heritage

Technical Note

Preservation of Digital Images: Question of Fixity

Alexey Tikhonov

ROSPHOTO, B.Morskaya 35, 190000 St. Petersburg, Russia; alt@rosphoto.org; Tel.: +7-812-314-1214

Received: 13 February 2019; Accepted: 15 April 2019; Published: 17 April 2019

Abstract: One of the most important aspects of the long-term digital-image preservation strategy is maintaining data fixity, i.e., assuring the integrity and authenticity of original data. This article aims to highlight the limitations of the approaches used to maintain the fixity of digital images in the digital preservation process and to offer perceptual hashing as a way to alleviate some of the limitations of current approaches, as well as discuss some non-technical implications of the described problems. This paper is exploratory, and while it includes a simple experiment description, it only outlines the problem and testing environment for a possible solution that could be elaborated on in further research. The most commonly used fixity maintaining techniques are immutability of data and file checksums/cryptographic hashes. On the other hand, planning for long-term preservation necessitates the need to migrate data into new future formats to maintain availability and sustainability, and the concept of the file itself should not be assumed to remain forever, which calls for other tools to ascertain the fixity of digital images. The problem goes beyond one that is exclusively technical: bitstream content is not ready for human perception, and the digital preservation strategy should include all the necessary technical steps to assure the availability of stored images to human eyes. This shifts the perspective on what should be considered the digital image in digital preservation. It is not the file, but a perceptible object, or, more specifically—instructions to create one. Therefore, it calls for additional tools to maintain fixity, such as perceptual hashing, transformation logging, and others.

Keywords: fixity; preservation; digital; perception; hash

1. Introduction

One of the basic requirements of digital preservation is maintaining the fixity of digital content—the property of a digital file or object being fixed or unchanged [1]—because we need to have confidence that the objects we have in storage are the same as they were when we put them in there. However, the problem with a digital image is that it does not actually exist. What we usually call a digital image is a file with bitstream numeric representation of the image, which is raw data that the digital object consists of, for example, actual colour information for pixels in the digital raster image and headers or technical metadata used to determine the file format for image viewing software. To actually perceive it, we need to have access to that kind of software, be able to use it on some operating system/hardware, and finally have some output system that actually generates the image that we can see, be it a computer monitor, printer, or some kind of device that directly manipulates our optic nerves [2]. Therefore, even if we maintain the fixity of the original file, we cannot cast aside the problem of maintaining other parts of the infrastructure needed to present it as an actual image [3]. The purpose of this paper is to improve the relevancy of metrics used to validate digital images in long-term preservation by shifting the focus from files and data to the perceptive analysis of images. While it is a broad outline that needs further research, some existing technologies and tools can serve as the first steps in this direction.

Another important attribute of the digital image is its intrinsic non-uniqueness. The digital data lifecycle is a never-ending Theseus paradox: there are never 'original' bits, we only have copies.

An important distinction here is that in the digital domain, we can assume that it is possible to have verifiable perfect bit-to-bit copies of objects, which means that in a sense, we know everything there is to know about the object we store. There is a catch though: the same image can be stored with completely different bitstream data, for example, if it is saved in a different image format (Figure 1), which is very likely to happen. While ingesting data, we usually need to normalize it, for example, by converting initial files to formats suitable for long-term preservation. We can be certain that at some point, our data will need to be migrated to another software/hardware configuration that will require converting the files to another format, resulting in them losing bitstream fixity.

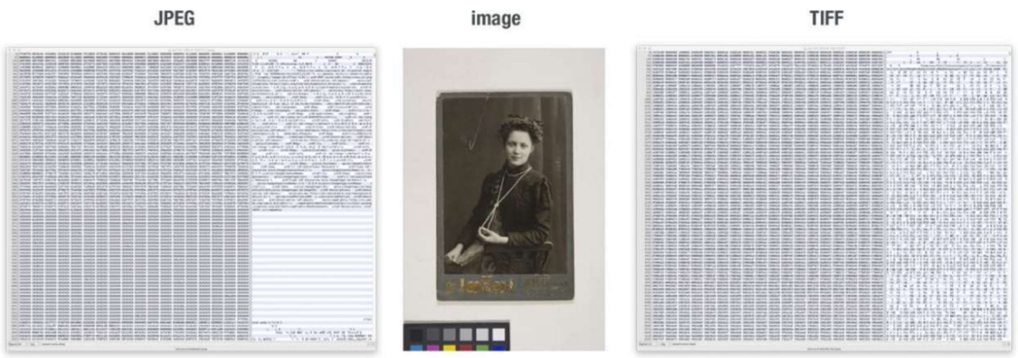


Figure 1. Bitstream representation of a single image in two different formats.

Therefore, it is very important to keep in mind the limitations of traditional tools used to maintain file fixity. One of the most powerful tools to maintain file fixity is the use of hash functions—mathematical functions that can be used to map data of an arbitrary size to data of a fixed size. They are commonly used to assert file fixity because even small changes in a file result in a completely different hash, and hashes are easily stored as small strings. For example, a commonly used hash resulting from the MD5 algorithm can be stored in a hexadecimal format as a 32-symbol string that creates small overhead in storage and can be stored alongside the original file. This approach is used in the widely adopted BagIt convention for storing digital objects. Keeping and checking file hashes is necessary, but not sufficient, for creating long-term policy. We can more or less arbitrary declare an image in one format/bitstream representation a 'master' and use file fixity tools before the next data migration, creating new hashes for files changed during migration after the migration process is completed. Every data migration is a risky process with potential data loss, so we need to be able to mitigate those risks. We need to be able to determine if the migration was successful and that the resulting image was not changed, even if bitstream data was, especially keeping in mind that actual migration is most likely an automatic process affecting a large number of objects.

2. Perceptual Hash in Image Fixity

Perceptual hash functions are in a sense diametrically opposite to cryptographic hash functions: while in cryptographic hashing, a small change in input leads to a drastic change in hashes (called avalanche effect), which makes them great tools for maintaining file fixity, perceptual hashes of similar images will also be similar, which makes them useful for maintaining content fixity [4,5].

Using cryptographic hashes is more straightforward since we only need to determine whether a file was changed, while the probability of collision (having another file with the same hash string) is quite low. There are two possible scenarios for the collision: data corruption and malicious attack. It is very unlikely that corrupted data would result in a hash collision, while a malicious attack, taking into account constant advances in processing power, is not out of the question, so we need to consider the possible intent of such an attack first [6]. If the goal is to substitute a stored image with another image with similar characteristics, described in stored object metadata, a collision attack seems to be extremely unlikely. If, on the other hand, the attacker's goal is to trick the archive into seeing garbage data as a valid stored image, the chance of success mostly depends on the collision resistance of a particular algorithm.

Compared to the cryptographic hashing perception, hashing is less robust and prone to false positives, but in our case, it is used to compare supposedly identical images so that we can mainly focus on the discriminability of the perceptive hashing algorithm [7]. For example, if we need to migrate our images from PNG to TIFF (lossless), md5, sha, and other cryptographic hashes will be completely different and will need to be recalculated, but a comparison based on a perceptive hash will show zero difference. Based on the availability and relative computational cheapness of perception hashing algorithms today (supported, for example, by the most commonly used open source image manipulation library ImageMagick), it could be argued that including perceptual hashes of digital images in archival information packages has not only long-term benefits, since it records information about the image itself, which is the actual object we want to preserve, not the file, but can also be incorporated in dupes/similar searching algorithms.

3. Case Study

A simple test case was used to demonstrate the difference between the phash and md5 cryptographic hash that can be used in maintaining image fixity through format conversion. The source image was converted to TIFF, PNG (lossless), and lossy JPG formats, and another image was created with a deliberate artifact: red box (Figure 2). ImageMagick 7.0.7-21 Q16 x86_64 2018-01-08 was used to compare different images using phash metrics.

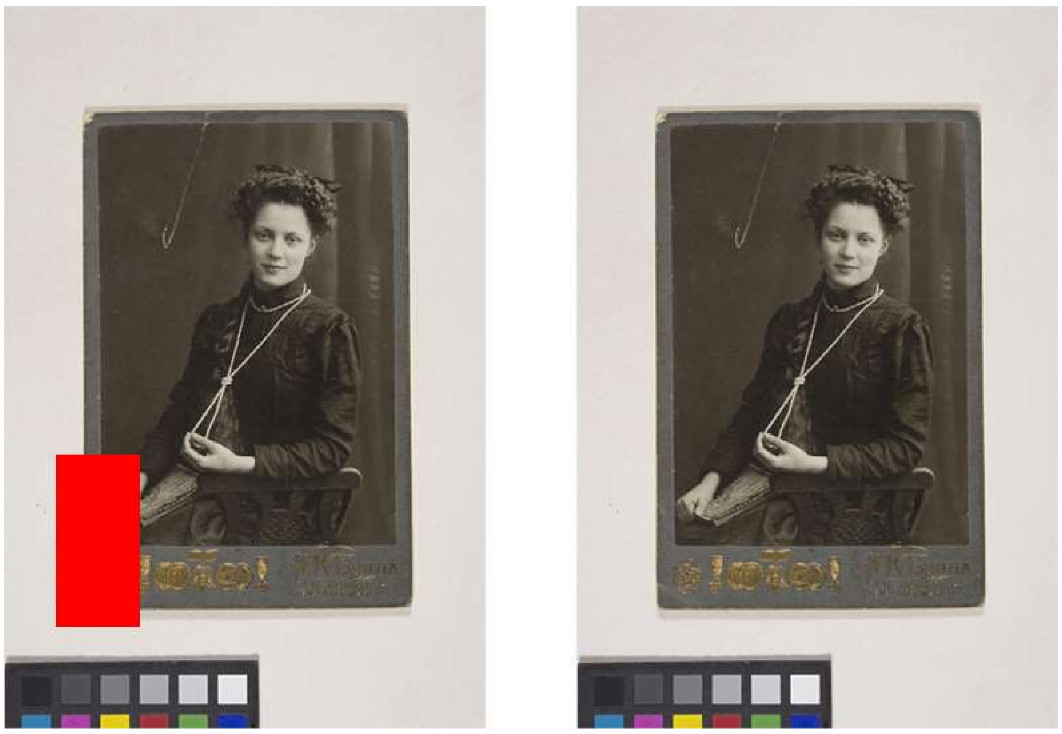
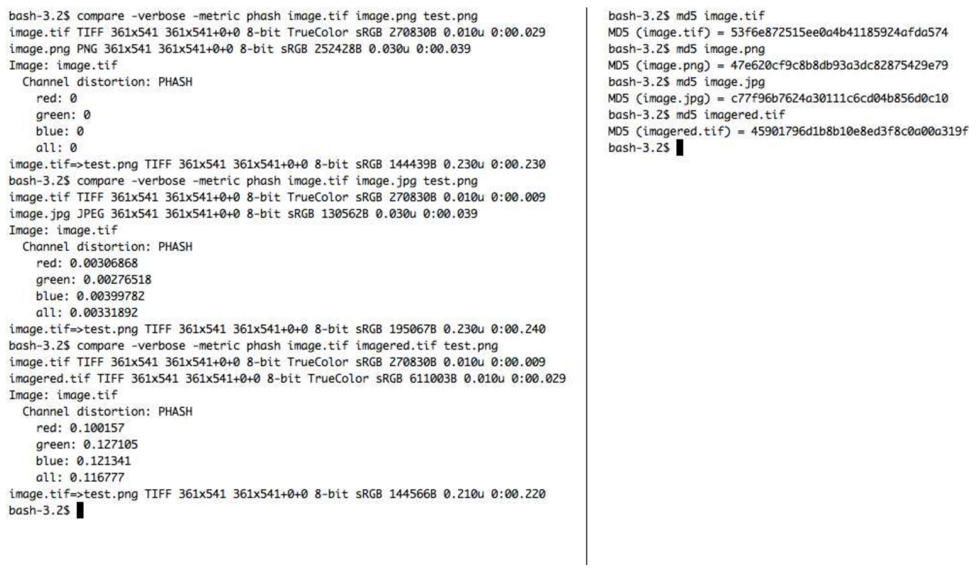


Figure 2. Test images used to test phash comparison. On the left: imagered.tif, and on the right: image.tif, .png, .jpg.

As seen in Figure 3, conversion to lossless PNG shows zero difference using phash metrics, while conversion to lossy JPG shows slight distortion, and a comparison to the image with the artifact shows prominent distortion. At the same time, for crypto hash functions, all files were equally different. What this test shows is that while phash cannot serve as the main fixity checking tool, it can be used to measure image preservation in the migration process. One important possible area of application is the creation of derivatives for dissemination, which are often compressed and resampled to a lower resolution than original images in storage. The use of the phash comparison allows for not just error checking in the derivatives creation process, but also for the limited measurement of derivatives' degradation compared to the original image.



```
bash-3.2$ compare -verbose -metric phash image.tif image.png test.png
image.tif TIFF 361x541 361x541+0+0 8-bit TrueColor sRGB 270830B 0.010u 0:00.029
image.png PNG 361x541 361x541+0+0 8-bit sRGB 252428B 0.030u 0:00.039
Image: image.tif
  Channel distortion: PHASH
    red: 0
    green: 0
    blue: 0
    all: 0
image.tif=>test.png TIFF 361x541 361x541+0+0 8-bit sRGB 144439B 0.230u 0:00.230
bash-3.2$ compare -verbose -metric phash image.tif image.jpg test.png
image.tif TIFF 361x541 361x541+0+0 8-bit TrueColor sRGB 270830B 0.010u 0:00.009
image.jpg JPEG 361x541 361x541+0+0 8-bit sRGB 130562B 0.030u 0:00.039
Image: image.tif
  Channel distortion: PHASH
    red: 0.00306868
    green: 0.00276518
    blue: 0.00399782
    all: 0.00331892
image.tif=>test.png TIFF 361x541 361x541+0+0 8-bit sRGB 195067B 0.230u 0:00.240
bash-3.2$ compare -verbose -metric phash image.tif imagered.tif test.png
image.tif TIFF 361x541 361x541+0+0 8-bit TrueColor sRGB 270830B 0.010u 0:00.009
imagered.tif TIFF 361x541 361x541+0+0 8-bit TrueColor sRGB 611003B 0.010u 0:00.029
Image: image.tif
  Channel distortion: PHASH
    red: 0.100157
    green: 0.127105
    blue: 0.121341
    all: 0.116777
image.tif=>test.png TIFF 361x541 361x541+0+0 8-bit sRGB 144566B 0.210u 0:00.220
bash-3.2$ █
```

```
bash-3.2$ md5 image.tif
MD5 (image.tif) = 53f6e872515ee0a4b41185924afda574
bash-3.2$ md5 image.png
MD5 (image.png) = 47e620cf9c8b8db93a3dc82875429e79
bash-3.2$ md5 image.jpg
MD5 (image.jpg) = c77f96b7624a30111c6cd04b856d0c10
bash-3.2$ md5 imagered.tif
MD5 (imagered.tif) = 45901796d1b8b10e8ed3f8c0a00a319f
bash-3.2$ █
```

Figure 3. Difference between phash comparisons using ImageMagick and md5 cryptographic hashes.

4. Digital Artifacts

As was already mentioned, dealing with objects in the digital domain, we imply that it is possible to have all the information there is about the object, and there is no 'dark matter' between pixels in the digital image. However, this does not mean that we know everything there is to know. The digital image is one of the most straightforward digital objects, and yet it can nevertheless contain some surprises. For example, TIFF is a container format, and there is a possibility that our ingestion tools might miss some of the data present in TIFF (for example, an additional image/previous version of an art object or non-standard metadata). An image may contain steganographic information that could be corrupted during normalization and migration processes. Images can be provided to the institution in the format not suitable for long-term preservation, while not being normalizable without a loss of data (e.g., camera raw files). Therefore, to maintain the fixity of a digital object from ingestion through migrations, we need to either use completely reversible normalization and migration procedures or, at the very least, maintain a pre-normalized copy in digital artifact status, meaning that there is no guarantee that it will be readily available due to possible format obsolescence. This is especially important for objects with incomplete provenance when there is no clear chain of ownership and preservation.

5. Digital Provenance as Fixity Assurance

While hash and other control sum-based techniques for fixity checking are intrinsically connected to digital content, maintaining a complete internal and external history of the object is an important and necessary component in assuring object fixity. The lifecycle of the digital object includes ingestion and sometimes migrations, which change the file and therefore create new hashes. Changing metadata is usually more frequent, and if metadata information is stored in files, as, e.g., according to BagIt specification, and has its own hashes, they too are not immutable. Therefore, maintaining the chain of events that affect stored hashes is as important as maintaining digital object fixity; otherwise, these events might become weak spots in digital cultural object preservation practices [8].

The most common practices include logging all user actions in the database. Many institutions require verified paper trails for changing crucial descriptive metadata. Similarly, the administrative way of maintaining fixity of the digital object involves keeping records of changes and techniques used in, e.g., the migration process.

6. Discussion

Shifting the accent from preserving files to preserving images, we can use perceptual hashes to describe what we actually intend to preserve. The distinction is very important: by using cryptographic hashing/checksums/technical metadata, we maintain material fixity, but we are talking about raw data, that is, in fact, circumstantial to maintaining image fixity since different bitstream objects can represent one image. Using perceptual hashes, on the other hand, is a small step towards being able to verify the fixity of the image itself as we perceive it, adding a new layer to the concept of trustworthy digital objects [9]. While right now it is mostly applicable for fixity checking during format conversion during data migration, using perceptual hashes is an important paradigm change, because now we are testing not what we have in this digital object, but whether it works as intended. That means we have a more robust model for the digital preservation process because it is less dependent on current formats and technologies. The paradigm could be researched further and extended to other types of digital heritage objects: we describe conditions for objects to fulfill their functions in relation to humans and test everchanging digital objects against these conditions.

When we create a long-term digital archive, we have to ensure that archive packages are accessible. It is not enough to store bitstream data and technical metadata for access. We have to be sure that instruments that can be used to access objects exist and are maintained because while physical objects represent themselves by being tangible, digital objects do not. This means that when we prepare a digital archival object, i.e., a digital image, we have to think about how it should present itself, we should store instructions and materials to create objects of perception, and we need to be able to interact with the object; otherwise, we have not prepared it properly.

Some of the complications arise from the fragmented nature of digital objects. Raw data can be stored in one place, metadata in other, while metadata links to common registries and standards and hardware are a completely different matter. A possible way to develop digital preservation could be the movement toward smart archival packages, which know how to represent themselves and help maintain their fixity. We have to store a code to view our content, or at least preserve specifications to build such a code. A possible logical step would be to include that code in the dissemination information package at the very least. As a next step, by peeking into the realm of what science fiction is, could fully self-representative archival objects that include instructions be employed to create hardware or configure some kind of smart matter?

Funding: This research received no external funding.

Conflicts of Interest: The author declares no conflict of interest.

References and Notes

1. *Digital Preservation Handbook*, 2nd ed.; Digital Preservation Coalition: London, UK, 2015; Available online: http://handbook.dpconline.org/ (accessed on 5 August 2018).
2. Phillips, M. The NDSA Levels of Digital Preservation: An Explanation and Uses. Available online: https://www.semanticscholar.org/paper/The-NDSA-Levels-of-Digital-Preservation-%3A-An-and-Phillips/97e881f881d48d4f8a587f9782c5c3bffb821510 (accessed on 12 July 2018).
3. Truman, G.; Aaron, C.; Louis, M.; Carol, K. NSDA 2017 Fixity Survey Report 2017, doi:10.17605/OSF.IO/SNJBV.
4. Zauner, C. Implementation and Benchmarking of Perceptual Image Hash Functions.
5. Hernandez, R.A.P.; Miyatake, M.N.; Kurkoski, B.M. Robust Image Hashing Using Image Normalization and SVD Decomposition. In Proceedings of the IEEE 54th International Midwest Symposium on Circuits and Systems (MWSCAS), Seoul, Korea, 7–10 August 2011. [CrossRef]
6. Kelsey, J. SHA3: Past, Present and Future. Available online: https://csrc.nist.gov/CSRC/media/Projects/Hash-Functions/documents/kelsey_ches2013_presentation.pdf (accessed on 5 August 2018).

7. Yang, B.; Gu, F.; Niu, X. Block Mean Value Based Image Perceptual Hashing. In Proceedings of the 2006 International Conference on Intelligent Information Hiding and Multimedia, Pasadena, CA, USA, 18–20 December 2006; pp. 167–172. [CrossRef]
8. Bralić, V.; Kuleš, M.; Stančić, H. A Model for Long-Term Preservation of Digital Signature Validity: TrustChain.
9. Gladney, H. Long-Term Preservation of Digital Records: Trustworthy Digital Objects. *Am. Arch.* **2009**, *72*, 401–435. [CrossRef]