Preserving a Hardware-Dependent Digital Artwork: Investigating Disk Imaging and Emulation Strategies

David Cirella, Claire Fox, Ethan Gates, Madeline "made" Smith

Yale University Library; The Museum of Modern Art

New Haven, USA; New York, USA

david.cirella@yale.edu; claire.fox@yale.edu; ethan.gates@yale.edu; madeline_smith@moma.org

Abstract

This paper summarizes the efforts and findings of a collaborative case study undertaken by a team of digital preservation and conservation staff to preserve legacy Apple hardware included as part of an accessioned artwork Yale University Art Gallery. Intended to test the capabilities of the EaaSI (Emulation-as-a-Service Infrastructure) framework for assessment and exhibition of digital art, a number of specific technical and logistic hurdles in pursuing emulation raised challenges for long-term preservation workflows involving unique hardware.

Keywords

Emulation, disk imaging, software preservation, hardware preservation, animation, media arts preservation, EaaSI

Introduction

In the summer of 2021, digital preservation staff from a Yale University Library (YUL) and a conservator from the Yale University Art Gallery (YUAG) selected *Tree Turbine* (2007), an artwork by Joseph Smolinski, from the gallery's collections as a case study for artwork assessment and access via emulation technology in a museum context. The artwork, an animated short video intended for exhibition, included mid-2000s era-specific hardware components and software applications, both of which required long-term preservation actions and planning in order to make future exhibition feasible.

Emulation, a technology tool and practice that has been used in preservation and conservation contexts for years, is still a strategy that is considered emerging or experimental in many preservation and conservation departments across cultural heritage institutions. In "The Australian Emulation Network: Accessing Born Digital Cultural Collections", presented at the Second Summit on New Media Art Archiving, Melanie Swalwell described growing community efforts to take advantage of emulation's potential by bringing together a group of practitioners and collecting organizations, all using the same remotely-accessible emulation platform, EaaSI (Emulation-as-a-Service Infrastructure). [1] This paper builds on those efforts by focusing on a case study to practically implement the same collaborative, web-based emulation framework described by that previous paper in the context of assessing and preserving a particular piece of digital artwork. The selected case study was an opportunity to test real-world implementation of emulation and the EaaSI platform at Yale - the host organization for a U.S.-based network of academic and museum partners (currently funded by the Mellon and Sloan Foundations) equivalent to the "AusEaaSI" group. [2] The collaborators on the *Tree Turbine* project anticipated the need to be adaptive in their work, documenting their processes and remaining open to alternative strategies when roadblocks were encountered.

Surprisingly, prior to reaching a stage when emulation would be incorporated into the process, the team realized that their regular processes for disk imaging or reformatting legacy storage media would also require alternative strategies from their typical workflows due to the conservation requirements for the artwork. In addition, the challenges encountered during emulation experimentation uncovered unique legacy hardware issues that had not yet been seen in previous EaaSI use cases. This required adapting typical preservation and conservation workflows for born-digital objects -- specifically, born-digital artworks with both hardware and software components -- from start to finish.

The following paper documents the case study, which includes disk imaging of the artwork and investigating emulation as an assessment and exhibition strategy, along with context for the artwork from the gallery's collection. While the actions taken by the *Tree Turbine* preservation team fell within the overall scope of a standard digital preservation workflow, it diverged in ways that opened new avenues for the study of preserving born-digital artworks in a way that preserves the era-specific integrity of these works so they can be experienced by the public for years to come.

Artwork Background

Joseph Smolinski's *Tree Turbine* is a high-definition digital animation video artwork. Created in 2007, the artwork is part of a series Smolinski created around the concept of what humankind's potential biotech future might look like. Smolinski began "working with the imagery of cellular communication towers disguised as trees," thinking about how to use the aesthetic of the common cell tower tree to build, as he describes it, "a spinning tree turbine that would generate usable electricity and camouflage into the landscape." [3] *Tree Turbine* is Smolinski's first envisioning of this functioning electric wind turbine, created as an animated concept video for a 20-foot tall "tree turbine" prototype Smolinski later built in 2008 as part of an exhibition at the Massachusetts Museum of Contemporary Art, United States.

Tree Turbine was created during a period in which Smolinski was beginning to explore how to translate his work into different mediums, including the use of 3D animation. 3D animation has since become a part of his regular artistic practice. [4] Joseph Smolinski is a multidisciplinary artist and educator based in New Haven, Connecticut. Through his artist practice, Smolinski questions "the shifting roles of technology within communication networks, energy and oil companies, and the industrial agricultural infrastructure, which indelibly shape the so-called natural environment." [5]

The digital animation runs on a 3 minutes 40 second loop, depicting a series of power-generating wind turbines in the form of fake pine trees that slowly spin in the wind as the animation moves through various scenes where the turbines are installed and generating power, including a suburban neighborhood, a landfill, a cabin, a roadside electric car charging station, and a view of the Los Angeles skyline. [6]

As a physical artwork, the core components of *Tree Turbine* consist of an H.264 .MOV digital video file stored on a 2006 Mac Mini computer. The Mac Mini is connected to a modified Apple 23-inch cinema display monitor, dating from between 2004 and 2008, that displays and plays the digital file, in past exhibitions using the Apple Quicktime application as the playback method. A dedicated mouse and keyboard was used to control the computer and monitor.

Process

Various qualities of the artwork required a multifaceted approach to preservation that protected the integrity of the current, functionally accessible piece, and also considered what future access, beyond the lifespan of the physical object, would include.

The integrated nature of the artwork, encompassing both the digital file of the animation and the playback environment (including the software, operating system, and computer hardware), presented various challenges.

Challenges

The requirement to maintain the physical integrity of the artwork precluded what would have been the library team's normal practice of removing the hard disk and imaging it using a write blocker while cloning the data.

Replicating the digital file of the animation fell short of preserving the full object, including playback software and operating environment. In addition, any attempt to duplicate the file from within the operating environment used for exhibition could result in unintended changes to the data or supporting environment.

Future access to the artwork as result of these preservation actions should be as close as possible to the original exhibition.

Disk Imaging

To strictly maintain the physical integrity of the host system while preserving the work, we devised a process using a Linux-based LiveCD to operate the original hardware to create a full copy of the internal disk to an external hard drive.

A LiveCD is a type of operating system that runs in read-only mode from a CD or DVD via the internal optical disc drive. All data generated during the session is held in system RAM and is flushed on power-down, protecting the internal disk from modification. [7] For compatibility with the specific computing hardware, the ubuntu-13.04-desktop-amd64+mac LiveCD was used to operate the Mac Mini. [8]

Once the LiveCD system was booted, the "lshw" command line utility was run to document the internal hardware components and provide details about the hard disk including, the size, partition listing, serial number, and disk UUID. [9] Next, a secondary external hard drive was attached via USB and mounted to serve as the destination for the image of the internal disk. As destination for the clone, the secondary hard drive needed to be reformatted to a filesystem, both compatible with the decade-old operating system chosen for our LiveCD and able to support file sizes large enough to store our intended disk image file. The ext4 filesystem, commonly used by Linux-based operating systems, satisfied both requirements. [10]

For capturing the highest fidelity copy of the work, the method of block-level copying was chosen to create a bit-stream duplicate of the internal disk. An imaging process of this type accesses all sectors of the source disk and replicates every block to a destination disk or file. During this operation, the source disk remains unmounted, negating the risk of any changes or new data being written to the source disk. The output of a block-level copy at the device level captures all sectors on the physical disk, including unused or previously used space, and all partitions regardless of what is visible to the host system performing the imaging.

This type of image is viable for use in various methods of long-term access. The full image can be dumped back onto a new physical hard disk, enabling a replacement drive to be used in the original hardware. The image can be mounted from a modern host system to gain file-level access to the disk image contents. Using a compatible emulator, the image can also be booted in a virtual, emulated computing environment configured to mimic the original hardware.

The “dd” command line utility was used to perform the block-level copy of the internal disk to the destination drive. The command takes the source and destination as parameters, along with options that specify block-size and conversion behavior. To safe-guard against accidental write to the source drive, output from the following command line utilities were evaluated to verify the device names, locations, and status of each drive: “fdisk” for reporting sector size, disk identifiers, and device path and “df” to verify the source drive had not been mounted before imaging. [11, 12]

After verifying our invocation, the dd command was run with the “conv=noerror,sync” option. These set error handling parameters in the case that any disk sector returns a read error. The “noerror” option instructs dd to advance to the next sector and keep imaging; sync instructs dd to pad a read error sector in the destination copy, keeping the sequence of data in the copy readable. [13] In this case, producing an image that captures read errors, should they exist, allows the maximum amount of data to be cloned. It is also possible that the error is returned when reading an empty sector, thus no data is lost despite the error.

Executing the dd command cloned 80026361856 bytes (80 GB) in 3328.56 seconds, at a rate 24.0 MB/s, producing our block-level copy of the internal disk.

With the cloning of the internal disk complete, we proceeded to verify the integrity of our clone and establish a hash value to be used in future verifications. The first step included creating a checksum of the source data by using the md5sum utility to compute a hash value for the internal disk drive. [14] Next, a checksum of our disk image file was computed using the same md5sum utility, before comparing the values from each, finding identical output. The checksum value has been stored alongside the image file and will be used to verify the integrity of the image file following moves between different systems and storage locations.

After confirming a bit-perfect copy of the disk contents, we performed verification of content access by mounting the image on a modern Linux system to access and copy individual files from the disk image. This proved the viability of the cloned disk, specifically the ability to mount the file system of the imaged partitions from an external system to access the original files.

Emulation

The open source emulator QEMU was selected for attempting to run the Mac Mini disk image in emulation, as it is currently the only emulator both compatible with the EaaSI framework and capable of running Intel x86-based systems similar to the Mac Mini (to the team's knowledge, no open source emulator is available to specifically recreate a 2006 Mac Mini). [15]

Working directly with the block-level raw disk image presented challenges for experimenting and testing the disk image in emulation. Sharing an 80 GB file between remote team members and systems (like EaaSI) would require lengthy upload/download times given bandwidth limitations, and create an unnecessary strain on EaaSI computing resources.

To mitigate these concerns, we used the QEMU project's disk image utility ("qemu-img") to create a sparsified and losslessly compressed copy of the raw disk image in the QCOW2 disk image format (essentially to serve as an "access copy" of the block-level disk image for sharing and access). Sparsification detects unassigned memory in the source disk image's file system, allowing the access disk image copy to only take up as much storage space on the host system (e.g. digital preservation workstation or EaaSI server) as is actually used by system and user data in the guest system/disk image; further lossless compression (using the zlib library) shrinks the size of the access disk image even more while still allowing it to be uncompressed on-the-fly when run in QEMU. [16]

The exact command used was:

$ qemu-img convert -O qcow2 -c *raw_disk_image.img access_disk_image.qcow2*

This resulted in an access disk image sized only approximately 14 GB to the original, raw disk image's 80 GB. The conversion was also reversed and verified against

the original, raw disk image, again using md5sum, to ensure the qemu-img compression was indeed lossless.

Unfortunately, from there, attempts to run the access disk image on a local workstation using QEMU stalled almost immediately. Though QEMU can nominally emulate the same Intel x86 processor architecture used by the Mac Mini, the specific version of Mac OSX installed on the Mac Mini - 10.4 - used a unique method for fetching processor information during the operating system boot process. [17] This method checked for, and expected, defined responses unique to the particular CPUs used by Apple on their hardware, and none of QEMU's emulated processors (neither its generic x86 emulator, nor a number of specific emulated CPU models) appear capable of returning the required response. In other words, the operating system realizes it is not running on real Mac Mini hardware and refuses to boot.

After much further experimentation, the only path found to run the access disk image in QEMU was to obtain installation media for the Intel x86 version of Mac OSX 10.6 - a later version that does not have the same specific processor checks built into its kernel - which could successfully boot in QEMU and be used to update the operating system on the access disk image from 10.4.10 to 10.6. [18] On subsequent attempts QEMU could then boot the updated access disk image and allow exploration and assessment (using QEMU's "snapshot" mode to avoid any further user alterations, accidental or otherwise). [19]

Though this path marginally moved experimentation forward, options for further assessment or access remained limited. Even booting Mac OSX 10.6 requires use of a custom bootloader not included in default/common distributions of QEMU. [20] Though this bootloader could be acquired and used on a local workstation, it is not currently available in the EaaSI framework, making it pointless at this point to attempt to upload, run, or share the updated access disk image in EaaSI. [21] And while initial evaluation made it appear that no meaningful user data was changed during the process of upgrading the operating system from 10.4.10 to 10.6 (the system's user/registration information, Applications folder, and general file hierarchy appeared to remain unaltered), the status of the original Mac Mini as an accessioned artwork once again called into question whether the OS changes performed would be deemed too significant by the artist, curators, or scholars.

Logistics - a limited amount of time available to further pursue the case study; distributed team members; restricted/limited access to physical space; and the inability to run the emulation remotely in EaaSI - prevented a side-by-side comparison of the access disk image emulated in OSX 10.6 in QEMU against the original hardware within the scope of this case study.

Conclusion

Over the course of this case study, the *Tree Turbine* preservation team encountered a range of challenges and insights related to digital preservation practice within a time-based media art conservation context. Primary takeaways from the work conducted included:

- Legacy Apple hardware presents unique, often highly specific challenges to both digital preservation best practices and long-term access, exhibition, and assessment.
- Working with era-appropriate tools required to operate legacy hardware requires finding documentation of technical limitations present in utilities, tools, and systems from their date of release.
- Pathways toward professional development are needed for staff to gain this technical, historical knowledge, and have the ability to apply it in practice.
- Collaboration between units and staff with varied expertise can help reduce the amount of redundant, siloed knowledge.
- Departmental policies and standards are needed in order to document these esoteric requirements in assessments, acquisition records, catalogs, and other areas where critical documentation is compiled.

Some of the challenges encountered were anticipated or have documented precedents. The need to disk image a hard drive while maintaining the physical integrity of the host system, for example, was a challenge that was unusual for a library digital preservation department, but standard within a museum context with time-based media collections where art works might include functional hardware as part of an art object. In that regard, the work conducted on *Tree Turbine* allowed the university library team to expand its disk imaging practices, and the university art gallery to refine its policies regarding the care of born-digital art works.

Still, some of the encountered challenges presented new, as-yet unresolved issues that will require further study, testing, and documentation. The challenges uncovered by experimentation with emulation technology and the EaaSI framework demonstrated the need for specific technical

and computing knowledge, the need for different units and staff to collaborate in service of sharing that knowledge and applying it within the boundaries of departmental policies and standards, and the capacity to expand existing digital preservation best practices in service of long-term access, exhibition, and assessment of born-digital artworks. Ultimately though, we hope further case studies and efforts in collaborative emulation services will foster Melanie Swalwell's vision for "A Community of Practice [that] will build confidence in the GLAM sector around born digital collecting." [22]

References

[1] Melanie Swalwell, "The Australian Emulation Network: Accessing Born Digital Cultural Collections", *ISEA 2022 Proceedings of the Second Summit on New Media Art Archiving*, https://isea-archives.siggraph.org/wp-content/uploads/2022/11/ISEA2022_Proceedings-of-the-Second-Summit-on-New-Media-Art-Archiving.pdf

[2] "What is EaaSI?", EaaSI program website, 2022, accessed December 20, 2022, https://www.eaasi.info/what-is-eaasi

[3] Joseph Smolinski, Yale University Art Gallery artists' questionnaire, 2021.

[4] Joseph Smolinski, Yale University Art Gallery artists' questionnaire

[5] Joseph Smolinski, "About", Smolinski Studio website, accessed December 19, 2022, http://www.smolinskistudio.com/about

[6] Joseph Smolinski, Yale University Art Gallery artists' questionnaire

[7] "Ubuntu Documentation", LiveCD - Community Help Wiki, June 2, 2012, accessed December 20, 2022, https://help.ubuntu.com/community/LiveCD

[8] "Ubuntu 13.04 (Raring Ringtail)", Ubuntu 13.04 server installation media, accessed December 20, 2022, https://old-releases.ubuntu.com/releases/13.04/

[9] Carla Schroder, "Getting Detailed Information About Your Computer Hardware", in *Linux Cookbook: Essential Skills for Linux Users and System & Network Administrators, 2nd Edition* (O'Reilly Media, 2021), https://go.oreilly.com/stanford-university/library/view/-/9781492087151/

[10] Carla Schroder, "Managing Disk Partitioning with parted", in *Linux Cookbook: Essential Skills for Linux Users and System & Network Administrators, 2nd Edition.*

[11] "Fdisk(8) – Linux Manual Page", Man7.org, 2022, accessed December 21, 2022, https://man7.org/linux/man-pages/man8/fdisk.8.html

[12] Mark Sobell, *A Practical Guide to Linux Commands, Editors and Shell Programming* (Pearson, 2017), 793.

[13] Shiva V.N. Parasam, "Evidence Acquisition and Preservation With Dc3dd And Guymager", *Digital Forensics with Kali Linux – Second Edition* (Packt Publishing, 2020), https://learning.oreilly.com/library/view/~/9781838640804

[14] "md5sum(1) – Linux Manual Page", Man7.org, 2022, accessed December 21, 2022, https://man7.org/linux/man-pages/man1/md5sum.1.html

[15] QEMU Project Developers, "x86 System emulator", QEMU documentation, 2022, accessed December 20, 2022, https://www.qemu.org/docs/master/system/target-i386.html

[16] QEMU Project Developers, "Disk image file formats", QEMU documentation, 2022, accessed December 20, 2022, https://www.qemu.org/docs/master/system/qemu-block-drivers.html?highlight=qcow2#cmdoption-image-formats-arg-qcow2

[17] Landon Fuller, "Mac OSX 10.4 under VMware Fusion on Modern CPUs", Landon Fuller's blog, December 17, 2013, accessed December 20, 2022, https://landonf.org/2013/12/index.html

[18] Ladon Fuller, "Mac OSX 10.4 under VMware Fusion on Modern CPUs".

[19] QEMU Project Developers, "Snapshot mode", 2022, accessed December 20, 2022, https://qemu.readthedocs.io/en/latest/system/images.html?highlight=snapshot#snapshot-mode

[20] Gabriel L. Somlo, "Running Mac OS X as a QEMU/KVM Guest", October 21, 2018, accessed December 20, 2022, http://www.contrib.andrew.cmu.edu/%7Esomlo/OSXKVM/

[21] "qemu-eaas", Emulation-as-a-Service/emulators source code repository, accessed December 20, 2022, https://gitlab.com/emulation-as-a-service/emulators/qemu-eaas

[22] Melanie Swalwell, "The Australian Emulation Network: Accessing Born Digital Cultural Collections"

Author(s) Biography(ies)

David Cirella (he/him) is a Digital Preservation Librarian at Yale University Library. In this role he works with stakeholders from around the institution towards the long-term preservation of their digital content. His areas of interest include digital forensics, programming, and information retrieval. He currently serves on the Documentation and Training Committee of the BitCurator Consortium.

Claire Fox (she/her) is a Digital Preservation Librarian at Yale University Library, where she oversees the administration, support, and expansion of Yale's instance of the Emulation-as-a-Service Infrastructure (EaaSI) program of work, with an aim to provide broader access to legacy born-digital collections at Yale. She is a member of the Software Preservation Network's Coordinating Committee and Metadata Working Group, and holds an MA from New York University in Moving Image Archiving and Preservation.

Ethan Gates (he/him) is a Software Preservation Analyst at Yale University Library and User Support Lead for the EaaSI (Emulation-as-a-Service Infrastructure) program of work. He is responsible for troubleshooting and documenting the EaaSI

platform, as well as providing training and community support to the U.S.-based EaaSI Network and digital preservation field at large on topics of emulation and software preservation. He wrote the entry on "Emulation" in The Handbook of Archival Practice (Rowman & Littlefield, 2021) and is a member of the Software Preservation Network's Community Engagement Collaborative, the BitCurator Consortium's Membership Committee, and the Association of Moving Image Archivists' Open Source Committee.

Madeline “made” Smith (they/them) is the David Booth Fellow in Media Conservation at The Museum of Modern Art (MoMA). They have worked with media collections at the Center for Constitutional Rights, ArteEast, and Ballet Tech, all in New York; the Los Angeles County Museum of Art; the Smithsonian American Art Museum, in Washington, D.C.; the Yale University Art Gallery, in New Haven, CT; and with media artists’ personal collections. made holds a B.A. in American Studies and English from the University of Virginia (2015), and an M.A. in Moving Image Archiving and Preservation from New York University’s Tisch School of the Arts (2020). Their master’s thesis was on the history of the Matters in Media Art web resource and the stewardship of time-based media in art museum collections.