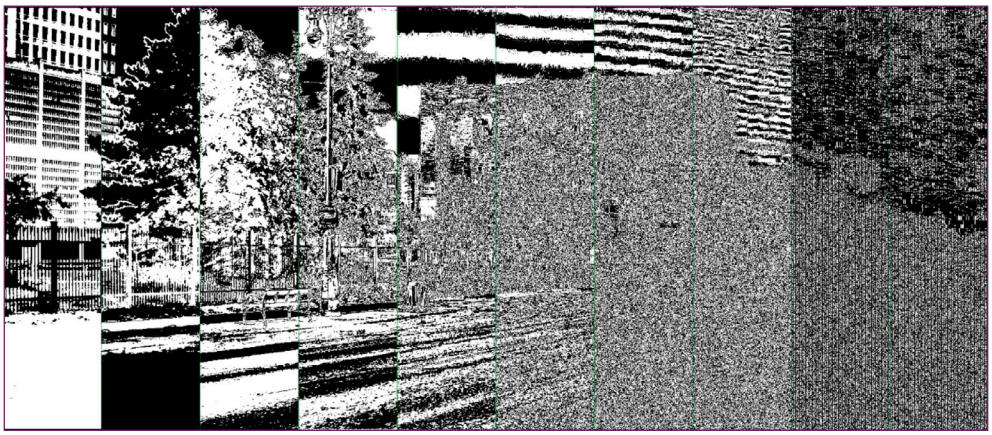


Figure 8. From HDCAM, Y=channel Bit Plane 10 Slices filter in QCTools

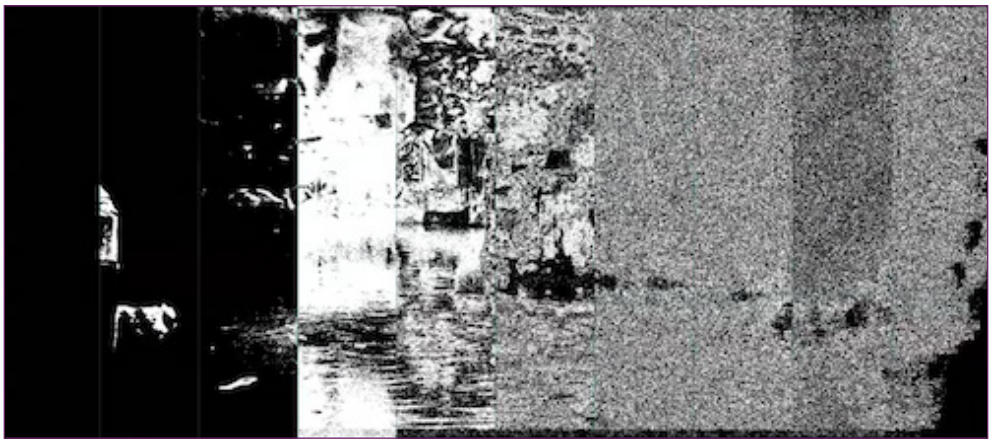


Figure 9. From HDCAM SR, Y-channel Bit Plane 10 Slices filter in QCTools

Typically, the information contained in each bit plane would become progressively more granular from left to right (1st bit plane to 10th bit plane). As these images illustrate, the 9th and 10th bit planes do not necessarily contain increasingly granular information. In the case of the HDCAM transfer in particular, the 9th and 10th "slices" on the right side of the frame appear to contain less unique visual information than the other eight. Keep in mind that these images depict the Y channel of the video signal, which contains both black and white information. While the 9th and 10th bit planes are darker, this does not in and of itself indicate that they contain less information. That being said, the contrast to the other eight bit planes is striking and suggests something unusual may have taken place.

I tweeted short gifs of these bit plane views to see if anyone had encountered similar issues, or had any idea what may have caused this discrepancy.[16] I had a few responses, indicating that there may have been an issue with the deck, or possibly, given that the

16 Colloton, E. 2019. "This looks wrong tho - less info in 9 and 10? From a vendor, so I would have to check [...] Twitter. https://twitter.com/EddyColloton/status/1196908477139968000 [Accessed December 30, 2019].

HDCAM tape only contains 8-bit depth video, the disparity between the 9th and 10th bit planes compared to the other 8 makes sense.

I hadn't really considered this issue when dictating the capture settings to the vendor. Videotape formats are commonly migrated to 10-bit uncompressed QuickTime for preservation.[17] We relied on this practice when selecting a target format for both HDCAM and HDCAM SR. The fact that HDCAM is an 8-bit depth format was overlooked, and may be the cause of the lack of information in the 9th and 10th bits of our preservation master file.

However, HDCAM SR does contain 10-bit depth video. Each tape was played back using an HDCAM SR deck (as HDCAM SR playback decks are backwards compatible with HDCAM). Would the video signal from an HDCAM tape sent out over HD-SDI from an HDCAM SR deck only contain 8-bit depth video or would it have been padded 10-bit depth video? If the source of the issue with the HDCAM transfer is solely due to the format, then why does less information appear in the 9th and 10th bit planes of the transfer from the HDCAM SR tape? Can we know for certain that the issues we're observing in our HDCAM SR transfer are a result of the transfer, and not representative of the information on the tape?

I'm currently working with the vendor to produce video files using the same equipment with different HDCAM and HDCAM SR tapes, to verify that this issue is from the transfer process. If the vendor uses the same transfer process to create files that are not exhibiting this issue, it is possible the issue is a result of the tapes' production, and our transfers are therefore representative of the information encoded onto the tapes by their content creator.

This case demonstrates the complexity of rectifying an issue with a video file once it has been discovered. The source of the issue can be difficult to identify, even harder to understand, and may not necessarily pose a risk, or even be an error at all. If indeed the 9th and 10th bit planes of the video from the HDCAM are simply padding, the additional file size is wasteful, but the original content is still preserved within the digital video file. The video characteristics may not be as representative *as possible* of the original work, but the underlying issue is imperceptible to the viewer, and a low risk to the long-term preservation of the file. Without a thorough understanding of the issue and its source, however, it is impossible to evaluate the issue's significance. Knowledge sharing through conferences, dialog on social media, and conversations with engineers fills a vital need to demystify the complexities of video and empowers stewards of cultural heritage to make more informed decisions. I would not have caught this issue at all if I hadn't seen a presentation about it, nor would I have realized that HDCAM is an 8-bit format if I hadn't tweeted about it; and I would not have been able to troubleshoot the issue without the help of our vendor. Granular and niche challenges, such as the ones described in this paper, present an opportunity to learn from one another, and to build better tools, practices, and communities.

17 For detailed discussion of target format considerations, see Section B of *Guidelines for the Preservation of Video Recordings*, 2019 revised version: https://www.iasa-web.org/sites/default/files/publications/IASA-TC_06-B_v2019.pdf

5. Dolby E Encoding

Shu-Wen Lin and Dan Finn

In 2018 the Smithsonian American Art Museum (SAAM) loaned out a single-channel digital video artwork with dedicated display equipment specified and provided by the artist. While on display, the artist-provided projector died and had to be temporarily replaced. After the loaned equipment was returned, our main objectives were to identify appropriate equipment for a new exhibition format, and to ensure we had everything we needed for long-term digital preservation of the artwork. For long-term preservation, we looked to transfer the HDCAM tape the artist provided at acquisition as a preservation format. However, the work's surround sound audio is encoded on the HDCAM using Dolby E, a proprietary encoding technology which requires specific hardware or software to decode.[18] While this tape format "recipe", HDCAM with Dolby E audio, is common enough in broadcast and production environments, it is not commonly used in a video art context. As a result, we discovered our lab was not prepared to deal with this recipe easily.

In order to ensure we had something preserved for the time being, we copied the exhibition file onto our museum's digital repository. The specifications of that file are lossy, however, so we wanted to generate a higher-quality preservation file by returning to the HDCAM tape.

The SAAM Media Conservation Lab has a Sony SRW-5500 deck that can play HDCAM and HDCAM SR tapes. "Plan A" was to produce a new preservation file by capturing the output of the SRW-5500 using the lab's BlackMagic UltraStudio 4K, the capture card we use to turn video content stored on tape into video content stored in digital files. However, the Dolby E encoded audio made that plan unworkable.

Dolby E is a legacy audio format designed primarily for production and broadcast environments. In the HDCAM context, Dolby E encodes 6 audio channels using only two audio tracks on the HDCAM tape. They are stored on the tape as "non-audio data." While the SRW-5500 can encode Dolby E, it cannot decode that data for capture without additional equipment. Decoding requires specific Dolby E hardware, the DP572, or a proprietary software solution such as Neyrinck's SoundCode or Minnetonka's SurCode.

Our vendor research showed a wide range of potential costs related to decoders. A used DP572 could cost upwards of $4000 USD, but there were also some eBay listings for $200. SoundCode is $2700 for both encoding and decoding of Dolby E, or $2000 for just the decoder. SurCode is $3500, but can be rented for $250 for a week. The rental is the likely choice moving forward. The other price points are achievable for us, but since this piece is the only work in our collection that has an element with Dolby E, we were hesitant to spend 20-40% of our annual lab budget on something that will not be reused. Another determining factor for holding off on either purchase or rental was that the piece is not scheduled for exhibition in the foreseeable future. We decided to first use the equipment and tools that we already had to experiment with no-cost alternatives.

18 For further information on Dolby E in broadcast environments, see de Pomerai, E. "Managing a Real World Dolby E Broadcast Workflow," http://downloads.bbc.co.uk/rd/pubs/whp/whp-pdf-files/WHP175.pdf.

We tried, for example, experimenting with FFmpeg, as it has some capability to decode Dolby E in certain instances. FFmpeg requires the Dolby data to be contained in a file, specifically a transport stream encoded per SMPTE 337 M.[19] Since we have Dolby E on HDCAM as non-audio data, FFmpeg, as predicted, failed in our one attempt.

Based on suggestions on Avid forums, we then tried capturing the non-audio data into two 20-bit 48 kHz channels in order to convert them to surround audio afterwards.[20] After capture, we analyzed the file in MediaInfo and FFmpeg to see if the captured audio was recognized as Dolby E data. The video was successfully captured, but not the audio. When we attempted to play back the captured non-audio data, we only heard hissing and random noise. Attempting to play back non-audio data is not a recommended practice, as the noise produced can damage one's monitors or hearing if the volume is not reduced. We were certainly surprised at how loud non-audio could be. After our experiments to date, there does not appear to be a way to decode and capture the Dolby E surround audio data without purchasing additional hardware or software.

Dolby E is no longer the de facto Dolby codec in use, adding an additional challenge of obsolescence to consider. Next steps will focus on coordinating with the artist's studio in the event they have the audio in a different preservation-level format. Failing that or any other chance discovery, we will most probably rent SurCode for capturing a preservation level audio file.

6. DV Capture and Erratic Display

Annie Schweikert

In 2018, Stanford University accessioned an archival collection that included a significant amount of material stored on hard drives. While surveying the audiovisual component of the collection in 2019, I found that certain video files on these drives did not play as expected. Those files, each created in or around 2005, flickered rapidly and erratically between 4:3 and 16:9 aspect ratios during playback (Figure 10). This issue was present in VLC and ffplay on Mac, Linux, and Windows, but not in QuickTime 7 or X.

Figure 10. 4:3 display aspect ratio (left) and 16:9 display aspect ratio (right).

19 See thread on Dolby E decoder capability ffmpeg-user mailing list archives (September 2017): https://lists.ffmpeg.org/pipermail/ffmpeg-user/2017-September/037112.html [Accessed March 4, 2021].

20 Avid Community, (2009). "Dolby E delivery:" http://community.avid.com/forums/p/69885/391655.aspx [Accessed March 4, 2021].

I wondered how widespread the problem was. Was it isolated to a particular type of file (and thus introduced upon creation), or did it represent a larger problem across the drives? I narrowed down the issue to a small set of files that clearly shared a similar provenance. The video codec of each file was DV, or Digital Video, a born-digital video codec, but one that (at the time) was typically recorded on videotape and then transferred onto a computer as files for editing.[21] The 2005 creation date suggested that the files were transferred from videotape via FireWire, a transfer strategy that would have preserved all technical metadata and data from the original digital video stream (as opposed to treating the original DV tape as solely audiovisual content and substituting new technical metadata).[22] The file format (also called the wrapper or container) is "original" QuickTime—i.e., the Apple specification of the file format, not the later standardized ISO QuickTime profile.[23] The videos were recorded from Israeli broadcast television, meaning the broadcast standard is PAL, and thus the expected display aspect ratio (DAR) is 4:3.[24]

The playback issue clearly stemmed from clashing DARs (4:3 and 16:9), suggesting the problem lay in conflicting technical metadata. At first, it seemed like the conflict could be between the video codec and the file wrapper. The fact that the file is wrapped in the QuickTime file format, and that the behavior disappears during playback in QuickTime, seemed to support this hypothesis, as the specifications for the player and file format stem from the same standards. For example, a QuickTime file header stores metadata in small chunks called "atoms"; perhaps the QuickTime player was reading aspect ratio information from an atom that VLC did not see or could not read.[25]

However, examining the files' technical metadata did not indicate that the wrapper was the source of the wrong aspect ratio. MediaInfo reported a 4:3 DAR for the video stream (i.e., the DV codec), while a tool called DV Analyzer reported a 16:9 DAR for the video stream. Neither tool reported a DAR stored in the wrapper.[26] A closer inspection of the QuickTime header atoms, using Atom Inspector, also did not reveal any DAR information stored in the wrapper.

21 For more information on DV, see "Digital Video Encoding (DV, DVCAM, DVCPRO)," *Sustainability of Digital Formats*, Library of Congress, updated 21 Feb. 2017, accessed 11 Oct. 2020. https://www.loc.gov/preservation/digital/formats/fdd/fdd000183.shtml

22 For more information on DV transfer strategies, see Dave Rice and Chris Lacinak, "Digital Tape Preservation Strategy: Preserving Data Or Video," AVP, 2 Dec. 2009, https://www.weareavp.com/digital-tape-preservation-strategy-preserving-data-video/.

23 The standardized ISO QuickTime profile is based on the earlier Apple specifications, and the use of one or the other is typically a question of date. For more information on the original QuickTime file format, see "Introduction to QuickTime File Format Specification," *Apple Developer Documentation Archive*, Apple Inc., 2004-2016, accessed 20 Jan. 2020. https://developer.apple.com/library/archive/documentation/QuickTime/QTFF/QTFFPreface/qtffPreface.html.

24 For more technical specifications and differences between the PAL standard and NTSC, its North American equivalent, see "Video Learning Guide for Flash," Adobe, 22 Feb. 2011, accessed 20 Jan. 2020. https://www.adobe.com/devnet/flash/learning_guide/video/part06.html.

25 "Metadata," *Apple Developer Documentation Archive*, Apple Inc., 2004-2016, accessed 20 Jan. 2020. https://developer.apple.com/library/archive/documentation/QuickTime/QTFF/Metadata/Metadata.html#//apple_ref/doc/uid/TP40000939-CH1-SW1.

26 MediaInfo reports technical metadata for audiovisual files, with metadata sorted by whether it applies to the file format, the video stream, or the audio stream. DV Analyzer is built around MediaInfo, but it was last updated in 2017 (to Version 1.4.2), and uses an earlier (unspecified) version of MediaInfo than the up-to-date version (CLI version 19.04, released 2019) I used as a standalone tool. "DV Analyzer," *MediaArea*, accessed 10 Jan. 2020. https://mediaarea.net/DVAnalyzer.

I then rewrapped the DV video stream in a new, generic QuickTime wrapper using the following ffmpeg command:[27]

```
ffmpeg -i FILE.mov -c:v copy NEW_FILE.mov
```

The resulting file played successfully in VLC and ffplay, with the expected 4:3 display aspect ratio—a result that seemed to suggest that the problem lay in the original wrapper. However, this result did not explain why different tools were reporting different DARs within the video stream, rather than between the stream and the wrapper.

I brought my issue up during a presentation at the No Time to Wait conference, and FFmpeg developer Carl Eugen Hoyos, who was also attending NTTW 4, offered to take a look at my file. He extracted and played the raw video stream in VLC, where the playback artifact reoccurred. This result proved that the conflicting problem was not in the wrapper, but in the codec itself, as it still occurred when the file was pared down to the raw video stream. He suggested that the file played fine in the QuickTime player, or when rewrapped in a new QuickTime wrapper, because the QuickTime architecture simply picks one display aspect ratio to favor so as not to encounter this error—while VLC does not favor one and therefore reproduces the error.[28]

In pinpointing and addressing this issue, I had to draw on my own knowledge of codecs, wrappers, and diagnostic tools; but especially on the knowledge of others. Though I am still unsure of how the conflicting display aspect ratios were introduced in the first place, I now have a working solution—either simply playing the file in QuickTime instead of VLC, or rewrapping the file to render it universally playable.[29] At the same time, the underlying issue is unresolvable; the original source video and carrier are long gone, and the file cannot be recaptured. I can only understand the files well enough to implement solutions with a layer of security.

Conclusions

Quality control of complex audiovisual signal chains can include any number of steps, but in the past years, many tools have made it easier for cultural heritage professionals to diagnose problems and safeguard their collections. There are now established protocols developed to meet the needs of the many institutions rushing to digitize and ingest at-risk audiovisual content before it's too late. However, bulk or automated workflows are not enough to identify complex problems or edge cases such as the ones discussed in this paper.

Audiovisual workflows necessitate a comprehensive understanding of the full signal chain, carrier history, and file path in order to distinguish and diagnose potential errors, as opposed to norms or issues that may simply be characteristics of the format with no remedy. The lack of documented workflows that fully and honestly engage in these gray areas make it difficult to gain this comprehensive understanding. As these case studies demonstrate, even knowledgeable audiovisual specialists need to rely on sharing

27 This command takes the original FILE.mov, removes the wrapper (demuxes), and creates the output file NEW_FILE.mov. NEW_FILE.mov is a QuickTime-wrapped file with generic default characteristics set by FFmpeg. (This behavior is implied in FFmpeg by giving NEW_FILE an ".mov" extension.) The flag "-c:v copy" ensures that the underlying video codec information remains untouched.

28 Thank you to Carl Eugen Hoyos for this analysis and assessment on 6 December 2019.

29 For a preservation copy, a rewrapped file would need to retain all metadata currently stored in the QuickTime wrapper. A typical access copy would not necessarily need to retain that same information.

and exchanging information with the community to crowd-source potential solutions or explanations. This is even more the case with especially rare or arcane formats, or with staff who do not regularly QC audiovisual files in their day-to-day work.

Collection stewards must accept a certain level of imperfection. Close examination of files can raise dismaying questions, not all of which can be authoritatively answered. No person can know everything, and no project or collection will ever be perfectly preserved. In an environment of limited time, resources, knowledge, and staff, "good enough" is still good, and sometimes has to be enough. Errors, failures, artifacts, loss, and other issues are inevitable, including in repositories where this level of scrutiny is impossible.

There may be no one-size-fits-all answer for QC processes; if the risk of failure is inevitable, then the best answer is to prepare for it. It is sound practice to have multiple points of QC, and validate and keep as much information as possible about files at every point along the signal chain to have a greater chance of pinpointing the exact source of failure when something goes wrong. Perhaps most importantly, this article underscores the importance of consulting with colleagues working on similar projects, sharing resources, and documenting challenges. The problem one institution spends weeks or months solving today could save another organization those weeks or months of headache in the future, or lead them to discover an issue they didn't know they had.

Acknowledgements
The authors would like to acknowledge the many audiovisual and digital file community members that have provided feedback. Special thanks to the No Time to Wait Community, where these case studies were first presented and shared. Thanks also for British organizational support from ITV, the Media Archive for Central England, and the University of Lincoln.

We'd also like to acknowledge and thank Joanna White, Blake McDowell, Caitlin Stewart, Erica Titkeymeyer, and Maddy Smith for providing us with invaluable feedback and comments.