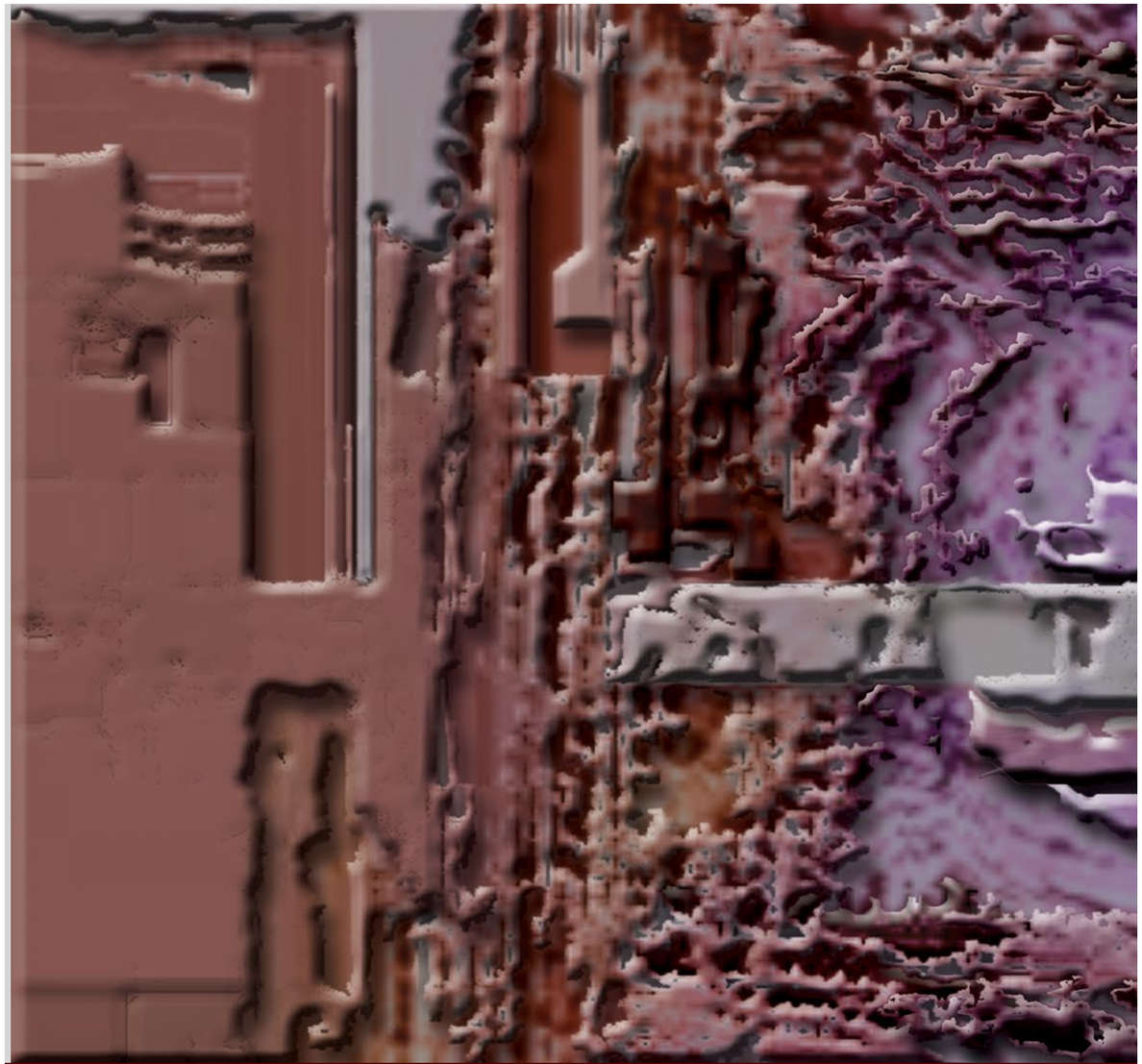
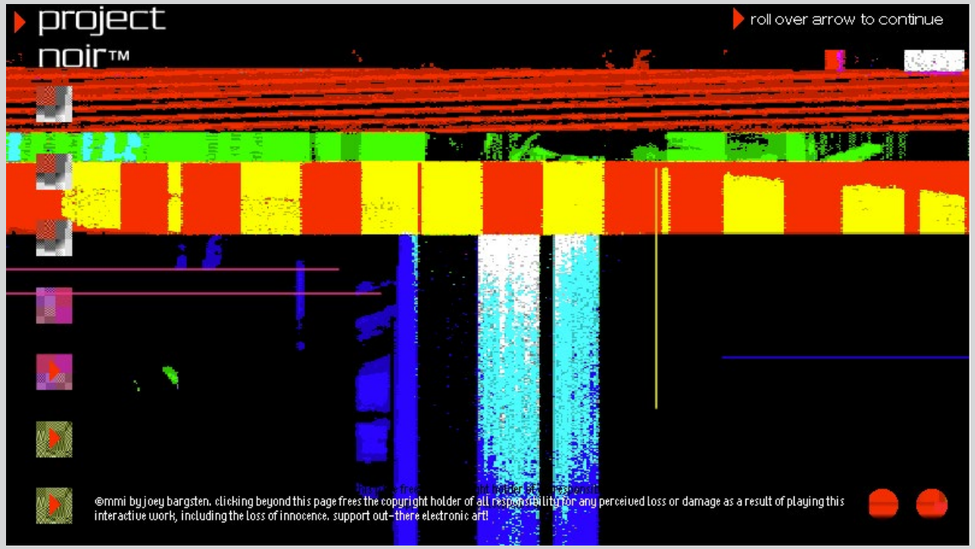


PRACTICUM: INTRODUCTION TO GLITCH ART AND DATAMOSH VIDEO

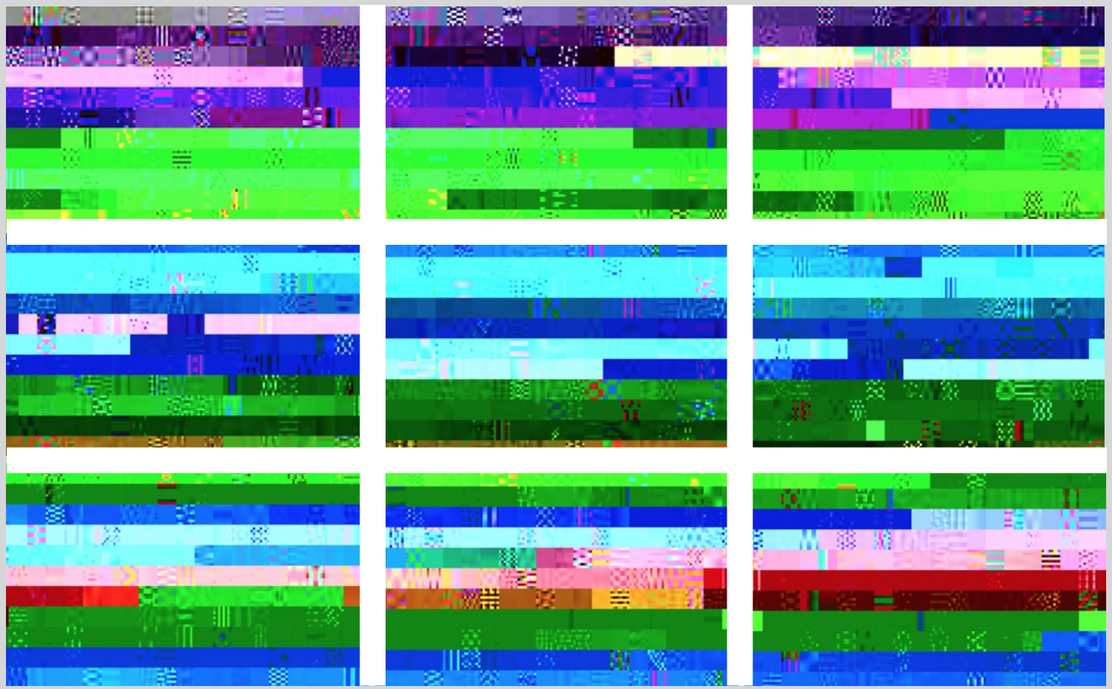
The "glitch aesthetic" is based to a large extent on the wide array of techniques and practices that enable us to create glitch art. By my definition, "glitching" art is the process of taking a visual image and distorting it, while also subverting the design app and production techniques used to create it. To some, this may seem unnecessarily rebellious, but art and design practices seem to be moving so far away from the "traditional" these days—in museums, marketing, and even on corporate Web sites.

My "puffy glitch" above is an example of basic glitch art. In Photoshop, I've selected individual colors and copied them to new layers, then applied an "Emboss" layer, and the result is a more dimensional texture.

My first official use of glitch as art dates back to 1997 in a module called *Project Noir* from my Web-Art site *BadMindTime*™. I'd just bought a scanner, and it broke, generating the groovy colors and visual compositions I decided to use as the splash page for this interactive work.



Here's a set of glitchy images I generated in PureData—primarily a sound-editing application.



For more background, here are two videos dealing with glitch art (typically referring to still images) as well as datamosh projects (video images).

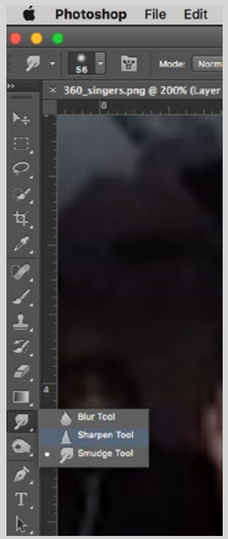
- Offbook (YouTube): *The Art of Glitch* [P.1.1]. This video, by PBS Digital Studios, gives you some historical perspective on glitch art and datamoshing. One of the designers it highlights is Anton Marini (a.k.a. DJ Vade), a really innovative guy who created a plug-in for Quartz Composer (called the Rutt-Etra) designed to alter video images in interesting unpredictable ways.

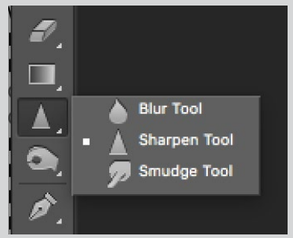
- *Chairlift: Evident Utensil* [P.1.2]. This is one of the earliest examples of the glitch art/datamosh aesthetic. In this case, it's applied to a music video, directed by Ray Tintori, who supervised the datamoshing. (You've probably seen some imitations of Tintori's work in videos by well-known singers.)

Let's try out some of my suggestions to get you started glitching—in this case using Photoshop, TextEdit and the Rutt-Etra-Izer. I'll also give you some links for learning the Datamosh and Rutt-Etra plug-ins for Quartz Composer. From there, it's up to you to keep going and continue sharing your aesthetic with others. Document your glitch processes, especially if you find some settings and techniques you really like and want to duplicate.

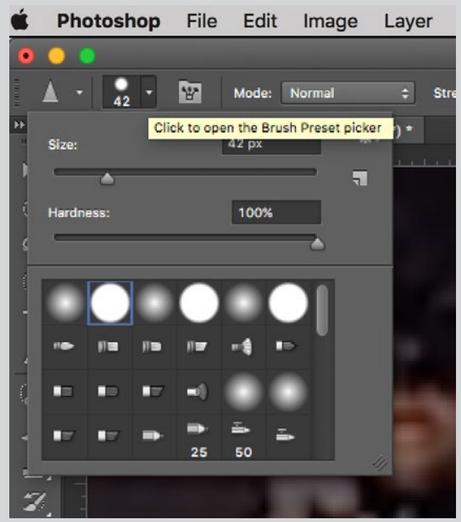
I. Glitching with Photoshop (PS)—Five Suggestions. Any version of PS should do. *Need an introduction to Photoshop? Check out the tutorials here:* [P.1.3].

Explore the Sharpen Tool (sharpening the pixels of an image). After you've opened an image in PS, go to the vertical toolbar on the left and find the icon for the Blur, Sharpen, and Smudge Tools. (I have a Sharpen icon [cone] on my toolbar, but you might see any one of those three icons if last used.)

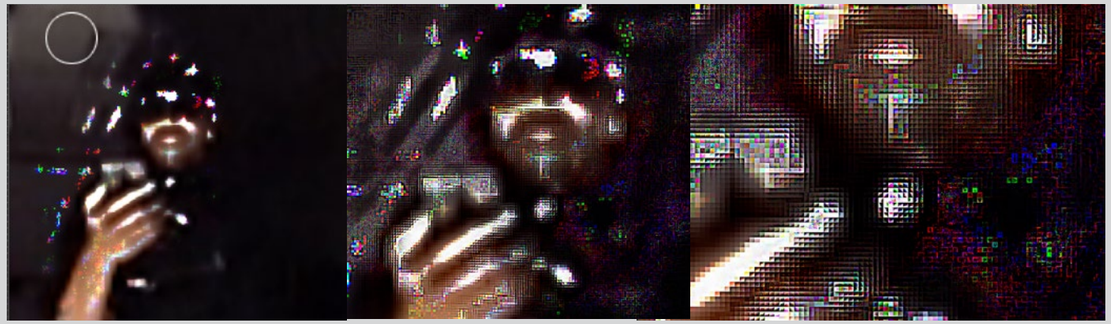




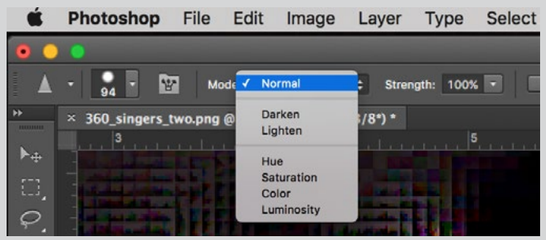
Then, select a brush by clicking on the icon (which has a numerical value) located next to the Sharpen tool. From there, choose a brush size that's fairly big:



Now "paint" with the brush on your image to sharpen its pixels.

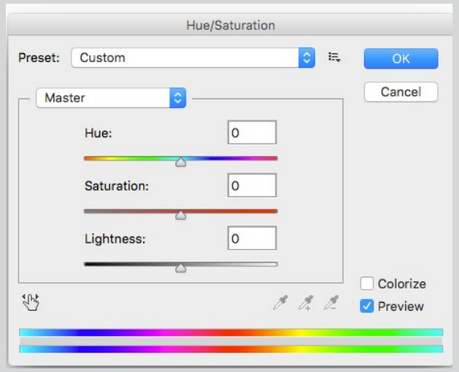


You can achieve a range of effects by changing the Sharpen Tool's "Mode" (button to the right of the brush). You can select "Lighten" to reveal and amplify the pixels in the shadowy, dark areas. For the opposite effect, select "Darken." We'll talk about Hue and Saturation below, approaching it differently. For glitching, I don't pay much attention to Color and Luminosity here, but you can certainly explore them. You can also Zoom-In (on toolbar), take a screenshot, and then work on this very detailed image.

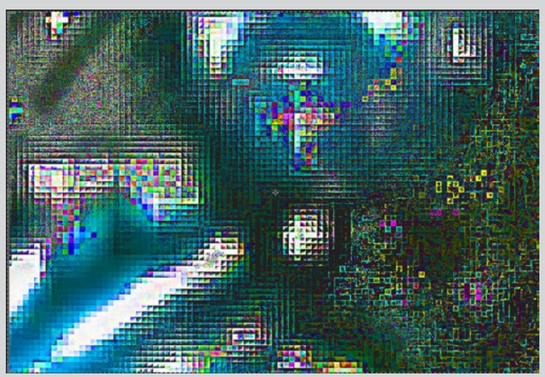


Subvert the Primary Colors. Basically, this means exploring, via PS, the primary colors (or lack of them) in your image. We're going to glitch these colors by amplifying them beyond the norm.

On the main PS menu, go to Image > Adjustments > Hue/Saturation. Shift the Hue (color) of an image by moving the Hue slider R or L—sliding it all the way out of normal range if you want. "Saturation" amplifies the intensity of the colors, and "Lightness" affects the overall light level in the image. If the Preview button is checked, you can see the immediate effects on your image.



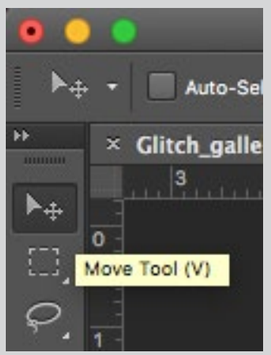
Here's what my sharpened image looks like with a wacky Hue/Saturation variable:



Shift the Channels. This makes for some interesting glitches because—as you may already know—every image is made up of red, green, and blue light; thus, when you move the corresponding PS channels out of range, this will change and disrupt the light content of an image. As shown in the graphic below, the RGB channel is the composite of the three other channels. (Note that the numbers on the channels are not values; they stand for shortcut commands.)

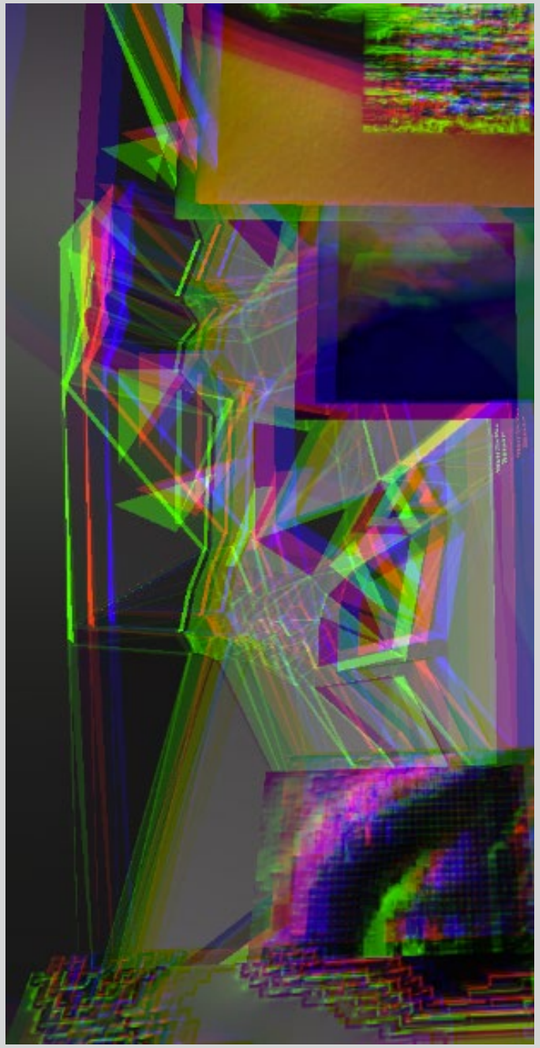
To use "Channels" to glitch an image's light content, go to the main PS menu > Window > Channels. Click on the Red Channel. This'll show you a B/W image on your main screen that represents the red light content of your image.

Select your image (main menu > Select >All) and click on the Move tool/icon on the toolbar on the left (usually the first icon).



Clicking on "Move" will, in essence, unlock the movement of the image's pixels. Then, click on the arrow keys on your keyboard to actually move the pixels, while you can watch the image on your screen shift around in response. To preview your actual changes, click on the RGB Channel in the Channels window. If you like the result, save the image as is. If not, play around with the blue and green channels. It's cumulative. What you've changed in red will be retained.

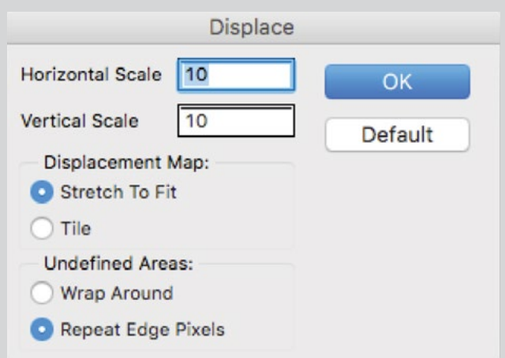
When you're done, your image will have offset images in red, green, and blue (in various combinations), similar to seeing a 3D-image without the clunky glasses.



Find Other Glitch Filters. Use Filters in PS to add visual "noise" or other disruptions (that mimic video interference, scanning problems, etc.). Go to Filter > Filter Gallery to compare all the filters at your disposal. The "noisiest" filters are 1) Brush Strokes > Spatter; 2) Distort > Diffuse Glow; 3) Sketch > Photocopy or Reticulation; 4) Stylize > Glowing Edges; and 5) Texture > Texturizer. All of these are great tools in exploring your glitch vocabulary.

Distort and Displace. PS comes with a Displacement filter that enables you to distort one image with the light/dark values of another. PS refers to the second image as the "displacement map." Try out this filter with two regular images. Then try it with two glitched images.

From the main PS menu, go to Filter > Distort > Displace. You'll get a Displace window with variables that you can play around with.



The Horizontal and Vertical Scale values increase/decrease the amount of displacement in pixels (and 10/10 default should work in general). Under "Displacement Map," "Stretch To Fit" will stretch the second image (displacement map) over the entire image. If you choose "Tile," your second image will be repeated multiple times until this fills in the original image. In "Wrap Around/Repeat Edge Pixels," I always pick "Wrap Around" because it seems to deliver more organic-looking results. (You won't see the values you choose implemented until you finish the steps below.)

Click "OK." This'll prompt you to pick your second image, the Displacement Map (which must also be in Photoshop's .psd file format). Click "Open" to incorporate the second image. Below are some displacement examples.

Note that "Distort and Displace" is a visual version of the "Convolute" exercise in the SoundSalad/Sound-Hacking tutorial of this book.

II. Glitch Images via TextEdit (Mac). Here's a way to create some interesting glitch accidents. In this process, you'll apply file extensions to image files that aren't normally used on them and then manipulate these images in an "inappropriate" application. Try it out. Actual results will vary!

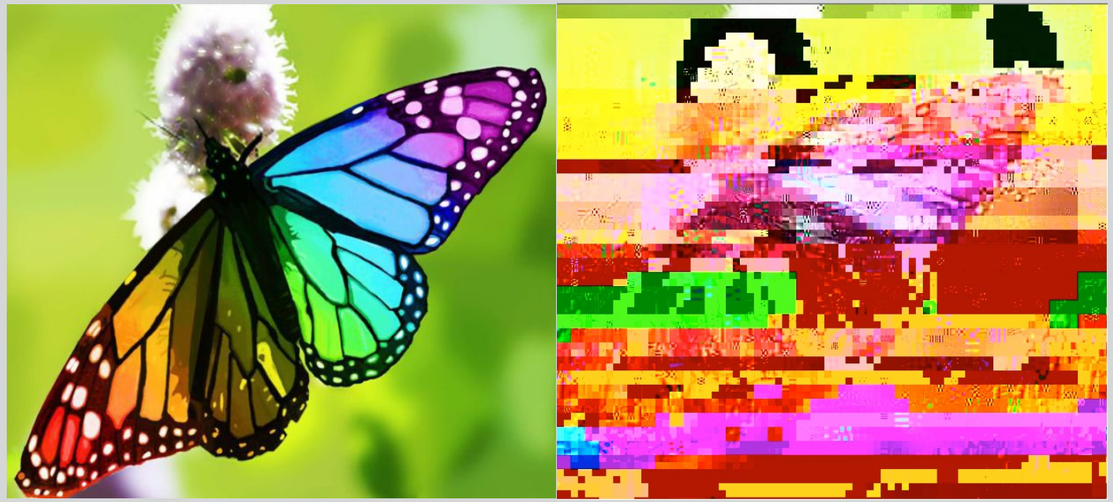
You'll need TextEdit open (free on Mac), hopefully already on your dock. Also, make sure you have your "Preview" App on your dock (also free with Macs). When you finish these .jpg glitches, you can drag them into "Preview" or open them up in Photoshop for more glitching.

Steps:

- Find a .jpg image (not .png or .gif) and copy it to the desktop.
- Change the file extension from .jpg/jpeg to .txt or .html.
- Open the new file in TextEdit—where your image will be transformed into gibberish text.
- Make some changes to that text: Add new gibberish and/or cut and paste small sections of the text onto other parts, etc.
- Save the file back to your desktop, and change the extension back to .jpg. (The thumbnail should appear to be some kind of image.)
- View the file by opening it in "Preview" or in Photoshop.

Note: Play around with a minimal amount of changes at first to get a sense of the "breaking point" of a file—at which point it won't open in PS. Take screenshots of your files often. If an over-glitched file won't open in PS, take the image's screenshot

and work with it in PS. If you're successful, your transformation might look something like this:



III. The Rutt-Etra-Izer Component. For a quick and easy glitch mechanism for still images, go to this free online app **[P.1.4]** and drop in your image. Note that this application is unrelated to the Quartz Composer Rutt-Etra plug-in for datamoshing videos, discussed below.

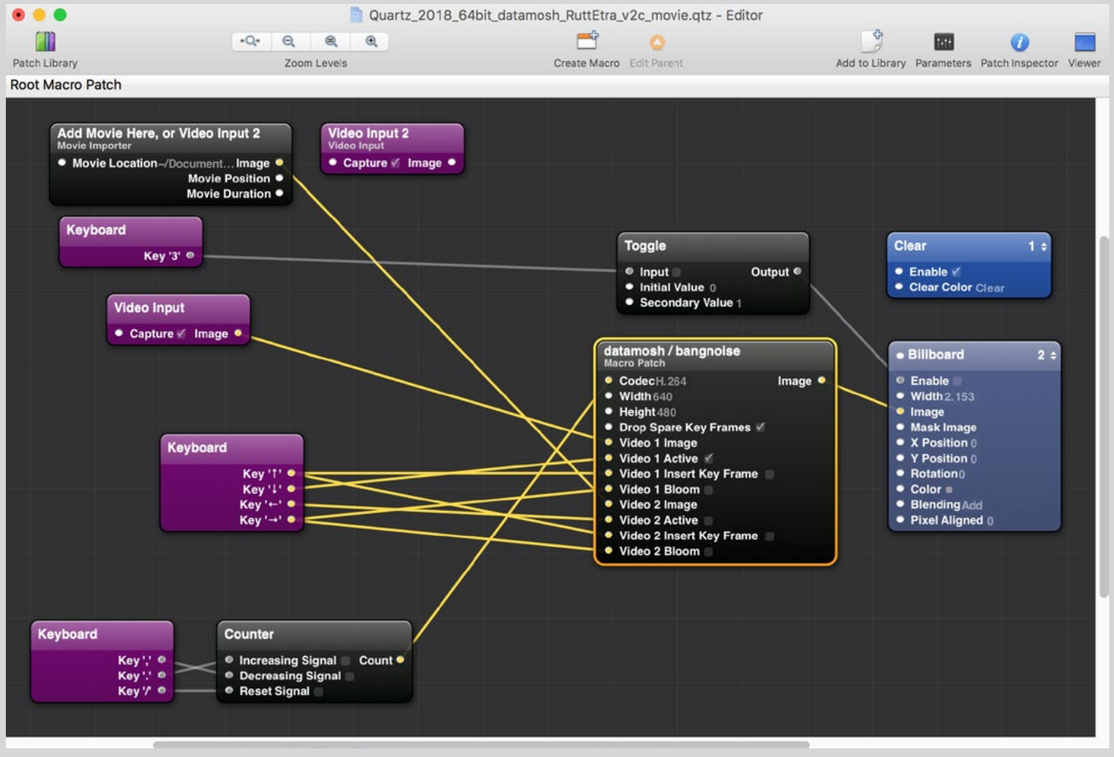
Here's my cat before and after:

IV. Quartz Composer with Datamosh and Rutt-Etra Plug-Ins. Quartz Composer is a free app (for the Mac) for making music visualizations, widgets, and screen savers, but this App—along with its plug-ins for Datamoshing and Rutt-Etra—also provides the raw material for glitchy video elements. I use the plug-ins, for instance, when I do live shows with video-jamming and/or to include interactive video elements to dance performances—i.e., dancers who interact in real-time with video elements generated by their live dance moves.

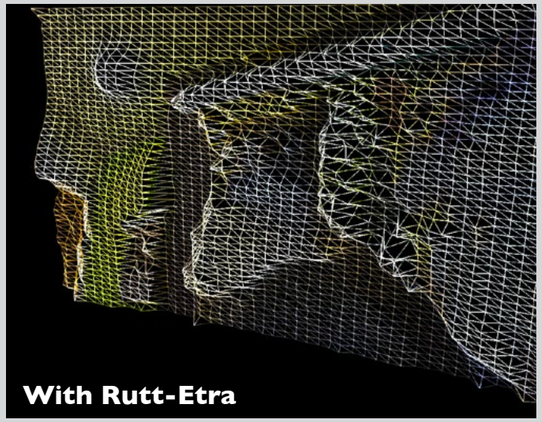
The basics of downloading and installing Quartz Composer (QC) is discussed in detail in a number of sites devoted to this tool. If you're a multimedia designer, I believe it would serve you well to walk through some tutorials in QC, then some further ones for the Rutt-Etra and Datamosh plug-ins for QC (where the real glitching takes place). Rutt-Etra and Datamosh are also currently free. Here are links for some tutorials to get you started:

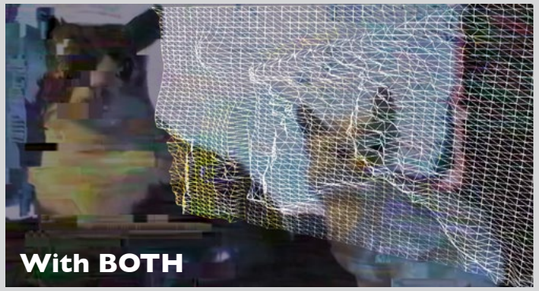
[P.1.5] Quartz Composer
[P.1.6] Rutt–Etra
[P.1.7] Datamosh

Once you take a look at these tutorials, you'll realize that QC looks a lot more complicated than it is. Yes, it's a programming environment, but no coding is required. You'll link your project's components together with "wires" or "spaghetti" that are created when you click on one element and drag/connect it to another. You'll be able to visualize and monitor the results of this schematic or "patch" on an adjacent screen. My patch below has purplish boxes for input (the videos I'll include in a jam, for example) and black boxes for instructions I want to apply to the video input. (Note that "keyboard" in my QC patch refers to commands for a computer keyboard, not a musical instrument.) The blue boxes represent my project's output. To the novice, it might not look worth the learning curve, but it's a quick and efficient way to spontaneously add a video, say, to a live video-jam or dance performance.



Based on the schematic above, here are some variable graphics.





Note: The Rutt-Etra-Izer and the QC R-E plug-in are based on the Rutt-Etra analog video synthesizer, built in the 1970s by Steve Rutt and Bill Etra. This synthesizer was put to good use by Nam June Paik and a whole generation of video artists.

PRACTICUM: SOUNDSALADS AND SOUND-HACKING WITH FREE SOUND SOFTWARE

This tutorial is for the complete (sound) novice, which many of my multimedia students are —even those who are undergrad animators, visual designers, and filmmakers! In this tutorial, you'll first put together a succession of unpredictable sound events from a variety of sources to include these events in a stereo track—a SoundSalad. You can use this track in a number of projects, including the Unity 3D scene project (see tutorial in this book). In the subsequent sound-hacking exercise, you'll make a richly textured, slowly evolving ambient track from some noisy source audio. Both of the tutorial exercises include what I'd call "visual sound editing." At this point, don't worry too much about what your tracks will sound like—it's one way of turning off any default preferences about how you define sound, noise, and music.

Remember—you're NOT creating a song (as in a multi-track production—lead vocals, backup harmonies, rhythm section, bass). You're exploring the very building blocks of sound—pitch, dynamics, and timbre (and see Appendix I for my explanation of those terms). The end result may be noisy and might sound a little "arbitrary." These projects may not fit your personal definition of music, but the results should be structured, original sound.