

Enhancing the Shroud of Turin

Team Awesome

Adam Brill

Brian Griffin

Janell Joyner

Tyler Phillippe

In our CSCI 410 class, Prof. Schneider showed us an image of the Shroud of Turin. He wanted the class to come up with a project idea and to execute it. The class brainstormed a few ideas about what could be done to help analyze the Shroud. Our original plan was to remove the cloth thread from the background of the image. We also noticed the different blood, burns, and scourge marks all over the Shroud and the image on the Shroud. It was decided that our goal would be to clean up the image on the Shroud by removing the cloth and replacing it with something better. In order to accomplish this, we broke down our idea into steps. The color ranges of the different parts of the image would have to be collected, removed, and then replaced with a false color. We later discovered how difficult our goal was and with our time constraint we decided to change our goal to enhancing the Shroud. We kept most of the checkpoints from our previous goal. The new checkpoints would be to identify the different parts of the Shroud and to replace them with a false color.

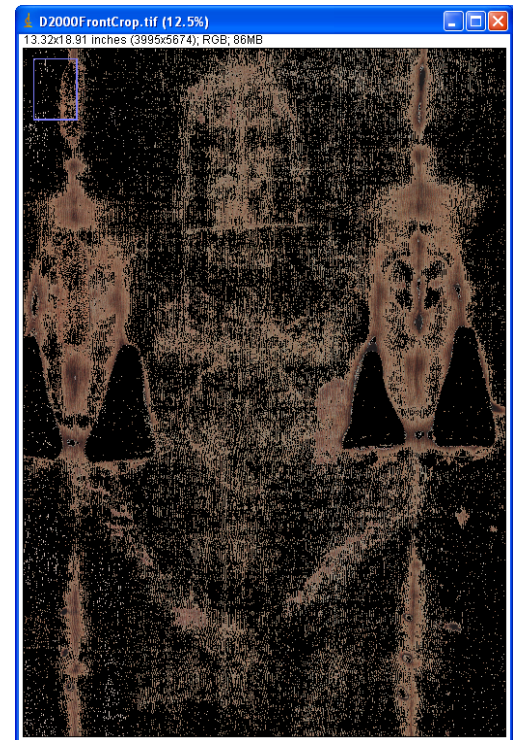
ImageJ was the image editing program that we used to edit the image. We did most of our work on a front crop of the Durante 2000 Shroud image and we needed an algorithm to help us remove the colors. We had several different checkpoints that we reached that allowed us to advance towards our goal. Once we had a plug-in to find the color ranges we needed to find a way to remove the color and replace it.

We created a plug-in that could identify the desired color ranges and then replace it with a false color. This plug-in would become the main plug-in used for our project, the FLC plug-in. The first attempt to replace colors went well. To get this to work we had to collect a set of RGB pixel values. We put these values into an array list that could be inserted into the FLC plug-in. From this array list, we choose the ranges that the pixels had to fall in to be replaced. Then the plug-in would randomly select a set of pixels from the array list to be used as the replacement

pixels. The plug-in replaced the cloth background pixels with black pixels. This mini success showed us that we had a decent amount of work ahead of us because there was a considerable amount of noise remaining in the image due to the similar color ranges of the pixels. There were also random pixels that did not match the pixel ranges around it. We needed to refine the values by giving it better values to work with.

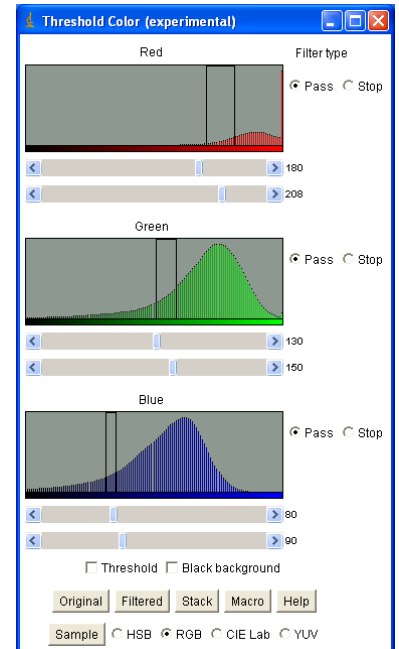
Before we began to refine our FLC plug-in, we decided to try editing the image before running the plug-in on it. Different filters with a number of parameters were run on the Shroud in an attempt to fix the image. We ran a median filter with a five pixel radius, a mean filter with a five pixel radius and three pixel radius, a minimum filter with a three pixel radius, a maximum filter with a three pixel radius, and a Gaussian Blur with a twenty pixel radius on the image. All of these filters blurred the image to a degree and made the thread marks of the cloth less noticeable. After trying a different combination of these filters, we decided that we would use an image that has been modified by a mean filter with a five pixel radius and an enhanced contrast filter with a five pixel radius. This brought the colors in the image and made it easier to pick them out.

To help refine the code, we began to search the Internet and ImageJ itself for different plug-ins and how to write additional plug-ins that might aid us. One plug-in that we created found the highest and lowest RGB pixel values. This told us where to limit our color range, but the plug-in gave too broad of a result to be used in the FLC plug-in effectively. While searching the internet for a plug-in, the RGB_Measure_Plus code was found. To work this plug-in, a threshold of the highest and lowest RGB had to be entered. The code would then print out the



“average”, “standard deviation”, and the “area fraction” of the RGB pixel values. Once again this was a good plug-in, but not what we were looking for.

The search for useful plug-ins continued with a search through a color image processing book and searching through ImageJ plug-ins. The book provided a more in depth description of what we had learned about color imaging, but did not provide us with any further inspiration. A thresholding plug-in within ImageJ was found. This plug-in was used to evaluate the image in different color spaces and it was used to aid in the completion of our goal. With the thresholding plug-in, we were now able to move forward with our plans.



Since we were able to identify and replace the cloth pixels, the next thing that we did was identify the blood, burns, and scourge pixel ranges. A section of code was added to our FLC plug-in to identify the blood and layer it to show the different densities in the blood on the image. This addition to the code did an exceptional job. It was able to identify the blood pixels correctly and give them a red false color, but it also colored in the burn pixels. The main plug-in would once again need some more refining.

To see what we had accomplished so far, the FLC plug-in was set to replace all of the components of the image with a false color. The blood and burns were turned red, the cloth was given a more uniform color, and the body was turned a yellow color. This revealed the problem spots on the Shroud that we needed to address. There were burn marks on the Shroud that were just about same color as the body along with other stray particles on the Shroud that matched the body



color. We needed a tighter threshold and more specific RGB pixel values to put into the FLC plug-in.

By using the thresholding plug-in and collecting multiple sets of RGB pixels values, we were able to get the FLC plug-in to give a better representation of the blood pixels in the image. It turned less of the burn pixels red and did not remove the blood pixels that we had already identified. We then replaced the cloth, blood, burns with a uniform cloth color and smoothed it over to get a natural look. This gave the image on the Shroud an interesting look, but not what we were looking for. As we continued to add RGB pixel values and continued to tighten the threshold on the image, there were less extraneous pixels on the image. The image looked a little cleaner, but there was still work to be done.

Prof. Schneider suggested to us that we try to roll over the pixels. That is, he wanted us to try to see if we could push the body pixels over the blood pixels to help clean up the image. The closest thing we could do, that would execute his idea, was to replace the blood pixel colors with body pixel colors. Half of the team began to collect RGB values of the body, while the other half worked on tightening the threshold even more on the image. When we combined all of what we found into the FLC plug-in, we got an image that had body colored blood pixels. These blood pixels came out darker than the actual body color.



We were unable to get the Shroud as clean as we had wanted to, but we did achieve other goals. The FLC plug-in that we used was able to find and replace color ranges that were identified as blood, burns, and scourge. For those who would like to continue on with our

project, they might try to attempt to find the slightest difference in the body and burn colors.

Another project might be to attempt to rollover the pixels, so that all the elements affecting the image are removed.

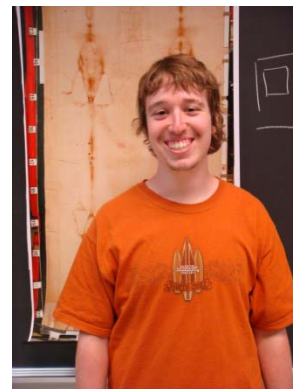
[FLC Plug-in code](#)

[Enhancing the Shroud ppt](#)

Who's Team Awesome?



Adam Brill is a computer science major at Bridgewater College. He was one of the primary programmers on the team as well as the developer of the FLC plug-in.



Brian Griffin is a computer science major and mathematics minor at Bridgewater College. He was a secondary programmer and helped to set up the array lists for the program.



Janell Joyner is a computer science major and mathematics minor at Bridgewater College. She was a secondary programmer and maintained documentation of the work by the team.



Tyler Phillippe is mathematics major and computer science minor at Bridgewater College. He was the other primary programmer and operated the thresholding plug-in.