Affiliation

Organization	Address	Country
Internet Based Communication Networks and Services (IBCN) Ghent University - IBBT	Gaston Crommenlaan 8 (Bus 201), B-9050 Ghent	Belgium

Publication Information

Pub ID	Pub Date
2011 Fall Conference Paper #1569466133	October 25-27, 2011

SMPTE Meeting Presentation

Automated File-Based Quality Control: A Machine-Learning Approach

Matthias De Geyter

VRT-medialab, Gaston Crommenlaan 10 (Bus 101), B-9050 Ghent, matthias.degeyter@vrt.be

Nick Vercammen

IBCN, Ghent University – IBBT, Gaston Crommenlaan 8 (Bus 201), B-9050 Ghent, nick.vercammen@intec.ugent.be

Dirk Deschrijver

IBCN, Ghent University – IBBT, Gaston Crommenlaan 8 (Bus 201), B-9050 Ghent, dirk.deschrijver@intec.ugent.be

Tom Dhaene

IBCN, Ghent University – IBBT, Gaston Crommenlaan 8 (Bus 201), B-9050 Ghent, tdhaene@intec.ugent.be

Piet Demeester

IBCN, Ghent University – IBBT, Gaston Crommenlaan 8 (Bus 201), B-9050 Ghent, piet.demeester@intec.ugent.be

Brecht Vermeulen

IBCN, Ghent University – IBBT, Gaston Crommenlaan 8 (Bus 201), B-9050 Ghent, brecht.vermeulen@ibbt.be

Written for presentation at the

SMPTE Technical Conference

Abstract. *In recent years, broadcasters successfully introduced file-based workflows to improve production efficiency. However, they are increasingly dealing with a proliferation of file formats, and many of them still have large archives that need to be digitized for reuse. To guarantee trouble-free workflows and long-term preservation in this quickly evolving digital domain, it is essential that media files adhere to well-described, established standards. Furthermore, their audiovisual quality should be up to broadcast level. A variety of content analysis tools checking container and encoding formats, as well as audiovisual quality, are available but often hard to configure, and frequently provide difficult-to-interpret results. In this research, a learning algorithm takes into account the results of several sources of content analysis to perform a reliable automatic interpretation, which is communicated as a traffic light decision to an operator who can then take further action if necessary. Thus, valuable time and money can be saved.*

Keywords. file-based workflows, quality control, machine-learning, automation.

(The SMPTE disclaimer is on a footer on this page, and will show in Print Preview or Page Layout view.)

Introduction

File-based workflows have become more and more commonplace lately, since they promise faster-than-realtime and concurrent media processing, and subsequently improved production efficiency. Most teething problems have been solved by now, but many challenges and opportunities still lie ahead [1]. One of the most important ones –and the focus of this paper- is the problem of quality control.

Because of file-based contributions from external production companies, and the introduction of affordable cameras such as some digital single-lens reflex (DSLR) or smartphone cameras in professional broadcasting -following the trend of consumerization- there is an ongoing proliferation of media formats within the broadcast facility. Standard-definition (SD) and High-definition (HD) formats coexist within the same environment, and different post-production solutions (e.g. Avid, Apple…) may be used within one company (depending on the program format or personal preference) again implying the application of various media formats or format conversions.

Furthermore, numerous broadcasters have a large archive of past programs, packed away on a variety of storage media such as analog or digital tape, or even film. To preserve these archives for the future, and above all to make them more accessible; e.g. to use extracts within new productions, digitization into current, mature and standardized formats is often the most adequate solution, although the original assets may remain very valuable.

This expansion of media formats may cause problems within existing file-based workflows on several fronts, as will be explained in the next section.

The Importance Of Quality Control

For most early adopters, the transition to file-based production was a long and painstaking process. Several problems arose, such as IT networks and storage that were not up to the task of transporting large media files, or integration issues between products from different vendors. As a result, the initial focus was on overcoming obstacles that prevented an efficient, unified workflow between a small number of selected components. Envisaging the future, probably no one was indifferent to standards compliance at the time, but some standards were not yet fully mature, or subject to interpretation (e.g. the Material eXchange Format (MXF)). On top of that, the digital world is rapidly evolving, resulting in the introduction of new formats for the acquisition, processing and distribution of both consumer and professional media. High-definition (HD) becomes more mainstream and 3DTV experiments are being conducted worldwide.

Thus, it becomes essential to ensure that files in new flavors don't collide with established workflows. One possibility to achieve this involves rewrapping or transcoding essence, but may come at the price of quality loss.

Additionally, long-term preservation poses another problem; many broadcasters are digitizing their tape- and film-based archives, but given the pace of the digital revolution, current state-of-the-art technology may already be obsolete by tomorrow, thereby introducing a risk of loss of media created with such technologies, simply because they can no longer be decoded. Therefore, it is essential that media files adhere to well-described, established standards.

Furthermore, since digitization and format conversions are typically lossy processes (the compromise of quality vs. bandwidth or storage), they could introduce (non-) perceivable quality impairments despite delivering standards-compliant files. This is something to consider for archival storage; e.g. a digitization process may be completely fine by today's norms, but may well prove inadequate in the future (a simplified example: some film footage was digitized before the advent of HDTV and displays no visual difference compared to the original on a Standard-definition (SD) screen, but on an HD monitor a loss of detail is clearly visible).

Naturally, incidental errors may occur as well upon digitization, transcoding, digital transport or storage of media assets. To ensure that media files are compatible with all involved production systems -and are not causing unexpected errors or even system crashes- and that the characteristics of aired programs are on par with broadcast quality, it is essential to set up some kind of verification process.

File-Based Quality Control: Opportunities and Challenges

With the previous section in mind, various essence features can be checked to avoid problems and ensure an optimal quality:

- File or wrapper format: e.g. does an MXF file conform to the standard, does it contain the necessary metadata?
- Compression format: e.g. does a file conform to the MPEG standard?
- Visual quality: this can be assessed by visual inspection or by calculations such as the peak-signal-to-noise ratio (PSNR) or preferably a no-reference quality measure that corresponds more closely to visual assessment
- Occurrence of specific artifacts, such as blockiness
- Audio properties such as loudness and lip sync (a combination of audio and video characteristics)

Detecting errors in the file or compression format by means of software products is quite well established, since these are not subject to interpretation. Visual quality and specific artifacts not directly related to encoding errors, on the other hand, represent a significant challenge. Audiovisual inspection by a human expert is one possible solution, but this is a time-consuming and labor-intensive process. Quite often, these viewers only examine short samples in the beginning, middle and end of a media file, instead of watching it entirely, which may result in undetected errors. Automation is highly desirable and a number of tools provide such automated content analysis. They can e.g. detect artifacts like blockiness, audio clipping (or audio levels or loudness above a certain threshold), silences, a period of black frames etc.

Although these tools can be very useful, they are quite hard to configure, since this involves setting a number of thresholds for a large amount of parameters, while the correlation between an observed audiovisual artifact and one or more parameters is not always very clear. For instance, upon digitization of old video tapes, dirty tape heads may cause errors that are detected as block artifacts, luma or chroma errors, or even field dominance errors, depending on the nature of the content, the severity of artifacts etc. Additionally, the interpretation of analysis results is still labor-intensive and difficult, since these tools generate a listing of all -from negligible to critical- detected errors, which can be cumbersome to look through. This means a trained expert is needed for both configuration and interpretation (Figure 1).

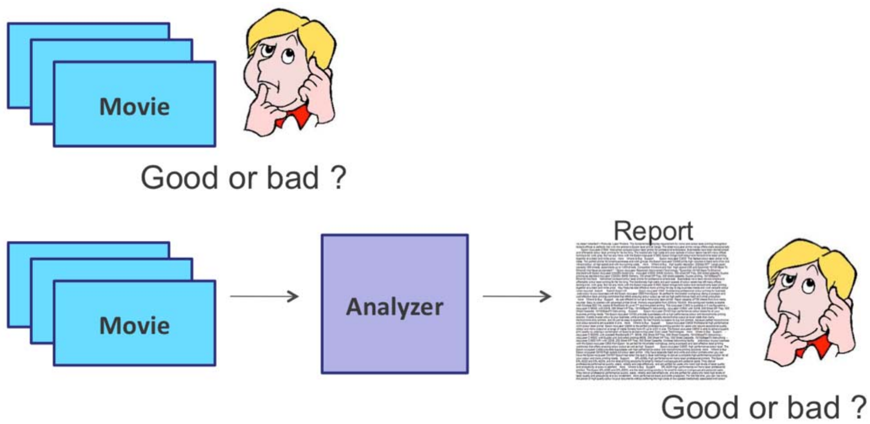

Figure 1: visual assessment of media files vs. software-based content analysis; both are time-consuming

In an attempt to overcome these difficulties, machine-learning techniques were applied to the error reports generated by content analysis tools, with the goal of presenting a simple “traffic light decision” to an operator instead of a listing of failed or succeeded tests, as will be explained in the next section.

Another important aspect of quality control, but outside the scope of this paper, is where to apply it. An obvious choice is to check files upon entrance in a facility, but this may be hard to achieve or not very efficient if there are different entrance points (e.g. journalists’ contributions vs. files from external production facilities, digitized archival material etc.). Another option is to check files before playout, but this involves the risk that there is not enough time left to correct detected errors before airing. Next to this location problem, there may be different requirements for different files; not only if they come in a different file or compression format (a facility may host several formats alongside each other, e.g. DV and IMX) but also depending on the type of content. The visual quality of breaking news content is not as important as for an expensive drama series, and it may very well be impossible to digitize some old archives in a quality that lives up to today’s expectations. Most file-based automated content analysis tools allow or force you to create different profiles for different types of content (they require different profiles for different wrapper or compression formats instead of detecting the format and performing the corresponding checks subsequently, and also allow you to link profiles to watch folders or application programming interface (API) calls), and although not discussed thoroughly (creating optimized profiles for several types of content is obviously very time-consuming), the learning algorithm was applied to two distinct types of content (DV25 and IMX50), using different profiles for the picked content analysis tools.

A Self-Learning Method For Quality Control

In our research, the outputs of several commercially available content analysis tools were combined and served as an input to a machine-learning algorithm. The tools used were a dedicated MXF checker (IRT MXF Analyser Professional) and two content verification products that analyze wrapper and encoding formats, and detect audiovisual artifacts (Interra Baton and Tektronix Cerify). Note that it’s not strictly necessary to execute multiple analyses; the main goal of our quality control is to make sure there are no false negatives (a negative being a file classified as containing no errors in our case), since it is crucial that all files exhibiting real errors are detected, while some misclassified good files (false positives) are acceptable. The combination of multiple tools allows us to improve reliability; the individual tests are independent, but the worst result (e.g. one tool detects an error, while the others let that error pass) is made decisive.

The learning algorithm parses the results of the individual tools and assigns weights to the generated alerts based on a training set classified by experts: if one of the analyzers raises an alert, but there is nothing wrong with the file under test according to the expert, the alert's score is altered to indicate it is less important. Before training, all alerts are regarded as critical errors, which contributes to the minimization of false negatives. The training set is simply divided into "good" and "bad" files; although the expert may have documented the reason for approving or rejecting the file, this information is not used to correlate with raised alerts. As discussed in the previous section, this relationship between audiovisual artifacts and quality analysis parameters (and therefore alerts) is hard to decipher, hence the algorithm relies on the size and variety of the test set to tune alert scores instead (taking a brute force approach instead of manually looking for correspondences between an expert's musings and possibly cryptic parameters, which could be quite time-consuming). Furthermore, some errors are always regarded as critical (e.g. failures against objective standards describing the wrapper or compression format).

When a score of 0 is assigned to alerts that correspond with critical errors, while a score of 100 represents insignificant errors, then all alert scores should ideally converge to these extremes (provided the training set is large enough). In reality, it's possible this convergence won't happen due to a variety of reasons:

- inconsistencies in the expert viewer's evaluation; a viewer may accidentally miss perceivable errors, e.g. due to concentration loss
- contextual information missed by the analysis tools: some alerts may be perfectly acceptable in some situations, but not in others; e.g. depending on their location in the video fragment (beginning, middle, end, although various tools allow to specify this), checkerboard patterns or skyscraper windows that are detected as block errors…
- false alerts: some actual errors are missed by some analysis tools, or sometimes errors are detected where there aren't any; tweaking the parameters during configuration or combining multiple tools allows to reduce this number of false alerts

However, by simple thresholding it is possible to provide an operator with an automated "traffic light decision" (e.g. files with one or more scores below 40 get a red light, files without scores below 70 get a green light, and everything else gets an orange light; see Figure 2). After training, the alert scores aren't changed anymore, and copied from the training phase, but it is always possible to add additional training data.

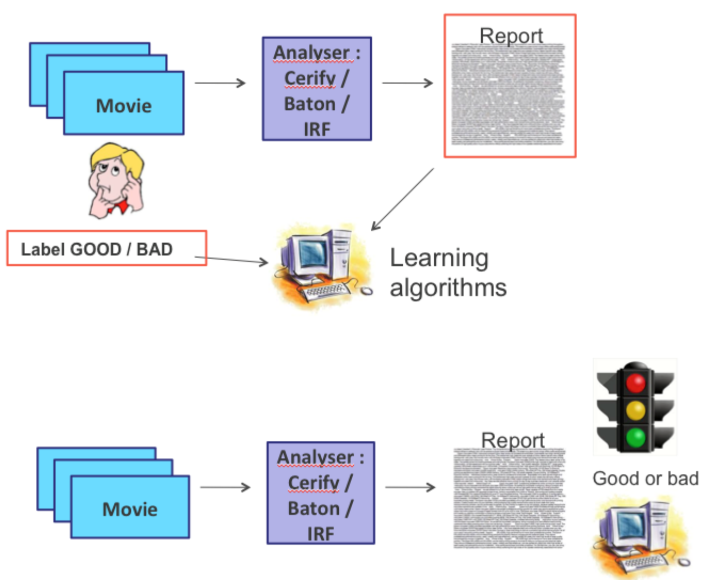

Figure 2: training and working phase of a learning algorithm

Initial tests with this algorithm look quite promising, although the training set was probably not large enough to draw definite conclusions (erroneous files are –luckily- quite rare, thus +/-100 of those were used during training). By combining multiple tools all files that contained errors according to an expert were detected. Using a single content analysis tool reduced accuracy by +/- 5%. With the combined (hence more strict) approach, 50% of the good files were correctly classified as containing no errors, meaning an operator can safely skip checking those, since all files with flaws are detected as such. A single analysis tool may classify more good files properly, but since some bad files are also labeled as correct, all files still need to be checked.

Of course, additional tweaking of parameters can improve results: to save time, most audiovisual checks were performed using the analysis tools' default settings. If a tool doesn't raise an alert for some error (e.g. because parameters are wrongly configured), the learning algorithm can't detect it either; its results are fully based on the generated reports.

Furthermore, it can be argued that the expert, who assessed the training set, may have made mistakes as well (apart from the inconsistencies described above), but our assumption is that if an expert doesn't see certain errors, the viewer at home will definitely miss them. Errors that could break system integrations are typically related to objective standards like the wrapper or compression format and are most likely not detected by viewing, but instead by the analysis software and are regarded as critical.

Conclusion

Quality control is one of the more important challenges in the rapidly evolving world of file-based production, not only for newly shot material, but also for digitized archives.

There are various aspects to quality control; some objective features like conformance to wrapper or compression standards are relatively easy to test and automate, but others such as video quality are a lot harder to assess. Nevertheless, software solutions that meticulously analyze a media file's characteristics exist and look quite promising. Yet the configuration of these content analysis tools, as well as the interpretation of their results is quite complex.

Therefore, a machine-learning algorithm combining the data of several analysis tools was introduced. The algorithm compensates for some of the configuration issues, and communicates an easy-to-interpret traffic light decision to the user, who can then take further action. Using this approach, all erroneous files were correctly classified, while fifty percent of correct files were categorized accurately too.

In the future, an enlarged training set, next to additional tweaking of some configuration parameters, may further improve results. Additionally, commercially available tools will continue to improve, and may even incorporate similar training possibilities some day.

Journal Articles

1. M. De Geyter, L. Overmeire, "File-Based Workflows: Key Challenges in Real-World Facilities" SMPTE Motion & Imaging Journal, 37-42, March 2011