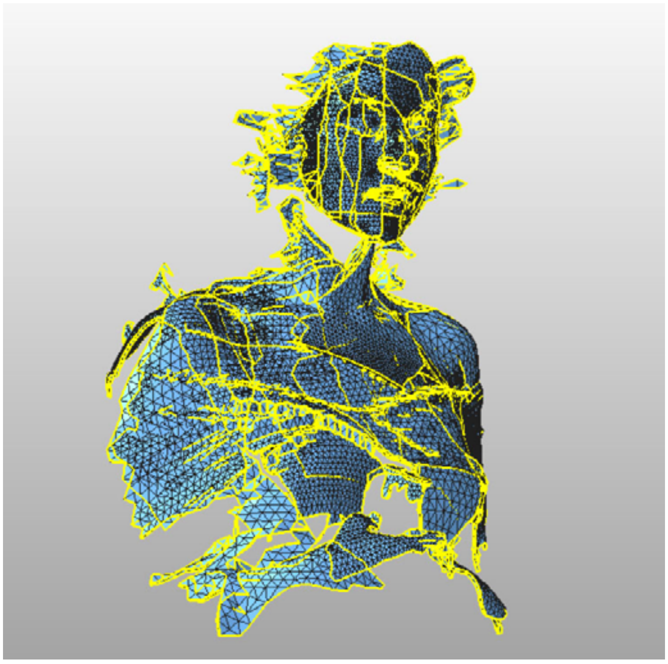
Tutorial for translating glitchy files into a form which 3D printers can understand

ON PRESERVING GLITCH

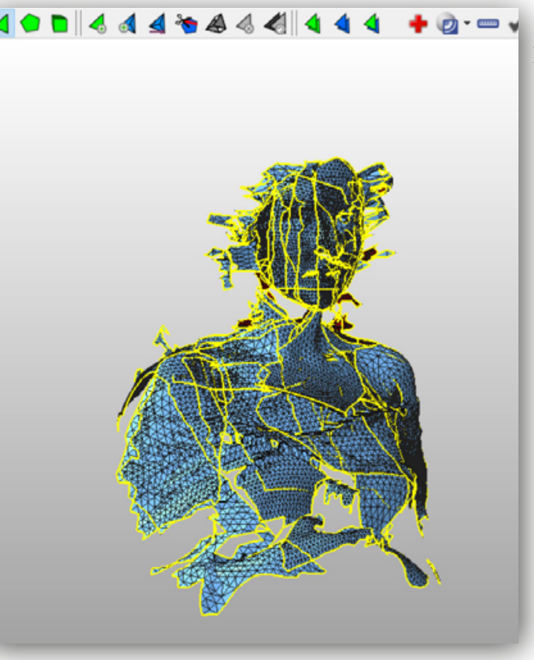


In a folder on my desktop, I keep a folder entitled ‘weird errors’. I also advise all my students to do the same. It’s inevitable that, in the course of learning to 3D model, scan or print, odd results can arise. Instead of committing these files to the wastebasket, I tell students to keep them all, along with a short note detailing the steps that led to their creation. As an artist I have always found it imperative to keep an eye open to the mistakes, glitches and strange detours that crop up while pursuing a project. (I now outsource most of my fabrication, but an unfortunate consequence of this post-studio practice is that errors are seen only by fabricators. I also ask my fabricators to send me images of any errors that happen and corrected before the artist can observe them and perhaps turn them into a new body of work.)
In my own practice, I misuse high-end 3D laser scanners in order to yield glitchy results. It has been a challenge to learn to 3D print my files without erasing their native aesthetic. Most tutorials on 3D printing focus on repair: patching holes, deleting areas of bad data, and smoothing surfaces in order to produce a result that looks ‘natural’ and closer to the original. In my practice, I do the opposite. Of course I manipulate my files, but I try to expose the glitches that arise along the way instead of ‘correcting’ them. An incomplete 3D scan reveals the incompleteness of human vision; made from a single perspective, it only takes a slight rotation to reveal the gaps and blind spots, all the things we do not see.
Below is a brief tutorial on how to process a glitchy file in order for a 3D printer to understand it.

STEP 1: Obtain a glitchy file (image 1)

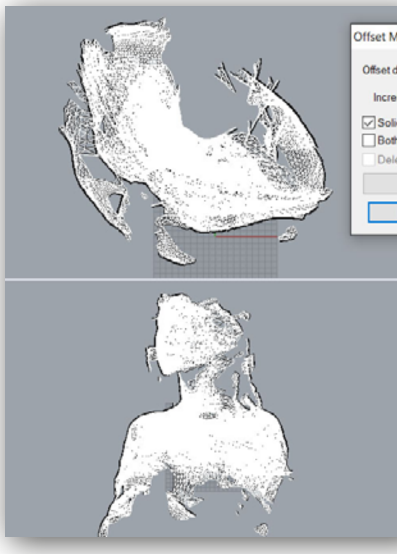
This could be a poorly done 3D scan, or a 3D file that arose when you did something uninspected: turned a sphere inside out, or ran an operation that caused your object to break out in a rash of polygons. The most interesting results often arise when you use a tool for a purpose for which it was not designed: in my case, try to 3D scan a moving body. My file is an incomplete 3D laser scan of my body, swathed in a bed-sheet and holding the 3D scanner's magnetic locator cube - which in the image appears to be some kind of ceremonial object. My eyes are closed to protect them from the scanner's not-quite-eye-safe laser beam - but it appears that I am deep in meditation.

STEP 2: ANALYSE THE FILE IN NETFABB BASIC (IMAGE 2)



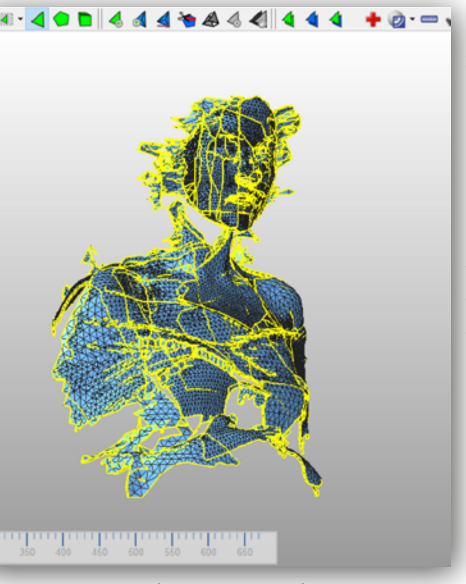
Import using part->add in order to view and get some basic information on your file. If the dimensions do not look right, use part->scale to adjust them. Then look to see if the surface is open or closed. If you hit the repair icon (red cross on the toolbar) Netfabb will provide some statistics, including the number of holes in the file. You will be able to see if it is an open surface as one side will be red (in Netfabb, green is 'outside' and green is 'inside'.) In order for a 3D printer to understand a file, it needs to meet certain basic conditions. It must be a single 'watertight' surface; all the polygons must be facing the same way (polygon faces have a direction called a 'normal' that isn't usually visible in the GUI, but the printer software will need to know which side is the inside and which is outside, and this can get scrambled in glitchy files); and it needs to conform to the limits of the material you want to use for printing. This is called wall thickness and is important so that the final piece does not break on printing, or on removing from the machine. If you did not create your file in a solid modeling program like TinkerCAD, chances are it has no inherent thickness (my scans are merely skins with a zero value for thickness). The first step is to turn your shape into a solid, three-dimensional object. The very quickest way to solidify without filling in holes or cracks is to offset the mesh. This creates a copy of the surface at a uniform distance. After extensive experimentation, the only way I have found to perform this operation (if you don't have $10,000+ to drop on Geomagic) is to use Rhinoceros. It's not free but it does have a 90-day trial and a Mac beta here (if your file is largely solid, e.g. a scan of an object in 360 degrees that has only small holes, you may omit this step).

STEP 3: Offset in Rhino (image 3, file - BustOffsetInRhino)



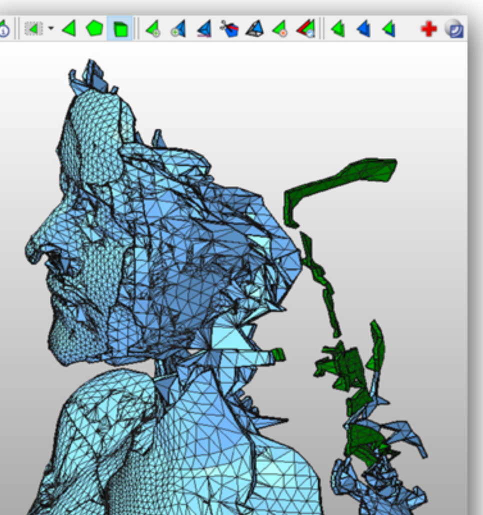
Make sure your file is the size that you want it to be, then offsetit by a minimal amount. Use OffsetMesh in the Rhino command line and make sure that 'solidify' is checked. Export it as an obj and import it back into Netfabb. Choosing the correct wall thickness is critical: if you are ordering through a service bureau like Shapeways, the file must pass wall thickness requirements or it will be rejected. There is little room for experimentation with large factories or service bureaus, as one print that does not go through correctly can crash their build and cost thousands in lost revenue. You may have more luck if you have access to your own 3D printer at a school or lab. Another reason to keep the piece as thin as possible, but no thinner, is that 3D printing material is usually priced by the cubic centimeter. You will also need to carefully consider wall thickness if you plan to cast the piece later: my foundry have had a challenging time with casting any prints that are under 3 or 4mm thick. (The irony and hubris of taking something as ephemeral as a digital glitch, then printing and immortalizing it in bronze, which will outlive us all – unless melted down for scrap when the shit inevitably hits the fan – is not lost on me.)

STEP 4: Netfabb Repair (image 4)



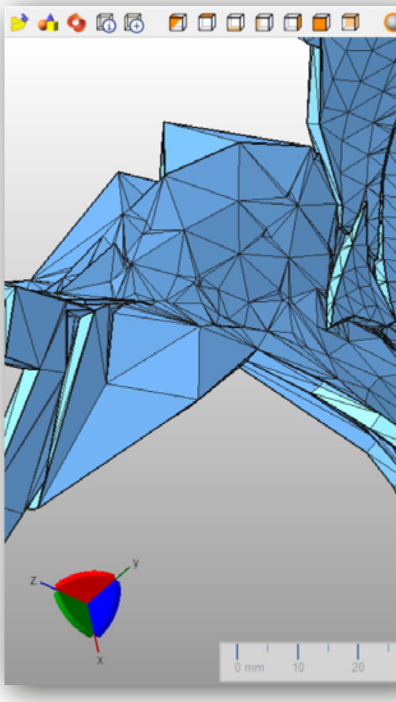
I have found that the most reliable repair workflow in Netfabb involves running repair operations in this order: stitch triangles, fill trivial holes, fill all holes, fix flipped triangles, remove degenerate faces. (Opt for this if you can rather than the automated repair. You may need to bump up the Stitch Triangles tolerance if you still see numerous small holes highlighted in yellow.

STEP 5: Remove Extra Shells in Netfabb (image 5)



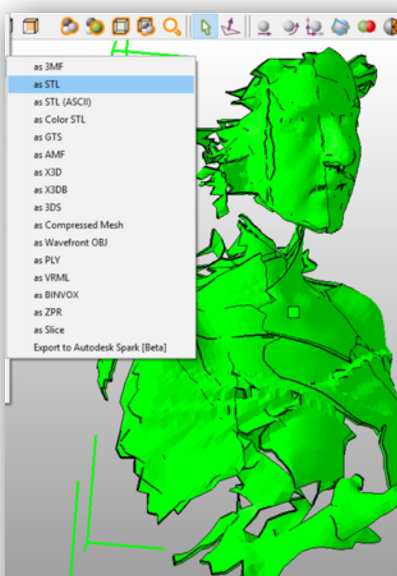
In the Netfabb repair module, hit 'select shells' to select the main object, then hit 'toggle selection' to select all the other extraneous shells. (This is also necessary in order to avoid rejections. If any pieces are now selected that you don't want to delete, you will need to join them on manually to the main shell by deleting faces and then adding triangles to join them on.) Hit the delete key to remove these. The shells window in the repair module should now have a count of 1.

STEP 6: Manual Repair (image 6)



The workflow above is a brief overview of the most minimal steps that you will need to obtain a printable stl file. I have omitted any automatic operations that fill, smooth or correct the appearance of the file (offsetting first in Rhino preserves any existing holes, instead of filling them in.) However, there are no medals for authenticity in glitch art, and you are within your rights to alter your file in any way you wish before printing it. Avoiding auto-repair means you may want to delete or rebuild areas of your model. You can do this by selecting individual triangles or surfaces (tip – change the surface tolerance slider to select smaller or larger areas at any time, or use CTRL+ or CTRL- to grow or shrink your selection) then delete those areas, and fill the holes either in the repair options or by hitting 'add triangles' and putting in new triangles to fill gaps.

STEP 7: Finbal Repair and Export (image 7)



Check that your file is now a single error-free shell in Netfabb. Then export as an STL file.